UNIVERSITI POLY-TECH MALAYSIA

UPTM Credit Transfer System

MOHAMMED MUQSIT BIN OSMAN

BACHELOR OF INFORMATION TECHNOLOGY (HONS) IN COMPUTER APPLICATION DEVELOPMENT

UNIVERSITI POLY-TECH MALAYSIA
Faculty of Computing & Multimedia

UPTM Credit Transfer System

MOHAMMED MUQSIT BIN OSMAN
AM2408016622

FYP4144

DECEMBER 2025

Declaration of Originality

This project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet, etc.) has been acknowledged within the main report to an item in the References or Bibliography lists.

I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Copyright Acknowledgement

I acknowledge that the copyright of this project and report belongs to Universiti Poly-Tech Malaysia.

Signed: 

Date: 08/04/2025

Office Stamp

Abstract

This project presents the development of the UPTM Credit Transfer System (CTS), a web-based platform designed to improve the management of Credit Transfer Applications (CTAs) at Universiti Poly-Tech Malaysia (UPTM). The existing credit transfer process relies heavily on manual document handling, fragmented record storage, and individual syllabus review practices, which contribute to delays, limited traceability, and inconsistent evaluation outcomes. In response to these issues, this project focused on three main objectives: to analyze the requirements of an automated credit transfer system, to design a centralized digital repository for credit transfer records, and to develop a staged syllabus comparison module using decomposed prompting. The system was developed to support the end-to-end credit transfer workflow for four main user groups, namely students, Head of Programme (HOP), Resource Person (RP), and Academic Affairs Students (AAS) staff. Students are able to submit applications, upload transcripts and syllabi, and monitor application progress. The system then extracts transcript information, matches courses against articulation records, and stores applications, supporting documents, and review outcomes in a centralized repository. For cases that require deeper academic evaluation, the system provides a staged syllabus comparison workflow consisting of preprocessing, standardization, mapping, verification, scoring, and summary generation. This structure helps RP users perform more transparent, reviewable, and consistent syllabus evaluation before making approval or rejection decisions. The completed system digitalizes the major stages of the credit transfer process, improves access to historical records, and supports more structured coordination between students and reviewing staff. By combining centralized record management with staged AI-assisted syllabus comparison, the proposed UPTM CTS provides a more efficient, traceable, and systematic approach to credit transfer administration and offers a stronger foundation for future process improvement at UPTM.

Table of Contents

List of Figures

List of Tables

Acknowledgements

Alhamdulillah, all praise and thanks are due to Allah SWT for granting me the strength, patience, and opportunity to complete this Final Year Project. There were moments when the work felt smooth and rewarding, and there were also moments when it felt mentally heavy and difficult to carry forward. Through all of it, I was given enough strength to continue. Completing the UPTM Credit Transfer System means more to me than finishing a university requirement. It represents a personal journey of growth, discipline, and perseverance.

My deepest gratitude goes to my beloved parents. No words can truly repay their love, sacrifice, and endless support throughout my life and throughout this journey. More than anything, I carry their dua with me. In moments of exhaustion, doubt, and pressure, it was their prayers and the thought of their belief in me that gave me the strength to keep going. Whatever good is found in this project is inseparable from the blessing of their support.

I am especially indebted to Miss Noornajwa binti Md Amin, my Final Year Project Supervisor and Head of Programme for the Bachelor of Information Technology (Honours) in Computer Application Development (CT204). Her guidance was present from the early stage of shaping the project idea until the completion of the system and report. She did not simply supervise this work, but continuously pushed me to think more carefully, improve the direction of the project, and maintain a better standard in both development and writing. Her patience, advice, and trust played a major role in shaping this project into what it is today.

My sincere appreciation also goes to Madam Norfazlina binti Johar, the Final Year Project Coordinator, for managing the FYP process with clarity, dedication, and care. Throughout the journey, her coordination, reminders, and support helped students stay on track during a period that could easily become overwhelming. Her contribution may often be seen behind the scenes, but it was important to the smooth progress of this project.

I would also like to extend my appreciation to Madam Zurina binti Jusoh, my examiner and Head of Programme for the Bachelor of Information Technology (Honours) in Cyber Security. I am grateful for the time, attention, and professional evaluation given to this work. Her comments and feedback helped me view the project from a sharper academic perspective and encouraged improvements that strengthened the final outcome of this study.

A special word of thanks goes to my housemates, Imran, Zafran, and Haidhir. They were not only people I lived with, but brothers in struggle throughout this Final Year Project journey. We shared ideas, discussed new information and technology, exchanged opinions on each other's work, and helped test one another's systems. Because we were all going through FYP at the same time, they understood the pressure,

the deadlines, and the frustration in a way few others could. Their companionship and support made the difficult days much easier to face.

This project was also developed during the month of Ramadhan, and that experience tested me in a different way. Trying to stay focused, think critically, solve problems, and keep the momentum of development while fasting was not easy. There were times when concentration came slowly and progress felt smaller than expected. Even so, that period taught me patience, consistency, and the importance of continuing the work with sincerity, even when the process felt demanding.

Finally, I would like to thank Universiti Poly-Tech Malaysia, the Faculty of Computing & Multimedia, and everyone who contributed directly or indirectly to the completion of this project. Being recognized among the Top 6 is something I receive with gratitude and humility, and I know it was only possible because of the guidance, prayers, feedback, and trust of many people around me. I sincerely hope that the UPTM Credit Transfer System will bring benefit to UPTM and, in its own way, open the path for even greater innovation in improving the university's academic processes in the future.

1 INTRODUCTION

1.1 Introduction

This chapter introduces the overall context and framework for the project. It begins with the project background, which describes the institutional setting and current practices in the credit transfer process at UPTM. The problem statement follows, identifying the specific issues and limitations that the project seeks to address. Next, the objectives of the project are outlined to provide clear direction for the proposed solution. The chapter also defines both the product and project scope, establishing the boundaries and intended functionalities of the system. target users also are repeatedly described with very detailed explanations. This is important to design requirements for credit transfer system.

1.2 Project Background

Universiti Poly-Tech Malaysia (UPTM) is a private higher education institution in Kuala Lumpur, Malaysia, offering a range of diploma and degree programs in fields such as computer science, information technology, and creative digital media. The Faculty of Computing & Multimedia (FCOM) is one of the core faculties at UPTM, providing academic programs including the Diploma in Computer Science (CC101), Bachelor of Information Technology (Honours) in Computer Application Development (CT204), and Bachelor of Information Technology (Honours) in Cyber Security (CT206).

In a study by Azizan et al. (2021),a credit transfer system is a process that enables course credits students have earned in one area of study or institution to transfer to another. This flexibility assists their educational journey without hampering how they can progress. Essentially, credit transfer aims to reduce the number of credits needed for a degree, making the process as streamlined as possible. This is the basis of the credit transfer system. If a student receives credit for one course and then takes the same course over again at another school, is not that unfair to him? In UPTM, credit transfer processes are divided into these two categories. As NACAC (2022) shared Horizontal (or lateral) credit transfer means students are transferring credits at the same academic level, either between programs or institutions. For example, the transfer of credits from one diploma program to another. Vertical credits transfer is the way that credit transfers from a lower level to a higher level. For example, credits which are granted advanced standing from the diploma program to bachelors' degree series.

Currently, the credit transfer process at UPTM is primarily manual, involving several administrative steps and multiple parties. Things usually begin with an FCOM student applying for credit transfer, often due to a change of program, re-admittance or advancement to a higher qualification. The student first of all

needs to make contact with Head of Program (HOP), who gives them an outline of the process and supplies what relevant forms. Forms like Credit Transfer Application (CTA), articulation list, and copies of program structure. The HOP then reviews student's academic transcript, by doing some preliminary assessment of transferable credits based on course titles and articulation outcomes or prior knowledge of articulation outcomes (Azizan et al., 2021).

In the case in which the students is from institution or courses that are not listed in the articulation list. HOP will contact the students and request the related syllabus after HOP identify courses that have higher chances of success for credit transfer. Currently, HOP will gather student's that eligible to credit transfer and during the Credit Transfer Application (CTA) process, students are required to bring their transcript as well as their related syllabus if applicable. After students have filled out CTA they will submitted along with the supporting materials, HOP then will revise each student's CTA to make sure everything is correct. After HOP obtained the syllabus, HOP will handle it to Resource Person (RP) and RP will do the syllabus evaluation. RP will refer to relevant resources and documents to make sure the judgement is correct. This manual review process may lead to the negative consequences as Chandrasekaran & Mago (2022) identified that manually comparing syllabus prone to human error and misjudgement resulting in unfair outcomes for students. After RP completes the syllabus evaluation, HOP will confirmed and review the syllabus evaluation form and then HOP will update the articulation list. Next, CTA will be reviewed by dean or deputy dean before those CTA got sent to Academic Affairs Students (AAS). AAS staff then will input all those finalized CTA into Campus Management System (CMS). Once AAS staff finish processing the CTA in the CMS, they will sign the CTA as a confirmation. Then students will retrieve the CTA at the AAS. The problem arise when there's too many CTA as it will pile up quickly and make the retrieval process quite harder for students.

Moreover, the current Credit Transfer Application (CTA) process remains largely manual and time-consuming because it involves multiple stakeholders and requires strict adherence to the workflow. A delay or error at any stage can affect the entire process, while each stakeholder faces different challenges in fulfilling their responsibilities. To simplify and standardize this workflow, the UPTM Credit Transfer System (CTS) is developed to digitalize CTA management and automate syllabus comparison. The system is implemented as a full-stack web application using Next.js, which handles the user interface, authentication, server-side processing, API routes, and core business workflows. Application data is stored in PostgreSQL through Prisma, files are managed in SeaweedFS via an S3-compatible interface, and AI-related processing is coordinated from the Next.js server using supporting services such as Stirling PDF for document text extraction and Ollama for transcript extraction and syllabus comparison.

1.3 Problem Statement

This section presents the problem statement of the research which is an integral part of any research work. The objectives the research aims to address are provided, these should be clear and of interest. As formulated by Srinivas et al. (2023), a good articulated problem statement specifies the exact gap or deficit in knowledge or practice, rationale for addressing it and with solid ground for constructing research question(s) and objective(s). The problem statement helps to establish the importance of the problem and keeps you focused on looking for a solution or new ideas throughout this research process.

1.3.1 Manual And Tedious Administrative Process

The current credit transfer workflow at UPTM relies heavily on manual processes, particularly for staff, including core activities such as handling the CTA and syllabus evaluation. As Abdul Aziz et al. (2025) stated that UPTM's credit transfer process is still manual, involving heavy paperwork, individual reviews, and long delays making it time consuming for staff and confusing or slow for students.

1.3.2 Lack Of Central System To Store Or Analyze Credit Transfer Records

Prior to the development of UPTM Credit Transfer System (CTS), finalized Credit Transfer Applications (CTAs) and supporting records such as syllabi, articulation mappings, and course evaluation outcomes were managed through manual and non-standardized storage practices, including physical documents and scattered files. This made historical decisions difficult to retrieve, increased the likelihood of repeated evaluations, and contributed to inconsistent syllabus assessment outcomes. More importantly, the fragmented handling of records prevented UPTM from using accumulated CTA data for reporting, trend analysis, and policy improvement. As Alotaibi (2021) notes, many higher education institutions collect abundant administrative data but fail to transform it into actionable knowledge because fragmented systems and limited analytical infrastructure cause decisions to be driven more by intuition than by evidence.

Without a centralized digital repository, UPTM had limited ability to convert credit transfer data into meaningful institutional information. For example, it was difficult to generate consistent reports on approval rates, processing time at each stage, or recurring workflow bottlenecks. As a result, decision-making remained reactive rather than evidence-based, opportunities for process improvement were missed, and institutional knowledge was not systematically retained. This gap in knowledge management justified the development of a centralized digital system capable of storing CTA records, managing supporting documents, and supporting data-driven analysis.

1.3.3 Evaluation Is Inconsistent Across Resource Person

The syllabus evaluation process at UPTM is not standardized, meaning there are no consistent, system-enforced guidelines for how RPs evaluate courses. While institutional policies define general criteria, their application varies significantly between individuals. This lack of uniformity leads to inconsistent decisions, reduces transparency, and undermines fairness in credit transfer outcomes. As Chandrasekaran & Mago (2022) identified that manually comparing syllabus prone to human error and misjudgement resulting in unfair outcomes for students.

1.4 Project Objectives

This section outlines the key objectives of the project, which define its purpose and guide the development and evaluation of the UPTM Credit Transfer System. Well-structured objectives provide clear direction, ensure alignment between activities and desired outcomes, and establish measurable criteria for success. To ensure these objectives are practical, focused, and actionable, the SMART framework which are Specific, Measurable, Achievable, Relevant, and Time-bound is applied. As Ayoub (2017) notes, this approach enables the formulation of goals that are not only clearly defined but also realistic and strategically aligned with institutional needs and long-term improvement goals.

1.4.1 To Analyze the Requirement of an Automated Credit Transfer System

This This objective involves identifying and documenting the functional and non-functional requirements of the proposed UPTM Credit Transfer System. It includes gathering input from key stakeholders such as AAS officers, HOPs and RPs through conducting interview and questionnaire. The outcome will be a comprehensive requirements specification that informs system design, ensures alignment with institutional workflows, and supports usability and scalability.

1.4.2 To Design A Centralized Digital Repository Platform For Credit Transfer Records

This objective focuses on designing a centralized digital repository platform to address the fragmentation and inefficiency in storing and using credit transfer records at UPTM. The platform will provide a structured and searchable environment for managing Credit Transfer Application (CTA) records, supporting documents such as syllabi, and related academic records such as articulation mappings and course evaluation decisions. In doing so, it will reduce reliance on physical documents and scattered digital copies while improving the consistency, traceability, and accessibility of historical credit transfer information. The system will also include reporting and analytical functions to generate insights such as approval or rejection trends, workload distribution, and average processing duration, thereby supporting more evidence-based decision-making and continuous process improvement.

1.4.3 To Develop a Staged Syllabus Comparison Module Using Decomposed Prompting

This objective focuses on developing a staged syllabus comparison module using decomposed prompting to evaluate source and target course syllabi in a more consistent, transparent, and reviewable manner. The module will process syllabus content through multiple stages, including preprocessing, standardization, mapping, verification, scoring, and summary generation, in order to produce structured comparison results. The generated outputs will support Resource Persons (RPs) in reviewing syllabus equivalency, adding comments, and making approval or rejection decisions. By structuring the comparison process into clear stages, the system improves consistency, reduces subjectivity, and provides better decision support for syllabus evaluation.

1.5 Scope and Target User

This section defines the boundaries of the proposed UPTM Credit Transfer System (CTS), the main users it is designed for, and the limitations that apply to both the project and the product. The primary target users are students, Head of Programme (HOP), Resource Person (RP), and Academic Affairs Students (AAS) staff, as these stakeholders are directly involved in the submission, review, evaluation, and completion of the Credit Transfer Application (CTA) process. This section also clarifies what will be developed in the prototype, what remains outside the scope of the project, and how the proposed system supports the transition from the current manual process to a more structured digital workflow.

1.5.1 Project Scope

The UPTM Credit Transfer System (CTS) will be developed as a web-based prototype using an appropriate software development life cycle (SDLC) to address inconsistencies, limited transparency, and inefficiencies in the existing CTA process. The project covers the major activities required to deliver a functioning prototype within fourteen weeks, including stakeholder engagement for requirements elicitation, workflow and interface design, database and system development, implementation of role-based modules, development of a staged syllabus comparison process using decomposed prompting, and testing using selected course and syllabus cases.

For academic delimitation, the project is limited to three programmes under the Faculty of Computing and Multimedia (FCOM), namely Diploma in Computer Science (CC101), Bachelor of Information Technology (Hons) Computer Application Development (CT204), and Bachelor of Information Technology (Hons) Cyber Security (CT206). Within this scope, the system is intended to support the CTA workflow from student submission until final export by AAS staff. Courses that can be resolved through existing articulation mappings may proceed without additional syllabus submission, while courses requiring further academic evaluation will follow the syllabus-based review process.

The project includes the development of a centralized digital repository for CTA records and supporting documents, a staged syllabus comparison module, workflow support for students, HOPs, RPs, and AAS staff, as well as reporting and export functions. However, the project does not include custom model training, fine-tuning of language models, direct integration with the Campus Management System (CMS), institutional agreement negotiation, or full-scale university-wide rollout. Finalized records can only be exported in formats such as Excel and CSV for manual entry by AAS staff. The project deliverables include a functioning web-based prototype, a project report, and presentation materials for evaluation.

1.5.2 Product Scope

The product scope covers the development of a web-based credit transfer management and syllabus comparison system. The product consists of five integrated modules:

I. **Centralized CTA Record and Document Management Module**

This module functions as a structured and searchable digital repository for CTA records and supporting documents, including applications, transcripts, source and target syllabi, articulation mappings, review outcomes, and export records. It supports role-based access control for students, HOPs, RPs, and AAS staff while improving record consistency, traceability, and institutional accessibility.

II. **Staged Syllabus Comparison and Review Support Module**

This module supports a more consistent and transparent evaluation of source and target course syllabus through a staged syllabus comparison process using decomposed prompting. In this process, syllabus content is handled through several structured stages, including preprocessing, standardization, mapping, verification, scoring, and summary generation. The resulting output is presented in a structured form to assist Resource Persons (RPs) in reviewing equivalency, recording comments, and making approval or rejection decisions. The module functions as a decision-support mechanism and does not replace academic judgment.

III. **Student CTA Submission and Tracking Module**

This module allows students to log into the system, complete onboarding, submit CTA information, upload transcripts, review extracted course mappings, and upload syllabi when required. It also enables students to monitor the progress of their applications, view application history, and track status throughout the workflow.

IV. Academic Review and Decision Management Module

This module supports HOP and RP responsibilities within the CTA workflow. HOP users can review applications, manage articulation mappings, manage target syllabi, request additional syllabus uploads, assign RP reviewers, and finalize decisions. RP users can review assigned comparison cases, inspect staged comparison results, provide comments, and submit approval or rejection decisions for courses requiring further academic evaluation.

V. Administrative Export and Reporting Module

This module supports AAS staff in managing finalized applications, exporting approved records for manual CMS entry, and marking applications as completed. It also provides reporting and export functions to support operational monitoring, including approval trends, processing duration, and workload visibility across the CTA workflow.

1.5.3 Target User

The intended users of the proposed credit transfer system are the main stakeholders directly involved in the Credit Transfer Application (CTA) process within the Faculty of Computing and Multimedia (FCOM) at Universiti Poly-Tech Malaysia (UPTM). Each stakeholder performs a specific role within the overall workflow, and the effectiveness of the process depends on coordination between these roles. The system is therefore designed to support the operational responsibilities of students, Head of Programme (HOP), Resource Person (RP), and Academic Affairs Students (AAS) staff. The following subsections explain the role of each target user in relation to the proposed system.

1.5.3.1 Faculty of Computing & Multimedia (FCOM) Students

Students from the Faculty of Computing and Multimedia (FCOM) at UPTM, particularly those enrolled in Diploma in Computer Science (CC101), Bachelor of Information Technology (Honours) in Computer Application Development (CT204), and Bachelor of Information Technology (Honours) in Cyber Security (CT206), are the main end users of the system. These students may apply for credit transfer when progressing through the academic ladder, such as from diploma to degree, or when seeking recognition for courses taken at previous institutions.

In the current manual process, students depend heavily on staff to initiate evaluations, submit physical forms, and provide updates on the progress of their applications. This often limits transparency and makes it difficult for students to know the status of their requests. In the proposed system, students initiate the application process themselves by entering the required information, uploading official transcripts, reviewing course mappings, and uploading syllabi when required for further academic evaluation. The system validates the completeness of the submission, highlights

missing requirements, and allows students to track the progress of their applications throughout the workflow. This gives students a clearer and more active role in the CTA process.

1.5.3.2 Head of Programme (HOP) of Faculty of Computing & Multimedia (FCOM)

The Head of Programme (HOP) is the main academic officer responsible for overseeing the credit transfer evaluation process within FCOM. In the existing manual workflow, HOPs often rely on fragmented records, previous experience, and manual document review when assessing course equivalency, which may affect consistency and delay decision-making.

In the proposed system, the HOP reviews student applications, verifies the mapped courses, manages articulation records, requests additional syllabus uploads when necessary, assigns Resource Persons (RPs) to cases requiring further evaluation, and performs final academic review before finalization. The HOP also manages supporting academic records such as articulation mappings and target syllabi used in the review process. By centralizing these functions in a single system, the HOP is able to make more traceable, structured, and transparent decisions.

1.5.3.3 Resource Person (RP)

The Resource Person (RP) is the academic expert responsible for evaluating syllabus equivalency for courses that require further review. In the manual process, RPs assess syllabus similarity primarily through personal judgment and physical or scattered digital documents, which may lead to inconsistent outcomes across different cases.

In the proposed system, RPs receive assigned review cases from the HOP and evaluate them using structured syllabus comparison results generated by the system. These results support RP review by presenting the comparison in a more organized and transparent manner, including mapped content, verification outputs, scoring, and summary findings. RPs may then record comments and submit approval or rejection decisions based on academic judgment. The system therefore supports RP evaluation while preserving the RP's role as the final academic evaluator for such cases.

1.5.3.4 Academic Affair Students (AAS) Unit

The Academic Affairs Students (AAS) unit is responsible for the administrative completion of the credit transfer process. In the current manual workflow, AAS staff handle finalized applications through repeated document checking, manual record handling, and re-entry of approved data into the Campus Management System (CMS), all of which are time-consuming and prone to error.

In the proposed system, AAS staff receive finalized applications after the academic review process has been completed. They are able to manage finalized and completed records, export approved application data in formats such as Excel or CSV for manual entry into CMS, and mark applications as completed within the system. This improves administrative efficiency, reduces duplication in document handling, and strengthens traceability and record management during the final stage of the CTA workflow.

1.6 Overview of This Report

In this chapter, we have presented the background, objectives, problem statement, scope and target users for the proposed automated syllabus comparison and credit transfer system for UPTM. Through AI-powered syllabus comparison and a centralized digital repository, the project enhances clarity, consistency and transparency for all stakeholders while addressing critical deficiencies of the current manual process.

The subsequent chapters will explain in detail what the project has achieved up to now. Chapter 2 presents a comprehensive review of literature on existing credit transfer systems, challenges faced by existing manual processes, need for semantic consistency in syllabus evaluation. The methodology happened to be the Agile methodology that was adopted for the project. The data gathering techniques used for the sites were interviews and questionnaires. The functional and non-functional requirements were mentioned too. System analysis of the project took place through use case model and flowchart. The report ends at Chapter 3.6, which fully defines the system design and requirements to allow implementation and development in subsequent stages.

2 LITERATURE REVIEW

2.1 Introduction

This chapter reviewed literature and existing systems related to the credit transfer process and automation in higher education. The review established the theoretical and practical foundation for the proposed UPTM Credit Transfer System by examining credit transfer regulations, manual workflow issues, digital record management, and technology-supported syllabus comparison. It also reviewed selected existing systems, namely Universiti Malaya's Pathway to Credit Transfer and Exemption (PaCE), the Automated Credit Transfer System (ACTS), and the CT204 Credit Transfer System, in order to identify relevant functions, strengths, and limitations. Through this review, the chapter guided the development of a system that addressed institutional needs at Universiti Poly-Tech Malaysia (UPTM), particularly in terms of efficiency, consistency, transparency, and traceability.

In addition, the chapter showed how digital workflow support, structured syllabus comparison, and user-centred system design could transform traditional credit transfer practices into a more streamlined, auditable, and equitable process.

2.2 Investigation

This study investigated the credit transfer process in higher education with reference to Universiti Poly-Tech Malaysia (UPTM). The investigation focused on three main areas, namely the regulatory structure of credit transfer, the operational inefficiencies of manual administration, and approaches to syllabus comparison that could support more consistent academic review. These areas were important because delays, inconsistency, and compliance issues did not arise from a single cause, but from the combined effect of fragmented workflows, subjective evaluation practices, and weak record management.

The findings from this investigation served as the empirical and conceptual basis for the proposed system. This section therefore examined the gap between regulatory expectations and actual institutional practice, and showed why a centralized digital repository, role-based workflow support, and a staged syllabus comparison module using decomposed prompting were needed.

2.2.1 Credit Transfer Systems and Regulations

Credit transfer systems allowed students to receive recognition for prior learning when moving between institutions or programmes. In doing so, they supported both vertical progression, such as from diploma to degree level, and horizontal mobility between qualifications at the same level. In the Malaysian context,

the Malaysian Qualifications Agency (MQA) stated that credit transfer could take place vertically or horizontally and had to be based on subject-to-subject mapping rather than course title alone. The same guidance also specified several general conditions, including a minimum passing grade of C, equivalent credit value, curriculum equivalence of not less than 80%, and accreditation or official recognition of the source programme (MQA, 2024). These requirements showed that credit transfer depended on documented curriculum comparability rather than superficial similarity.

Although the regulatory basis for credit transfer had already been defined, implementation remained difficult in practice. Heppner et al. (2019) described transfer credit assessment as a labour-intensive task that was constrained by time and susceptible to human bias. Chandrasekaran and Mago (2021) similarly found that transfer credit assessment often required domain experts to compare learning outcomes manually, making the process administratively complex. More recently, Kwak et al. (2026) reported that articulation work remained time- and resource-intensive because candidate articulations were still reviewed manually. These studies suggested that the main problem in higher education was not the absence of policy, but the difficulty of carrying out credit transfer consistently and efficiently within real institutional workflows.

.

2.2.2 Manual Workflow Inefficiencies

The manual implementation of credit transfer at higher education institutions created several interconnected operational inefficiencies, including administrative burden, fragmented record management, and limited process visibility. In such environments, students commonly submitted forms and supporting documents through disconnected channels, while academic and administrative staff tracked progress through paper files, email, or informal communication. As a result, the process often became slow, repetitive, and difficult to monitor.

Prior studies supported this observation. Heppner et al. (2019) noted that transfer credit assessment was labour-intensive, while Chandrasekaran and Mago (2021) found that manual implementation was not only resource-intensive but also affected by administrative complexity. These conditions increased the risk of delay, repeated verification work, and inconsistent handling across cases. In practice, the absence of a centralized workflow also made it harder for students and staff to know the current status of an application or to trace how a particular decision had been made.

Record management presented an additional weakness. When syllabi, prior decisions, articulation references, and application documents were stored in physical folders or scattered digital locations, retrieval became difficult and institutional memory became weak. This problem was also relevant from a quality-assurance perspective. The MQA's Institutional Quality Audit Framework required higher education providers to ensure that records were readily accessible to authorised users, protected through

appropriate security measures, and supported by audit logs and backup protocols (MQA, 2025). In a heavily manual or fragmented environment, meeting these expectations became more difficult. For that reason, the literature supported the need for digitization, centralized record storage, and structured workflow tracking in the proposed system.

2.2.3 Staged Approaches for Syllabus Comparison

To reduce subjectivity in manual syllabus evaluation, this project adopted a staged syllabus comparison approach using decomposed prompting rather than an embedding-based retrieve-and-re-rank architecture. This decision was supported by two main strands of literature, namely research on transfer credit automation and research on multi-step prompting methods for complex tasks.

First, existing studies on credit transfer and articulation showed that course equivalency assessment was inherently complex and could not be reduced to simple title matching or a single similarity score. Heppner et al. (2019) introduced a semi-automated approach to articulation using natural language processing and highlighted that transfer credit assessment was labour-intensive and vulnerable to bias. Chandrasekaran and Mago (2021) further explained that transfer credit assessment required domain experts to compare learning outcomes and aggregate those comparisons into course-level similarity, while still retaining tunable decision parameters to preserve flexibility. This was important because academic equivalency decisions often depended on more than textual similarity alone. They also depended on how course components were interpreted, weighted, and reviewed in context. More recently, Kwak et al. (2026) reported that articulation work remained time- and resource-intensive because candidate articulations were still reviewed manually. Taken together, these studies suggested that any technology used to support syllabus comparison needed to be transparent, inspectable, and flexible enough to accommodate academic judgment rather than replace it.

Second, prompting literature supported the use of decomposition when handling complex reasoning tasks. Khot et al. (2022) proposed decomposed prompting as a modular approach in which complex tasks were divided into smaller sub-tasks, allowing each sub-task to be handled by a dedicated prompt and improved independently. Their findings showed that decomposition was useful not only for improving performance, but also for making complex processing pipelines easier to control and refine. Zhou et al. (2022) similarly showed that least-to-most prompting was effective because a difficult task could be solved through a progression of simpler subproblems, where each partial output supported the next stage of reasoning. This was especially relevant for tasks in which a single direct response could miss important intermediate steps. Wang et al. (2023) reinforced this point by showing that planning before solving reduced missing-step errors in multi-step reasoning tasks. Their work suggested that structured intermediate stages improved both the completeness and the reliability of the final output.

These findings were directly relevant to syllabus comparison because equivalency assessment was not a single-step task. It involved several interdependent activities, including preprocessing syllabus text, standardizing extracted content, mapping source and target syllabus components, verifying possible matches, assigning scores, and producing a summary for academic review. In other words, the problem was not only one of semantic similarity, but also one of process structure. A staged approach therefore provided a stronger fit for the operational needs of the project than a single-pass comparison method.

Based on this literature, the present project adopted a staged syllabus comparison module using decomposed prompting. This approach was more consistent with the implemented system and more suitable for transparency because it produced intermediate outputs that could be inspected during academic review. Instead of relying on a single opaque similarity score, the process was structured into clear stages that supported verification and revision by Resource Persons. This made the module more appropriate for the credit transfer context, where consistency and efficiency were important, but final academic judgment still had to remain with the responsible reviewer. For this reason, the staged design was justified not only as a technical choice, but also as a workflow design choice that better supported traceability, transparency, and human review in course equivalency assessment.

2.2.4 Structured Review Support in Credit Transfer Assessment

In credit transfer assessment, fully automated decision-making would not have been appropriate because course equivalency involved more than text similarity alone. It also required academic interpretation, institutional policy alignment, and professional judgment when reviewing course learning outcomes, syllabus coverage, and supporting records. Earlier studies on transfer credit automation had already shown that computational methods could reduce manual effort, but expert review still remained necessary when determining final equivalency decisions (Heppner et al., 2019; Chandrasekaran & Mago, 2021). This suggested that the most suitable role of automation in the credit transfer context was not to replace academic evaluators, but to support them with more structured and consistent comparison outputs.

The literature on explainability also supported this position. Ribeiro et al. (2016) argued that explanations were important because users needed to understand the basis of a system output before deciding whether it should be trusted or acted upon. Miller (2019) further explained that useful explanations were not only technically correct, but also needed to be understandable from a human perspective. Jin et al. (2023) similarly found that non-technical end users had clear explanation needs and that systems became less effective when those needs were ignored. In education settings, this issue was especially important because acceptance of automated support depended not only on usefulness, but also on transparency, confidence, and perceived fairness. Karran et al. (2024) showed that the acceptability of technology-supported educational systems varied across stakeholder groups and was

influenced by factors such as explainability, justice, and confidence. Pitts and Motamedi (2025) likewise showed that trust directly affected whether users were willing to rely on educational systems and perceive them as useful. These findings indicated that a syllabus comparison module would need to produce outputs that reviewers could inspect and understand, rather than only presenting a final recommendation.

This justification aligned closely with the proposed UPTM Credit Transfer System. The staged syllabus comparison module was therefore more appropriate than a single-step output because it allowed intermediate results to be reviewed, verified, and commented on during the academic evaluation process. By structuring the work into preprocessing, standardization, mapping, verification, scoring, and summary generation, the system made the comparison process more traceable and reviewable for Resource Persons and Heads of Programme. This human-in-the-loop design also aligned with recent credit mobility research, which showed that adoption improved when system design took stakeholder expectations and review practices into account rather than treating automation as a complete substitute for human decision-making (Kwak et al., 2026). For this reason, explainability and reviewer oversight were not secondary design preferences, but core requirements for a system intended to support fairer, more transparent, and more accountable credit transfer assessment.

2.3 Related Works

The research, practices of credit transfer and academic mobility vary among regions and institutions. Different projects have attemted to simplify things by looking at a policy-based credit system or implementing a digital platform. It is interesting to look into these related works as they suggest how other initiatives addressed issues of consistency, transparency and automation of the credit transfer process. The subsequent sections describe three interrelated but distinct projects that will aid in the research.

2.3.1 Pathway to Credit Transfer and Exemption (PaCE)

The University of Malaya also has a relevant initiative known as Pathway to Credit Transfer and Exemption (PaCE) system. Students can apply for credit transfer and course exemption to the university through the online system, Maya through PaCE. PaCE rather than using a fully automated approach, focuses on policy compliance and procedural clarity, ensuring that students can transparently navigate the complex requirements for credit recognition.

The credit transfer under PaCE are of two types. They are credit transfer with grades and credit transfer without grades. In the first scenario, students will have to carry credits along with the previously acquired grade. On the other hand, in the second instance, just the credit would be recognized without the grade being transferred. You may apply PaCE to either vertical transfers (for example, from diploma to degree) or horizontal transfers (between programs or institutions of the same level).

The PaCE system shows that using a policy-driven portal can make credit transfer decisions more consistent and transparent. Making the rules standardized, classification clear, and submission electronic form will support student mobility and lifelong learning.

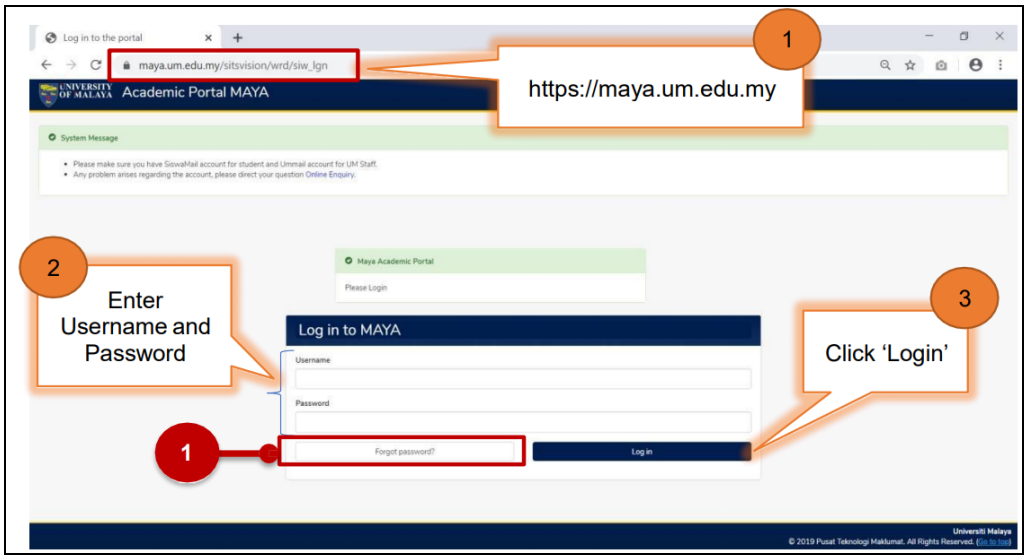


Figure 2.1: Login Page (Universiti Malaya, 2025)

Figure 2.1 shows the login page of the Maya portal, where students access the PaCE system. Users enter their username and password issued by their university. There is a "Forgot Password" option to recover your account.

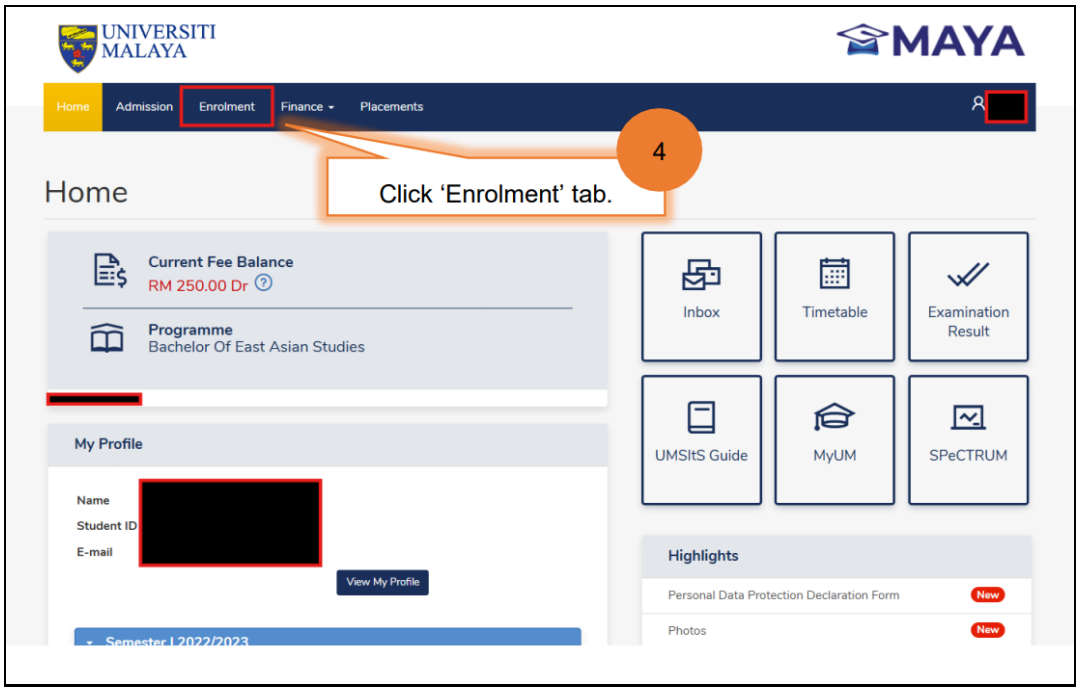


Figure 2.2: Main Dashboard (Universiti Malaya, 2025)

The main screen after log in is shown in figure 2.2. Using this interface, a student can operate the system's enrolment functions.

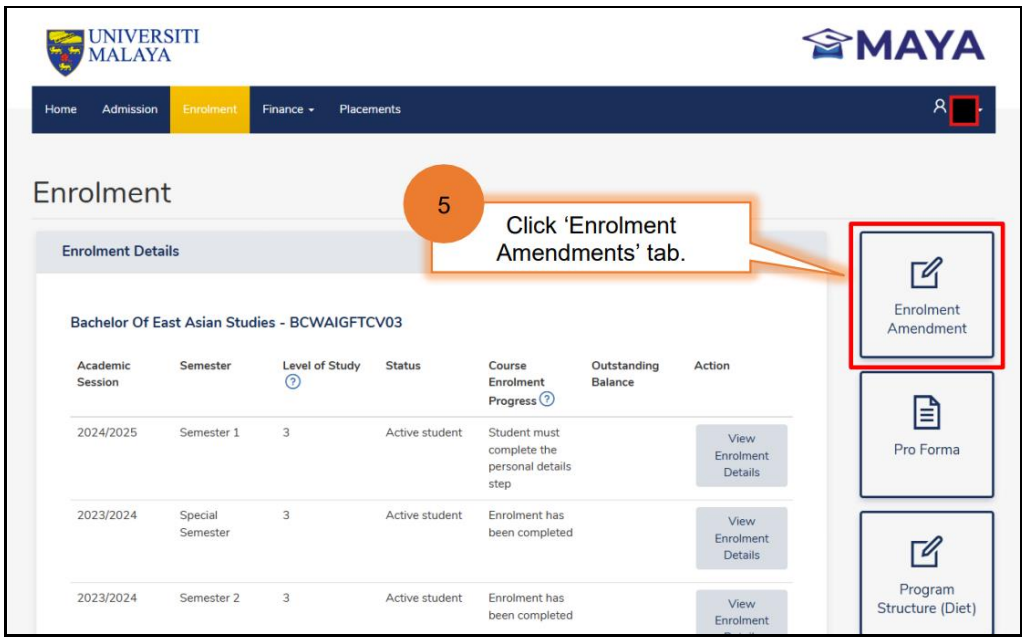


Figure 2.3: Enrolment Page (Universiti Malaya, 2025)

The Enrolment page shown in Figure 2.3 is where student select the "Enrolment Amendments" tab to start the application for credit transfer or exemption.

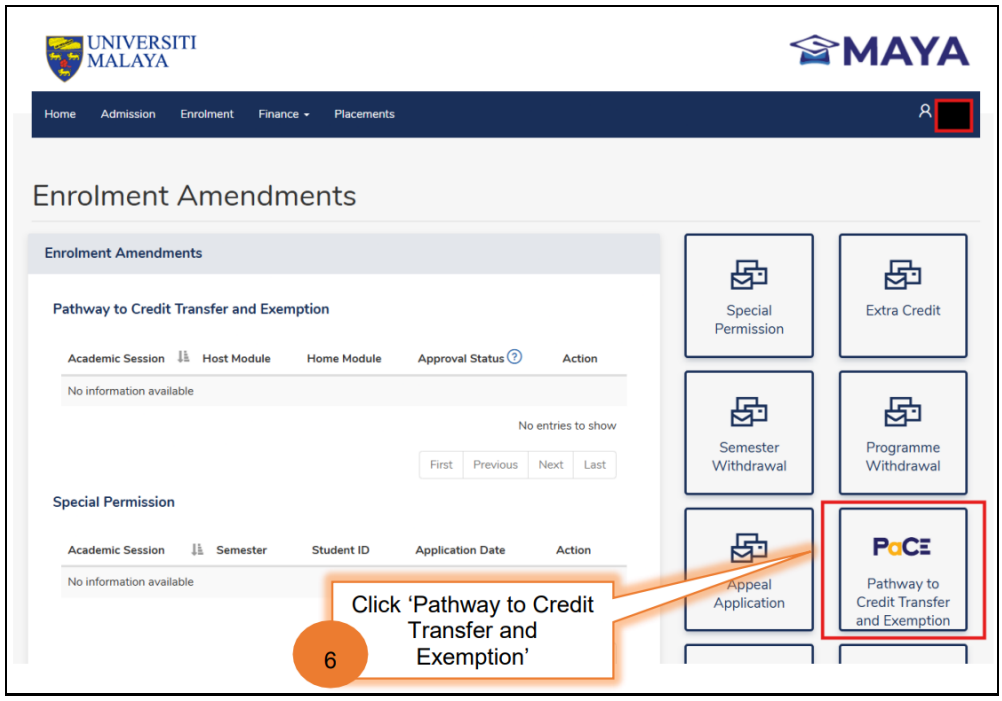


Figure 2.4: Enrolment Amendments Dashboard (Universiti Malaya, 2025)

The enrolment amendments dashboard is shown in figure 2.4. In this section, the option “Pathway to Credit Transfer and Exemption” is selected to begin a new application.

Figure 2.5: Transfer of Credit Application Form (Universiti Malaya, 2025)

As shown in Figure 2.5, the Transfer of Credit Application process begins with a form. Students starting entering required details of host institutions. This includes institution name, transfer method (one to one, many to one), and level of study or source transfer. Additional fields are module name, credit value, grade and completion period for complete academic information. You can upload supporting documents, such as course outlines, grading schemes, or exam questions by clicking the Browse My Computer button. The digital application form will standardize data collection and minimize manual entry errors in transferring credit.

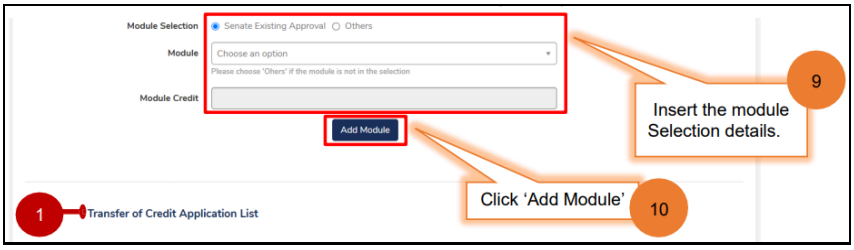


Figure 2.6: Module Selection and Upload (Universiti Malaya, 2025)

The module entry process in the web-based Transfer of Credit Application system is illustrated in Figure 2.6. The students begin by selecting which module type they want, Senate Existing Approval or Others, depending on earlier articulation. Next, they will enter the module details namely title, code, and credit value before clicking on Add Module to add them in the application list. The design of this interface allows for the entry of multiple modules but makes sure each submission is accompanied by evidence.

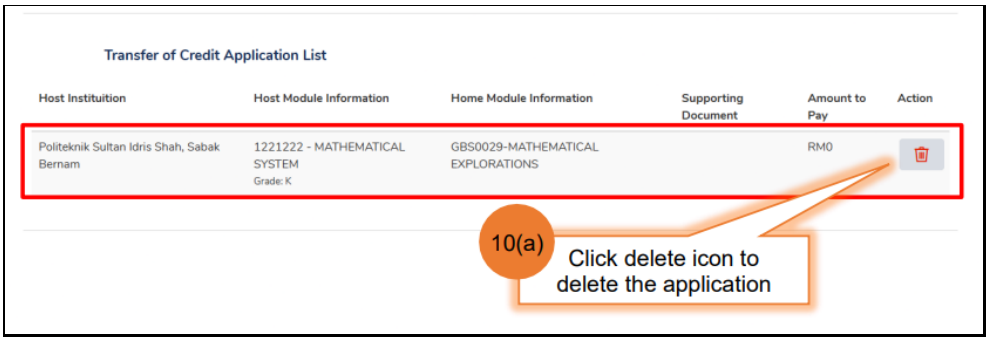


Figure 2.7: Managing the added modules (Universiti Malaya, 2025)

As can be seen from figure 2.7, students go through the module management stage to view their added modules and change it if necessary. The characteristics shown are the host institution; host module; home module; grade; and fee number. A student may click the delete icon for an incorrect or no longer relevant entry to remove it before submission. With this, a user can manage and maintain accuracy without restart, thereby giving flexibility and control over records.

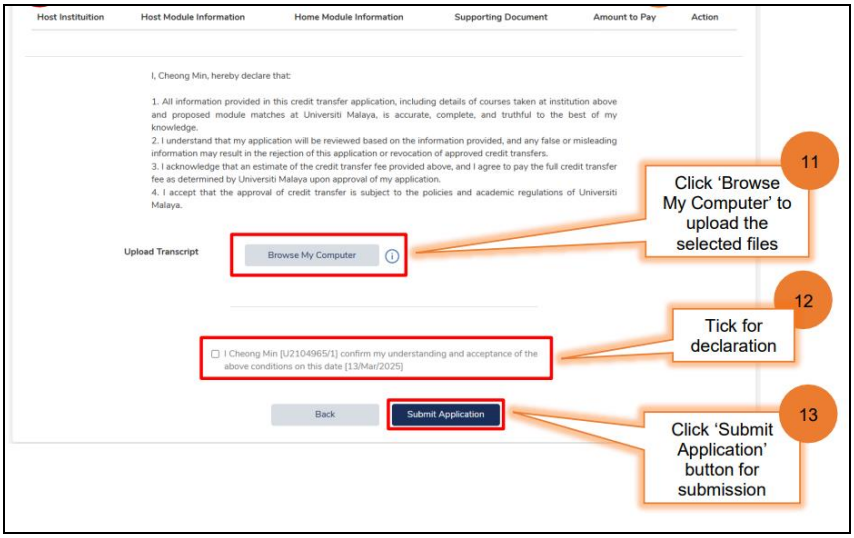


Figure 2.8: Declaration and Submission (Universiti Malaya, 2025)

The last part of the submission and declaration is shown in figure 2.8. Applicants are required to upload their transcript prior to submitting the application and clicking on declaration box stating that all information is true and correct. The user will be taken to the Confirm page where user click the Submit Application button to submit our application for review. This process of digital acceptance ensure that staff have agreed to institutional terms, and be an empowering glow in the automated credit transferring model. 96,236 Transforming submission can checks in paper-based form to digital form increases administrative efficiency and decreases human error.

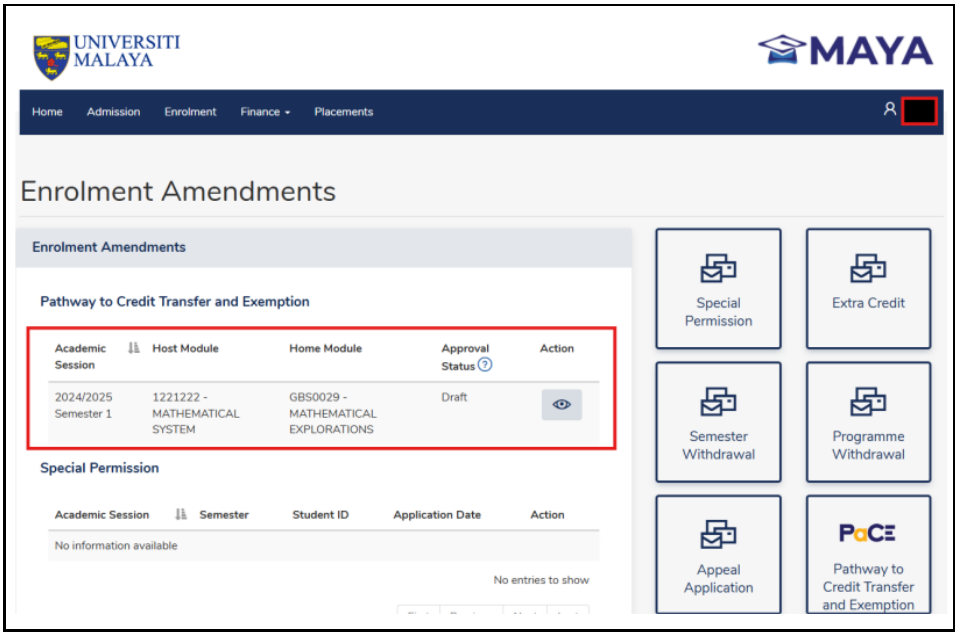


Figure 2.9: Application Source and Status Monitoring (Universiti Malaya, 2025)

The dashboard interface is presented to a student (when submitting a credit transfer application) on his/her screen as shown in Fig. 2.9. In the Pathway to PaCE tab users can view the modules they have

submitted and its academic session. Users also have the ability to view host module, home module and current approval status. Newly submitted applications are labelled Draft by the system until students can verify review status or submission details using the View (eye) icon. Status application overview: This dashboard is a great way to show an overview of application status and visibility. In this way, they can monitor the status of approvals and make an audit trail on every academic credit transfer.

2.3.2 Automated Credit Transfer System (ACTS)

One of the prominent works is Automated Credit Transfer System (ACTS) at Universiti Poly-Tech Malaysia (UPTM). The coded system was conceived as a prototype to digitalise and automatize the process of transfering academic credits in an NLP background, using Term Frequency-IDF (TF-IDF) and cosine similarity mechanism. Through an automated process of syllabus comparison, ACTS sought to solve systemic problems with the inefficiency, inconsistency and subjectivity credits transfer evaluations by humans.

To combat these challenges, the ACTS was designed with cosine similarity using TF-IDF vectorization. The system was developed in Python (using various NLP libraries, including NLTK, scikit-learn, and difflib). All of the above components are compatible with Streamlit as well and it was selected as the development framework to allow fast prototyping and interactive visualization of syllabus similarity scores. The application facilitates hand entry of data, and it also allows users to take documents down and re-upload them if there are submission errors.

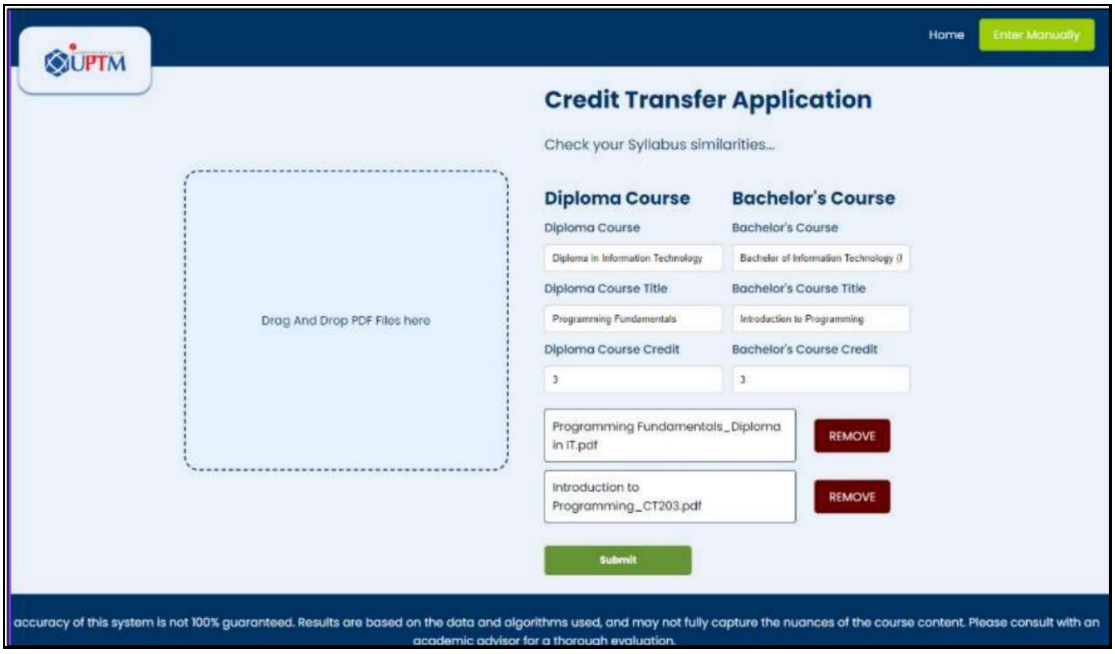


Figure 2.10: Syllabus Upload Module (Abdul Aziz et al., 2025)

The Syllabus Upload Module allows similar uploading of the syllabi of the diploma and bachelor programs in PDF format as shown in 2.10. Provided input fields for program, course title, and credit hours helps to guide users to get the correct values. A drag & drop feature is also given for convenience. Errors while entering data are greatly reduced by built-in validation features.

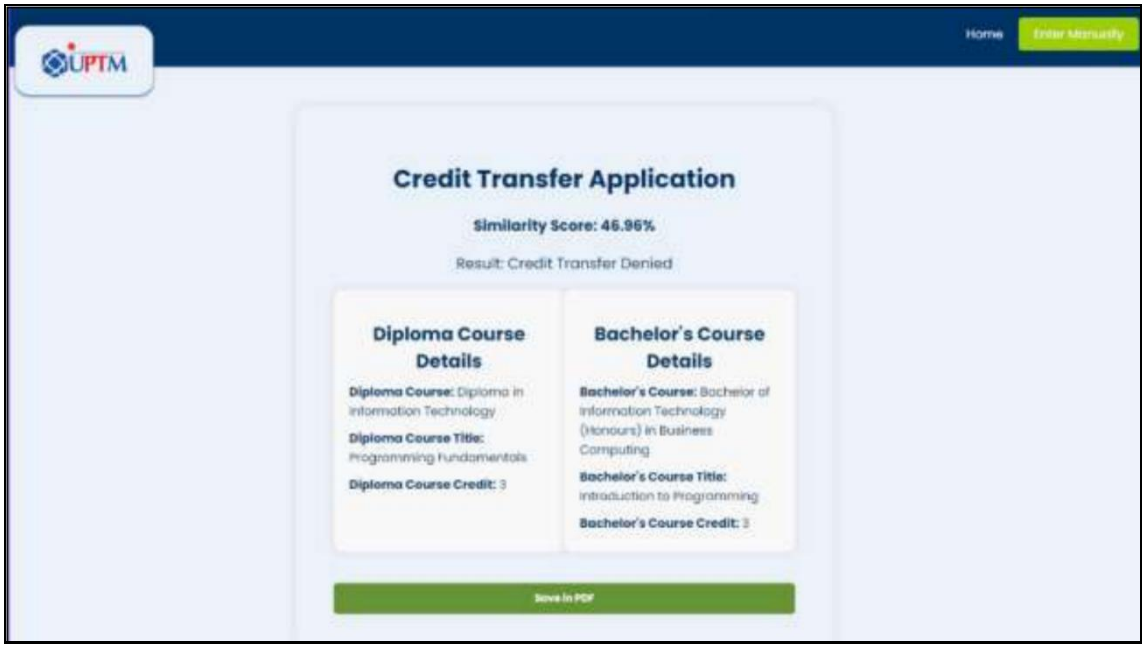


Figure 2.11: Summary Results (Abdul Aziz et al., 2025)

The Similarity Score Viewer Module shown in Figure 2.11 calculates and presents the summary results that shows similarity score between two syllabi. The comparison produces a score calculated as a percentage through the application of the TF-IDF and cosine similarity algorithms to the text. The system decides whether a credit transfer request is approved, which is based on the measured similarity. This automatic estimating may be more objective and transparent than manual estimations in decision making.

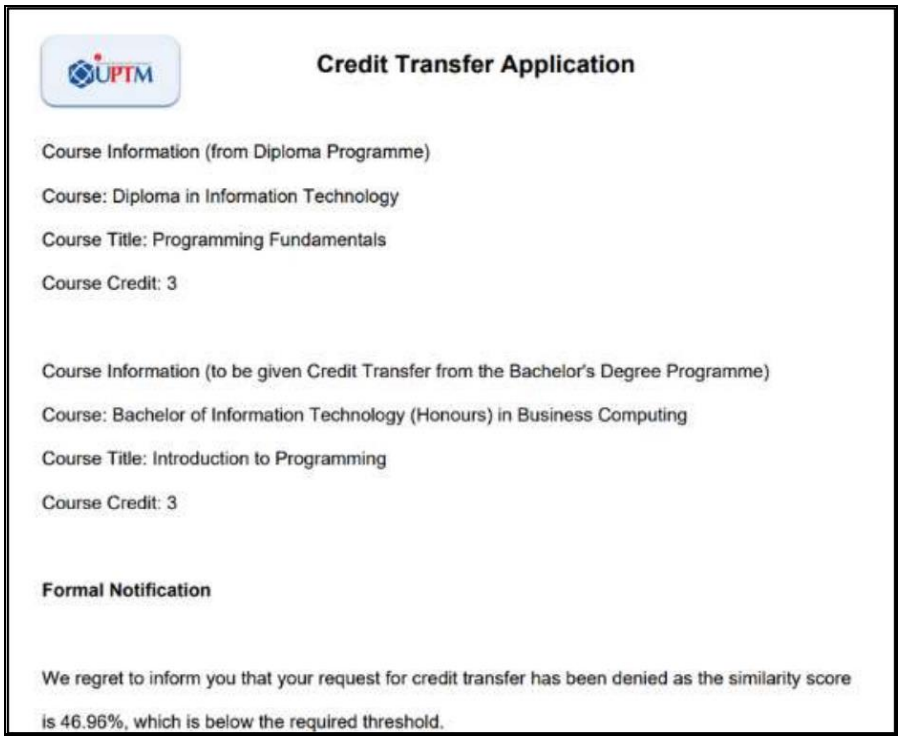


Figure 2.12: PDF Report Generator Module (Abdul Aziz et al., 2025)

Lastly, the PDF Report Generator Module is shown in Fig. 2.12. This option generates a formal report that gives an overview of the evaluation, both courses from each programme as well as the match score and result (approved or denied). It also offers an official notice that can be downloaded and retained for reference. This facilitates traceability, encourages uniform documentation and eliminates administrative.effortmaking.

2.3.3 CT204 Credit Transfer System

Another project beside the Credit Transfer System is the CT204 Credit Transfer System. It was originally implemented as an unpublished project for a diploma at Universiti Poly-Tech Malaysia (Osman, 2023) to serve as solution for the problem faced by traditional paper based credit transfer process of CT204 program. It is based on this manual system and the processes of having paperwork filled out, sent in to coordinators requiring a Dean's approval, sent to Admission and Records for approval which led to problems with delays from lost paperwork (and) lack of tracking an applicant's status in the process.

To access the system students could log in to the site with their school Google account, upload a validated transcript and select from an articulated list of courses and complete electronically and submit a transfer form. Furthermore, the process permitted to know in real time the status of an application for credit transfer; therefore a high level of transparency and involvement on behalf of students was attained. The system allowed coordinators to control the selection process (including verification of supporting documentation) and approval or rejection of course transfers directly over the systems interface.

Built on Nuxt 3, Supabase (PostgreSQL Database & Authentication) and Tailwind CSS using modern web technologies it followed an Agile approach to software development. This provided the project team with continual input from co-ordinators and monitors. The system was tested in three iteration as: High-frequency cycle testing, integration test phase and user acceptance test phase. The students indicated that they appreciated the system because it meant increased transparency and ease of use, while HOP's are finding that the structured workflow relieves them of some administrative burden.

The CT204 Credit Transfer system was not used outside of the CT204 program, but it illustrated for us that it would be possible to design a credit transfer system specifically geared to faculty. The effort demonstrated how real-time digital monitoring, secure transcript handling and user centered design can be applied to modern academic administrative processes. Thirdly, the CT204 Credit Transfer System will provide a base-line for the present bachelors project that can guide its conceptual model, system architecture and implementation approach.

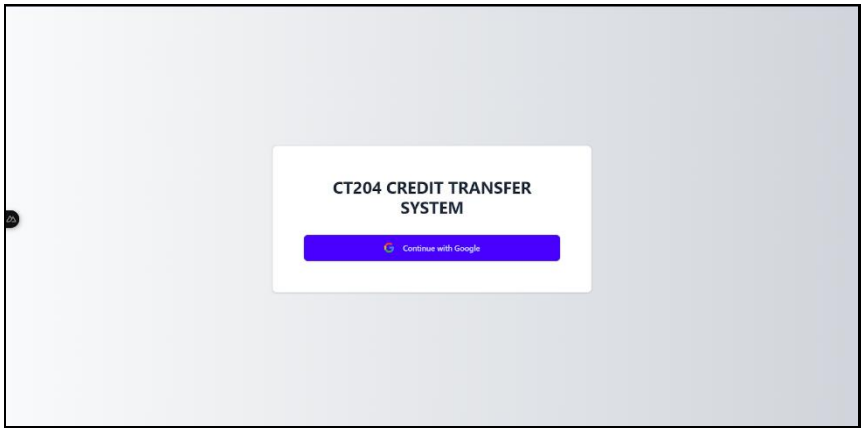


Figure 2.13: Login Page (Osman, 2024)

The secure login interface as shown in Figure 2.13 of the CT204 Credit Transfer System utilizes the OAuth 2.0 authentication process with Google Sign-In that provides the student with the option to use their Institutional Google account to access the credit transfer system (and therefore eliminates the need for manually registering for an account). In addition to providing enhanced protection of student information due to OAuth's ability to provide additional layers of data security; it also streamlines the management of access for both the students and administrator by reducing the number of steps required to manage access to the credit transfer system.

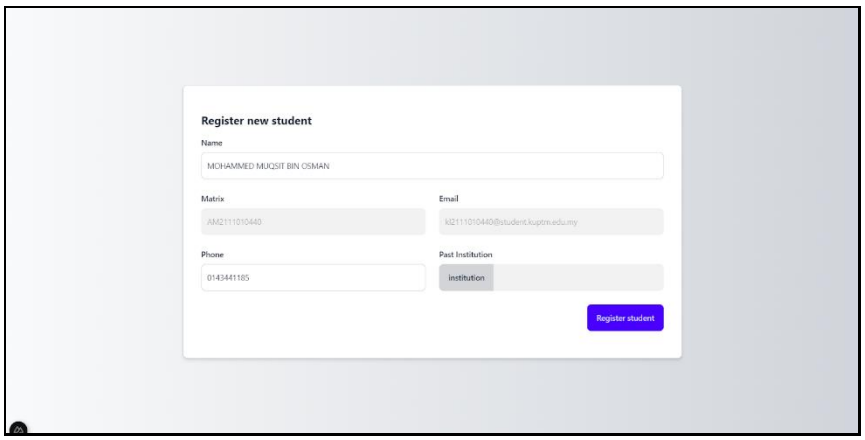


Figure 2.14: Student Registration Interface (Osman, 2024)

Once a user has successfully logged in after clicking the Google login button from Figure 2.14, to student part of the CT204 Credit Transfer System, he/she is asked to provide some details that are specific only such as his/her name and matric number and office (i.e contact telephone) numbers, an alternative email address and then followed by past institution(s). The registration form will be prepopulated with the most pertinent information, and it will validate during the OAuth sign-in process to ensure that there are no manual entry mistakes made. After creating an account, the user will land on the main dashboard and can now initiate the application for credit transfer. Having a structure to the registration process helps

ensure that information put into the system is valid and links to both institutional identity of the student and the record in the system.

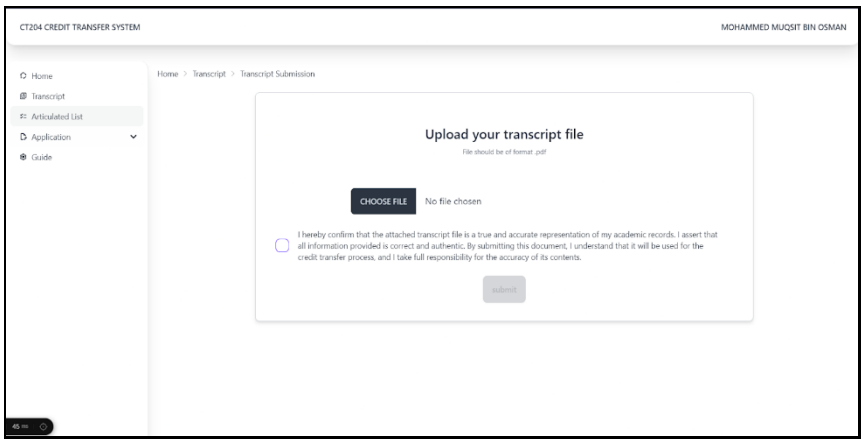


Figure 2.15: Transcript Upload Interface (Osman, 2023)

The student is shown the "transcript submission" part of the CT204 Credit Transfer System in Figure 2.15. In order for a student to begin a credit transfer request, they will need to add their official academic transcript (in PDF) to the system. This is important so that all the work being done by the credit transfer evaluators can be based on the same documents and therefore accurate. A declaration check box is also available to ensure the user has reviewed and certified the accuracy and authenticity of the document prior to submitting it. This is intended to help promote academic integrity, reduce the likelihood of fraudulent submissions and help to prevent the submission of false or altered information. Once a student has submitted their transcript to the system, it will become accessible to the coordinators for review and use as reference material for the purpose of evaluating the course equivalency.

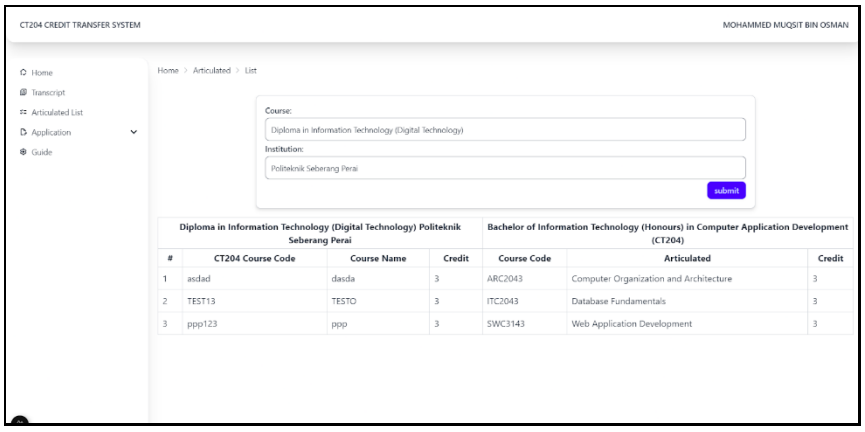


Figure 2.16: Articulated List Interface (Osman, 2023)

The Articulated List Page of the CT204 Credit Transfer System (See Figure 2.16) illustrates how students can view the pre-existing course equivalence(s) that exist between their former school/previous education experience and the CT204 program at Universiti Poly-Tech Malaysia. Students use this process by first

selecting their course and/or institution; subsequent to which the system will provide an articulated list with all the corresponding courses, course codes, and credits that were successfully transferred. Clearly displayed on the user interface are the articulations from both diploma level and bachelor's level, allowing students to easily identify what subject(s) they may be able to transfer credits for. The student is provided transparency throughout the articulation process and is better able to understand how their prior learning has been applied toward the requirements of their degree.

The Credit Transfer Application page of the CT204 Credit Transfer System is organized in three primary sections; each area is created to allow the student to follow an orderly and accurate application procedure.

1. Student Information Section (Application Information) automatically captures student information from the registration file.
2. Course Information Form for Credit Transfer - Students will be able to input course equivalence information on-line.
3. Application Review Section (Final Application Information) - Students will have the opportunity to review, verify and submit their complete application.

The organization of this site ensures that all credit transfers submitted by the student will be consistent, verifiable, and easily managed within one window. This is illustrated with detailed examples in figures 2.17 through 2.19.

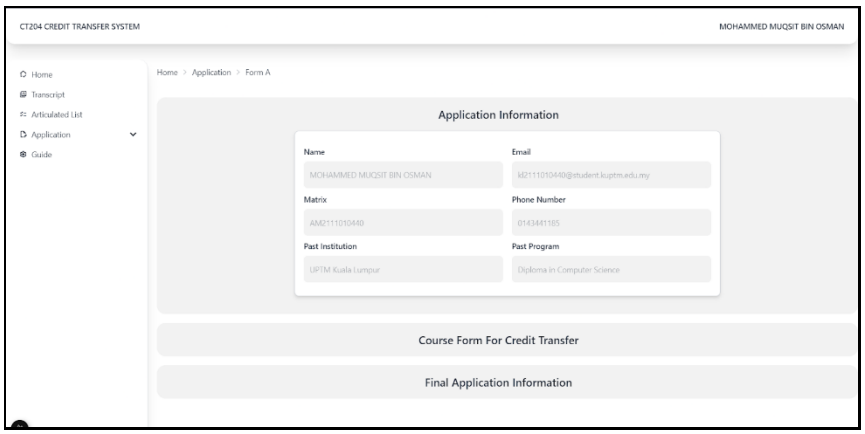


Figure 2.17: Application Information Section (Osman, 2023)

The “Application Information” section of the CT204 Credit Transfer System is shown in Figure 2.17. The application information section will automatically pull a student’s personal and academic details into this area (name, matric number, institutional email address, phone number, previous institution attended and previous program) through verified data from the student’s registration records. Auto-populating applicant information with minimal risk of manual error and ensuring accurate applicant information enhances system efficiency by allowing students to proceed immediately to the credit transfer form and eliminate re-keying of applicant's basic information.

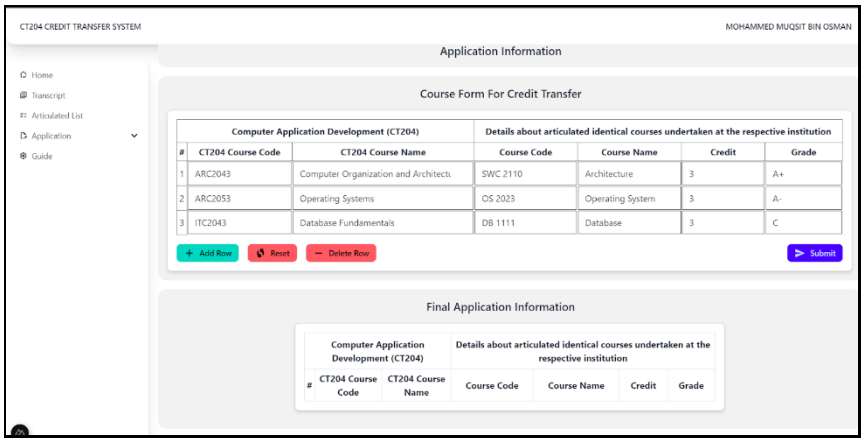


Figure 2.18: Course Form for Credit Transfer (Osman, 2023)

The Course Form for Credit Transfer shown in Figure 2.18 lets students input and administer course equivalence data for classes at the student's previous institution and the CT204 program. The dynamic table lets students add rows so they may enter all course equivalencies with one entry. In addition to the Add Row button, other options (Reset, Delete Row) allow for students to review, edit or delete individual rows prior to submitting the form. For each row, students will need to enter course numbers, course name, number of credits, and grade received at both institutions. The dynamic nature of this form format

provides users greater ability to have control over the process, and reduces the amount of time it takes to record many different class to class equivalent subjects for a credit evaluation.

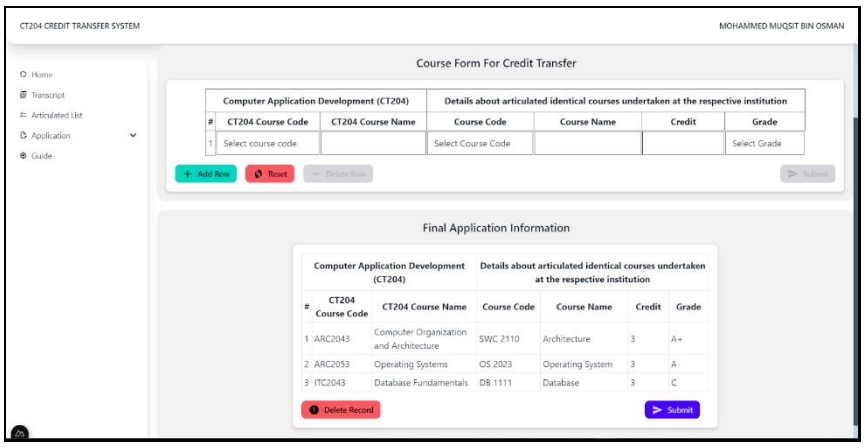


Figure 2.19: Final Application Submission Section (Osman, 2023)

The Final Application Information section, as shown in Figure 2.19, is a consolidation of all student-entered information to allow students to view, confirm, and edit their application prior to submission. This section will be used by the students to verify the accuracy of the information entered in the application process and to make any additional edits prior to submitting it to the Coordinator for review. The Final Application Information section provides the students with two options; "Submit" which will send the application into the Coordinator's queue for review, and "Delete Record" which will remove the entire record if there were errors made during the application process. With this feature, the students will be able to ensure the accuracy of the submitted applications and prevent them from sending an application to the Coordinators for review when they have identified errors in the application process. Through providing all three functions within one interface, the system increases the student's responsibility/accountability and reliability in completing the credit transfer process.

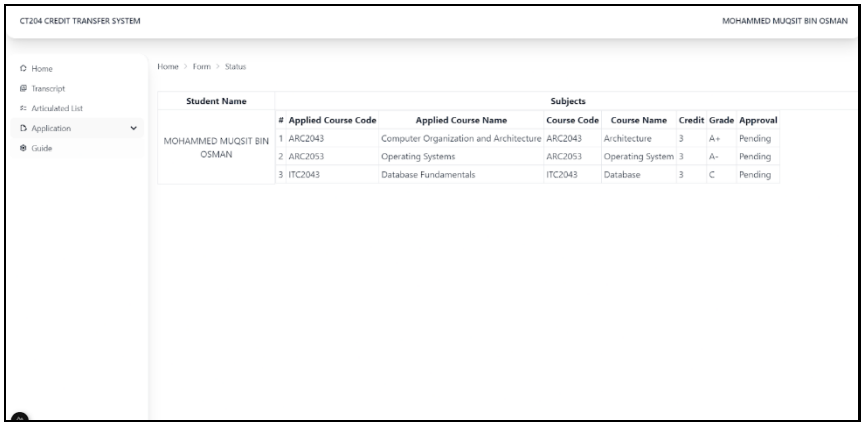


Figure 2.20: Credit Transfer Status Page (Osman, 2023)

The Credit Transfer Status Page (see Figure 2.20) is used by students to check on the status of their submitted applications for credit. Once an application has been submitted, all of the course entries are displayed in a formatted table which includes the course code, subject area, value of the credits, grade achieved and the current status of approval. The initial status for each entry is listed as pending; this indicates that the entry is waiting to be reviewed by either the program coordinator or a designated faculty member. The Credit Transfer Status Page also provides clarity and allows students to follow the evaluation process without having to rely upon personal contact with others.

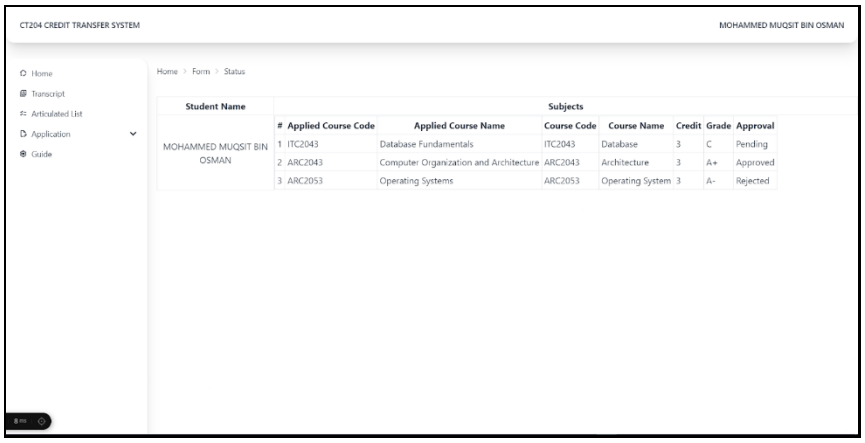


Figure 2.21: Updated Approval Status (Osman, 2023)

After a Coordinator reviews each application, Figure 2.21 shows the Credit Transfer Status page as it is now in an updated format. As each application is approved or rejected, the approval status will automatically be shown to reflect either Approved or Rejected for each subject. Thus, students can access information about their transferable credits in real time. Furthermore, by implementing automatic approval tracking within the system, the CT204 Credit Transfer System improves efficiency of communication, responsibility, and general student satisfaction with the credit transfer process.

Figure 2.22: User Guide Page (Osman, 2023)

The Guide Page (CT204 Credit Transfer System) is depicted in Figure 2.22. As stated previously, this page was developed to provide a methodical and step-by-step approach for students to navigate the credit transfer process. Each section; Step 1, Step 2, and Step 3 are expandable and provide detailed and explicit instructions to complete major tasks including uploading transcripts, submitting an application, and checking the status of their approvals. A self-help tool with this integrated guide reduces the need for manual supervision by a coordinator. The integration of instructional content into the user interface allows the system to allow new users to successfully and correctly perform the steps of the credit transfer process on their own.

2.4 Comparison

In order to gain a better understanding on the strengths and weakness of the current credit transfer system, it was compared against four similar projects such as the Pathway to Credit Transfer and Exemption (PaCE) which was developed by Universiti Malaya, CT204 Credit transfer system proposed by Osman (Osman, 2023), Automated Credit Transfer System (ACTS) and UPTM credit transfer systems. Each system was assessed according to their platform's design, authentication mode, the scope of functions and user level of automation as well as overall transparencies. The aim of this comparison entities is to show how UPTM complements/enhances the existing functionality, by incorporating artificial intelligence, centralised data management and advanced user interaction.

Table 2.1: Comparison of Existing Project

Feature / System	**Pathway to Credit Transfer & Exemption (PaCE)** ***(Universiti Malaya)***	**CT204 Credit Transfer System** ***(Osman, 2023)***	**Automated Credit Transfer System (ACTS)** ***(UPTM)***	**UPTM Credit Transfer System (Current Project)**
Platform Type	Web portal integrated with UM Maya system	Prototype web app (Nuxt 3)	Research prototype built in Python & Streamlit	Full-stack Next.js web application
Authentication Method	UM single sign-on (institutional login)	Google OAuth 2.0 with multi-role access	Local admin login	Google sign-in with role-based access control

Feature / System	Pathway to Credit Transfer & Exemption (PaCE) *(Universiti Malaya)*	CT204 Credit Transfer System *(Osman, 2023)*	Automated Credit Transfer System (ACTS) *(UPTM)*	UPTM Credit Transfer System (Current Project)
Core Functions	- Manual CTA submission - Status tracking	- CTA workflow - Status tracking - API integration	- Automated syllabus comparison - Syllabus comparison reports	- Online CTA workflow - syllabus comparison - status tracking - reporting, and export
Syllabus Comparison Method	N/A (manual review)	Manual review (no automation)	TF-IDF with cosine similarity (1st check only)	Staged syllabus comparison using decomposed prompting
Data Storage	UM database integration via Maya	Supabase (PostgreSQL hosted)	Local JSON / CSV storage	PostgreSQL and SeaweedFS (S3-compatible file storage)
CTA Tracking	Real-time status in dashboard	Real-time status in dashboard	Displays similarity scores only	Application and review status tracking
Target Users	UM students and administrators	CT204 students & coordinators (prototype)	Research evaluation only (no student access)	Students, HOPs, RPs, and AAS staff
Limitations	Evaluation remains manual and time-consuming	– Prototype only – manual review of syllabus	- No semantic context - limited to keyword matches	Depends on reviewer validation and manual CMS entry
Main Contribution	Digitalized manual submission and approval through an online portal.	- Demonstrated CTA workflow - API integration - Tracking Features	Showed AI feasibility using TF-IDF and cosine similarity for syllabus comparison.	Digitalized CTA workflow and standardized syllabus evaluation

The comparative analysis showed a gradual development in the digitalization of credit transfer processes within Malaysian higher education institutions. Universiti Malaya's PaCE system demonstrated that credit transfer applications could be moved from a manual process into an online portal by supporting digital submission and application tracking. However, syllabus evaluation still remained manual, which limited

improvements in consistency and standardization. Similarly, Osman's (2023) CT204 Credit Transfer System showed that a prototype system could support workflow management and online processing, but course equivalency assessment still depended on manual academic review. In contrast, the Automated Credit Transfer System (ACTS) represented an earlier effort to support syllabus comparison through automated text-based matching using TF-IDF and cosine similarity. Although this approach contributed toward automation, it remained limited in semantic depth, was mainly research-oriented, and was not integrated into a complete role-based CTA workflow.

Based on this comparison, the UPTM Credit Transfer System presented a more integrated solution for managing the end-to-end CTA process. The system was implemented as a full-stack Next.js web application with role-based access for students, HOPs, RPs, and AAS staff. It supported centralized CTA management, application and review status tracking, reporting, export functions, and a staged syllabus comparison module using decomposed prompting. Application data was managed in PostgreSQL through Prisma, while supporting files such as transcripts, syllabi, and exports were handled through an S3-compatible storage layer. Although final academic decisions still depended on reviewer validation and downstream CMS entry remained manual, the current system provided broader workflow coverage, more structured record management, and more transparent review support than the earlier systems discussed in this study.

2.5 Discussion

The review of existing systems showed that progress had been made in digitalizing the administrative side of credit transfer, but important gaps still remained in workflow integration, structured record management, and transparent syllabus evaluation. This project therefore drew on the strengths of earlier systems while addressing the limitations that were still evident in manual review processes and isolated prototype implementations.

One of the main practical strengths observed in the CT204 Credit Transfer System was its use of secure login and role separation. This showed the importance of controlling access according to user responsibility so that different stakeholders could interact with the system without overlapping functions or creating conflicts in the workflow. For the UPTM Credit Transfer System, this principle was retained through Google sign-in and role-based access control for students, HOPs, RPs, and AAS staff. This approach reduced the need for separate manual account registration, improved access control over sensitive academic records, and supported clearer responsibility boundaries throughout the CTA process.

From the perspective of workflow design, Universiti Malaya's PaCE system provided useful insight into how credit transfer applications could be managed through an online submission and tracking environment. Its structured portal-based approach demonstrated the value of moving application progress away from paper-based handling and fragmented communication. However, because syllabus evaluation in PaCE still depended on manual academic review, improvements in efficiency did not fully resolve issues of consistency and traceability in equivalency assessment. This highlighted the need for a system that not only digitalized submission and tracking, but also provided structured support for syllabus review.

The ACTS prototype contributed a different perspective by showing that syllabus comparison could be supported computationally rather than being handled entirely through manual reading and judgement. Its use of TF-IDF and cosine similarity demonstrated an early step toward comparison support, but the approach remained limited in semantic depth and was not integrated into a complete role-based credit transfer workflow. In the present project, this limitation was addressed by adopting a staged syllabus comparison module using decomposed prompting. Compared with a simpler text-matching approach, the staged process allowed the comparison to be handled through clearer intermediate steps such as preprocessing, standardization, mapping, verification, scoring, and summary generation. This made the comparison output more structured and reviewable for academic users rather than relying on a single similarity result.

Based on these findings, the UPTM Credit Transfer System was designed as a more integrated solution than the earlier systems reviewed. The system combined role-based access, centralized CTA management, application and review tracking, articulation support, reporting, export functions, and a staged

syllabus comparison module within a single full-stack web application. Application data was managed in PostgreSQL through Prisma, while supporting files such as transcripts, syllabi, and exports were handled through an S3-compatible storage layer. Although final decisions still depended on reviewer validation and CMS entry remained manual, the system provided broader workflow coverage, better record organization, and more transparent review support than the earlier systems discussed. In this way, the discussion showed that the present project did not simply replicate existing approaches, but synthesized their useful design ideas into a system that was more aligned with UPTM's operational context.

2.6 Conclusion

This chapter reviewed the literature and existing systems related to credit transfer management and established the theoretical and practical basis for the development of the UPTM Credit Transfer System. The review showed that although credit transfer had clear regulatory foundations and several institutions had already digitalized parts of the process, important challenges still remained in manual workflow handling, fragmented record management, and consistent syllabus evaluation.

The review of PaCE, ACTS, and the CT204 Credit Transfer System provided useful insight into different aspects of system design. PaCE demonstrated the value of an organized online submission and tracking workflow. ACTS showed that syllabus comparison could be supported through automated text-based methods, even though its approach remained limited in scope and semantic depth. The CT204 Credit Transfer System contributed practical ideas in relation to system structure, secure access, and role-based workflow organization. These findings helped identify the functional and architectural elements that were most relevant to the present project.

By synthesizing these lessons, the proposed UPTM Credit Transfer System was positioned as a more integrated platform for managing the end-to-end CTA process. The chapter therefore justified the need for a centralized system that combined structured workflow support, role-based access control, centralized record management, and a staged syllabus comparison approach to support academic review. The findings from this chapter directly informed the design and development approach described in the following chapter.

3 METHODOLOGY

3.1 Introduction

This chapter presented the methodology adopted for the development of the UPTM Credit Transfer System. It explained the framework, strategies, and methods used throughout the project lifecycle to ensure that the system was planned, designed, implemented, tested, deployed, and reviewed in a systematic manner. The methodology emphasized iterative development, continuous refinement, and alignment with stakeholder needs so that the resulting system could satisfy both functional and non-functional requirements.

Agile methodology was adopted as the principal framework for this project. This approach was considered suitable because the development of the UPTM Credit Transfer System involved evolving workflow requirements, multiple stakeholder groups, and several interconnected modules, including authentication, CTA submission, transcript handling, syllabus management, staged syllabus comparison, reporting, and export. An iterative development model was therefore more appropriate than a rigid linear process, as it allowed the system to be refined progressively based on observations made during development. This chapter also described the six phases used in the Agile methodology for the project, namely Requirements, Design, Development, Testing, Deployment, and Review. Although these phases were presented separately for clarity, they were applied in an iterative manner throughout the development process.

3.2 Agile Methodology

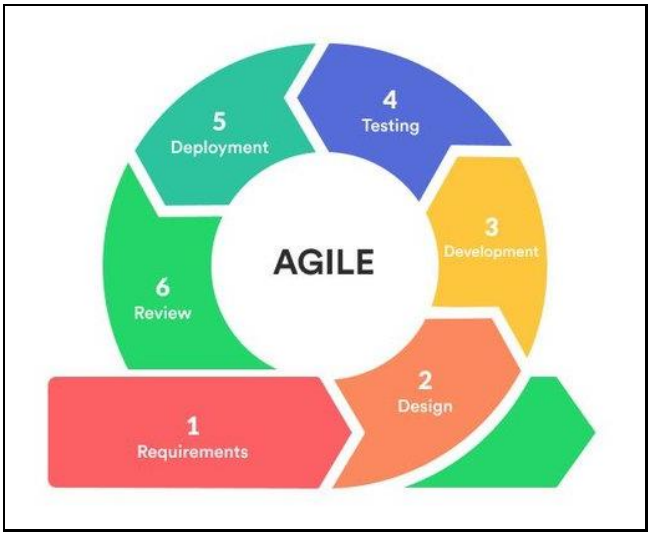


Figure 3.1: Agile Methodology Diagram (Okeke, 2021)

Agile methodology was selected for this project because of its flexibility, iterative structure, and emphasis on stakeholder feedback. Unlike traditional linear methodologies such as Waterfall, Agile supported continuous refinement through repeated development cycles, allowing improvements to be introduced as the system evolved. This characteristic made Agile particularly suitable for the UPTM Credit Transfer System, where requirements were influenced not only by technical considerations, but also by institutional workflow rules and the needs of students, Heads of Programme (HOPs), Resource Persons (RPs), and Academic Affairs Students (AAS) staff.

Under Agile methodology, the system was developed through incremental cycles in which working modules could be implemented and improved progressively. This enabled the identification of issues at an earlier stage, reduced the risk of major redesign late in the project, and allowed functional refinements to be made without disrupting the overall system structure. Agile was also appropriate because the project required gradual refinement of several connected areas, including role-based workflow support, centralized record management, articulation handling, staged syllabus comparison, and export functions.

In this project, the Agile process was organized into six phases: Requirements, Design, Development, Testing, Deployment, and Review. These phases formed the basis of the development cycle illustrated in Figure 3.1. Although shown as separate stages, the phases informed one another continuously and supported ongoing improvement throughout the project.

3.3 Phases in Agile Methodology

This project followed a six-phase Agile methodology consisting of Requirements, Design, Development, Testing, Deployment, and Review. The methodology represented a structured but iterative cycle in which the output of one phase could inform later refinements in the others. This approach was important because the UPTM Credit Transfer System needed to reflect actual institutional workflow requirements rather than a purely theoretical design. The phases, activities, objectives, methods, and deliverables applied in this project are summarized in Table 3.1.

Table 3.1: Phases in Agile Methodology

Phase	Activities	Objectives	Methods	Deliverables
Requirement	**Activity 1:** Identify and analyze UPTM's current manual credit transfer workflow.	Understand workflow issues, redundancies, and process pain points.	Document review, workflow mapping, and process analysis.	Current process flow and list of identified issues.
	Activity 2: Gather user requirements from students, HOPs, RPs, and AAS staff.	Define functional and non-functional requirements for the proposed system.	Stakeholder discussion, requirement analysis, and backlog definition.	System requirements and prioritized development backlog.
Design	**Activity 3:** Design system architecture, database schema, and user interface mockups.	Develop a secure, scalable, and role-based design for the system.	ERD design, wireframing, workflow modeling, and interface planning.	Database schema, UI mockups, and workflow design.
	Activity 4: Define the staged syllabus comparison workflow and supporting service integrations.	Establish a structured and transparent comparison design for syllabus evaluation support.	Process modeling, stage decomposition, and integration planning.	Comparison workflow design, stage definitions, and integration plan.
Development	**Activity 5:** Implement the full-stack web application and core CTA modules.	Build an integrated system with role-based access and end-to-end CTA functionality.	Agile iterations, version control, and module integration.	Functional modules for authentication, CTA workflow, review, reporting, and export.
	Activity 6: Integrate transcript handling, file storage, extraction support, and staged syllabus comparison.	Enable complete processing flow for documents, syllabus evaluation support, and workflow execution.	Service integration, iterative refinement, and functional implementation.	Working document flow, staged comparison module, and supporting integrations.

Phase	Activities	Objectives	Methods	Deliverables
Testing	**Activity 7:** Perform functional, integration, and workflow testing.	Verify system behavior, module interaction, and workflow correctness.	Script-based testing, manual validation, and issue tracking.	Test findings, corrected issues, and refined modules.
	Activity 8: Conduct user-oriented validation of the system workflow.	Evaluate usability, workflow clarity, and practical suitability for target users.	Guided walkthroughs, observation, and structured feedback collection.	Validation feedback summary and improvement list.
Deployment	**Activity 9:** Deploy the system in a live environment for controlled use.	Ensure stable access to the implemented system in an operational setting.	Environment configuration, deployment setup, and service verification.	Live deployed system and deployment configuration.
Review	**Activity 10:** Collect feedback, assess system performance, and identify future improvements.	Evaluate the system against project objectives and determine enhancement priorities.	Feature review, limitation analysis, and stakeholder feedback.	Review summary and future enhancement plan.

3.3.1 Requirements

The Requirements phase focused on identifying the main problems in the existing manual credit transfer process and determining what the proposed system needed to support. This phase examined the current CTA workflow, the roles of the stakeholders involved, the handling of records and supporting documents, and the points at which delays, repeated work, or inconsistency commonly occurred. Particular attention was given to issues such as fragmented record storage, limited visibility of application progress, repeated syllabus evaluation, and dependence on manual coordination across different stakeholders.

The purpose of this phase was to define both the functional and non-functional requirements of the system. Functional requirements included user onboarding, CTA submission, transcript upload, course review, syllabus upload, articulation handling, RP assignment, staged syllabus comparison, reporting, export, and completion workflow. Non-functional requirements included usability, security, traceability,

maintainability, and role-based access control. The output of this phase was a clearer understanding of the existing process and a prioritized set of requirements to guide the subsequent phases.

3.3.2 Design

The Design phase translated the identified requirements into a technical and workflow blueprint for the system. This included the design of the application architecture, database structure, workflow logic, user access rules, and interface layout. In the implemented project, the selected architectural direction was a full-stack Next.js application in which the same application handled the user interface, server-side business logic, route handlers, and workflow orchestration. The database was designed using PostgreSQL through Prisma, while supporting file management was planned through an S3-compatible object storage layer.

This phase also covered the role structure for students, HOPs, RPs, and AAS staff, as well as the status flow of CTA records from draft until completion. In addition, the syllabus comparison component was designed as a staged process using decomposed prompting rather than a retrieval-based pipeline. This allowed the comparison workflow to be separated into stages such as preprocessing, standardization, mapping, verification, scoring, and summary generation so that the outputs would be more transparent and reviewable. The major deliverables from this phase included the database schema, workflow models, user interface mockups, and the design of the staged comparison process.

3.3.3 Development

The Development phase involved the incremental implementation of the UPTM Credit Transfer System based on the outputs produced in the Design phase. Development was carried out iteratively so that the system could be expanded module by module while allowing adjustments to workflow behavior and interface logic as needed. The project was implemented as a full-stack web application using Next.js, with PostgreSQL managed through Prisma for structured application data. Authentication was handled through Better Auth with Google sign-in, while transcripts, syllabi, and export files were managed through an S3-compatible storage layer.

Core development activities included implementing the student onboarding flow, CTA creation and step-based submission process, transcript ingestion, articulation matching, syllabus upload handling, HOP review workflow, RP review workflow, AAS export flow, and reporting features. Supporting integrations were also developed for document text extraction and staged syllabus comparison. As development progressed, shared syllabus handling, shared review grouping, workflow status synchronization, export generation, and scheduled deadline enforcement were incorporated to support

the complete CTA lifecycle. Version control was maintained through Git so that system changes could be introduced incrementally and managed systematically.

3.3.4 Testing

The Testing phase was conducted to verify that the system behaved correctly, supported the intended workflow, and remained usable for its target users. Because the project contained several interconnected modules, testing did not focus only on isolated functions, but also on whether the system behaved correctly across complete workflow scenarios. This included validating form handling, authentication flow, transcript processing, syllabus upload behavior, staged comparison flow, review decisions, reporting functions, and export generation.

Testing in this project involved a combination of script-based verification, manual functional checking, and workflow-oriented validation. Selected helper functions and reporting components were tested directly, while the larger workflow was validated through guided use of the implemented features. This approach was appropriate because the project involved multiple user roles and dependent workflow states that were difficult to assess through isolated testing alone. The main output of this phase was the identification and correction of workflow issues, interface problems, and integration errors before live deployment.

3.3.5 Deployment

The Deployment phase focused on making the system available in a live environment for practical use and controlled evaluation. The implemented system was deployed on a DigitalOcean droplet and managed through Coolify. In this deployment environment, the main Next.js application operated together with the supporting services required for database access, document handling, and staged syllabus comparison.

The purpose of this phase was to ensure that the system could operate reliably outside the local development environment and support realistic usage conditions. Deployment activities included environment configuration, database connectivity setup, authentication configuration, object storage access, and service-to-service communication for document extraction and syllabus comparison support. This phase also ensured that the CTA workflow, file management process, reporting features, and export functions could operate consistently in the live environment. The main deliverable of this phase was a working deployed system aligned with the implemented project architecture.

3.3.6 Review

The Review phase evaluated the implemented system in relation to the project objectives and the operational context of its intended users. This phase considered whether the system addressed the main issues identified earlier, namely the manual handling of CTA workflows, fragmented record management, and inconsistency in syllabus evaluation support. It also assessed whether the implemented system remained practical and appropriate for the institutional setting in which it was intended to operate.

The review focused on the completeness of the implemented modules, the clarity of the workflow, the usefulness of the staged syllabus comparison outputs, and the extent to which the system improved visibility and traceability across the CTA process. From this review, a number of strengths could be identified, including centralized CTA management, clearer role separation, improved record organization, and more structured review support. At the same time, certain limitations remained, such as continued dependence on reviewer validation for final academic decisions and the need for manual downstream entry into the Campus Management System. The Review phase therefore provided not only an assessment of the implemented system, but also a basis for identifying future enhancements and longer-term improvements.

3.4 Requirement

This section described the methods and outcomes of the requirements gathering process for the UPTM Credit Transfer System. The purpose of this phase was to ensure that the proposed system reflected the actual operational workflow at Universiti Poly-Tech Malaysia (UPTM) and addressed the needs of its intended users. Both functional and non-functional requirements were identified through user interaction and document analysis. The main user groups involved in this phase were students, Heads of Programme (HOPs), Resource Persons (RPs), and Academic Affairs Students (AAS) staff.

3.4.1 Data Gathering Techniques

Data gathering techniques were important in obtaining accurate information about stakeholder needs, workflow issues, and system expectations. Tahai and Moitra (1994) noted that software projects often failed because of incomplete or inaccurate understanding of user requirements. More recent studies also emphasized that appropriate elicitation methods were necessary to obtain relevant and high-quality requirements for successful software development.

For this project, data collection was conducted to ensure that the UPTM Credit Transfer System accurately reflected the workflow and expectations of its end users. The requirement gathering process combined several complementary techniques, namely interviews, questionnaires, and document

analysis. Each of these methods served a different purpose in understanding stakeholder perspectives, identifying operational difficulties, and defining system requirements.

3.4.1.1 Interviews

Semi-structured interviews were conducted with key stakeholders from the Faculty of Computing and Multimedia (FCOM) at Universiti Poly-Tech Malaysia (UPTM). This approach was selected because it allowed predetermined questions to be used while still providing flexibility for follow-up probing when stakeholders described workflow challenges or operational needs in greater detail.

The interview participants included three Heads of Programme (HOPs) from CT204, CT206, and CC101, who explained how they reviewed student transcripts, identified eligible UPTM courses for credit transfer, and assigned syllabus evaluations to Resource Persons. Two Resource Persons (RPs) provided input on the manual syllabus comparison process and highlighted issues such as inconsistent evaluation criteria, high cognitive workload, and the need for more structured comparison support. In addition, two officers from the Academic Affairs Students (AAS) unit described administrative issues involving physical document handling, repeated data entry into the Campus Management System (CMS), and the difficulty of tracking application progress. Collectively, these interviews provided perspectives from academic decision-makers, evaluators, and administrative staff, thereby forming a strong basis for defining system requirements.

3.4.1.2 Questionnaires

Questionnaires were used to collect structured feedback from a wider group of potential users, particularly students. In requirements analysis, questionnaires are useful for obtaining both quantitative and qualitative information about user needs, expectations, and experiences with current processes. They also allow broader user representation and reduce the risk of overlooking important requirements.

For this project, a digital questionnaire was prepared using Google Forms and distributed to students from Diploma in Computer Science (CC101), Bachelor of Information Technology (Honours) in Computer Application Development (CT204), and Bachelor of Information Technology (Honours) in Cyber Security (CT206) at UPTM. These students were selected because they represented the primary users of the CTA process and were directly affected by the limitations of the current manual workflow. The questionnaire consisted of structured questions focusing on usability and accessibility of the current process, preference for digital submission and application tracking, communication difficulties during the credit transfer process, and awareness of articulation references. The feedback obtained provided a broader view of student expectations and frustrations, and these findings were used together with the interview results to define system requirements.

3.4.2 Functional Requirement

Functional requirements described the services and features that the system needed to provide in order to support the intended credit transfer workflow. These requirements were derived from the interviews, questionnaires, and document analysis conducted during the requirements phase. In the UPTM Credit Transfer System, the main user roles were students, Heads of Programme (HOPs), Resource Persons (RPs), and Academic Affairs Students (AAS) staff. The functional requirements were therefore organized according to these user groups in order to clearly distinguish their responsibilities within the system.

Table 3.2: Functional Requirements for Students

Functional Requirement	Description
Login and Authentication	Students must be able to sign in securely to access the system.
Complete Onboarding	Students must be able to provide the required academic and personal details before starting the CTA process.
Submit Credit Transfer Application	The system must allow students to create and complete a CTA through the online workflow.
Upload Transcript	Students must be able to upload transcript documents for course extraction and review.
Review Extracted Courses	Students must be able to review the extracted or matched course information before proceeding.
Upload Syllabus	Students must be able to upload source course syllabi when further academic review is required.
Track Application Status	Students must be able to monitor the progress of their application throughout the CTA workflow.
Receive Notifications	The system should provide updates when important workflow actions or status changes occur.
Download Final Credit Transfer Slip	The system must allow students to download the final credit transfer slip after the application has reached completed status.
Withdraw Application	Students must be able to withdraw an application before final completion, subject to workflow conditions.

Table 3.3: Functional Requirements for Head of Program (HOP)

Functional Requirement	Description
Secure Login and Access	The HOP must be able to access the system according to role-based permissions.
View and Review Student Applications	The HOP must be able to review application details, transcript data, course mappings, and supporting documents.
Review Suggested Course Matches	The HOP must be able to review and adjust system-supported course mapping outcomes.
Manage Articulation Records	The HOP must be able to create, review, update, and maintain articulation mappings for future use.
Manage Syllabus Records	The HOP must be able to upload or manage target syllabus records used in comparison and review.
Request Additional Syllabus	The HOP must be able to request additional syllabus evidence where required.
Assign Resource Person (RP)	The HOP must be able to assign RPs for cases requiring further academic evaluation.

Table 3.4: Functional Requirements for Resource Person (RP)

Functional Requirement	Description
Secure Login and Access	The RP must be able to access assigned review cases according to role-based permissions.
View Assigned Review Cases	The RP must be able to view the syllabus comparison cases assigned for evaluation.
Run or Review Syllabus Comparison	The RP must be able to inspect the structured comparison output generated by the system.
View Comparison Summary	The RP must be able to review summary findings, mapped content, and supporting comparison information.
Provide Evaluation Feedback	The RP must be able to record comments and academic remarks regarding syllabus equivalency.
Approve or Reject Equivalency	The RP must be able to approve or reject the evaluated equivalency based on academic judgment.

Table 3.5: Functional Requirements for Academic Affairs Students (AAS) Unit

Functional Requirement	Description
Secure Login and Access	AAS staff must be able to access the system according to administrative permissions.
View Finalized Applications	AAS staff must be able to view applications that have completed academic review and have been finalized.
Export Application Data	AAS staff must be able to export approved application data for downstream administrative use.
Manage Export Records	AAS staff must be able to view and manage generated export records.
Mark Application as Completed	AAS staff must be able to mark the CTA process as completed after downstream administrative action has been performed.

3.4.3 Non-Functional Requirement

Non-functional requirements described the quality attributes and constraints that the system needed to satisfy in addition to its functional behavior. These requirements were important because they influenced the usability, dependability, security, and maintainability of the UPTM Credit Transfer System. The main non-functional requirements identified for this project are summarized in Table 3.6.

Table 3.6: Non-Functional Requirement

Functional Requirement	Description
Usability	The system must provide a clear and user-friendly interface that enables students, HOPs, RPs, and AAS staff to perform tasks with minimal confusion.
Performance	The system should provide acceptable response time for normal operations such as login, form submission, status retrieval, and record loading under typical usage conditions.
Security	The system must enforce secure authentication, role-based authorization, and controlled access to sensitive student and academic records.
Reliability	The system must preserve data consistency and remain operational without unexpected failure during normal use.
Maintainability	The system must be implemented in a structured and modular manner so that future updates and improvements can be introduced with minimal disruption.
Traceability	The system should maintain clear record linkage and workflow status visibility to support review, monitoring, and auditability.

These non-functional requirements ensured that the UPTM Credit Transfer System was not only functionally complete, but also usable, secure, reliable, and maintainable for practical institutional use.

3.4.4 System Requirement

System requirements described the hardware and software environment needed to develop, test, deploy, and operate the UPTM Credit Transfer System. These requirements were important in ensuring that the system could function reliably, support its intended workflow, and integrate with the supporting services used in the project. Clearly defined system requirements also helped ensure that the technical environment remained aligned with the actual implementation of the system.

3.4.4.1 Hardware Requirements

The hardware requirements specified the physical computing environment used during the development and testing of the UPTM Credit Transfer System. As the system involved a full-stack web application, database operations, document handling, file storage integration, and staged syllabus comparison support, sufficient processing capability, memory, and storage were required to ensure stable local development and testing. The project was developed and tested on the system configuration shown in Table 3.7.

Table 3.7: Hardware Requirements Specification

Component	Specification
Processor (CPU)	AMD Ryzen 7 5700X3D, 8-Core Processor @ 3.0 GHz (Boost up to 4.5 GHz)
Installed RAM	16 GB DDR4 3600 MHz
Graphics Card (GPU)	AMD Radeon RX 6800 XT (16 GB GDDR6 VRAM)
Storage	1 TB NVMe SSD
Operating System	Microsoft Windows 11 Pro, Version 23H2 (64-bit)

This hardware environment served as the primary local development and testing platform for the project. It supported the main application, database access, supporting service integration, and local validation of the implemented workflow. For institutions or individuals intending to deploy a similar system, the minimum recommended hardware requirements are shown in Table 3.8.

Table 3.8: Minimum Hardware Requitement Specifications

Component	Specification
Processor (CPU)	Modern 8-core processor (AMD Ryzen 7, Intel Core i7, or equivalent)
RAM	8 GB
Graphics Card (GPU)	Discrete GPU optional, but beneficial for AI model operations (e.g., NVIDIA RTX series)
Storage	512 GB SSD or higher
Operating System	Windows 11 (64-bit), Linux Ubuntu 22.04 LTS, or macOS equivalent

These hardware requirements were considered sufficient for supporting development, document handling, database operations, and the main application workflow. Although a discrete GPU was not mandatory for all environments, it could be beneficial in contexts where local model-serving or heavier comparison workloads were required.

3.4.4.2 Software Requirements

The software requirements referred to the main tools, frameworks, and platforms selected to support the development and operation of the UPTM Credit Transfer System. These tools were chosen based on their suitability for building a full-stack web application, handling structured data, supporting secure access, managing uploaded files, enabling staged syllabus comparison, and supporting deployment in a practical environment.

I. Next.js

Figure 3.2: Next.js (Vercel, 2025)

Next.js was selected as the primary application framework because it supported both front-end and server-side development within a single full-stack environment. This made it suitable for the UPTM Credit Transfer System, which required user interfaces, route handling, form processing, and workflow logic to operate in an integrated manner. Its architecture reduced the need to maintain

separate application layers for the frontend and backend, making development more manageable and consistent with the project scope.

II. Better Auth

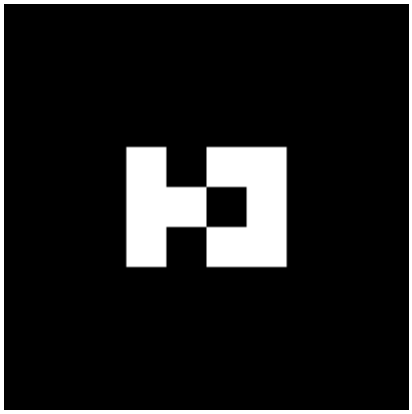


Figure 3.3: Better Auth (Better Auth, 2025)

Better Auth was selected to support secure authentication and session management. As the system involved multiple user roles with different levels of access, a structured authentication solution was necessary to ensure that only authorized users could access role-specific functions and sensitive records. Better Auth was appropriate for this purpose because it supported session handling and integration with the application's role-based access requirements.

III. PostgreSQL

Figure 3.4: PostgreSQL (PostgreSQL Global Development Group, 2025)

PostgreSQL was selected as the primary relational database because the system required reliable storage of structured data such as user records, applications, course details, syllabus metadata, articulation mappings, and workflow states. PostgreSQL was suitable for this project because of its stability, strong support for relational data, and ability to manage complex application records consistently.

IV. Prisma

Figure 3.5: Prisma ORM (Prisma ORM, 2026)

Prisma was selected as the database access and schema management tool because it simplified interaction between the application and PostgreSQL. It provided a structured way to define data models and improved development efficiency through type-safe database access. This made it suitable for a project with many related workflow entities and role-based records.

V. SeaweedFS

Figure 3.6: SeaweedFS (SeaweedFS, 2026)

SeaweedFS was selected as the file storage solution through its S3-compatible interface. The system required storage for transcripts, syllabus, and generated export files, and this storage needed to be separate from the main relational database. SeaweedFS was suitable because it allowed file handling to be managed through an object-storage approach while still integrating with the application workflow.

VI. Ollama

Figure 3.7: Ollama (Ollama, 2026)

Ollama was selected to support the staged syllabus comparison process and related extraction workflows. It was considered suitable for this project because it provided a practical model-serving environment that could support both local and externally hosted execution contexts when required. This flexibility made it appropriate for the staged comparison design adopted in the UPTM Credit Transfer System.

VII. Coolify

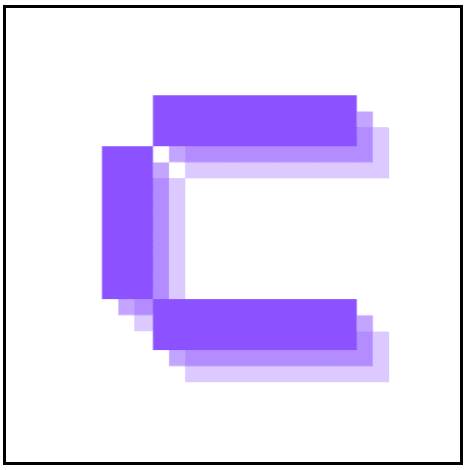


Figure 3.8: Coolify (Coolify, 2026)

Coolify was selected as the deployment management platform because it provided a practical way to host and manage the application and its supporting services in a live environment. It was suitable for this project because it simplified deployment management while supporting the hosting arrangement used for the implemented system.

3.4.5 Conclusion

This section presented the key requirements for the development of the UPTM Credit Transfer System. The requirement gathering process, carried out through interviews, questionnaires, and document analysis, provided useful insight into the operational challenges experienced by students, Heads of Programme (HOPs), Resource Persons (RPs), and Academic Affairs Students (AAS) staff. These findings formed the basis for defining the functional and non-functional requirements of the system, including user responsibilities, workflow needs, security expectations, usability considerations, and record management requirements.

In addition, the section outlined the system requirements in terms of hardware and software environment. The hardware requirements identified the computing resources needed to support development, testing, and system operation, while the software requirements described the main technologies selected to support the application, database, authentication, file storage, comparison support, and deployment environment. Together, these requirements established the technical and operational foundation needed for the system to function effectively and to support the intended credit transfer workflow.

Overall, the requirements presented in this section provided a clear foundation for the design and implementation of the UPTM Credit Transfer System. They ensured that the proposed solution remained aligned with user needs, institutional workflow expectations, and the selected technical environment. The following chapter builds on these requirements by presenting the system design in greater detail and showing how the selected components were structured into a complete credit transfer platform.

3.5 Analysis

This section described the analysis of the collected data and the conceptual basis used to model the proposed UPTM Credit Transfer System. The findings obtained through questionnaires, interviews, and document analysis were interpreted in order to identify user needs, workflow issues, role interactions, and system expectations. These findings then informed the modelling of system behavior, user roles, and process flow, which later supported the development of diagrams and design artifacts such as use case diagrams, workflow models, and database structures.

3.5.1 Data Gathering Analysis

The purpose of data gathering analysis in this project was to interpret the information collected from stakeholders and translate it into system understanding. In the context of system development, this process was important because the quality of the proposed solution depended on how accurately user

needs, workflow difficulties, and operational expectations were understood. The analysis therefore focused not only on collecting responses, but also on identifying recurring problems, user priorities, and opportunities for process improvement.

As described in Section 3.4.1, this study used a combination of interviews, questionnaires, and document analysis. These methods provided first-hand input from the intended users of the system, namely students, Heads of Programme (HOPs), Resource Persons (RPs), and Academic Affairs Students (AAS) staff. Each method contributed a different perspective on how credit transfer applications were currently managed and where improvements were needed. The following subsections present the analysis of the collected questionnaire and interview data.

3.5.1.1 Questionnaire Analysis

The questionnaire was prepared and distributed using Google Forms in order to provide an accessible and practical way to collect responses from students. This approach made it easier to distribute the questionnaire digitally, collect responses efficiently, and review the results through automatically generated summaries and charts. It also enabled broader participation from students across the selected programmes in the Faculty of Computing and Multimedia (FCOM) at Universiti Poly-Tech Malaysia (UPTM).

A total of 72 responses were collected from students enrolled in Diploma in Computer Science (CC101), Bachelor of Information Technology (Honours) in Computer Application Development (CT204), and Bachelor of Information Technology (Honours) in Cyber Security (CT206). These students were selected because they represented the main target users of the credit transfer process and were expected to have relevant experience or awareness of the existing workflow. Their responses were therefore useful in identifying the usability issues, communication problems, and workflow limitations experienced from the student perspective.

The questionnaire consisted of twelve structured questions, including both closed-ended and open-ended items. These questions were designed to examine student familiarity with the credit transfer process, measure their experience with the existing manual workflow, and identify their expectations for a more structured digital system. The following discussion presents the interpretation of the collected responses question by question, beginning with Question 1 as shown in Figure 3.8.

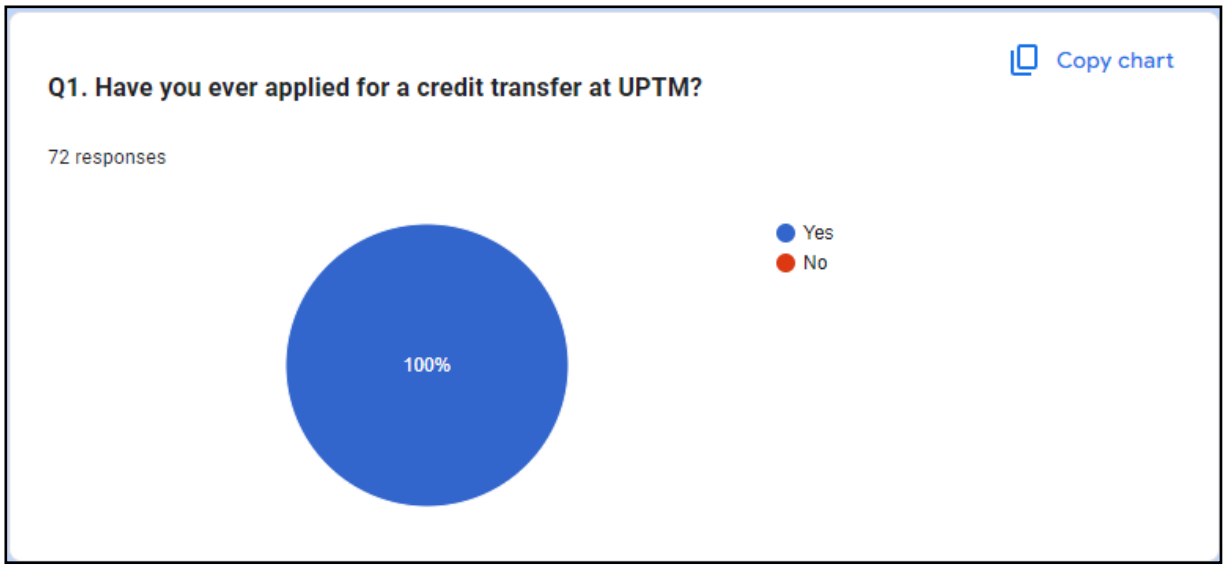


Figure 3.9: Result of Demographic Question 1

As shown in Figure 3.9, all 72 respondents, representing 100% of the questionnaire participants, indicated that they had previously applied for credit transfer at UPTM. This question was included to ensure that the respondents had direct experience with the process under study and were therefore able to provide relevant feedback based on actual participation rather than assumption.

This result was important because it showed that the questionnaire data reflected the experiences of users who had gone through the credit transfer process themselves. As a result, the responses obtained from the subsequent questions were more credible for identifying real workflow problems, communication issues, and user expectations. The finding therefore strengthened the value of the questionnaire as a source of requirement and workflow analysis for the proposed system.

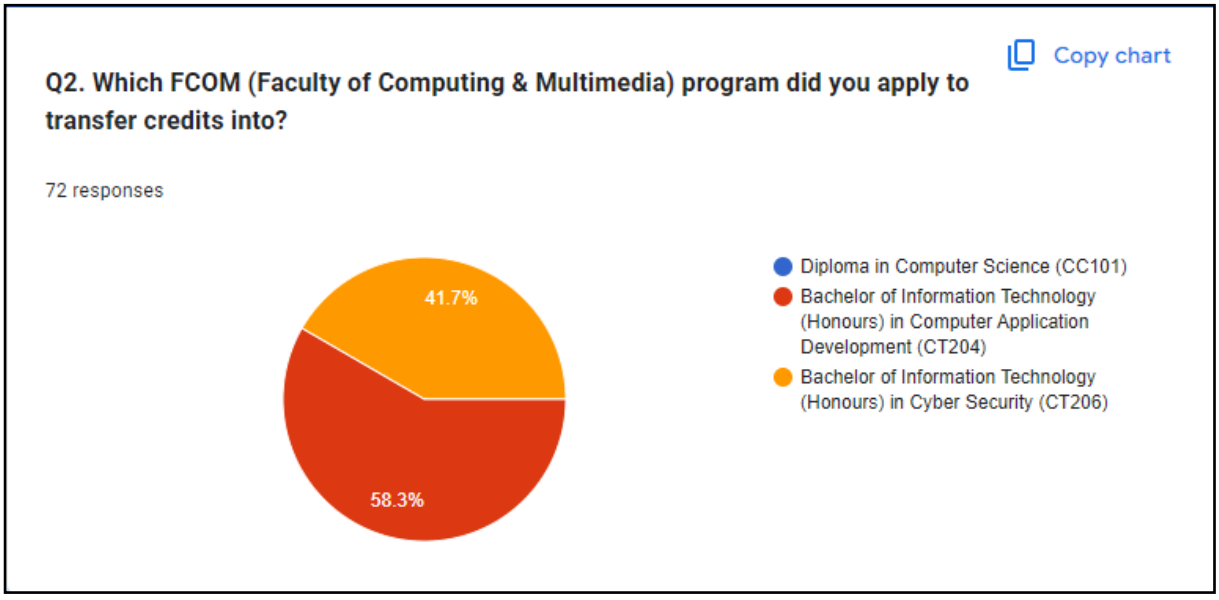


Figure 3.10: Result of Demographic Question 2

It can be seen from Figure 3.10 that there is the large proportion of the respondents (58.3%) are student from Bachelor of Information T echnology (Honours) in Computer Application Development (CT204) as compared to another one which is Bachelor of Information Technology (Honours) in Cyber Security programme(CT206), which belongs to 41.7% only. Participants were not drawn from the Diploma in Computer Science (CC101) since it was found that the majority of credit transfers were from diploma-level studies to degree studies. This pattern of transfer also suggests that the credit transfer function in UPTM is often used by students who are transferring to undergraduate programs and underlines the need for an automated system for evaluating prior learning when it comes to degree-level courses. Placements in the most impacted academic programs are identified, enabling an improved focus of the data management, subject mapping and course equivalency aspects of the suggested automated system addressing learners at a higher level.

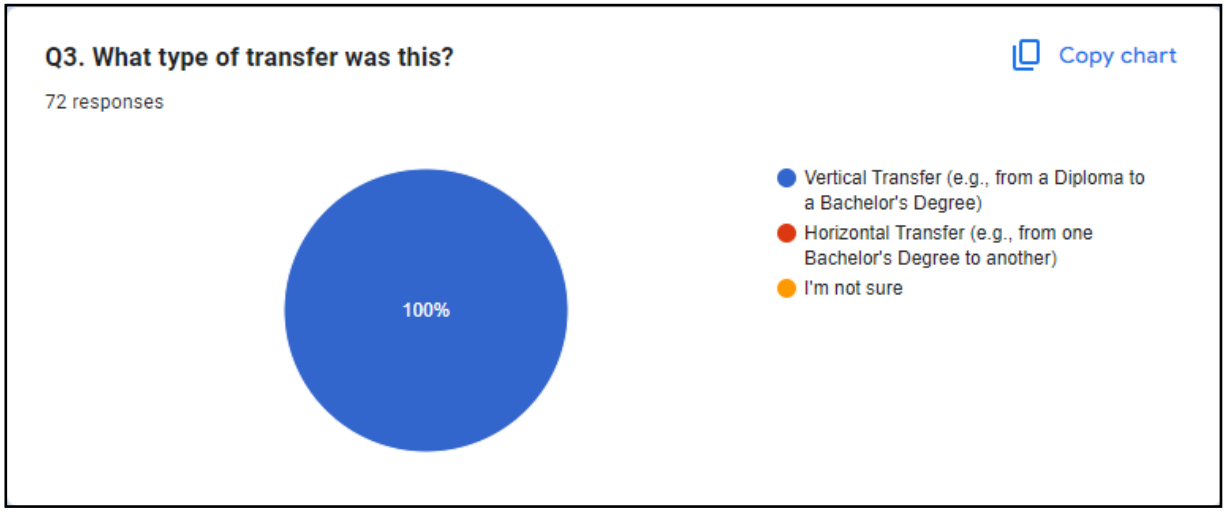


Figure 3.11: Result of Finding Question 3

As shown in Figure 3.11, all seventy-two (72) respondents, representing 100% of the sample, indicated that their credit transfer involved a vertical transfer from diploma level to bachelor's degree level. None of the respondents reported a horizontal transfer between programmes at the same qualification level. This finding suggested that the credit transfer cases most commonly experienced by the respondents were related to academic progression rather than programme switching. For the proposed UPTM Credit Transfer System, this result was important because it indicated that the system should primarily support vertical transfer scenarios, particularly the evaluation and mapping of diploma-level subjects to degree-level courses. This also reinforced the importance of syllabus comparison and structured review support in the context most frequently encountered by the target users.

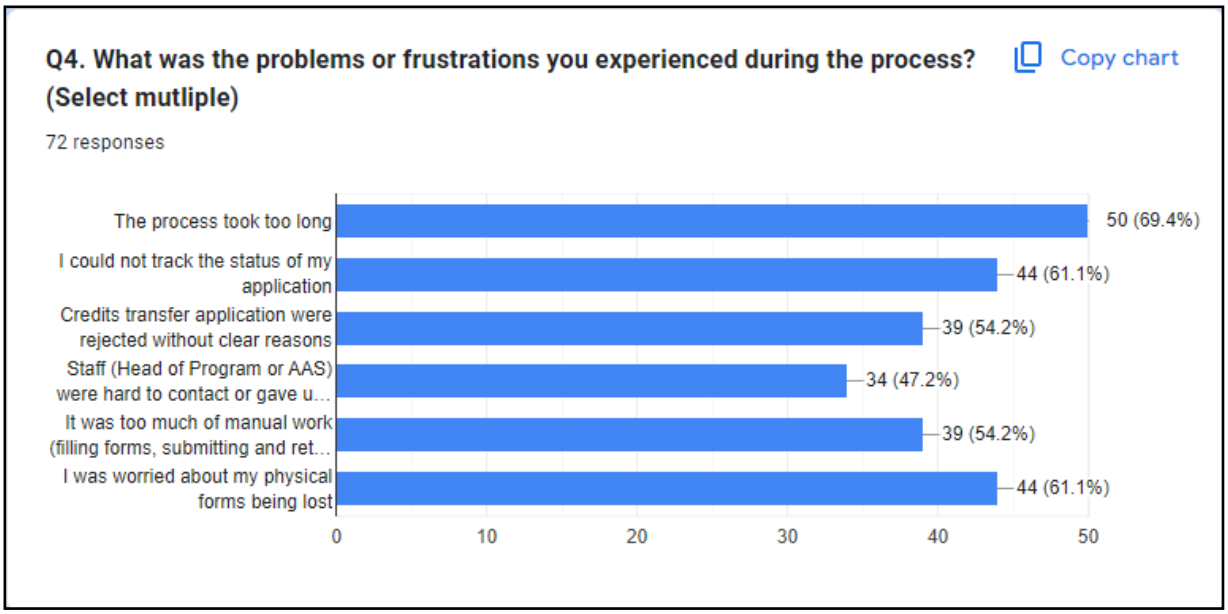


Figure 3.12: Result of Problem Experience Question 4

As shown in Figure 3.12, the most frequently reported problem was that the credit transfer process took too long, with 69.4% of respondents selecting this issue. This was followed by difficulty tracking the status of the application and concern that physical forms might be lost, both reported by 61.1% of respondents. In addition, 54.2% stated that applications could be rejected without clear explanation, while the same proportion highlighted the inconvenience of paper-based submission and retrieval. A further 47.2% reported that staff were sometimes difficult to contact or unable to provide sufficiently clear information. Taken together, these findings showed that the current process was affected by delay, limited transparency, heavy dependence on physical documents, and communication gaps. These issues supported the need for a centralized digital platform that could provide online submission, clearer status tracking, and more structured communication throughout the CTA workflow.

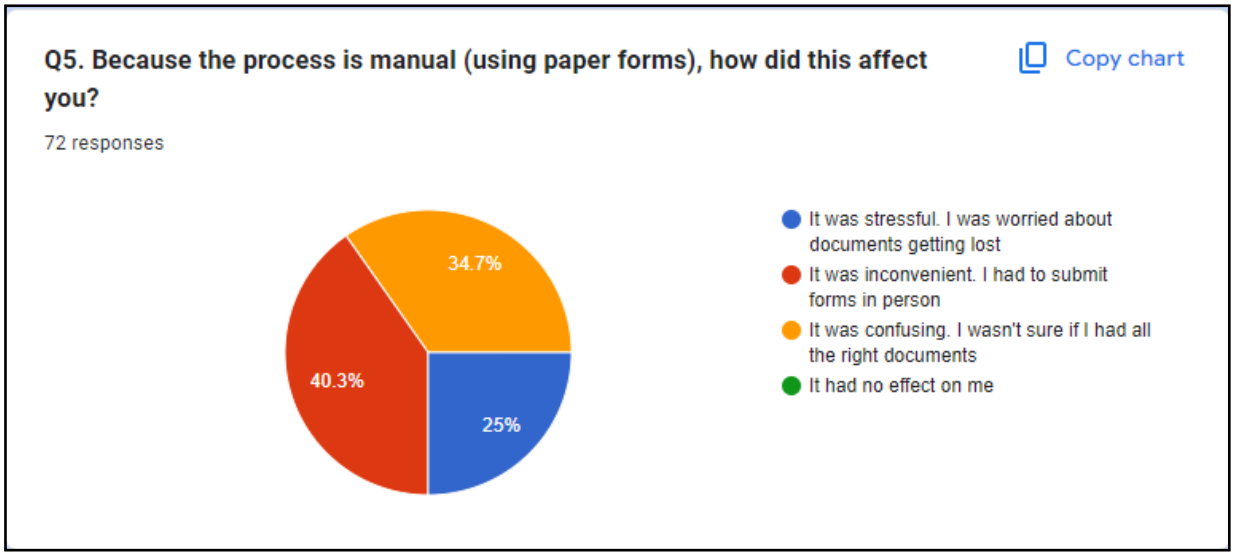


Figure 3.13: Result of Problem Experience Question 5

The results shown in Figure 3.13 indicated that the manual, paper-based process negatively affected students in several ways. Of the respondents, 40.3% stated that the process was inconvenient because forms had to be submitted in person, while 34.7% found it confusing because they were unsure whether all required documents had been prepared correctly. Another 25.0% reported that the process created unnecessary stress, particularly because of concern that submitted documents might be lost or misplaced. Notably, no respondents indicated that the manual process caused no frustration. This suggested that all participants experienced at least some level of difficulty as a result of the paper-based workflow. The finding therefore strengthened the justification for a digital system that could reduce manual handling, improve document security, and provide clearer guidance during submission.

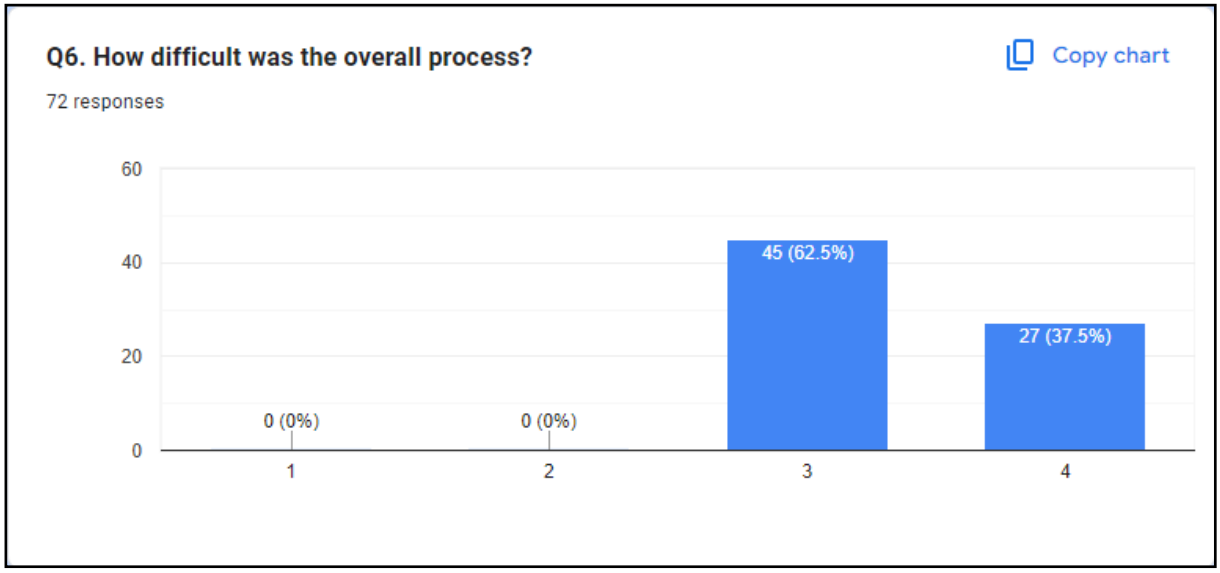


Figure 3.14: Result of Problem Experience Question 6

As illustrated in Figure 3.14, all respondents perceived the overall credit transfer process as difficult. A total of 62.5% selected “Difficult,” while 37.5% selected “Very Difficult.” No respondents rated the process as easy or very easy. This result showed a consistently negative user perception of the current workflow and suggested that the manual process imposed a substantial burden on students in terms of clarity, effort, and convenience. The finding further justified the need for a more structured and user-friendly digital workflow that could simplify submission steps, clarify required documents, and provide better visibility of application progress.

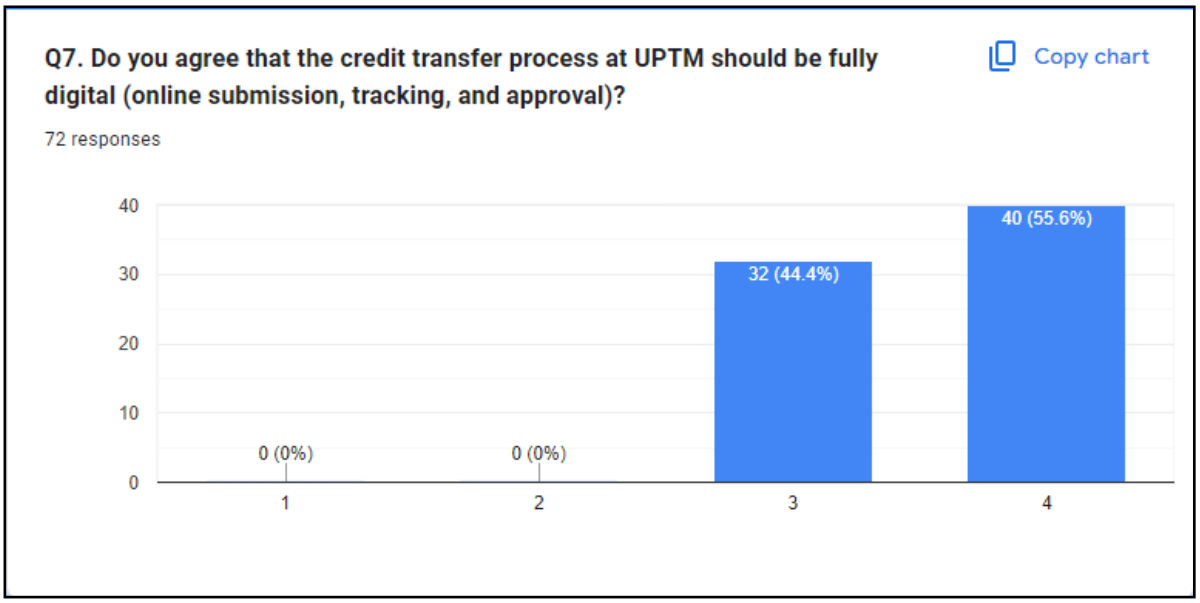


Figure 3.15: Result of System Need Question 7

As presented in Figure 3.15, all seventy-two (72) respondents expressed agreement that the credit transfer process at UPTM should be fully digital. Among them, 55.6% selected "4 – Strongly Agree," while 44.4% chose "3 – Agree." None of the participants disagreed or strongly disagreed, indicating unanimous support for a digital transformation of the current manual procedure. This result demonstrates that students overwhelmingly recognize the potential benefits of digitalization, including easier document submission, real-time tracking, and faster approval workflows. The consistency of positive responses highlights a clear demand for modernization and supports the core objective of this project is to design and develop a centralized automated credit transfer system that enhances transparency, efficiency, and accessibility. The findings also suggest strong user acceptance and readiness for adopting a digital platform, which increases the likelihood of successful implementation and long-term usage of the proposed UPTM Credit Transfer System.

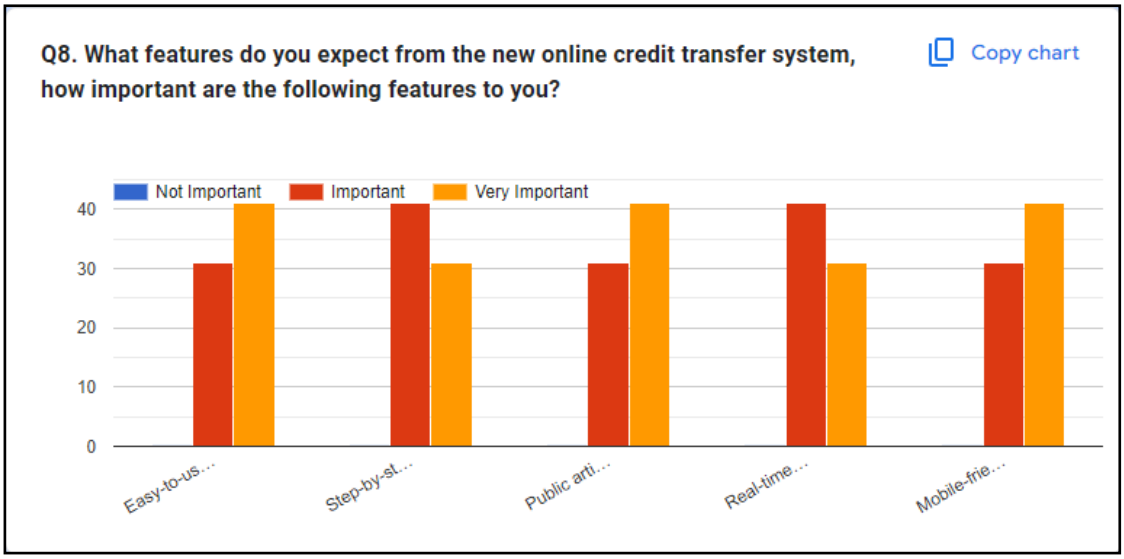


Figure 3.16: Result of Feature Expectation Question 8

As we can see in Figure 3.16, respondents were asked to rate several important elements of the proposed online credit transfer system according to their importance. Most students rated an online portal as important or very important to be easy and intuitive, highlighting the need for an accessible design that caters to diverse levels of student technology skill. The inclusion of step-by-step user guidance (e.g., videos; PDF tutorials) was another respected feature, suggesting that students want explicit instructions and direction throughout the application. The vast majority of respondents appreciated having access to a public articulation list where course equivalencies had previously been approved as doing so promoted transparency and provided students with information which would inform their decisions about whether or not it was appropriate to apply for credit transfer. The capability to track the status in real-time was also recognized as crucial for the system, enabling applicants to follow up their application through push notifications which could include statuses such as “Application Received”, “Under Review” or “Approved”. In general, it can be said that as the students' priority is in usability, transparency and accessibility, this feature will be included in UPTM Credit Transfer System design that influences digital reliability and efficiency.

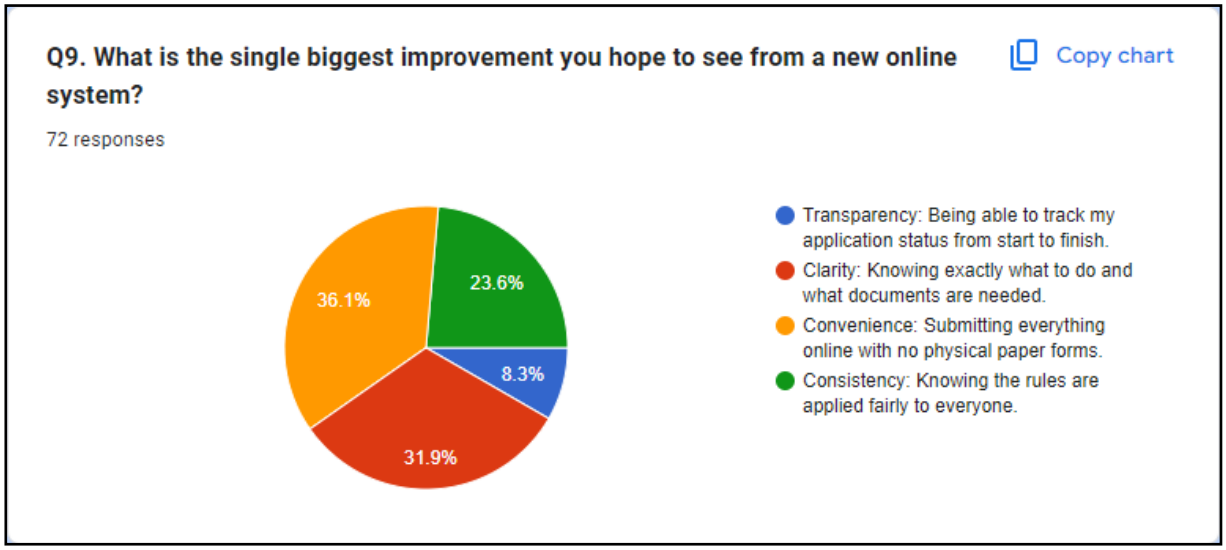


Figure 3.17: Result of System Improvement Question 9

As illustrated in Figure 3.17, the highest ranked improvement by survey participants was convenience (36.1% of students indicated a need to be able to turn all required documentation into an online submission without paper documents). Closely following this was clarity, 31.9% of respondents wanted more clearly defined tasks, step-by-step guidance and greater guidance on documentation required. At the same time, 23.6% specified consistency, meaning there is the need for a universally agreed-upon evaluation process so that all applications are assessed under the same conditions. Finally, 8.3% of students valued transparency—specifically around following the progress of their application from submission through final approval. Taken together, it appears that students anticipated the new online system to be convenient, easy to comprehend, and operated

clearly. These insights directly inform design priorities for the proposed UPTM Credit Transfer System (CTS): particularly, digital submission and intuitive user navigation are important, as well as standardized decision-making with support by automated comparison tools in the interest of fairness and efficiency.

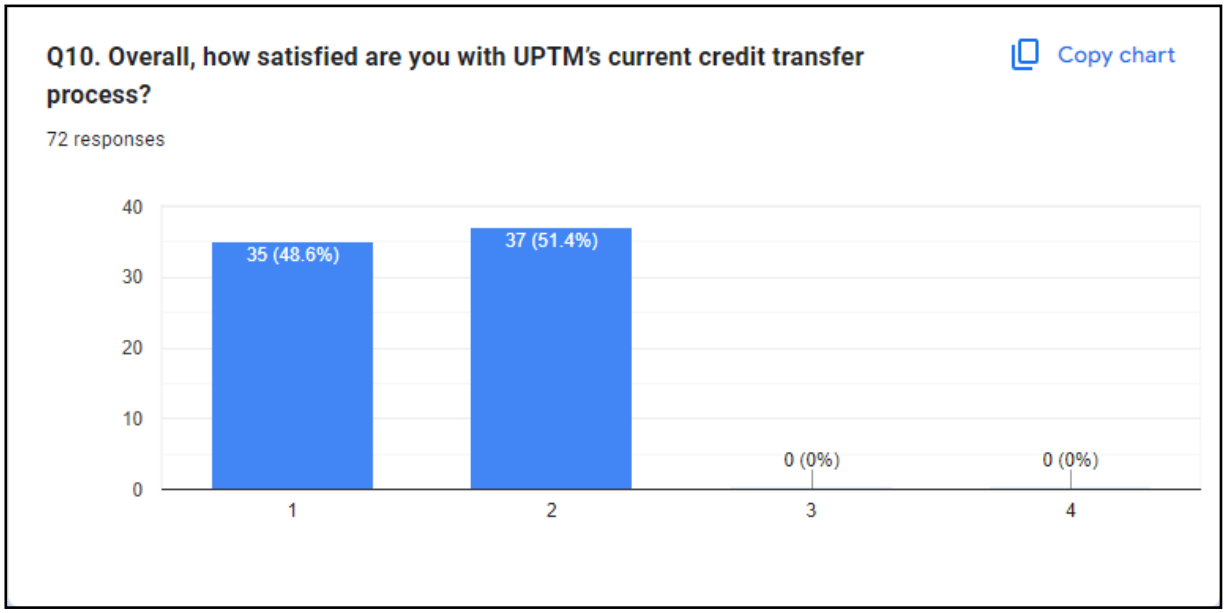


Figure 3.18: Result of Satisfaction Question 10

As illustrated in Figure 3.18, one hundred percent (100%) of the seventy-two (72) respondent s stated that they were not satisfied with UPTM's existing credit transfer system. In particular, 51.4% of respondents chose "2 – Dissatisfied" to rate their satisfaction level while 48.6% rated "1 – Very Dissatisfied." Not a single participant reported either neutral or above satisfaction, when there seems to be such widespread dissatisfaction of the current manual process. These findings also underline how the current system does not satisfy elements of efficiency, transparency and accessibility for students. The nearly unanimously negative response reiterates the previous responses in Questions 4 and 5, where excessive paperwork, long waiting periods for approvals, and an inability to track status were large pain points. This discontent highlights the imperative to transform the existing paradigm into an automated, transparent and user-friendly system for credit transfer. Challenges which needed to be deal with the proposed UPTM Credit Transfer System are real-time tracking, online document submission, automated equivalency evaluation in order to achieve the optimum user satisfaction and system efficiency.

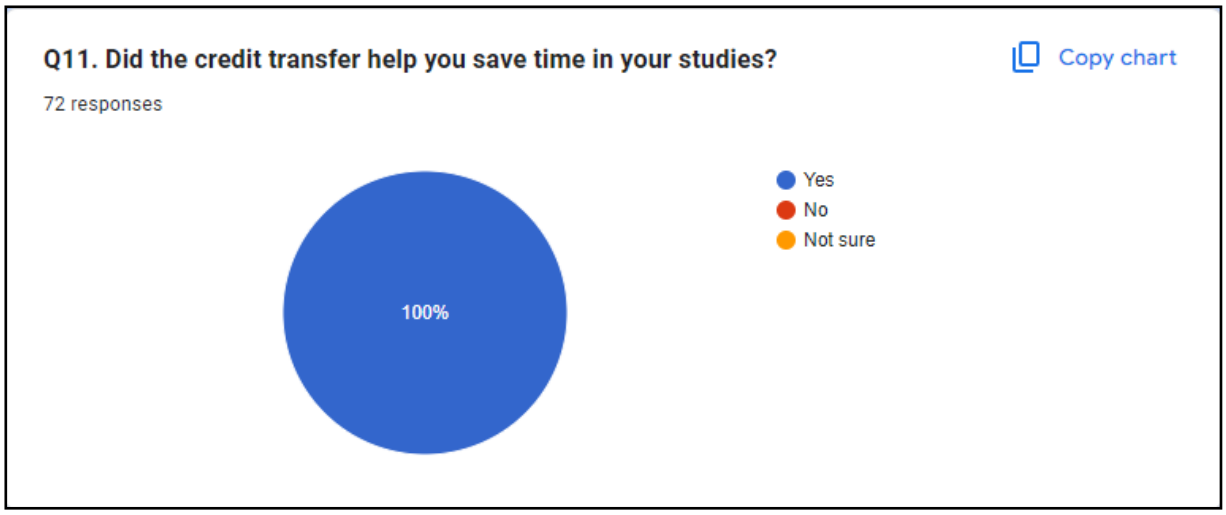


Figure 3.19: Result of Outcome Perception Question 11

As indicated in Figure 3.19, all (72) respondents who fell within the bracket of the 100% total participants were sure that credit transfer saved their time for studies. The above finding shows that the credit transfer system does indeed help in shortening the study period since students are able to take advantage of credits earned earlier. For instance, a three year programme could be made into two years by way of approved equivalent subjects. This result demonstrates the main advantage of a system of credit transfer. Thereby, permitting educational advancement without unnecessary repetition of subjects. In general, the students do acknowledge that credit transfer offers a real time-saving and greater academic freedom.

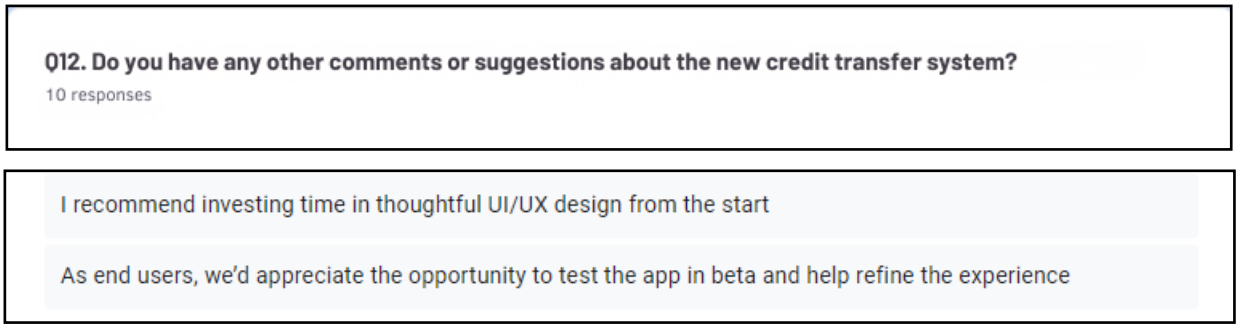


Figure 3.20: Result of Open Suggestion Question 12

For the final open-ended question as show in Figure 3.20, respondents were invited to share additional comments or suggestions regarding the proposed online credit transfer system. Out of seventy-two (72) participants, ten (10) students submitted written feedback, offering a range of perspectives on how the new system could be improved. Among the responses, two key comments were identified as the most constructive and relevant to the project's objectives. The first respondent emphasized the importance of investing time in thoughtful UI/UX design from the start, highlighting that a well-designed interface would significantly improve user accessibility and overall experience. The second respondent suggested that end users should be given the opportunity to test the system

in its beta stage, allowing real users to provide feedback and help refine the system before its official release. These comments reflect a user-centered perspective, emphasizing the importance of usability, inclusivity, and iterative testing in software development. The insights gathered from this question will be valuable during the design and evaluation phases of the proposed UPTM Credit Transfer System, ensuring that it not only meets technical requirements but also aligns with user expectations and practical needs.

3.5.1.2 Interview Analysis

The academic and administrative staff of Faculty of Computing and Multimedia (FCOM) in Universiti Poly-Tech Malaysia (UPTM) were interviewed. These interviews were conducted to acquire detailed qualitative feedback on the present current credit transfer process, understand current process biggest pain points, and receive suggestions for how to improve it. All interviews were face-to-face to enable open discussion and clarification where required. The participants were chosen according to their position in the credit transfer process, so that different practitioner perspectives are covered.

The interviews were categorized into three primary user groups:

I. Heads of Programme (HOP)
II. Resource Persons (RP)
III. Academic Affairs Students (AAS) Officers

Table 3.9: Interview Participant Summary

Participant ID	Position / Role	Program / Unit	Date of Interview
HOP 1	Head of Program	CT204 – Bachelor of Information Technology (Hons) in Computer Application Development	22 October 2025
HOP 2		CT206 – Bachelor of Information Technology (Hons) in Cyber Security	24 October 2025
HOP 3		CC101 – Diploma in Computer Science	29 October 2025
RP 1	Resource Person	Faculty of Computing and Multimedia	22 October 2025
RP 2			31 October 2025
AAS 1	AAS Officer	Academic Affairs Students Unit	6 November 2025
AAS 2			

Table 3.10: Analysis of Interview Questions Summary (HOP)

Question	Answer	Analysis
How long have you been involved as a Head of Programme, and how familiar are you with the credit transfer process?	- Involved between 1–5 years - Most are very familiar with the credit transfer process	Shows strong familiarity; confirms that problems are not due to lack of experience but due to manual workflow limitations
Can you describe the main steps when a student applies for credit transfer?	- If the past institution is not from UPTM (special case): • HOP checks the student's transcript and contacts the student to request syllabus (if the course is not in the articulation list). - A meeting with students is usually conducted in the first week of the semester. - Students fill in the CTA, mapping courses based on the articulation list. - Students submit syllabus (if not from UPTM). - HOP rechecks all submitted documents. - If the past institution is not from UPTM (special case): • HOP hands the syllabus to the RP. • RP performs syllabus comparison and fills in the articulation form. • HOP reviews and confirms the articulation form. - HOP submits the completed CTA to AAS. - Students check CMS to see whether the transfer is updated. - Students retrieve the processed CTA.	The steps reveal a highly manual and multi-stage workflow involving repeated document checking, manual syllabus retrieval, and physical form handling. The process depends heavily on coordination between students, HOPs, and RPs, which increases delays and risks of missing information. This highlights the need for automated workflow routing, digital submission, and centralized tracking to streamline the credit transfer process.

Question	Answer	Analysis
What documents do you collect from students?	- Student transcript - Course syllabus (required only if the previous institution is **not** UPTM)	The required documents are minimal but crucial; however, obtaining syllabi from external institutions adds extra steps and often causes delays. This indicates the need for a digital document upload system with clear requirements and automated checks to ensure completeness before processing begins.
How do you communicate with RPs and AAS?	- WhatsApp (primary communication method)	Using WhatsApp shows communication is informal and unstructured; a system with built-in notifications would make coordination clearer and more reliable.
What are the most common challenges or delays?	- Students submit wrong or missing information - Slow and time-consuming manual syllabus comparison	These challenges show that errors in student submissions and manual syllabus evaluation are major bottlenecks, highlighting the need for clearer digital forms and automated or assisted syllabus comparison to speed up the process.
How do you track the application progress?	- Tracked manually by asking RP, students, or AAS	Tracking is entirely manual and depends on asking different parties, showing the need for a centralized system that displays real-time progress without requiring constant follow-ups.
How do you verify all steps are completed before approval?	- Double-check each form manually	Verification depends entirely on manual checking, indicating the need for a system that enforces step-by-step completion and reduces the risk of overlooked requirements.

Question	Answer	Analysis
Problems retrieving or reviewing past records?	- Difficult because records are stored physically - Must check documents one by one	Physical record storage makes retrieval slow and inefficient, highlighting the need for a searchable digital database with organized historical records.
Desired features in an automated system?	- Report generation (statistics) - Digital storage in a database instead of physical documents	The need for automated reporting and digital record storage shows a desire for better data management and analytics, emphasizing the importance of a centralized system that reduces paperwork and improves accessibility.
How could a digital system improve efficiency and transparency?	- Easier to track application progress - Less paperwork - Faster overall processing	A digital system would streamline tracking, reduce manual paperwork, and accelerate processing, clearly improving both efficiency and transparency in the workflow.

Table 3.11: Analysis of Interview Questions Summary (RP)

Question	Answer	Analysis
How long have you been serving as a Resource Person, and how often are you involved in credit transfer evaluations?	- Serving as Resource Person between 5-10 years - Frequently involved in credit transfer evaluations, especially for courses taught such as Operating System	Shows extensive experience and continuous involvement, indicating that inefficiencies in the process are not due to lack of expertise but to the manual workflow.
Can you describe your role in the credit transfer process and what tasks you perform during an evaluation?	- Ensure diploma syllabus has at least 80% content similarity - Record the findings in the credit transfer evaluation form	The role is detail-intensive and heavily dependent on manual comparison and documentation, highlighting the need for structured digital tools.
What criteria do you normally use to determine whether two courses are equivalent?	- Minimum 80% average content similarity - Review of reference books if - syllabus content is unclear - Credit hours must be equivalent	Evaluation relies on multiple criteria that require careful manual judgment, emphasizing the value of a system that standardizes and assists in applying these criteria.
How do you usually obtain the syllabi or course outlines needed for your comparison?	- All syllabus documents are provided through the Head of Program	Dependence on HOP for documents slows down the process and suggests the need for centralized digital document access.
What are the main challenges you face when comparing syllabi manually?	- Unclear or overly simple syllabus content makes evaluation difficult - Need to review additional references or materials - Have to refill documents multiple times for repeated IPT evaluations - Sometimes receive inconsistent documents from different HOPs	Challenges show that manual comparison is time-consuming, error-prone, and inconsistent, supporting the need for AI-assisted comparison and standardized digital forms.

Question	Answer	Analysis
How do you currently record or document your evaluation results?	- Fill in a table-format credit transfer form - Compare topic-by-topic content and assign similarity percentages - Compute average percentage of equivalent topics - Sign the form before it is verified by Dean/Deputy Dean - Record credit hours, SLT, IPT course code, and other details	Recording is highly manual and repetitive, indicating that the system should automate calculations, form generation, and data entry.
Would you prefer if the system stored comparison data and documents in the cloud or kept them as physical copies?	- Prefers cloud system that can generate similarity percentages - Wants the system to search existing IPT records - Cloud system should summarize unclear content - Hardcopy usage depends on HOP requirements	The preference for cloud storage and automated features shows readiness for digital transformation and the need for centralized, searchable data.
Do you agree if the syllabus comparison process is automated using AI technology? Why or why not?	- Agrees because it simplifies the process	Strong support for AI automation reinforces the necessity of integrating AI tools to reduce manual workload and speed up evaluations.

Table 3.12: Analysis of Interview Questions Summary (AAS)

Question	Answer	Analysis
Can you walk me through what you usually do after receiving a completed credit transfer form from a coordinator or student?	- Verify the CTA is fully completed - Update student information in CMS - Stamp/sign the CTA	The process is entirely manual and involves multiple verification steps, indicating the need for automated checks and digital form submission.
After a credit transfer application is approved and entered into the CMS, how do you store or archive the physical form?	- File the form in the AAS physical storage cabinet	Relying on physical storage creates long-term retrieval and preservation issues, suggesting the need for digital archiving.
Have you ever experienced cases where physical credit transfer forms were damaged, misplaced, or lost? What happened?	- Yes, forms have been misplaced during busy intake periods - Required HOP or students to resubmit copies	Misplacement of physical forms highlights significant risks in the current system and reinforces the need for secure digital record management.
Do you find it difficult to process applications when the handwriting on the form is hard to read? Has this caused any issues?	- Unclear handwriting slows down data entry - Sometimes causes incorrect CMS entries - Requires contacting students or HOP for clarification	Handwritten forms introduce frequent errors and delays, emphasizing the need for standardized digital inputs.
What usually happens if a student submits a form with incorrect or incomplete information?	- Contact the student for corrections - Causes delays in processing	Frequent corrections indicate that students need clearer digital guidance and validation to prevent errors.
Frequent corrections indicate that students need clearer digital guidance and validation to prevent errors.	- Duplicate entries in CMS - High processing backlog during peak intake periods	These issues show a lack of workflow control and automated tracking, pointing to the need for better system validation and workload management.
What's your opinion on digitalizing the credit transfer application process for example, using an online form instead of paper?	- Strongly supports digitalization - Online forms reduce handwriting issues - Faster to process and update in CMS - Easier for tracking and storing	The officer's strong support for digitalization confirms that an online system would significantly improve efficiency, accuracy, and record management.

3.5.2 Use Case Model

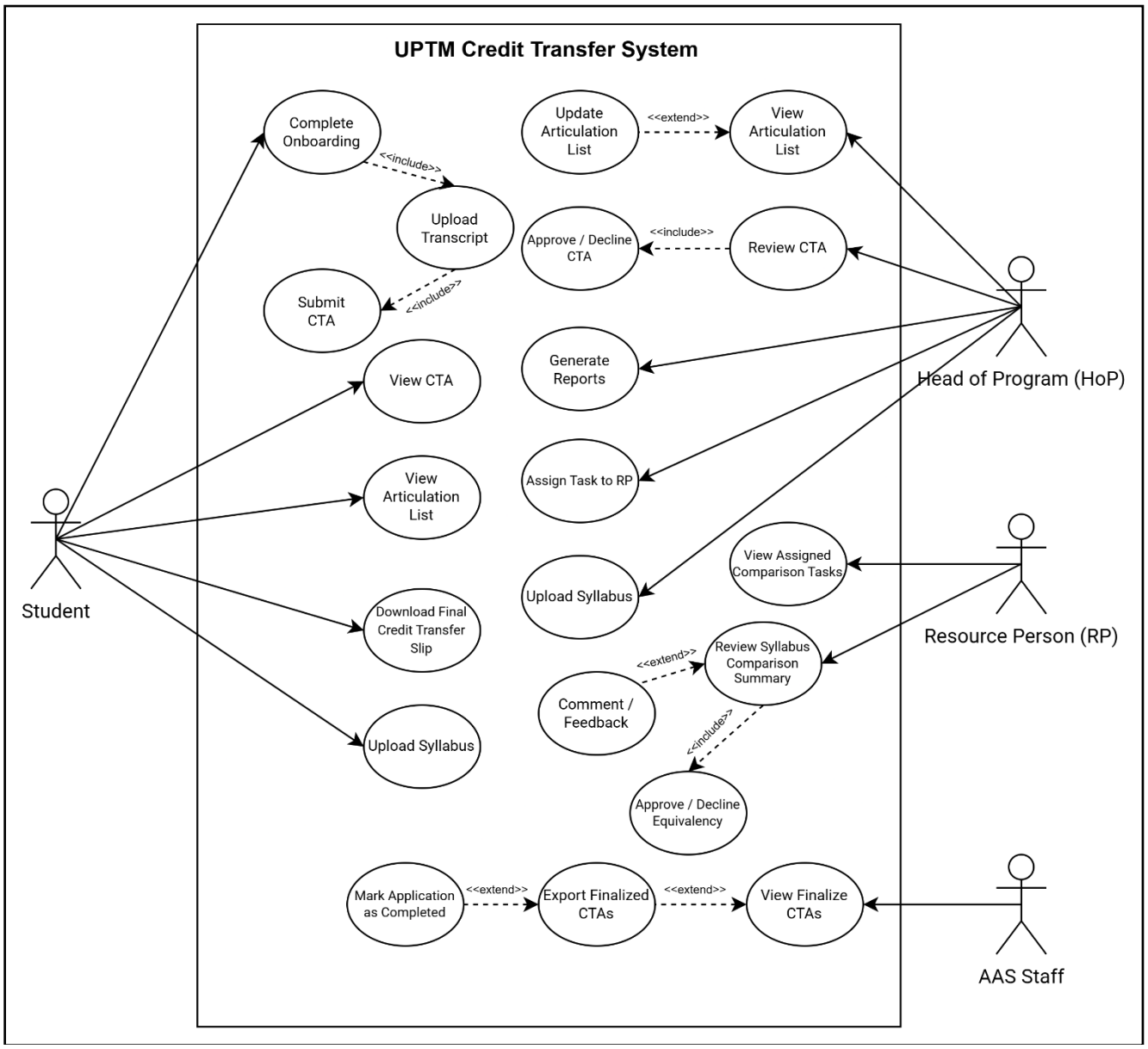


Figure 3.21: Use Case Diagram of UPTM Credit Transfer System

Figure 3.21 illustrates the use case diagram for the UPTM Credit Transfer System and shows the interactions between the four primary actors in the system, namely Student, Head of Programme (HoP), Resource Person (RP), and Academic Affairs Students (AAS) staff. The diagram presents the major functions available to each actor within the system boundary and reflects the overall structure of the proposed CTA workflow.

The Student actor represented the main initiator of the credit transfer process. The student interacted with the system to begin the application process, submit a CTA, upload transcripts and syllabus, view application details, refer to the articulation list, receive notifications, and download the final credit transfer result after completion. These use cases reflected the student's responsibility in initiating and monitoring the application from submission until the final outcome.

The Head of Programme (HoP) actor represented the main academic reviewer responsible for overseeing the CTA process at programme level. The HoP interacted with the system to review CTAs, upload syllabus records when necessary, assign review tasks to Resource Persons, update the articulation list, generate reports, and approve or decline CTA decisions. These use cases reflected the HoP's role in managing academic review, articulation maintenance, and final academic decision-making within the system. The Resource Person (RP) actor represented the subject-matter evaluator responsible for reviewing syllabus equivalency cases assigned by the HoP. The RP interacted with the system to view assigned comparison tasks, review syllabus comparison summaries, provide comments or feedback, and approve or decline course equivalency recommendations. These use cases reflected the RP's role in academic evaluation and decision support during the syllabus review stage.

The Academic Affairs Students (AAS) actor represented the administrative role responsible for handling finalized applications after academic review had been completed. AAS staff interacted with the system to view finalized CTAs, export or print records for downstream administrative use, and mark applications as completed. These use cases reflected the final administrative stage of the credit transfer workflow. Overall, Figure 3.21 showed the functional boundaries of the UPTM Credit Transfer System and demonstrated how each actor interacted with the digital platform to support a more structured, transparent, and manageable credit transfer process.

3.5.3 Flowchart

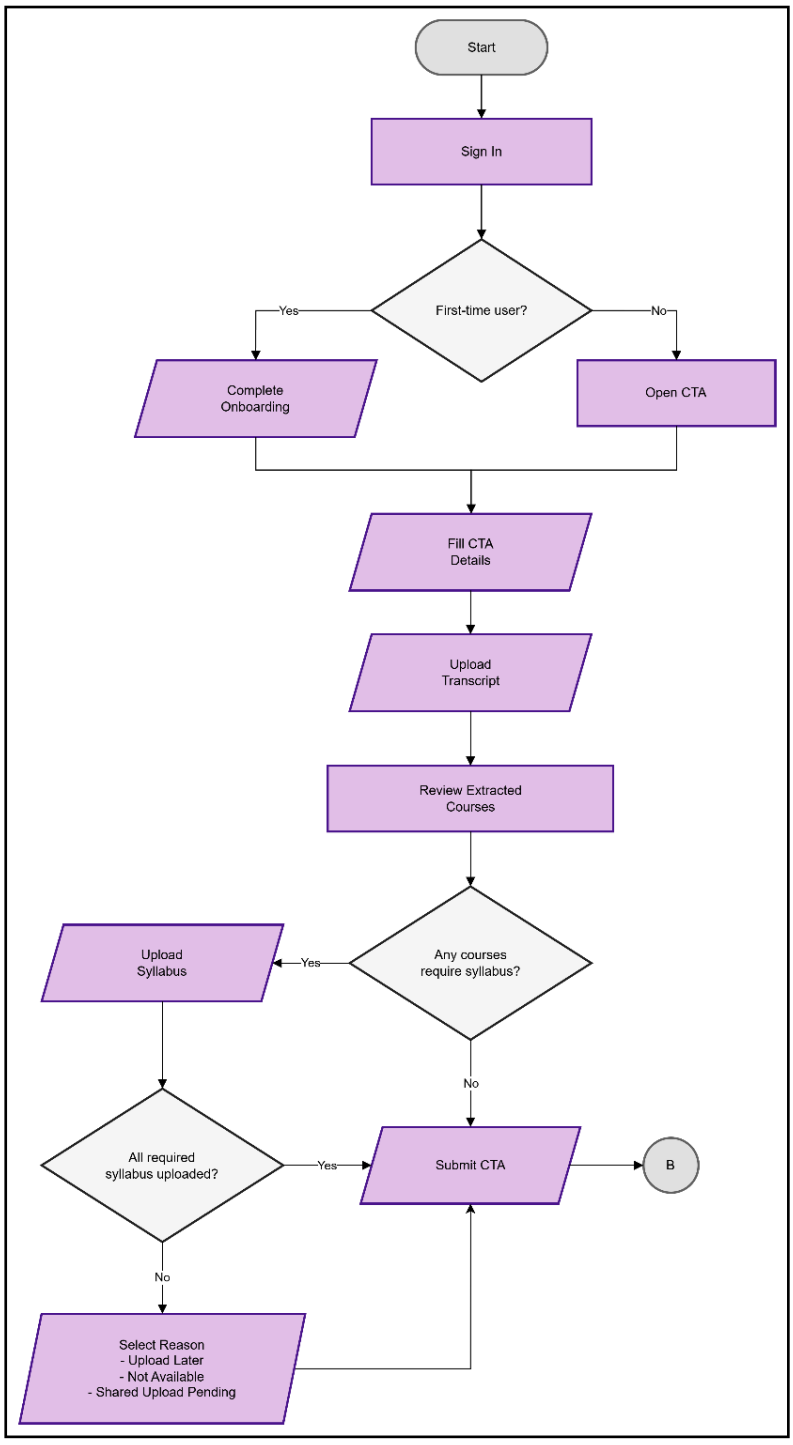


Figure 3.22: Student Submission Flow

Figure 3.22 illustrates the student submission flow in the UPTM Credit Transfer System. The process begins when the student signs in to the system and completes onboarding if it is their first time using the platform. After this, the student opens the Credit Transfer Application (CTA) and fills in the required application details. The student then uploads the transcript, which supports the next stage of the application process by allowing the system to identify the source courses that will be reviewed for transfer.

Once the transcript has been provided, the student reviews the extracted course information and the proposed course list before proceeding further.

The next stage of the flow concerns syllabus submission. If any courses require additional academic review, the student uploads the relevant syllabus documents for those courses. If the required syllabus documents cannot be fully provided at that point, the student may select an appropriate reason, such as uploading later, syllabus not available, or shared upload pending, before continuing with the submission. The flow then ends with the student submitting the CTA for the next stage of review. Overall, Figure 3.22 shows that the student process is not limited to a single form submission, but instead follows a guided sequence that includes data entry, transcript submission, course review, and syllabus handling before the application is formally submitted.

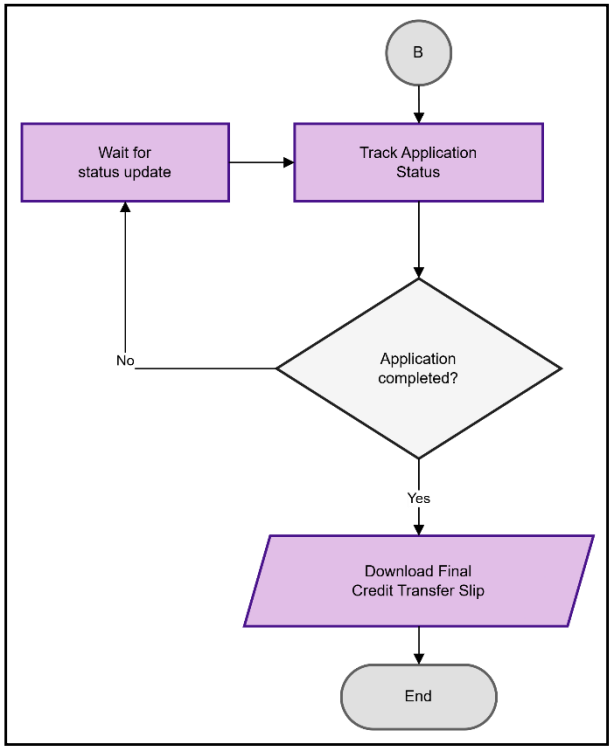


Figure 3.23: Student Post-Submission Flow

Figure 3.23 illustrates the student flow after the CTA has been submitted. Once the application is submitted, the student's main role shifts from data entry to monitoring the progress of the application throughout the review process. Through the system, the student can track the application status and observe how the CTA progresses through the subsequent workflow stages.

This post-submission flow remains active until the application reaches completed status. At that point, the student is able to download the final credit transfer slip as the concluding output of the process. In this way, Figure 3.23 complements the earlier submission flow by showing that the student's involvement does not end immediately after submission, but continues through status monitoring until the credit

transfer process is fully completed. Together, Figures 3.22 and 3.23 provide a complete representation of the student journey in the UPTM Credit Transfer System, from initial application preparation to final result retrieval.

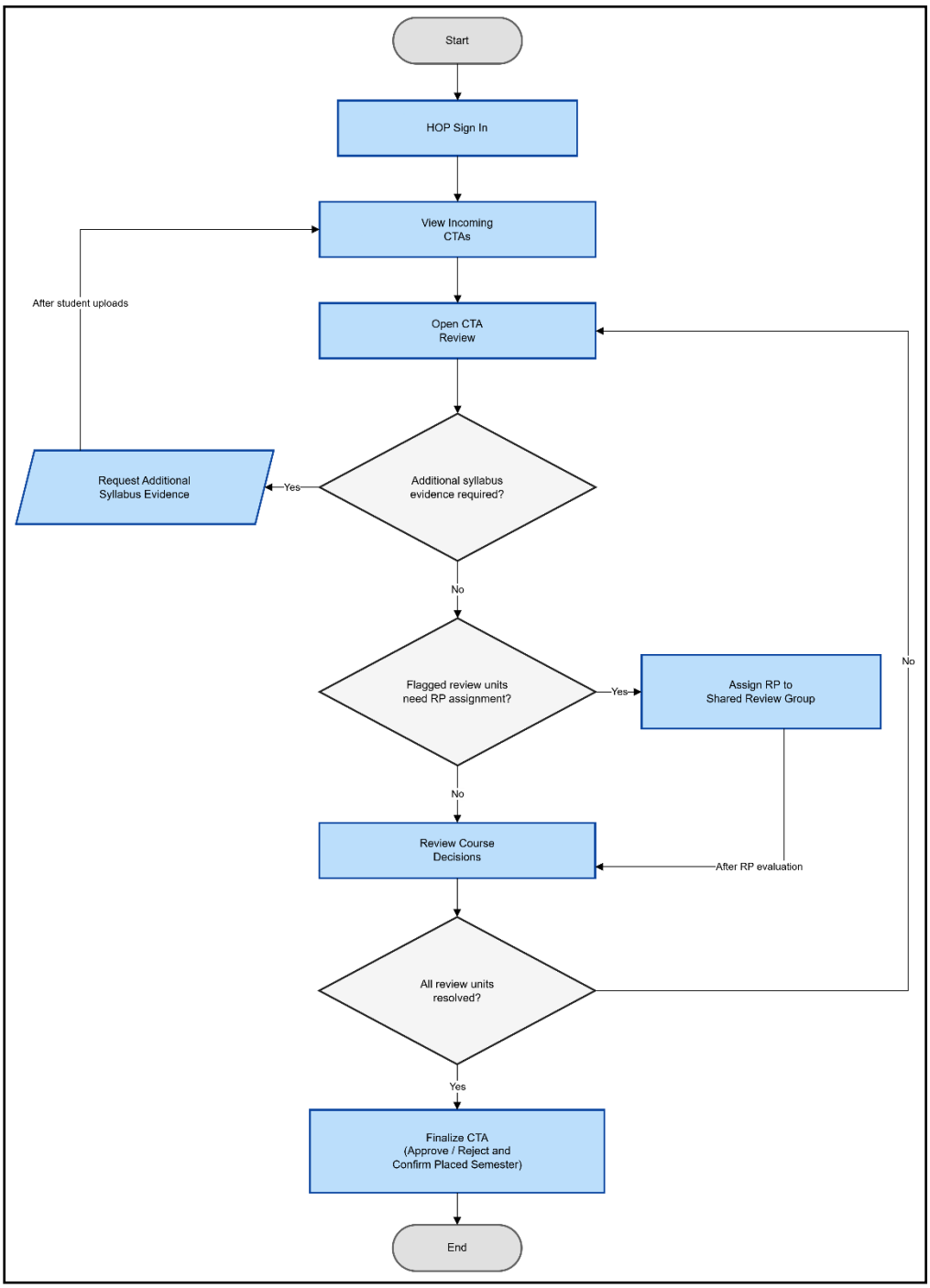


Figure 3.24: Head of Program Flowchart

Figure 3.24 illustrates the main review and decision flow performed by the Head of Programme (HoP) in the UPTM Credit Transfer System. The process begins when the HoP signs in to the system and accesses the incoming Credit Transfer Applications (CTAs) awaiting academic review. The HoP then

opens a selected application and examines the relevant information, including the submitted CTA details, extracted course records, flagged review units, and the availability of supporting syllabus evidence.

At the first decision stage, the HoP determines whether additional syllabus evidence is required before the review can continue. If further supporting documents are needed, the HoP requests additional syllabus evidence from the student. In such cases, the application does not proceed immediately and returns to the review queue until the required evidence has been submitted. If no further syllabus evidence is needed, the HoP continues with the review process. The next decision concerns whether any flagged review units require assignment to a Resource Person (RP). Where deeper academic evaluation is necessary and the relevant syllabus evidence is available, the HoP assigns the case to an RP for review. After the RP completes the evaluation, the application returns to the HoP for further review. If no RP assignment is required, the HoP may proceed directly to reviewing the course decisions within the application.

Once the course-level outcomes have been reviewed, the HoP determines whether all review units in the application have been fully resolved. If some parts of the application are still unresolved, the CTA remains in the review cycle until all required academic decisions have been completed. If all review units have been resolved, the HoP finalizes the CTA by confirming the academic outcome, including approval or rejection and the placed semester where applicable. At this stage, the application completes the main academic review flow and is ready to proceed to the next administrative stage. Overall, Figure 3.24 shows that the HoP acts as the central academic decision-maker in the system by reviewing student applications, requesting additional evidence when required, assigning RP evaluation for complex cases, and making the final academic decision. Although the HoP also manages the articulation repository elsewhere in the system, that responsibility is separate from the core CTA review flow shown in this figure.

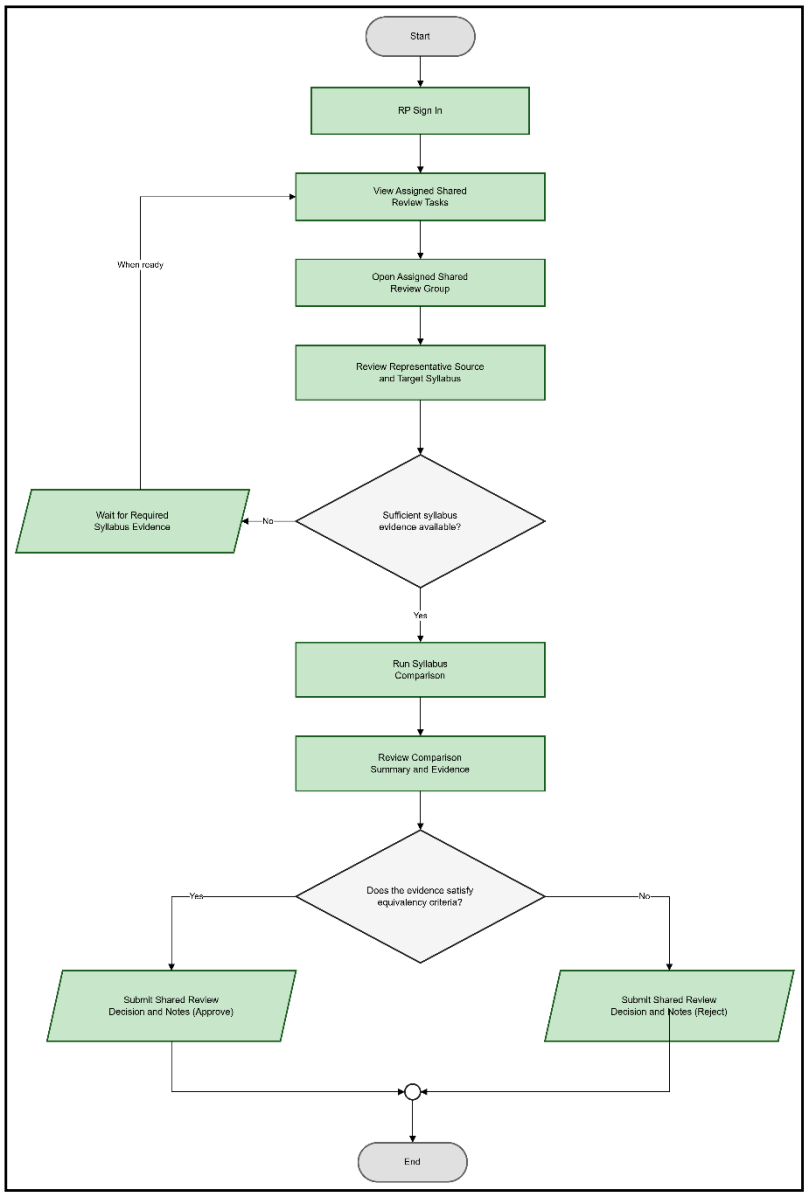


Figure 3.25: Resource Person Flowchart

Figure 3.25 illustrates the main review flow performed by the Resource Person (RP) in the UPTM Credit Transfer System. The process begins when the RP signs in to the system and accesses the assigned shared review tasks. The RP then opens the assigned shared review group and reviews the representative source syllabus together with the corresponding target syllabus. At this stage, the RP determines whether sufficient syllabus evidence is available to continue the evaluation. If the required evidence is still incomplete, the review is delayed until the necessary syllabus materials are available.

Once sufficient evidence is available, the RP proceeds to run the syllabus comparison and then reviews the comparison summary and supporting evidence produced for the shared review group. Based on this review, the RP determines whether the evidence satisfies the equivalency criteria for credit transfer. If the criteria are satisfied, the RP submits a shared review decision and notes for approval. Otherwise, the RP submits a shared review decision and notes for rejection. The process then ends after

the RP decision has been recorded. Overall, Figure 3.25 shows that the RP is responsible for the academic evaluation of flagged syllabus-dependent review cases. The flow focuses on reviewing syllabus evidence, conducting comparison, and recording a shared academic decision that supports the subsequent decision-making process in the system.

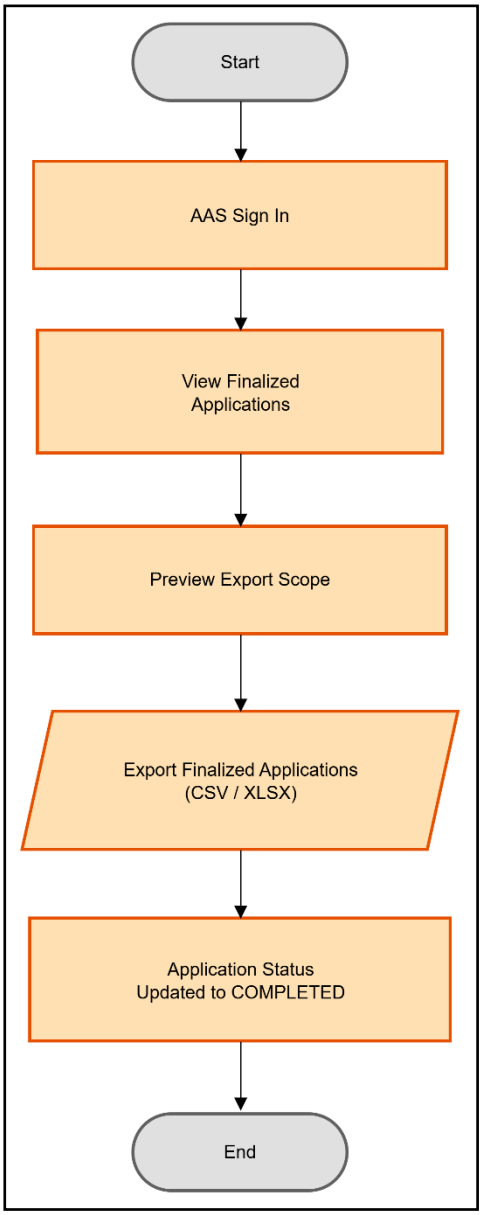


Figure 3.26: AAS Flowchart

Figure 3.26 illustrates the main operational flow performed by the Academic Affairs Students (AAS) role in the UPTM Credit Transfer System. The process begins when the AAS user signs in to the system and accesses the list of finalized applications. These applications have already completed the academic review stage and are ready for administrative export. Before proceeding, the AAS user may preview the export scope to determine whether the output should cover a single application, a selected set of applications, or the current filtered set.

Once the export scope has been confirmed, the AAS user exports the finalized applications in CSV or XLSX format for downstream administrative use. During this process, the system generates the export records and updates the application status from FINALIZED to COMPLETED. The flow then ends after the applications have been operationally closed.

Overall, Figure 3.26 shows that the AAS role is responsible for the final administrative handoff of credit transfer applications. Unlike the HOP and RP roles, AAS does not make academic decisions, but instead manages the export and completion stage that concludes the application lifecycle in the system.

3.6 Conclusion

This chapter presented a comprehensive analysis of the requirements for the UPTM Credit Transfer System through the use of multiple data-gathering techniques, including interviews, questionnaires, and document analysis. The findings from Heads of Program, Resource Persons, AAS officers, and students provided a clear understanding of the existing challenges in the current manual credit transfer process, such as inconsistent documentation, communication delays, difficulty in retrieving records, and the high administrative workload involved in syllabus comparison and CTA management. These insights were critical in defining the functional and non-functional requirements necessary to support a modern, efficient, and transparent credit transfer workflow.

The chapter also outlined the proposed system structure using use case diagrams, data flow diagrams, and detailed flowcharts for each user role. These models represent how students, HoPs, RPs, the Dean or Deputy Dean, and the AAS unit interact with the system and how each component contributes to a streamlined credit transfer process. Overall, this chapter establishes a solid foundation for the system design and architecture described in Chapter 4, ensuring that the proposed solution is aligned with real operational needs and institutional requirements.

4 DESIGN

4.1 Introduction

This chapter introduces the overall context and framework for the project. It begins with the project background, which describes the institutional setting and current practices in the credit transfer process at UPTM. The problem statement follows, identifying the specific issues and limitations that the project seeks to address. Next, the objectives of the project are outlined to provide clear direction for the proposed solution. The chapter also defines both the product and project scope, establishing the boundaries and intended functionalities of the system. target users also are repeatedly described with very detailed explanations. This is important to design requirements for credit transfer system.

4.2 Interface Design

This section presents the wireframe design of the UPTM Credit Transfer System. The wireframes were developed to show the arrangement of interface components, navigation structure, and main workflow areas for each user role before the final visual design was applied. These wireframes are important because the system supports several different actors, namely Student, Head of Programme (HOP), Resource Person (RP), and Academic Affairs Students (AAS), and each role requires a different interface based on its responsibilities in the credit transfer process.

4.2.1 Shared Interface Components

Across the wireframes, several interface components are shared to maintain consistency throughout the system. These shared components include the top navigation header, the left sidebar menu, page titles, search bars, filter controls, summary cards, tables, and action buttons. Keeping these elements consistent helps users move between modules more easily and supports a clearer workflow from application submission until final export.

4.2.2 Student Interface Design

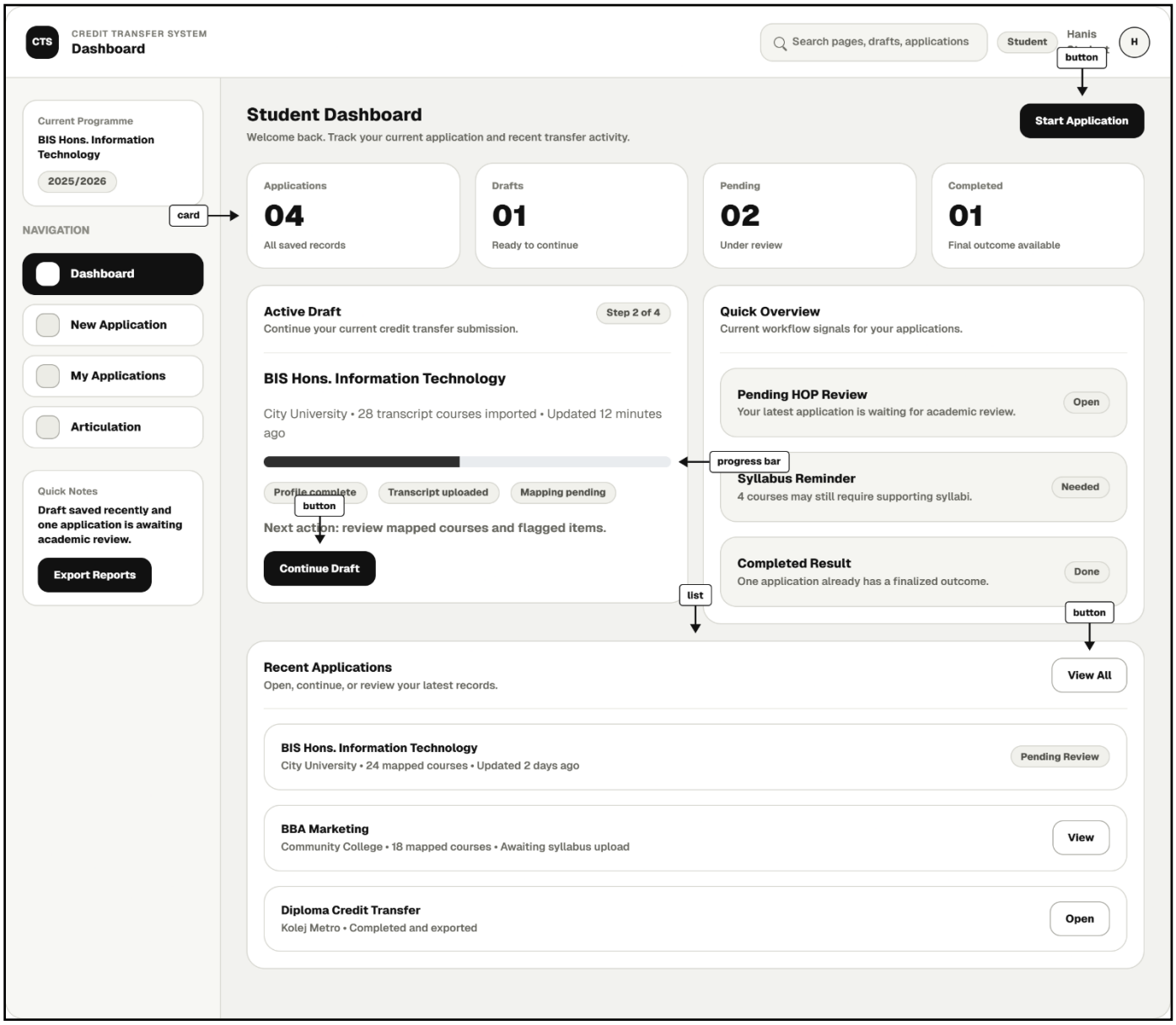


Figure 4.1: Student Dashboard Page

Figure 4.1 shows the Student Dashboard page. This page provides the student with a summary of current credit transfer activity, including application counts, draft progress, and recent application records. It helps the student identify the current stage of the application and decide whether to continue a draft, review an existing submission, or open the applications list for more detail.

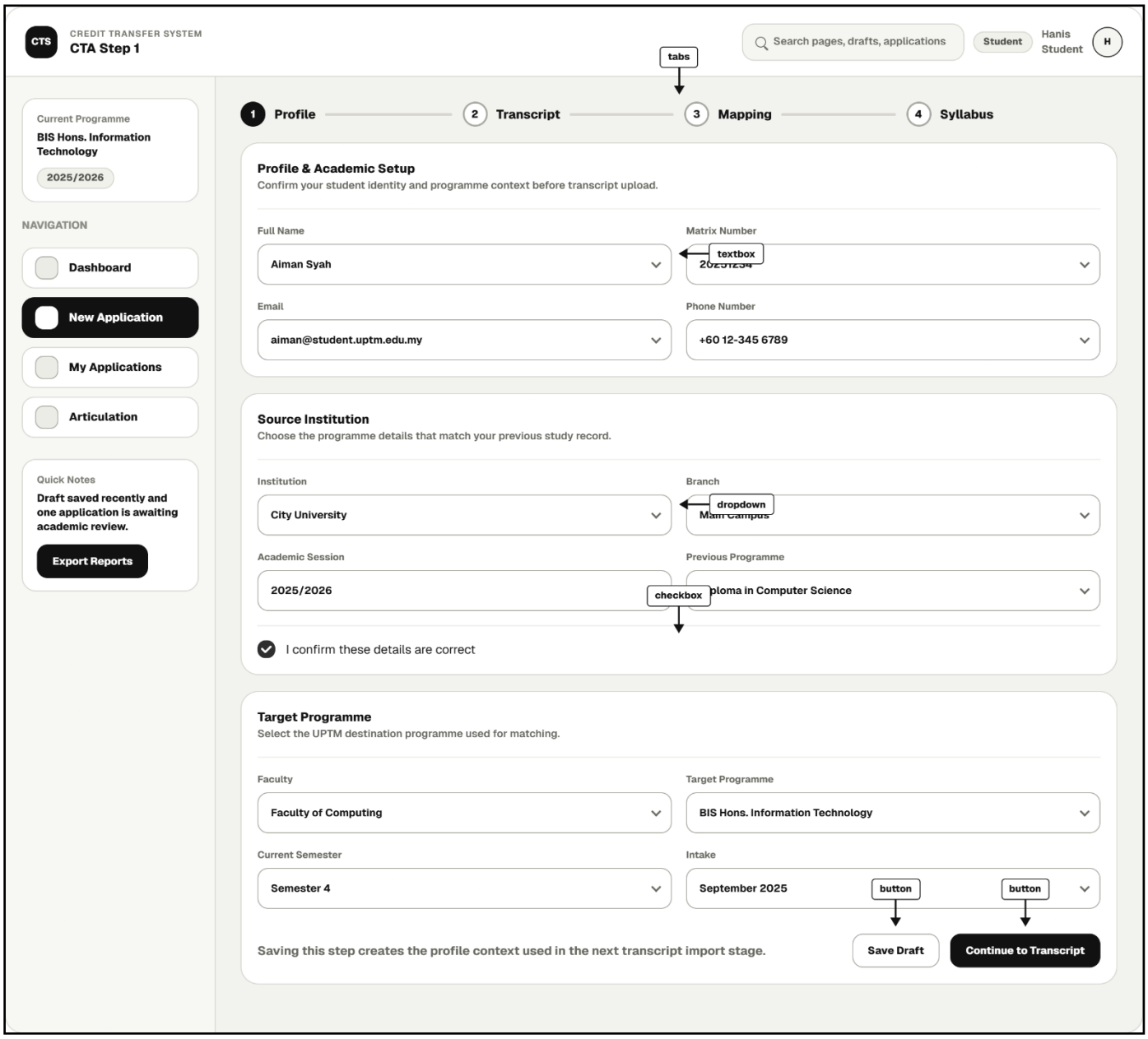


Figure 4.2: Student CTA Step 1 Page

Figure 4.2 shows the Student CTA Step 1 page. This page is used to enter and confirm the student's academic and application information, such as source institution, target programme, academic session, and personal details. The information entered here forms the application context that will later be used for transcript extraction and articulation matching.

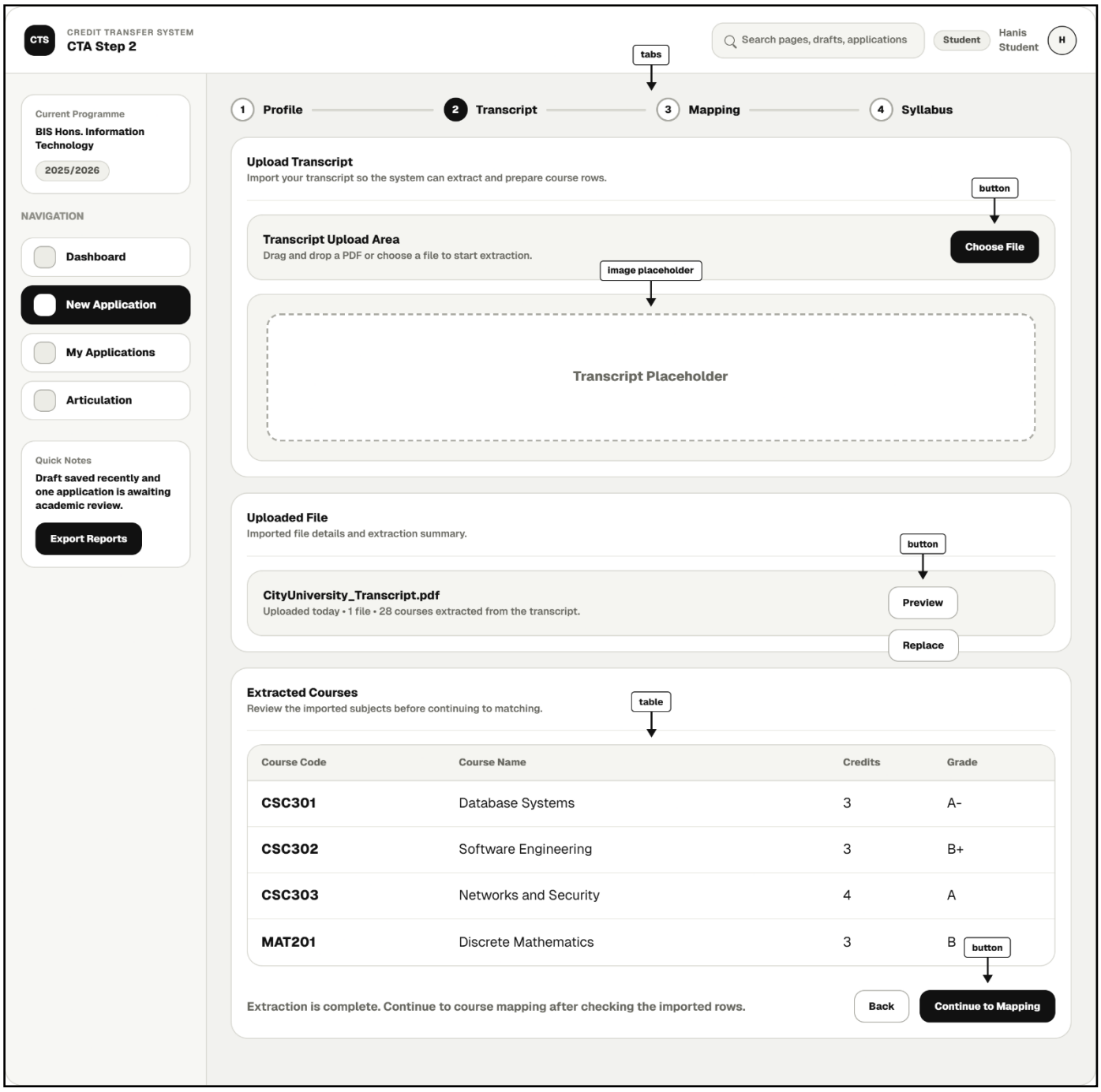


Figure 4.3: Student CTA Step 2 Page

Figure 4.3 shows the Student CTA Step 2 page. This page allows the student to upload the transcript and review the extracted course records generated by the system. It is the main page for turning the uploaded transcript into structured course data that can be matched against articulation rules.

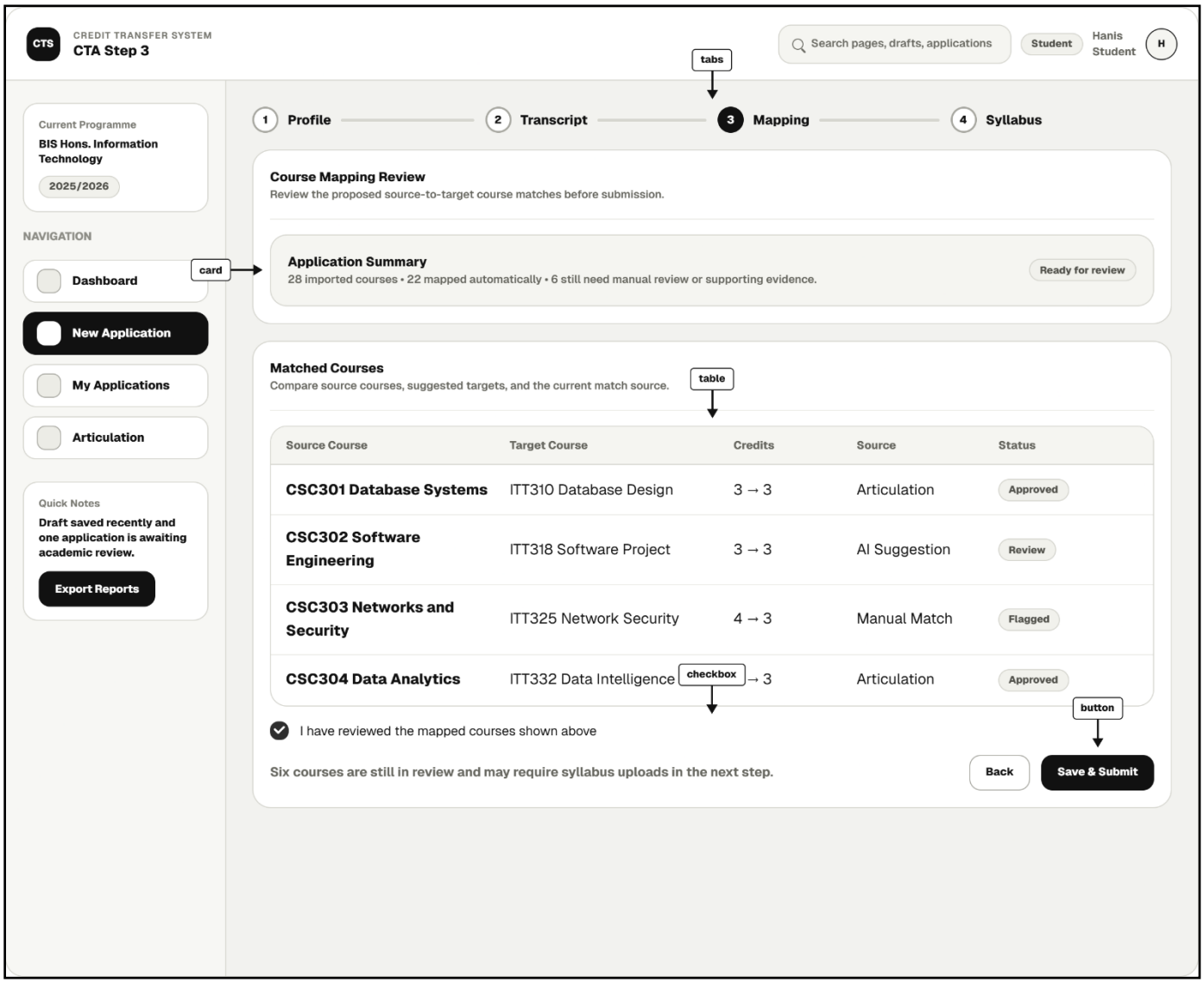


Figure 4.4: Student CTA Step 3 Page

Figure 4.4 shows the Student CTA Step 3 page. This page presents the matched course list together with the eligibility and mapping outcomes produced after transcript extraction. It helps the student review which courses were automatically approved, which courses require additional evidence, and which courses are not eligible for transfer.

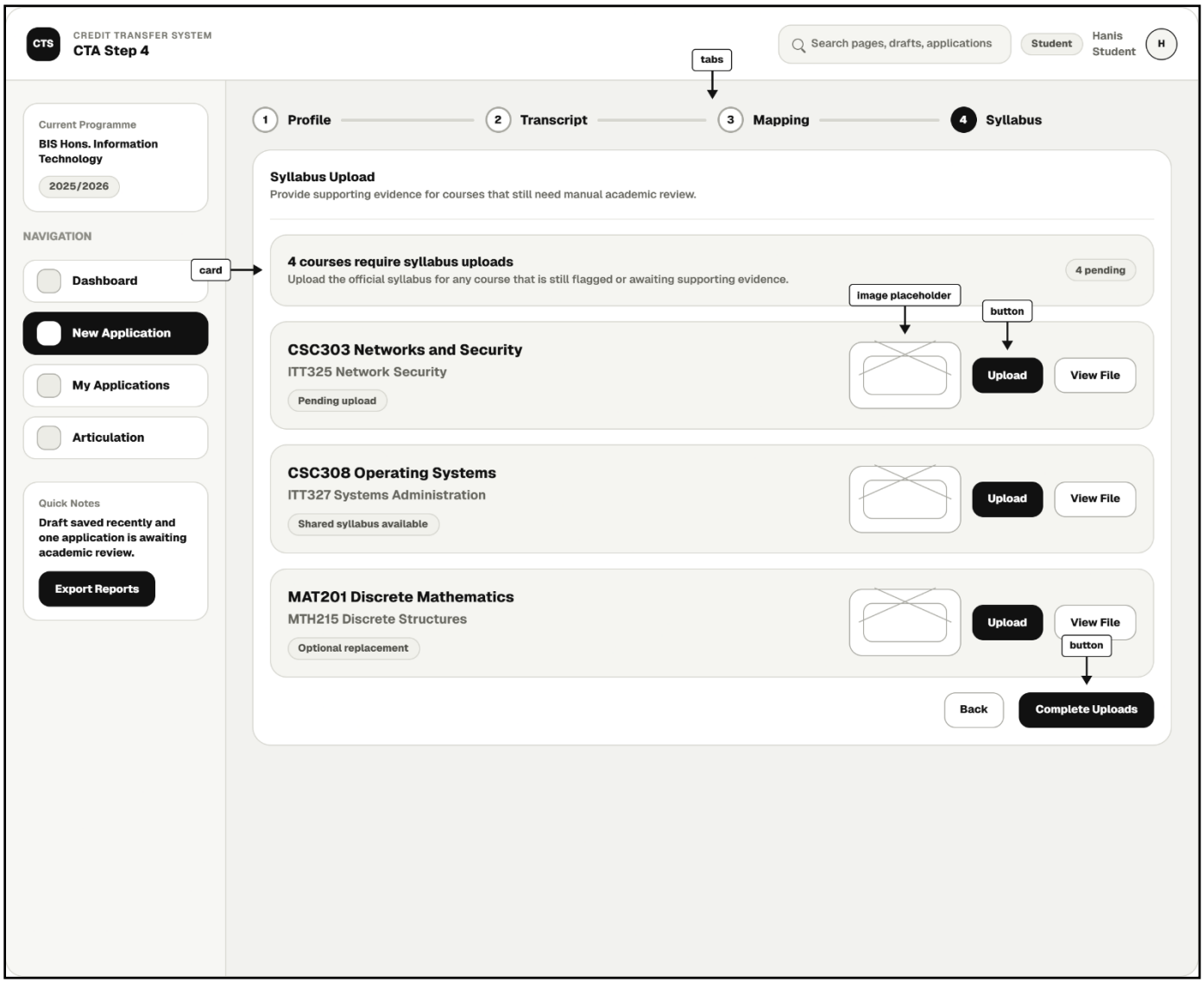


Figure 4.5: Student CTA Step 4 Page

Figure 4.5 shows the Student CTA Step 4 page. This page is used to upload source syllabi for courses that still require supporting evidence. It allows the student to complete the remaining academic requirements for flagged courses before the application can fully proceed into the staff review workflow.

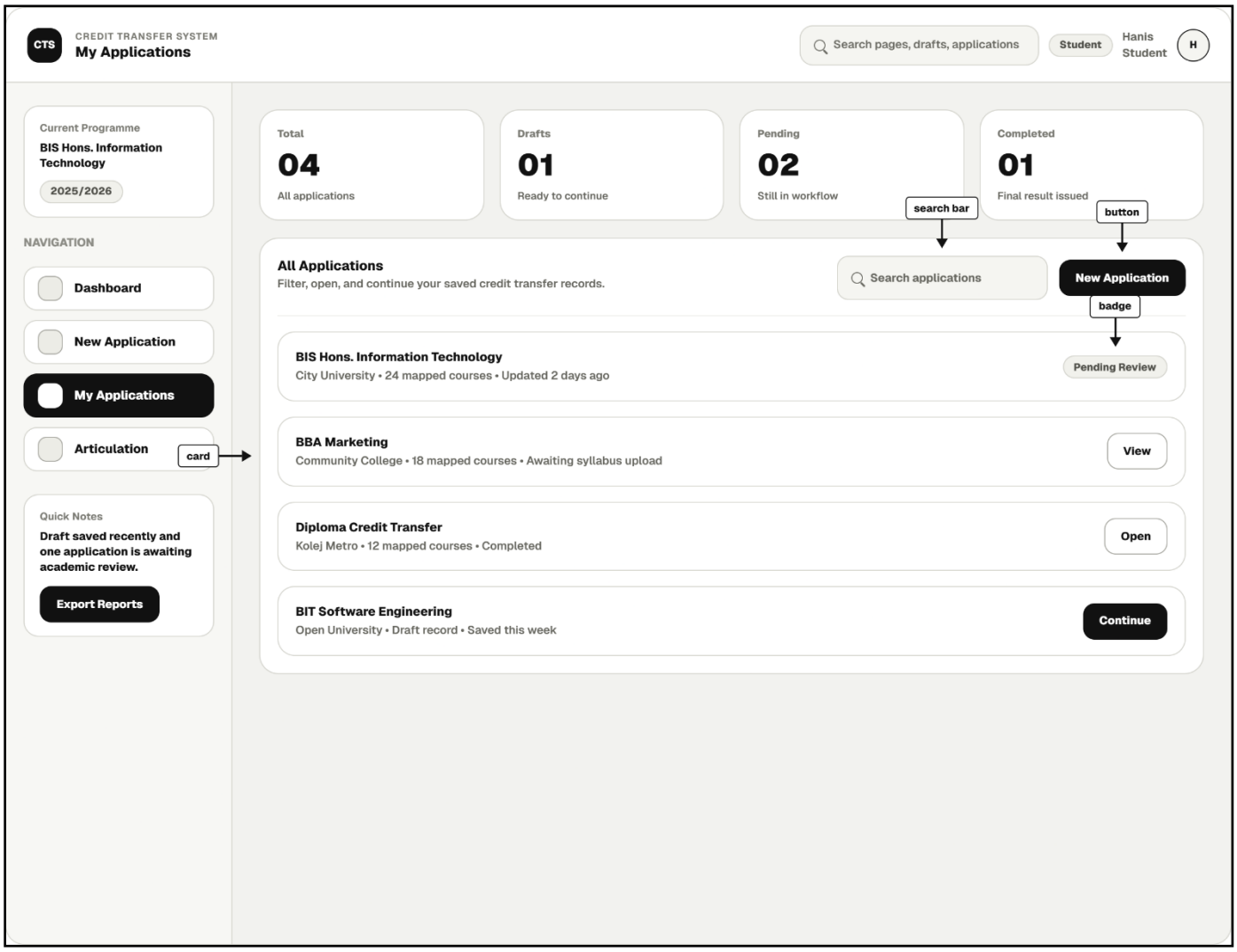


Figure 4.6: Student My Applications Page

Figure 4.6 shows the Student My Applications page. This page lists all saved and submitted credit transfer applications together with their current status and available actions. It helps the student manage existing records, continue unfinished applications, and monitor the progress of submitted applications.

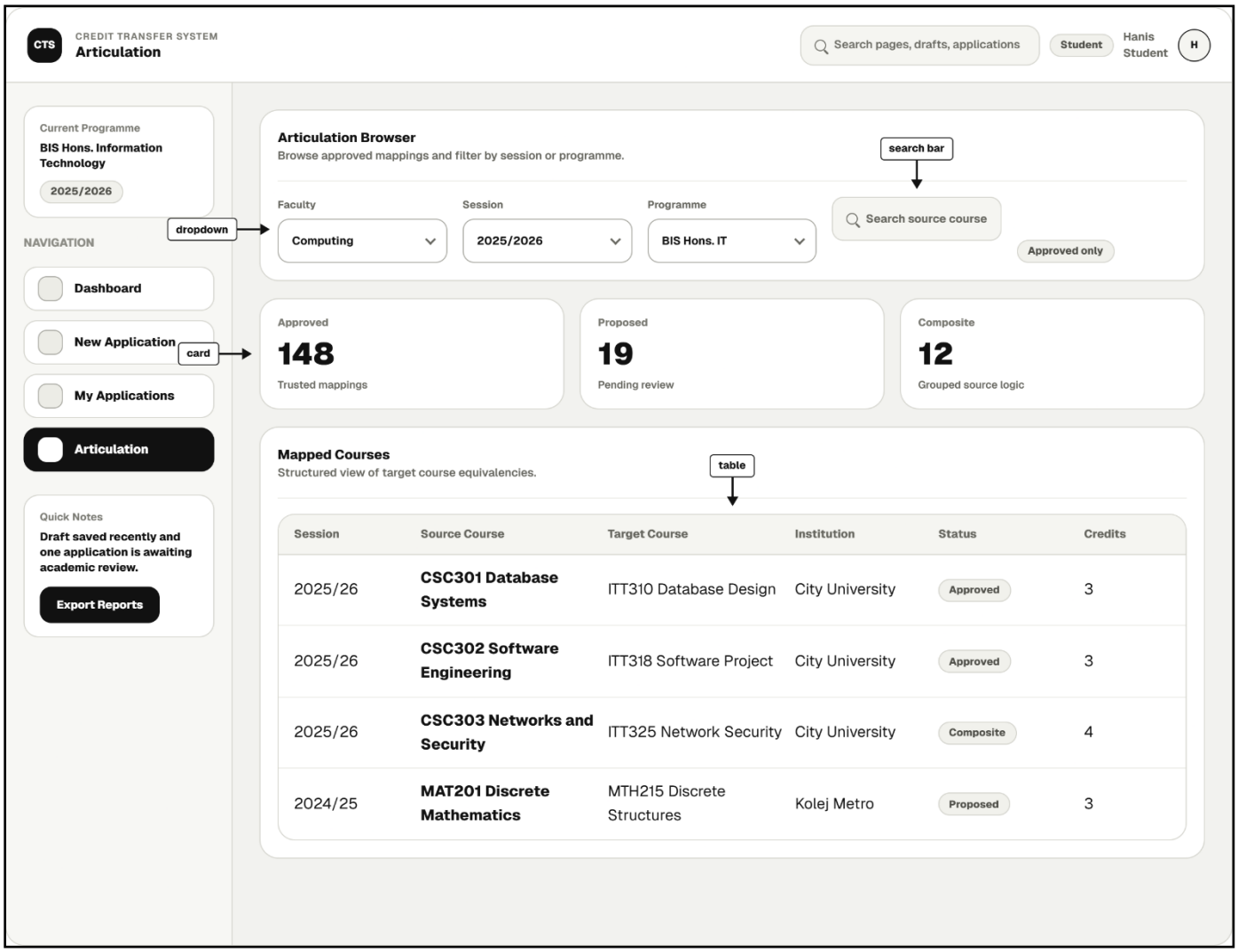


Figure 4.7: Student Articulation Page

Figure 4.7 shows the Student Articulation page. This page allows the student to browse articulation mappings by using filters and search controls. It helps the student understand possible source-to-target course equivalencies before or during the application process.

4.2.3 HOP Interface Design

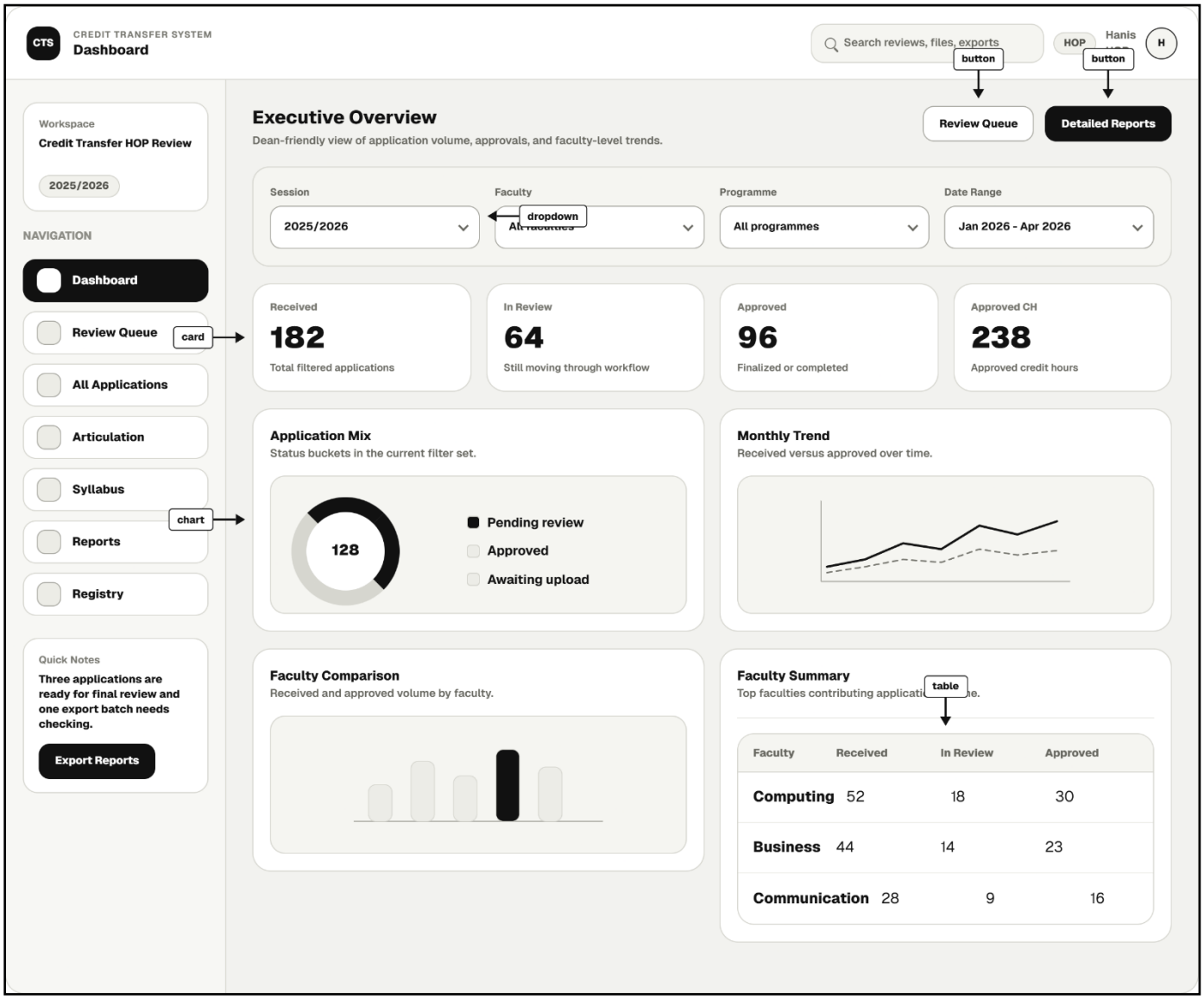


Figure 4.8: HOP Dashboard Page

Figure 4.8 shows the HOP Dashboard page. This page provides the Head of Programme with a summary of current workload, approval trends, and review activity through cards, charts, and summary tables. It helps HOP monitor the overall academic review situation and identify areas that need immediate attention.

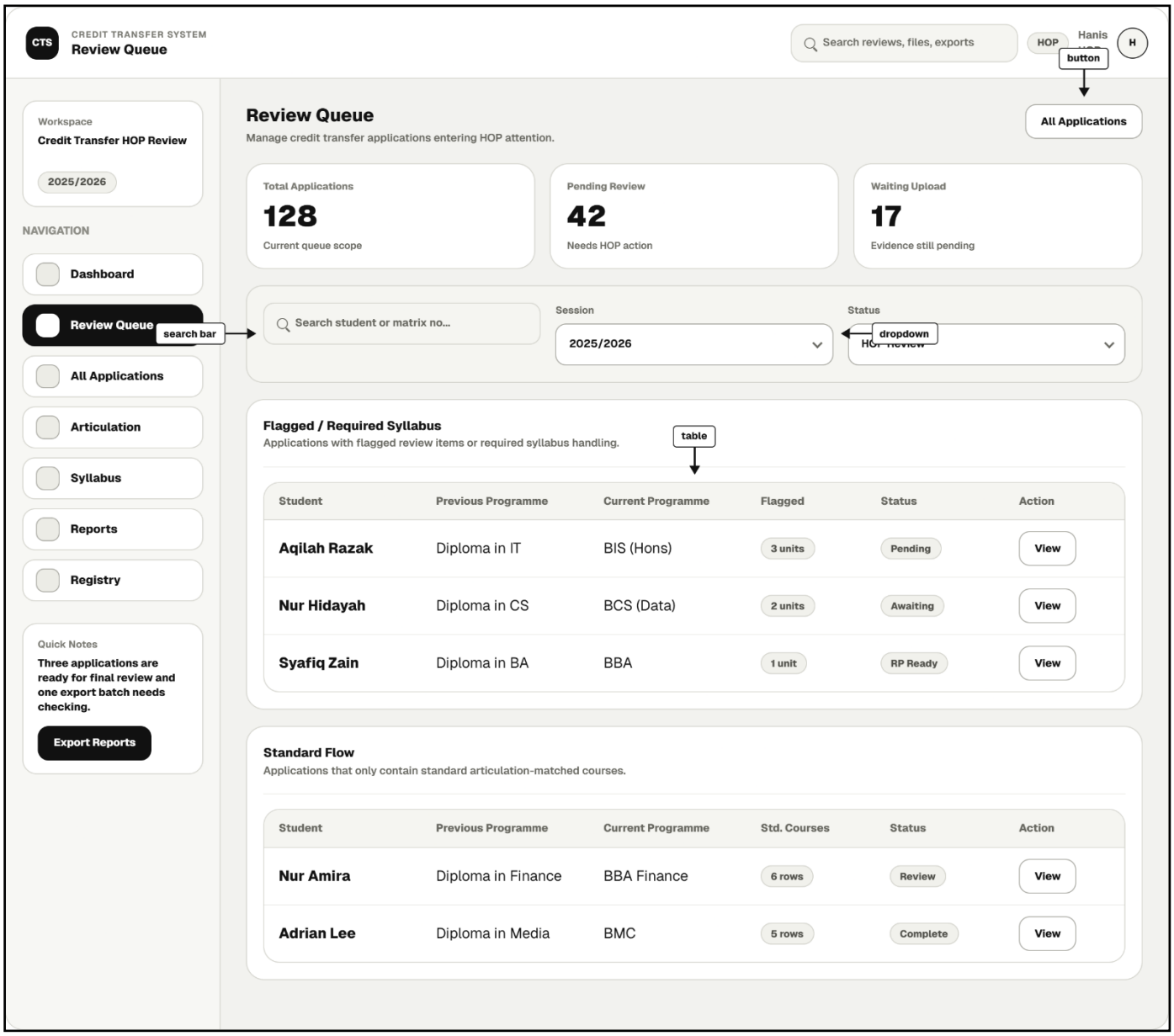


Figure 4.9: HOP Review Queue Page

Figure 4.9 shows the HOP Review Queue page. This page displays applications that are currently waiting for HOP action, including flagged cases and standard review cases. It helps HOP prioritize academic review work and manage the progress of applications that are ready for review.

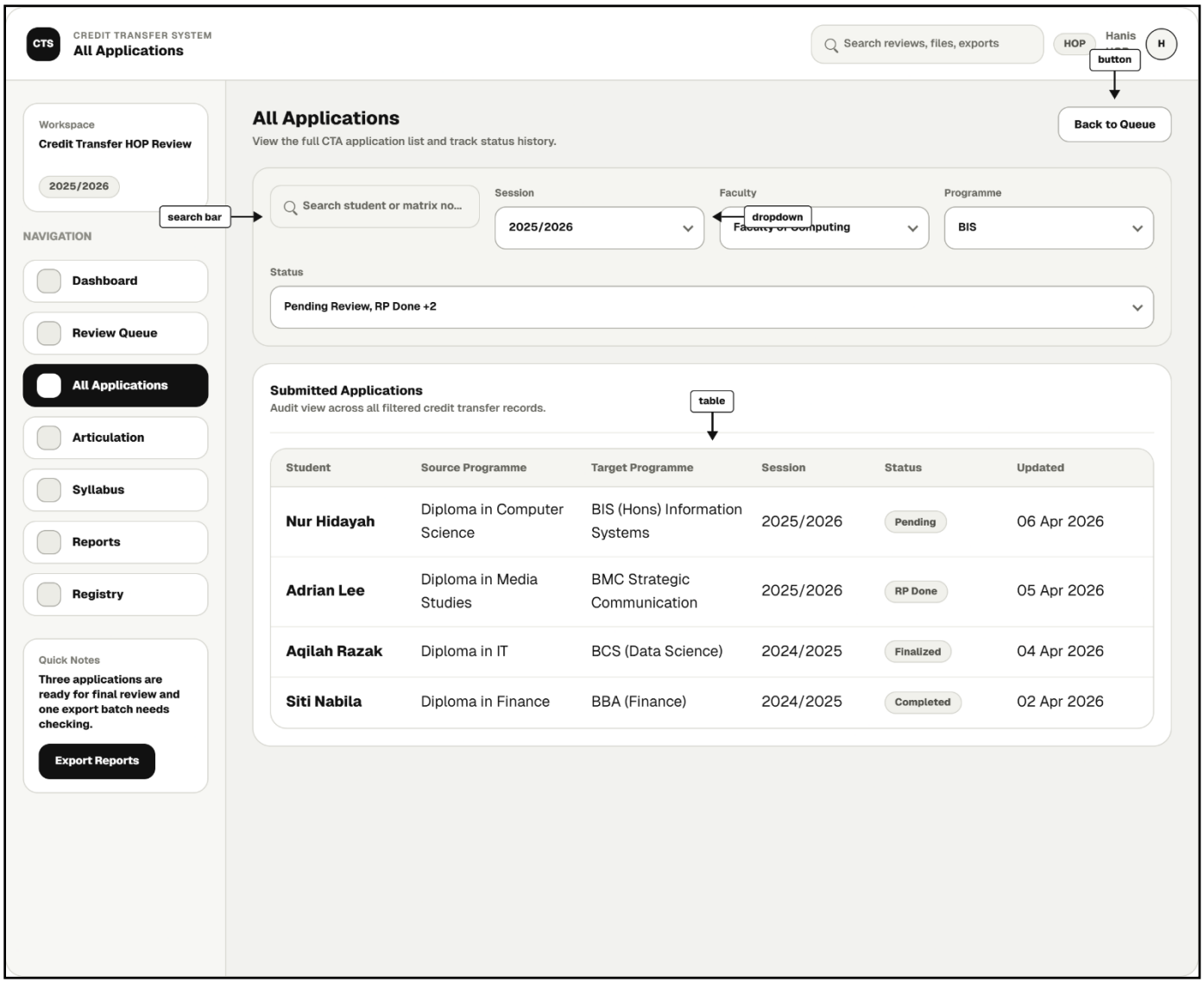


Figure 4.10: HOP All Applications Page

Figure 4.10 shows the HOP All Applications page. This page presents a broader view of submitted applications with filtering and status visibility across the workflow. It helps HOP inspect records outside the immediate queue and monitor application movement at a wider operational level.

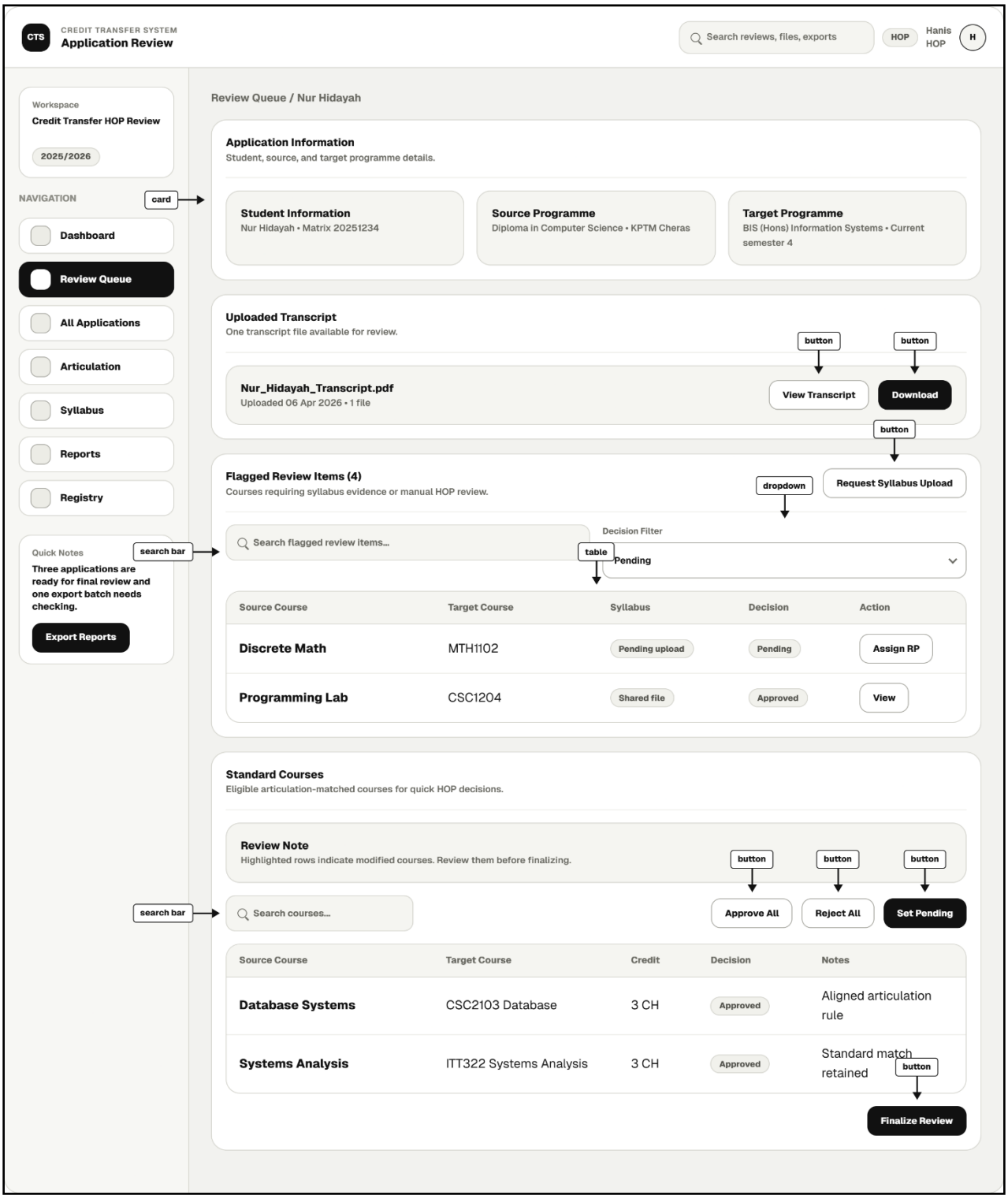


Figure 4.11: HOP Application Review Page

Figure 4.11 shows the HOP Application Review page. This page is the main academic review workspace for HOP, containing application details, transcript references, flagged courses, standard course decisions, and final review actions. It allows HOP to request additional evidence, assign RP where necessary, and prepare the application for final academic decision-making.

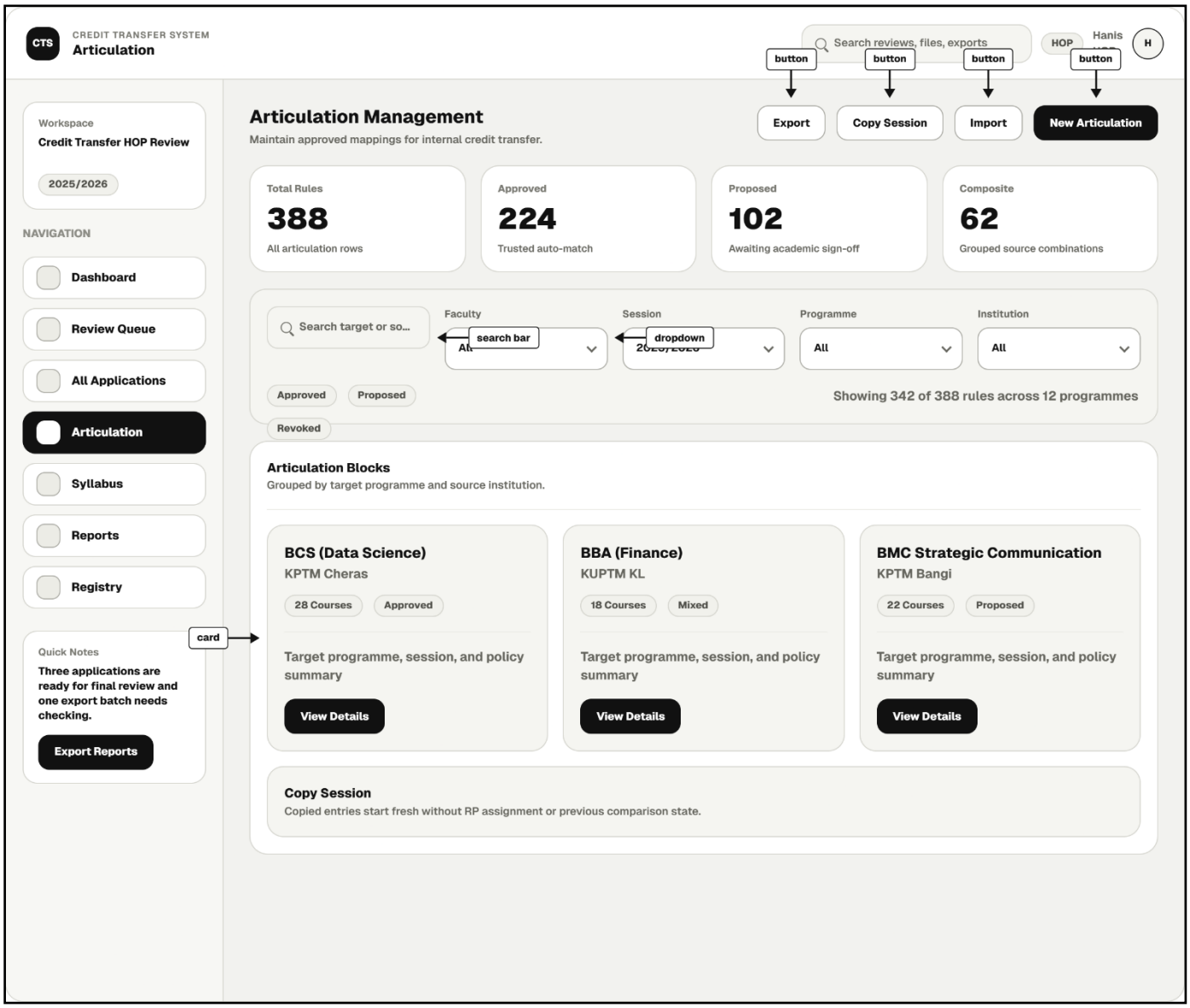


Figure 4.12: HOP Articulation Page

Figure 4.12 shows the HOP Articulation page. This page is used to manage articulation mappings, including creating, copying, importing, and reviewing articulation rules. It supports the maintenance of approved and proposed mappings that influence later student matching and academic recommendation outcome

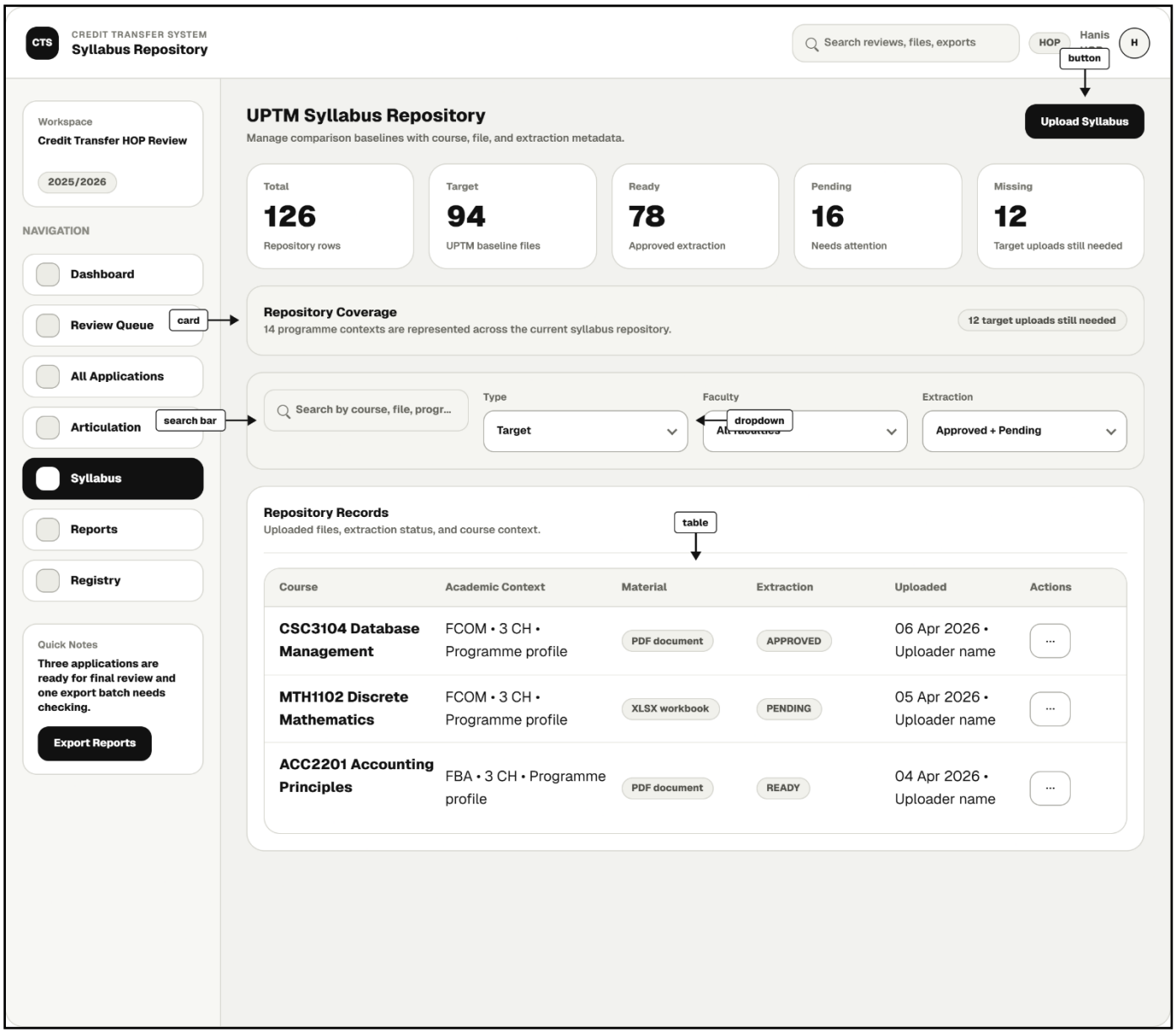


Figure 4.13: HOP Syllabus Repository Page

Figure 4.13 shows the HOP Syllabus Repository page. This page manages the target syllabus repository used in academic comparison workflows. It helps HOP upload target syllabi, monitor extraction status, and ensure that valid comparison material is available for subsequent RP evaluation.

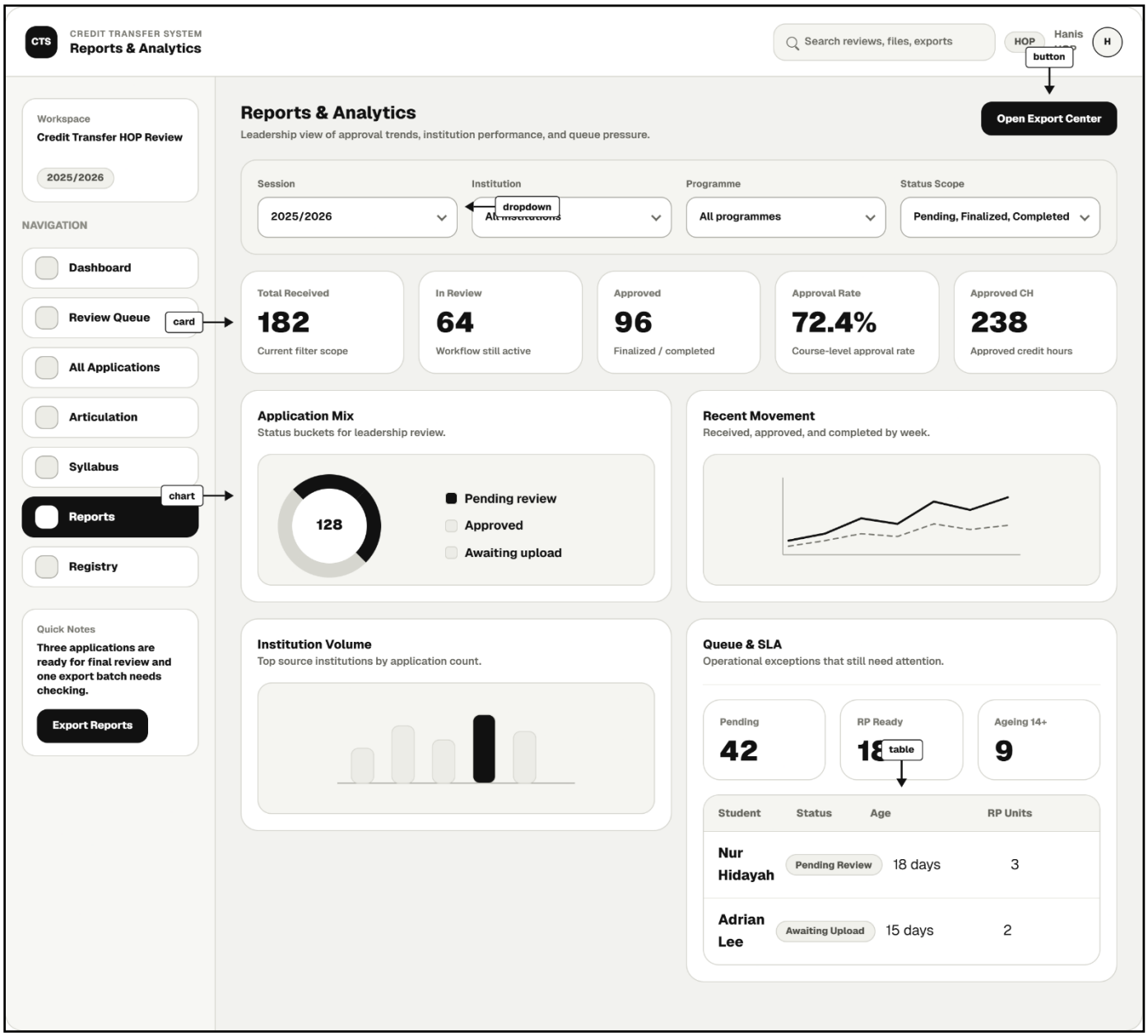


Figure 4.14: HOP Reports and Analytics Page

Figure 4.14 shows the HOP Reports and Analytics page. This page provides charts, metrics, and reporting summaries related to application counts, approval patterns, queue pressure, and institutional activity. It helps HOP evaluate operational performance and observe trends across the credit transfer workflow.

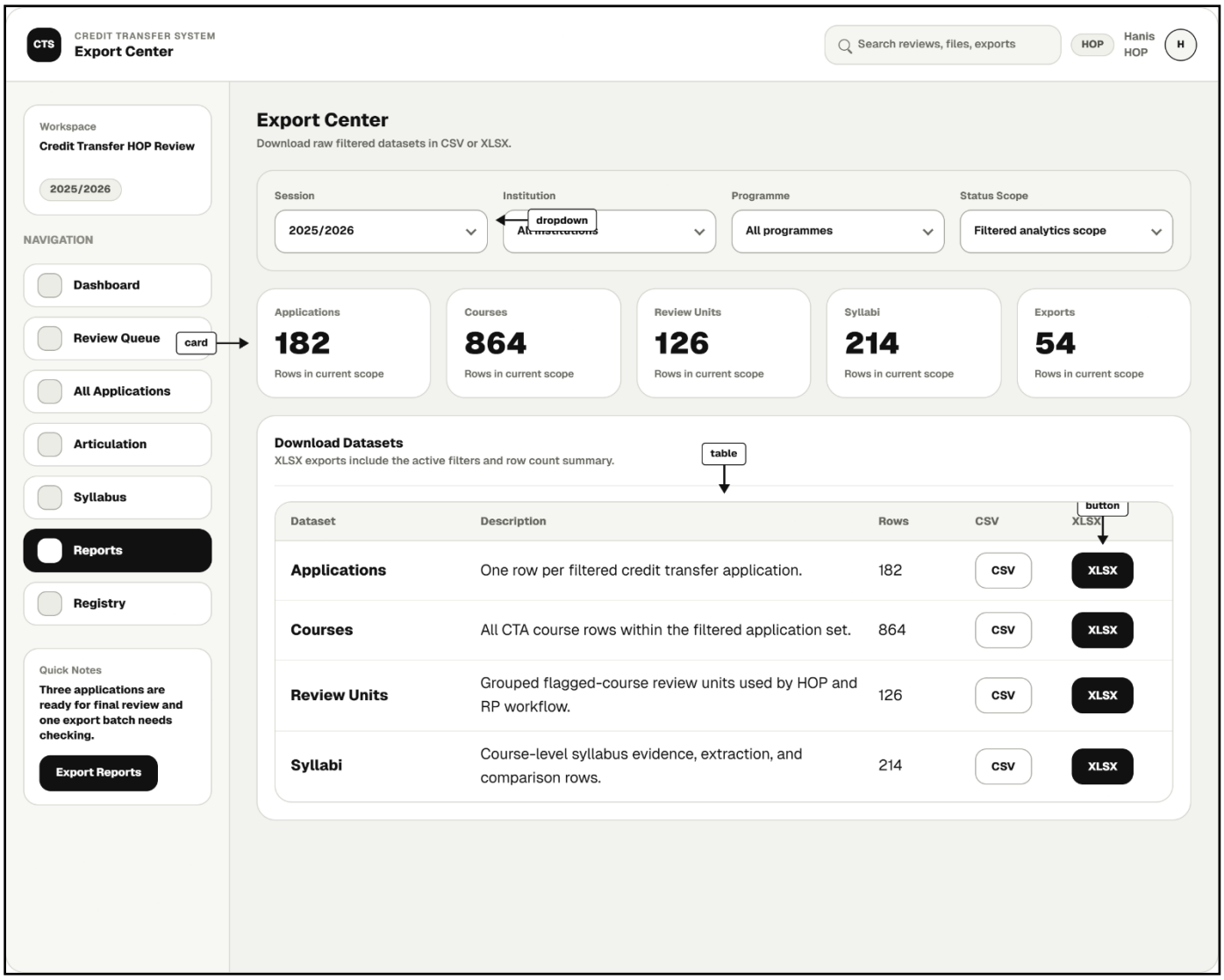


Figure 4.15: HOP Export Center Page

Figure 4.15 shows the HOP Export Center page. This page provides downloadable datasets in CSV and XLSX format based on the selected report scope. It supports administrative reporting, operational review, and downstream data handling.

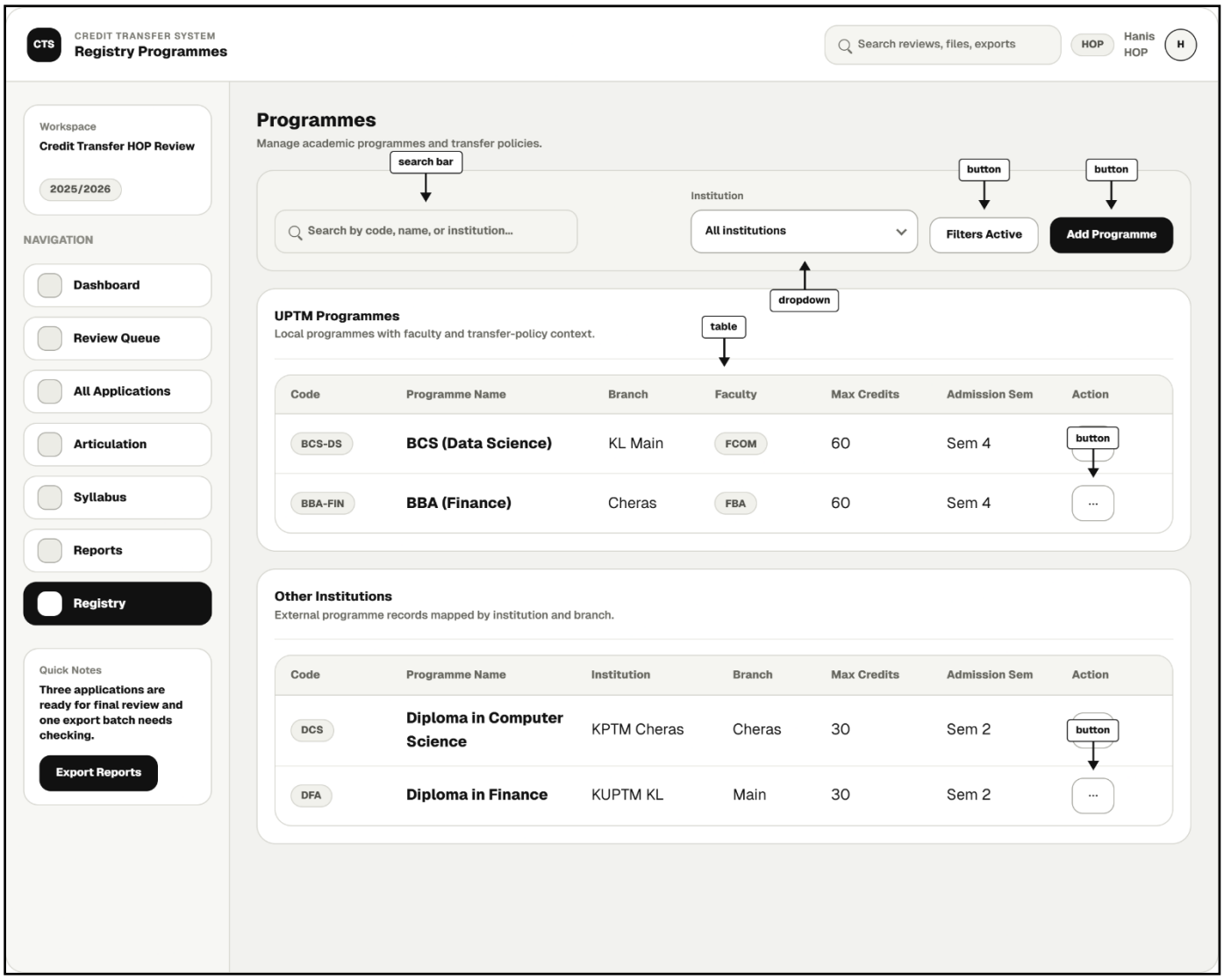


Figure 4.16: HOP Registry Programmes Page

Figure 4.16 shows the HOP Registry Programmes page. This page manages UPTM and external programme records together with institutional and transfer-policy information. It helps HOP maintain the programme master data that supports onboarding options, articulation scoping, and reporting relationships across the system.

4.2.4 RP Interface Design

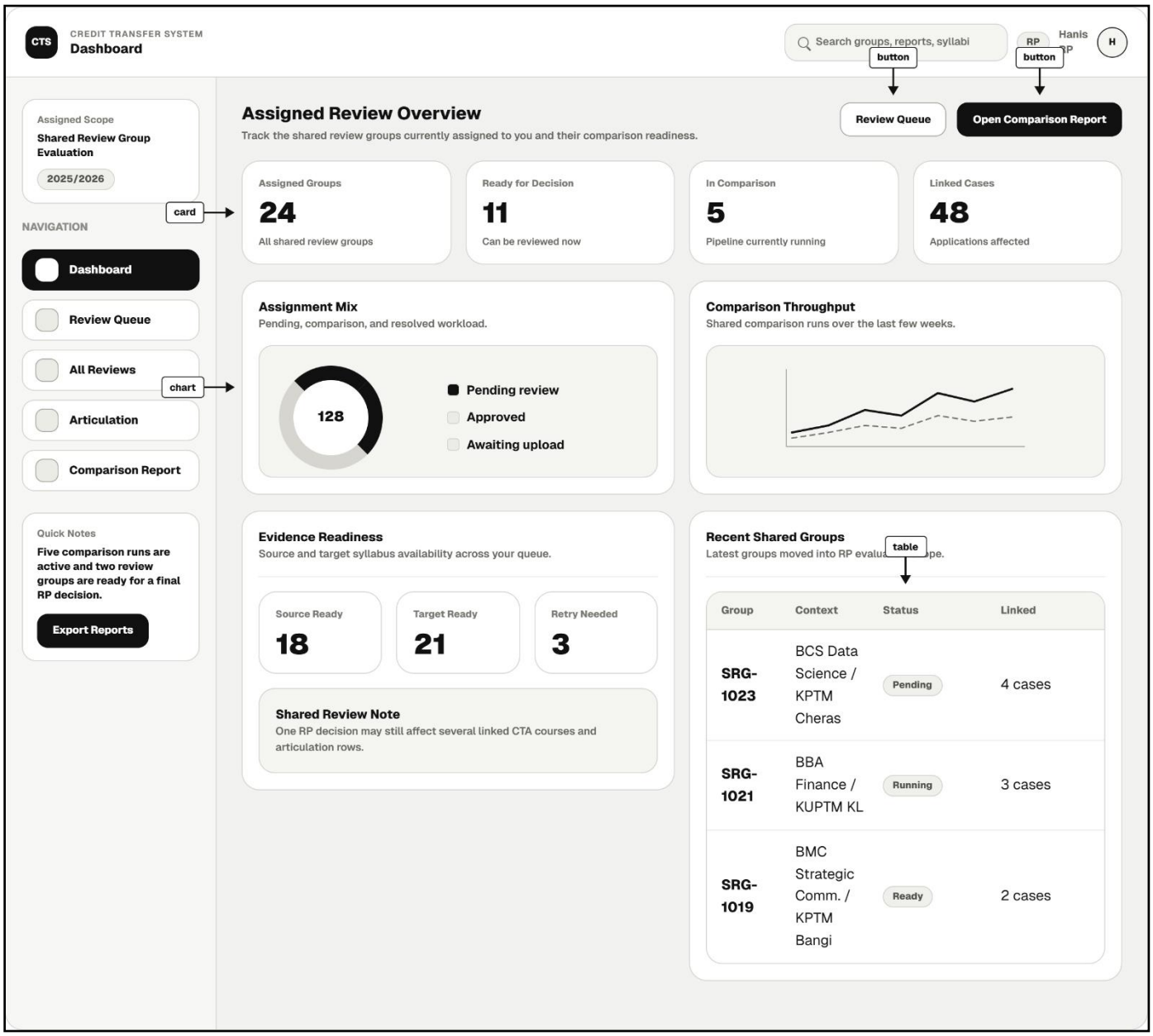


Figure 4.17: RP Dashboard Page

Figure 4.17 shows the RP Dashboard page. This page gives the Resource Person an overview of assigned shared review work, comparison readiness, and recent review activity. It helps RP identify which review groups are ready for evaluation and which groups still depend on evidence preparation or comparison processing.

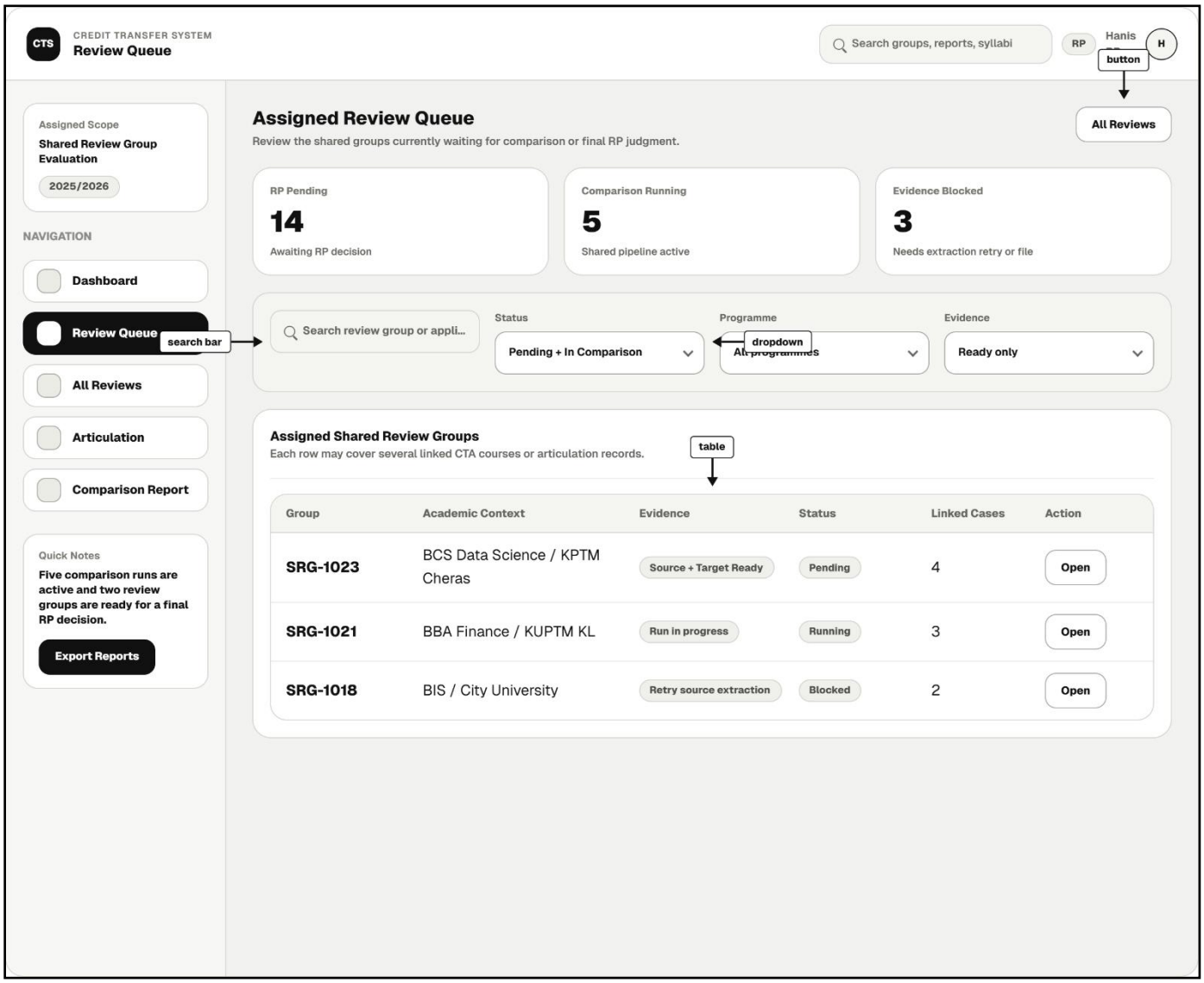


Figure 4.18: RP Review Queue Page

Figure 4.18 shows the RP Review Queue page. This page displays the shared review groups currently assigned to RP across pending and comparison-related states. It helps RP manage the queue of academic review units that require syllabus-based evaluation.

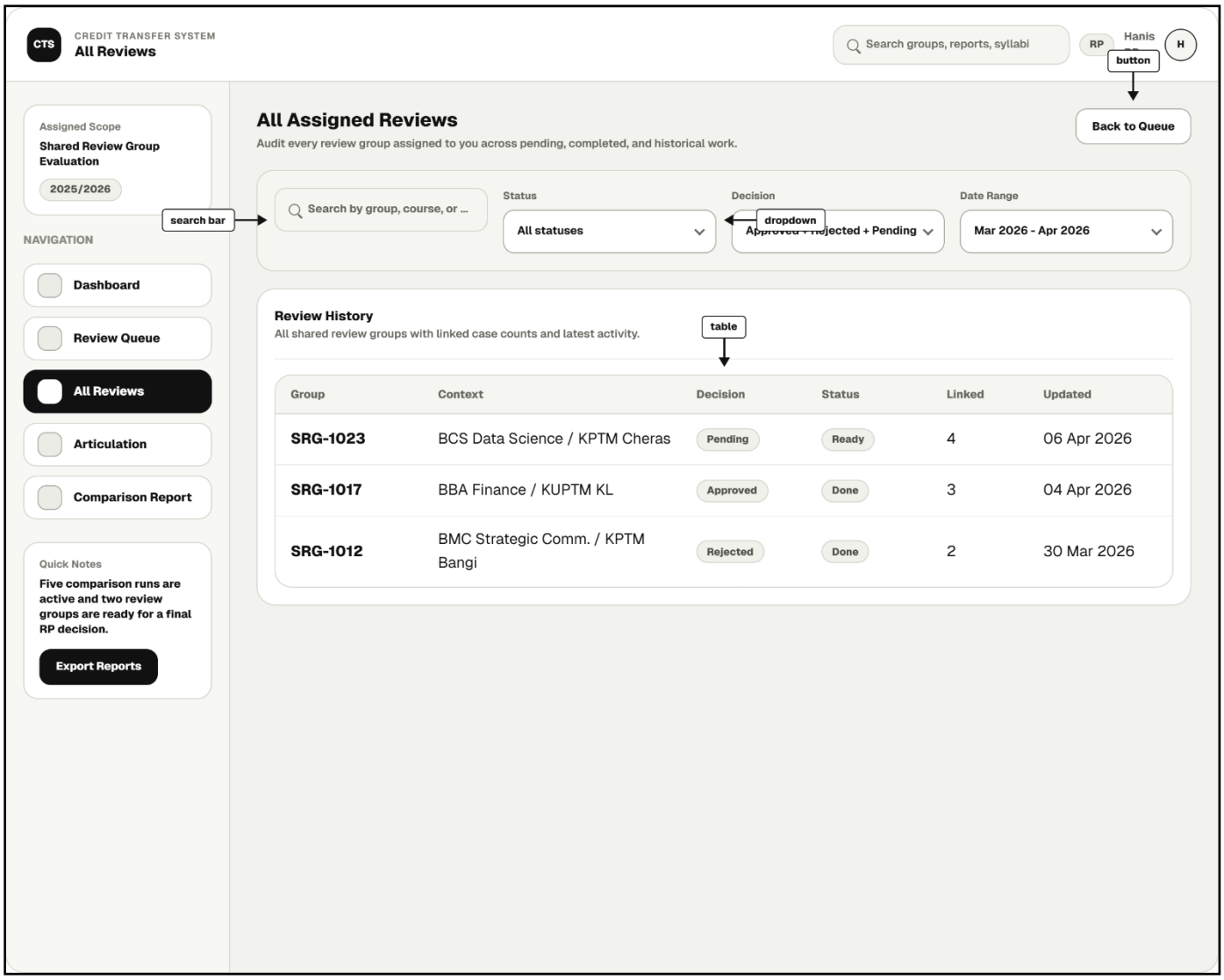


Figure 4.19: RP All Reviews Page

Figure 4.19 shows the RP All Reviews page. This page provides a wider audit view of all review groups assigned to RP, including active and historical records. It helps RP track earlier decisions, review linked case volumes, and revisit completed or in-progress evaluations.

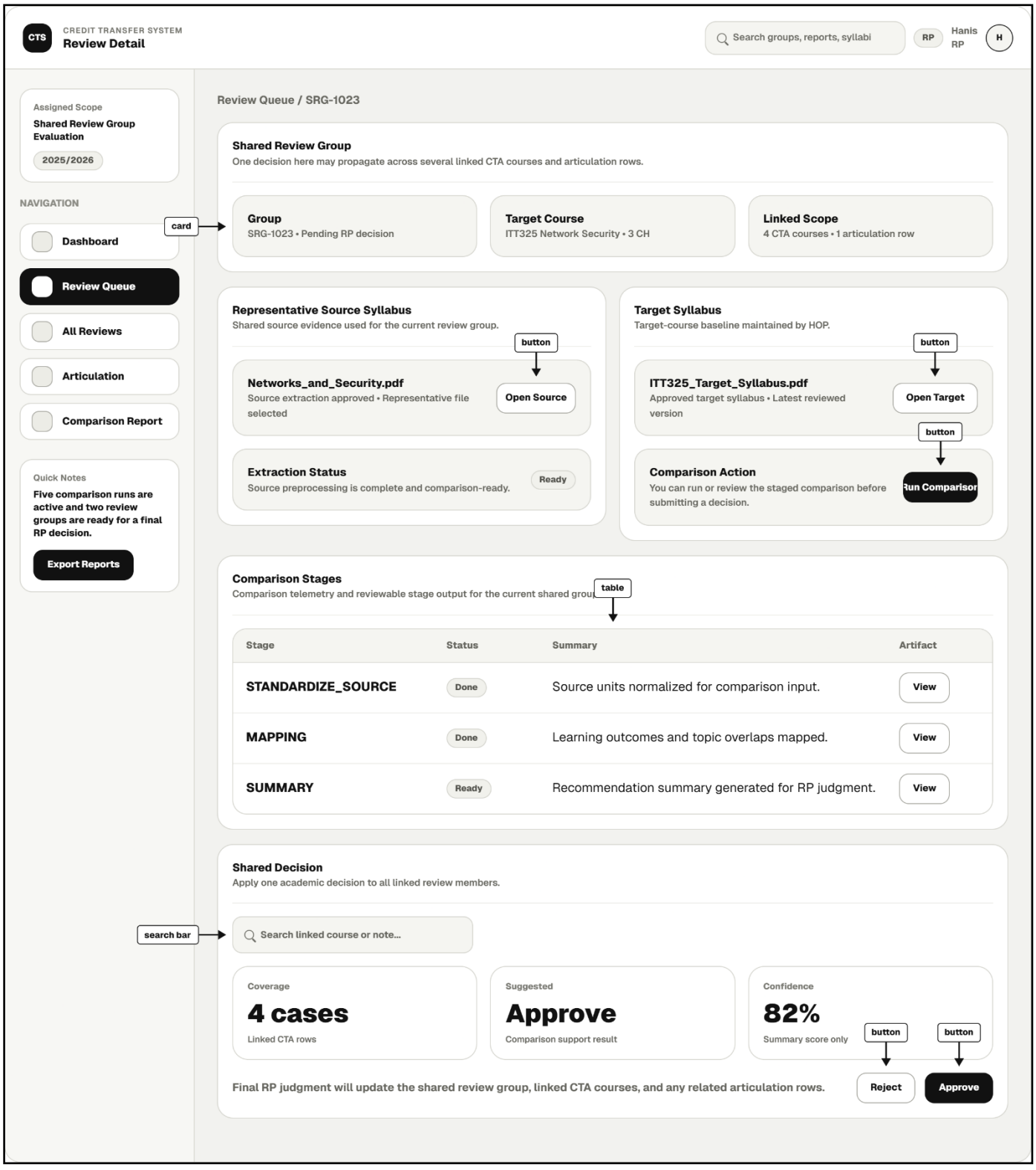


Figure 4.20: RP Review Detail Page

Figure 4.20 shows the RP Review Detail page. This page is the detailed evaluation workspace for a shared review group and includes representative source syllabus evidence, target syllabus evidence, comparison status, and shared decision controls. It helps RP study the academic evidence carefully before recording an approval or rejection decision that may affect multiple linked records.

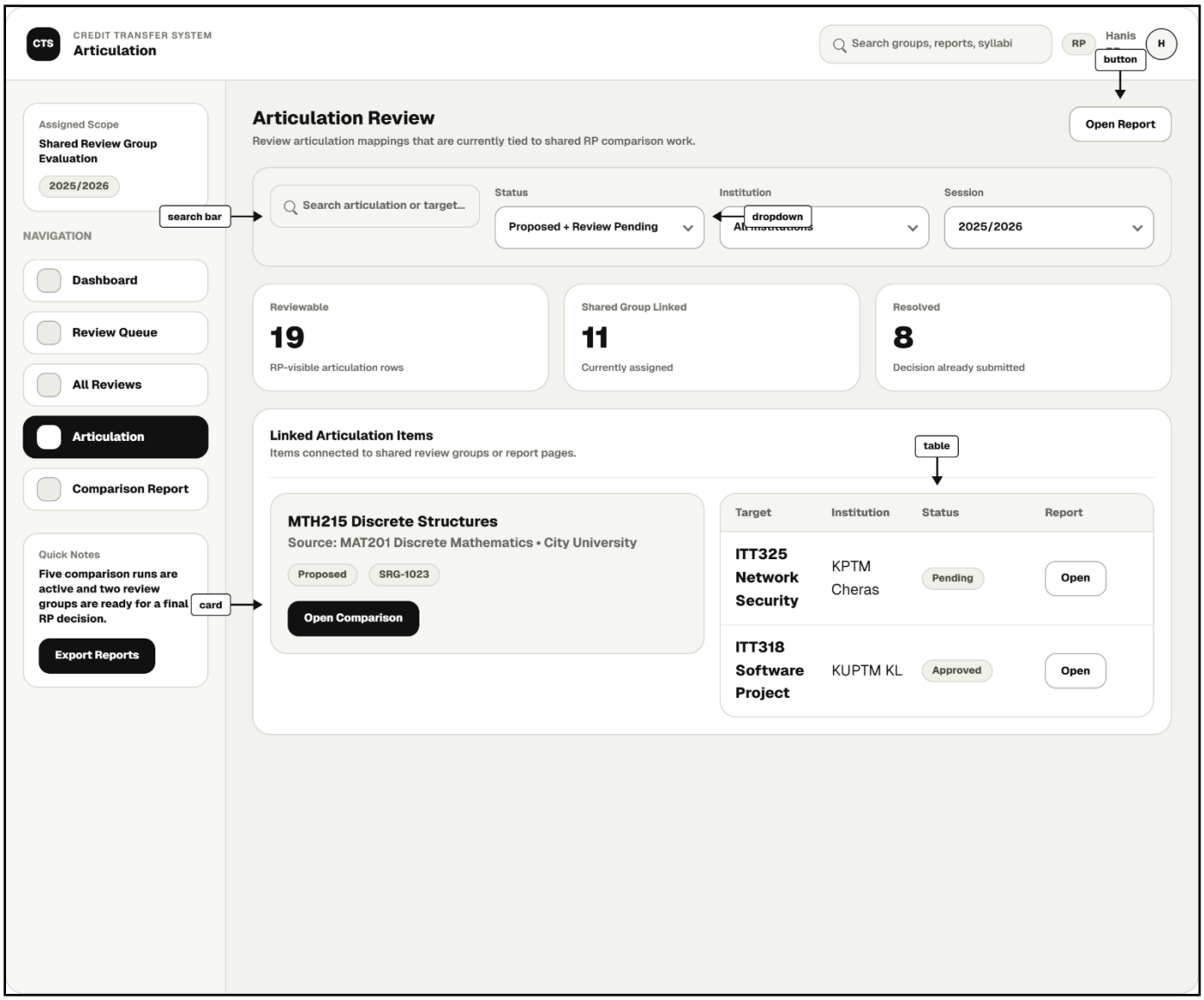


Figure 4.21: RP Articulation Page

Figure 4.21 shows the RP Articulation page. This page presents articulation mappings that are connected to shared comparison work. It helps RP inspect which articulation items are affected by the same evidence set and understand how the evaluation result relates to articulation status.

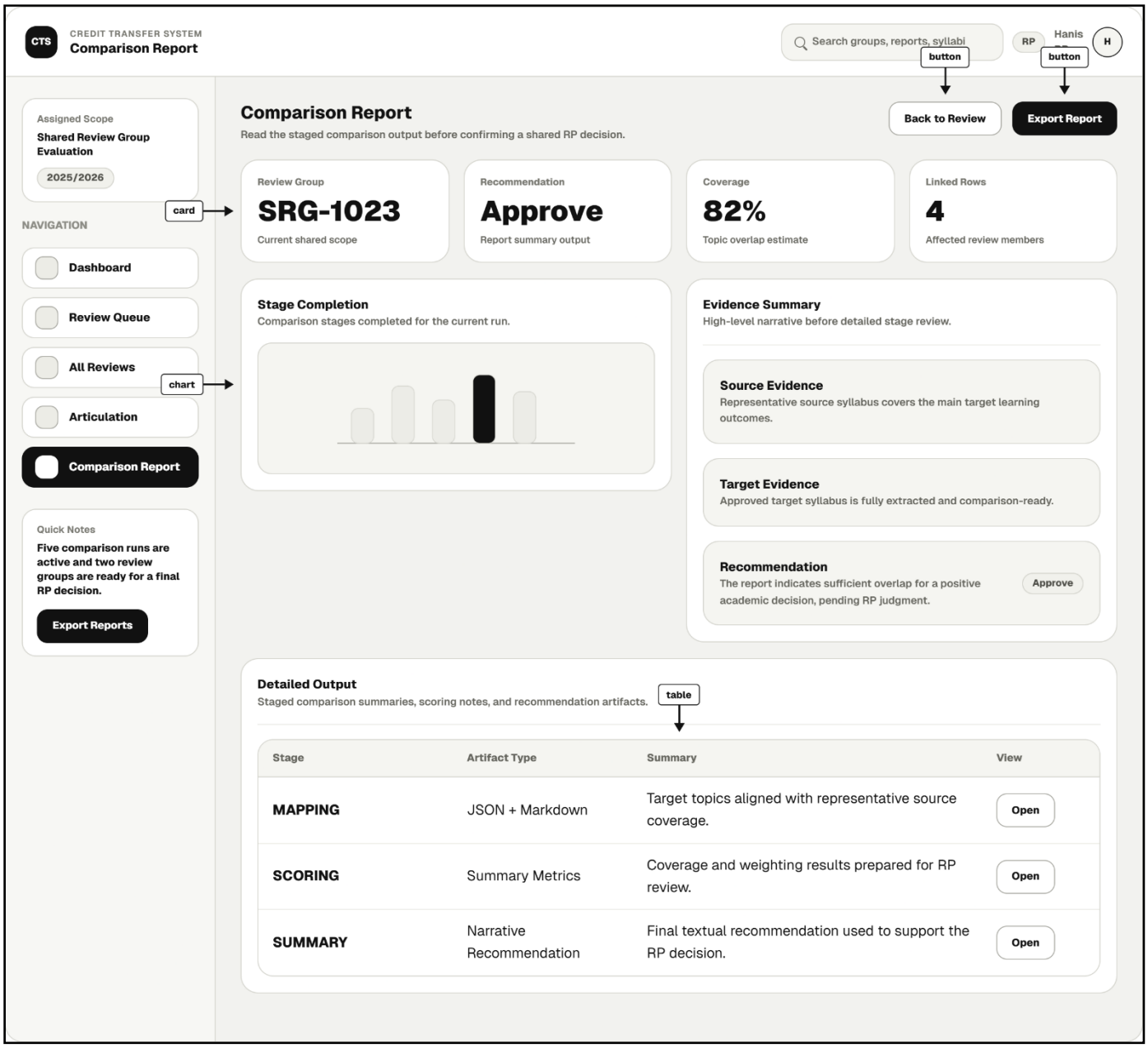


Figure 4.22: RP Comparison Report Page

Figure 4.22 shows the RP Comparison Report page. This page displays staged comparison output, evidence summaries, and recommendation details produced by the comparison pipeline. It helps RP review the system-supported analysis before confirming the final academic evaluation decision.

4.2.5 AAS Interface Design

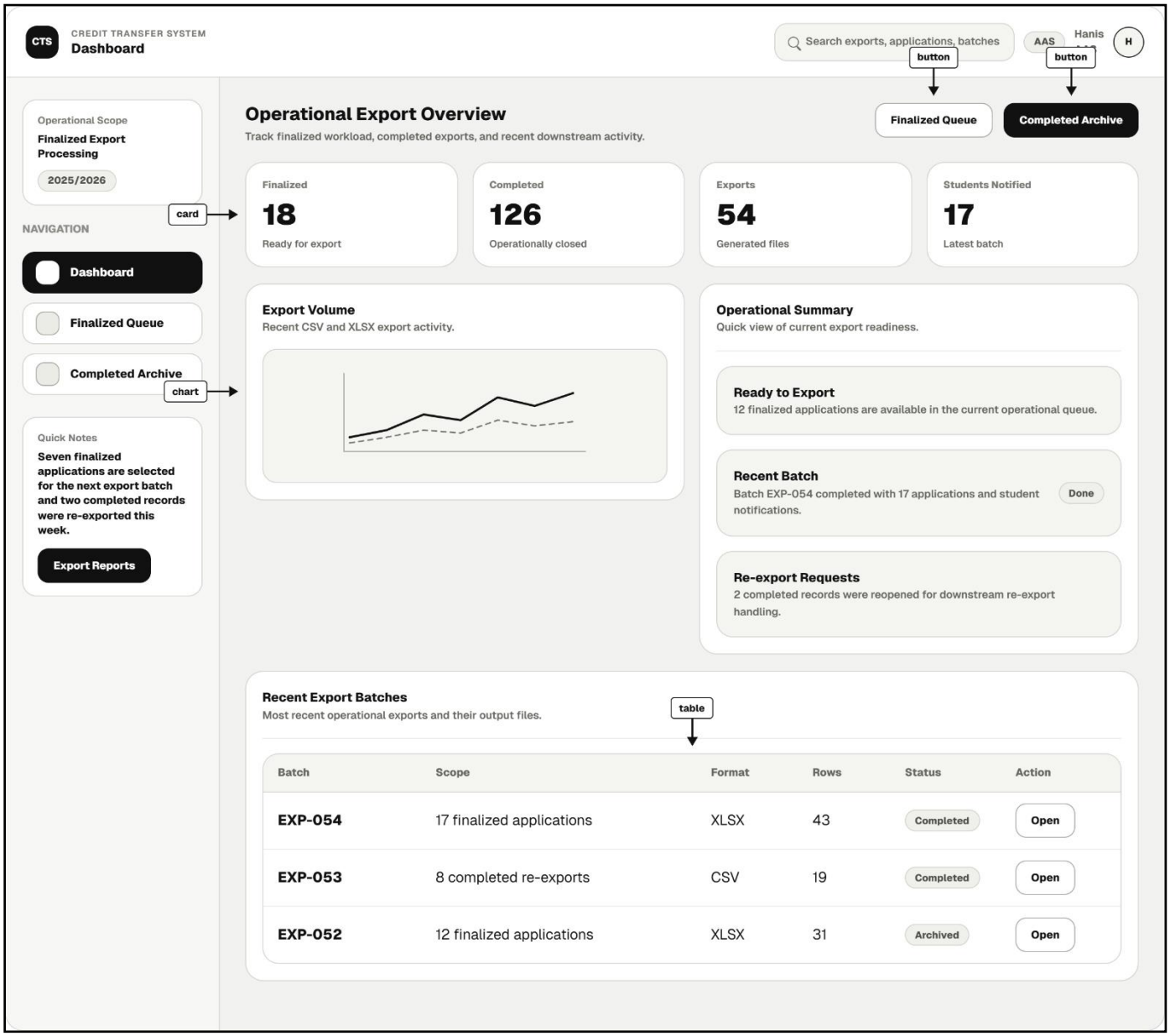


Figure 4.23: AAS Dashboard Page

Figure 4.23 shows the AAS Dashboard page. This page provides an operational overview of finalized applications, completed records, and recent export activity. It helps AAS monitor export workload and identify which applications are ready for downstream processing.

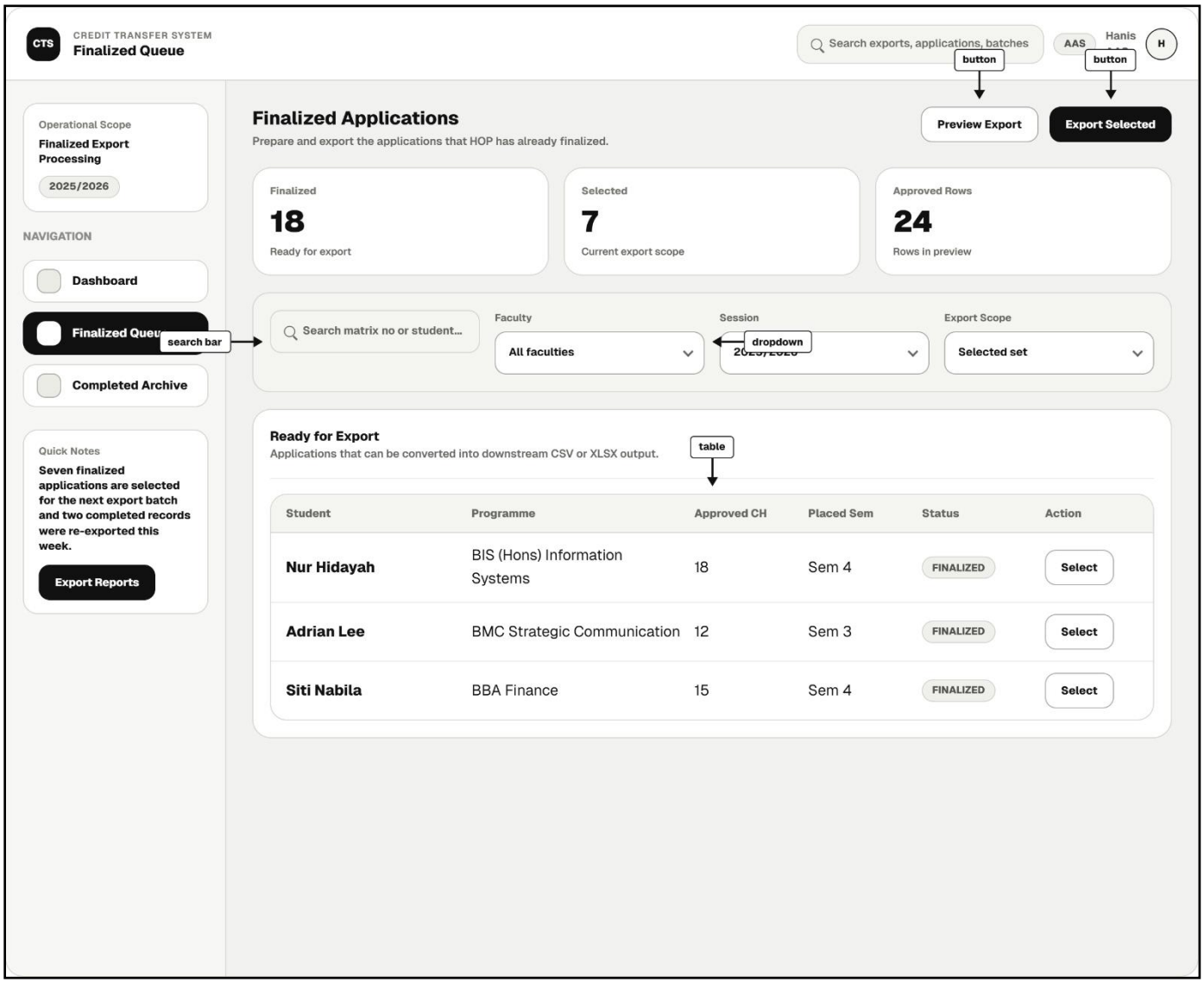


Figure 4.24: AAS Finalized Queue Page

Figure 4.24 shows the AAS Finalized Queue page. This page lists the applications that have already been finalized academically and are ready to be exported by AAS. It helps AAS select the required applications and generate the next export batch in spreadsheet or CSV format.

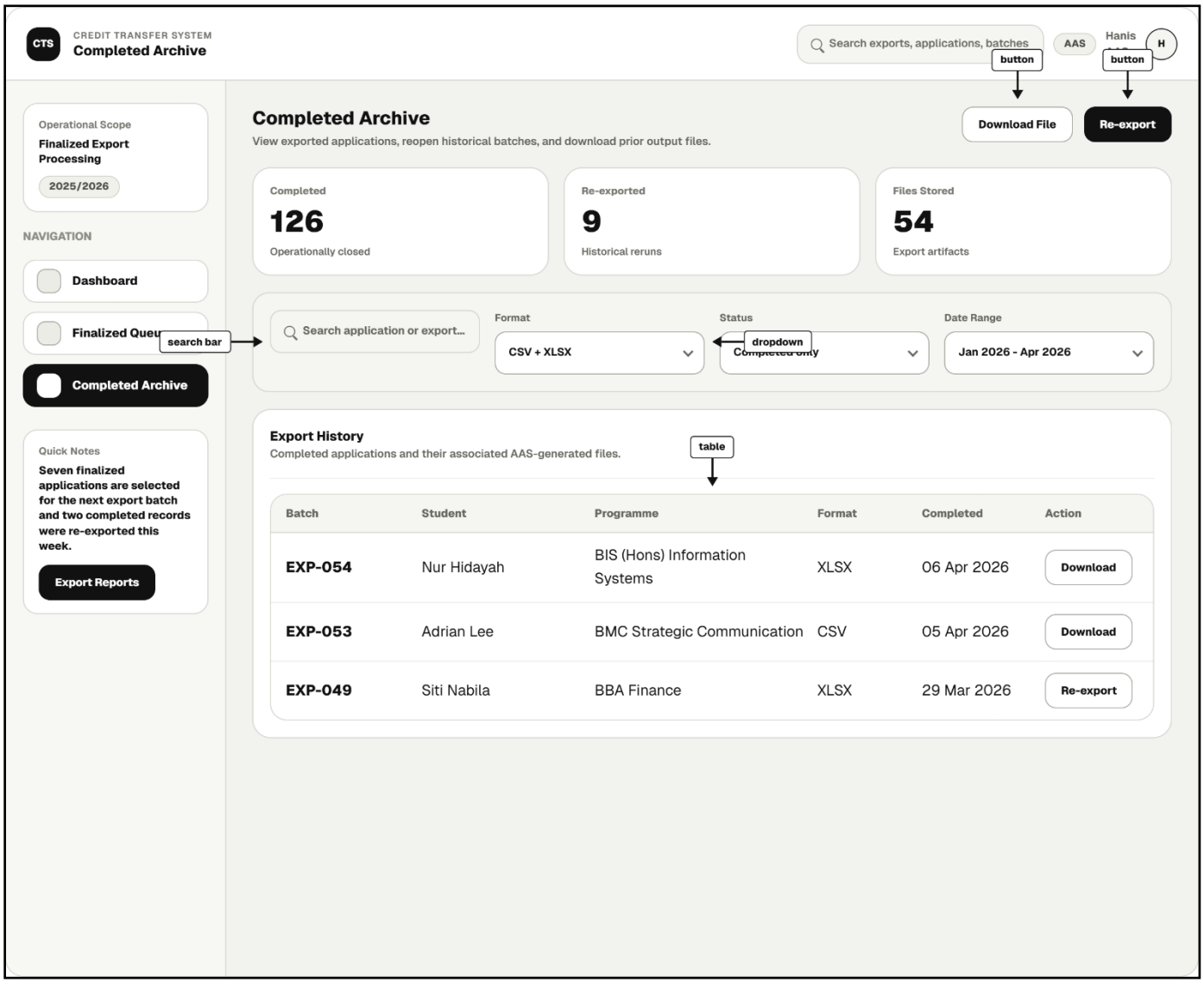


Figure 4.25: AAS Completed Archive Page

Figure 4.25 shows the AAS Completed Archive page. This page stores previously exported applications together with their export history and re-export actions. It helps AAS review completed operational records, download previous files, and perform re-export tasks without changing the academic decision state.

4.3 Database Design

The database design was performed in order to enable the credit transfer process from start to finish. Beyond storing information regarding the users that submitted the applications for credit transfer, the database will store information regarding academic master data, articulation data, uploaded files, the shared review of representatives of the schools, and the exported information from the database. This database is necessary in order to track the information about each representative teacher, each shared review group, and each decision of the representatives regarding which courses can be transferred between community colleges.

The database will be implemented within PostgreSQL using the Prisma ORM. Based upon the workflow diagram for the system and the database schema, the main databases for the system may include databases for authentication and users, academic master data, student CTA (credit transfer application) data, files and syllabi for those students, shared reviews of those syllabi, and telemetry data regarding the comparisons between community colleges syllabi and the exported data from the database.

4.3.1 Data Dictionary

The following subsection will explain the tables that are implemented within the database. Only the physical stored tables within the database are explained within this dictionary; relations that exist within Prisma but that do not create a database table are not explained as attributes of the objects within the data dictionary. Each of the tables will be grouped together in a way that indicates the relationship between those tables to provide an overall understanding of the database.

Table 4.1 demonstrates the structure of the User table within the database. This table stores the records of the users that are authenticated into the database. Each user can be identified by their name, email address, and role within the system. Additionally, academic masters (such as the HOP, the RP, and the AAS) can also have additional information stored regarding their academic scope. This table is utilized by the application to associate applications, files, reviews, and export information with the individuals within the community college that submitted and reviewed the applications for credit transfer.

Table 4.1: Data Dictionary for User

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each authenticated user.
name	String	Not Null	Stores the full name of the user.
email	String	Unique, Not Null	Stores the user's email address used for login.
emailVerified	Boolean	Default false	Indicates whether the user's email has been verified.

Column	Data Type	Constraints	Description
onboarded	Boolean	Default false	Indicates whether the user has completed onboarding.
image	String	Nullable	Stores the profile image URL if available.
role	Enum(Role)	Default STUDENT, Not Null	Stores the role assigned to the user in the system.
programmeId	String	Foreign Key -> Programme.id, Nullable	Links a staff user to an academic programme scope when needed.
academicSessionId	String	Foreign Key -> AcademicSession.id, Nullable	Links a staff user to an academic session scope when needed.
createdAt	DateTime	Default now(), Not Null	Stores the date and time when the record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the most recent update time.

Table 4.2 shows the structure of the Session table, which stores active authentication sessions created by the Better Auth layer. This table links each session token to a specific user and keeps expiry, client, and update information needed to maintain secure login state. When a user accesses protected pages in the system, the session table is used to validate whether the login is still valid.

Table 4.2: Data Dictionary for Session

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each authenticated session.
expiresAt	DateTime	Not Null	Stores the session expiry timestamp.
token	String	Unique, Not Null	Stores the session token used by the authentication layer.
createdAt	DateTime	Default now(), Not Null	Stores when the session record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest session update time.
ipAddress	String	Nullable	Stores the client's IP address if captured.
userAgent	String	Nullable	Stores the browser or device user agent string.
userId	String	Foreign Key -> User.id, Not Null	Links the session to the authenticated user.

Table 4.3 shows the structure of the Account table, which stores external authentication account links for each user. This table keeps provider identifiers and token-related information so the system can support Google sign-in and other provider-managed login flows. In practice, it extends the user identity model by connecting internal users to external authentication services.

Table 4.3: Data Dictionary for Account

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for the linked account record.
accountId	String	Not Null	Stores the provider-side account identifier.
providerId	String	Not Null	Stores the external authentication provider name.
userId	String	Foreign Key -> User.id, Not Null	Links the account record to the user.
accessToken	String	Nullable	Stores the provider access token when available.
refreshToken	String	Nullable	Stores the provider refresh token when available.
idToken	String	Nullable	Stores the provider identity token when available.
accessTokenExpiresAt	DateTime	Nullable	Stores the access token expiry timestamp.
refreshTokenExpiresAt	DateTime	Nullable	Stores the refresh token expiry timestamp when known.
scope	String	Nullable	Stores the granted OAuth scope string.
password	String	Nullable	Stores password data if a provider requires it.
createdAt	DateTime	Default now(), Not Null	Stores when the account link was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the most recent account update time.

Table 4.4 shows the structure of the Verification table, which stores verification records used by the authentication process. This table contains identifiers, verification values, and expiry timestamps that support secure confirmation flows such as email verification or one-time token validation. It is a supporting table, but it remains important because it helps protect account access and verification-related actions.

Table 4.4: Data Dictionary for Verification

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for the verification record.
identifier	String	Indexed, Not Null	Stores the identifier being verified, such as email.
value	String	Not Null	Stores the verification token or value.
expiresAt	DateTime	Not Null	Stores the expiry timestamp of the verification record.
createdAt	DateTime	Default now(), Not Null	Stores when the verification record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.5 shows the structure of the Institution table, which stores institution master records used throughout the credit transfer system. This table supports source institution selection during student application, programme registry management, articulation scoping, and reporting. Because the system compares academic records across institutions, this table provides one of the main reference points for institutional identity.

Table 4.5: Data Dictionary for Institution

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each institution.
code	String	Nullable	Stores the institution code when available.
name	String	Not Null	Stores the institution name.
slug	String	Unique, Not Null	Stores a URL-safe version of the institution name.
type	String	Nullable	Stores the institution category if used.
isActive	Boolean	Default true, Not Null	Indicates whether the institution is active in the system.

Column	Data Type	Constraints	Description
createdAt	DateTime	Default now(), Not Null	Stores when the institution record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.6 shows the structure of the Branch table, which stores branch or campus records under an institution. This table allows the system to distinguish between institution-wide rules and branch-specific academic contexts, especially when articulation or shared syllabus decisions differ by campus. It therefore improves the precision of source institution modeling.

Table 4.6: Data Dictionary for Branch

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each branch.
institutionId	String	Foreign Key -> Institution.id, Not Null	Links the branch to its institution.
name	String	Not Null	Stores the branch name.
slug	String	Unique with institutionId, Not Null	Stores a URL-safe branch label under one institution.
createdAt	DateTime	Default now(), Not Null	Stores when the branch record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.7 shows the structure of the Programme table, which stores academic programme master data used by the system. This table is important because target programme selection, transfer-credit limits, admission semester, faculty assignment, and registry views all depend on valid programme records. In the workflow, it supports both operational data entry and academic decision constraints.

Table 4.7: Data Dictionary for Programme

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each programme.
code	String	Not Null	Stores the programme code.
name	String	Not Null	Stores the programme name.
institutionId	String	Foreign Key -> Institution.id, Not Null	Links the programme to an institution.
branchId	String	Foreign Key -> Branch.id, Nullable	Links the programme to a branch when applicable.

Column	Data Type	Constraints	Description
facultyCode	String	Default FCOM, Not Null	Stores the faculty code for UPTM context.
facultyName	String	Default Faculty of Computing & Multimedia (FCOM), Not Null	Stores the faculty name.
maxTransferCredits	Int	Nullable	Stores the maximum credits allowed for transfer into the programme.
admissionSemester	Int	Nullable	Stores the semester placement baseline for admission.
createdAt	DateTime	Default now(), Not Null	Stores when the programme record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.8 shows the structure of the AcademicSession table, which stores the academic session values used across the system. This table standardizes session labels so that applications, articulation records, staff scope, and reporting can all refer to the same academic period consistently. It acts as a lightweight but important reference table in the overall schema.

Table 4.8: Data Dictionary for AcademicSession

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each academic session.
label	String	Unique, Not Null	Stores the academic session label.
createdAt	DateTime	Default now(), Not Null	Stores when the academic session record was created.

Table 4.9 shows the structure of the CreditTransferApplication table, which is the main transaction table for the student credit transfer process. This table stores the student's source-study background, target programme context, application workflow state, and final review outcome fields. Because the entire process begins from a draft application and ends with finalization and export, this table functions as the central container for one student submission.

Table 4.9: Data Dictionary for CreditTransferApplication

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each credit transfer application.
studentId	String	Foreign Key -> User.id, Not Null	Links the application to the student who created it.
studentMatrixNo	String	Not Null	Stores the student's matrix number.
myKadPassportNo	String	Nullable	Stores the student's MyKad or passport number.
currentSemester	Int	Not Null	Stores the student's current semester.
contactNo	String	Not Null	Stores the student's contact number.
sourceGraduationYear	Int	Nullable	Stores the graduation year from the source institution.
sourceCgpa	Decimal(3,2)	Nullable	Stores the student's source CGPA.
sourceProgrammeCode	String	Nullable	Stores the source programme code.
sourceProgrammeName	String	Nullable	Stores the source programme name.
creditTransferType	Enum(CreditTransferType)	Nullable	Stores whether the transfer is vertical or horizontal.
sessionCode	String	Indexed, Not Null	Stores the academic session used for the application.
targetProgrammeCode	String	Indexed, Not Null	Stores the selected UPTM target programme code.
targetProgrammeName	String	Not Null	Stores the selected UPTM target programme name.
sourceInstitutionId	String	Foreign Key -> Institution.id, Indexed, Nullable	Links the application to the student's source institution.

Column	Data Type	Constraints	Description
sourceBranchId	String	Foreign Key -> Branch.id, Indexed, Nullable	Links the application to the student's source branch.
targetProgrammeId	String	Foreign Key -> Programme.id, Nullable	Links the application to the target programme master record.
isInternalUPTM	Boolean	Not Null	Indicates whether the source institution is internal UPTM.
status	Enum(ApplicationStatus)	Indexed, Default DRAFT, Not Null	Stores the current workflow status.
currentStep	Int	Default 1, Not Null	Stores the current UI step in the student process.
reviewedByHOPId	String	Foreign Key -> User.id, Nullable	Stores the HOP responsible for the final academic review.
totalApprovedCreditHours	Int	Nullable	Stores the final approved credit hours.
placedSemester	Int	Nullable	Stores the final semester placement.
createdAt	DateTime	Default now(), Not Null	Stores when the application was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest application update time.
withdrawnAt	DateTime	Nullable	Stores the timestamp of withdrawal if the student withdraws the application.
withdrawnBy	Enum(Role)	Nullable	Stores the role that triggered the withdrawal.
withdrawReason	String	Nullable	Stores the reason for withdrawal when provided.

Table 4.10 shows the structure of the ApplicationStatusHistory table, which stores the audit trail of application status changes. This table records when an application moved to a new state, who triggered the change, and any notes attached to that update. It is important for traceability because the credit transfer workflow involves several stages and multiple staff roles.

Table 4.10: Data Dictionary for ApplicationStatusHistory

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each application status history record.
applicationId	String	Foreign Key -> CreditTransferApplication.id, Indexed, Not Null	Links the status entry to one application.
status	Enum(ApplicationStatus)	Not Null	Stores the status recorded at that point in time.
actorId	String	Foreign Key -> User.id, Indexed, Nullable	Stores the user who triggered the status change.
notes	String	Nullable	Stores additional context about the status change.
createdAt	DateTime	Default now(), Not Null	Stores when the status entry was recorded.

Table 4.11 shows the structure of the CTACourse table, which stores the individual source courses under a credit transfer application. Each row represents one course extracted from a transcript or added manually, together with matching results, review flags, and decision information. This table is one of the most important operational tables because academic approval is ultimately made at course level, not only at application level.

Table 4.11: Data Dictionary for CTACourse

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each course row within an application.
applicationId	String	Foreign Key -> CreditTransferApplication.id, Not Null	Links the course to its parent application.
sourceInstitutionId	String	Foreign Key -> Institution.id, Indexed, Not Null	Stores the source institution for the course.

Column	Data Type	Constraints	Description
sourceBranchId	String	Foreign Key -> Branch.id, Indexed, Nullable	Stores the source branch for the course when applicable.
sourceProgrammeName	String	Nullable	Stores the source programme name associated with the course.
sourceProgrammeCode	String	Nullable	Stores the source programme code associated with the course.
sourceCourseCode	String	Not Null	Stores the source course code.
sourceCourseName	String	Not Null	Stores the source course name.
sourceCreditHours	Int	Not Null	Stores the source course credit hours.
sourceGrade	String	Nullable	Stores the student's grade for the source course.
targetCourseCode	String	Nullable	Stores the matched target course code.
targetCourseName	String	Nullable	Stores the matched target course name.
targetCreditHours	Int	Nullable	Stores the target course credit hours.
isFlagged	Boolean	Nullable	Indicates whether the course requires further review or evidence.
recommendationSource	Enum (Recommendation Source)	Nullable	Stores whether the recommendation came from articulation or manual logic.
notes	String	Nullable	Stores remarks or reasons for non-eligibility or review state.
decision	Enum (CourseDecision)	Default PENDING, Indexed, Not Null	Stores the current decision for the course.
compositeId	String	Nullable	Stores the composite mapping group identifier when several courses map together.

Column	Data Type	Constraints	Description
orGroup	Int	Nullable	Stores alternative grouping inside a composite mapping set.
isUserModified	Boolean	Default false, Not Null	Indicates whether the student manually edited the course row.
rpNotes	String	Nullable	Stores notes entered by RP during evaluation.
evaluatedByRPId	String	Foreign Key -> User.id, Nullable	Stores the RP who evaluated the course or linked shared decision.
sharedSyllabusCaseId	String	Foreign Key -> SharedSyllabusCase.id, Indexed, Nullable	Links the course to a shared representative syllabus case.
sharedReviewGroupId	String	Foreign Key -> SharedReviewGroup.id, Indexed, Nullable	Links the course to a shared RP review group.
comparisonStatus	Enum (ComparisonStatus)	Default NOT_STARTED, Not Null	Stores the comparison state for the course's review path.
comparisonScore	Float	Nullable	Stores a comparison score when available.
syllabusDeadline	DateTime	Indexed, Nullable	Stores the deadline for student syllabus submission.
autoRejectedAt	DateTime	Nullable	Stores the timestamp if the course was auto-rejected after deadline.
syllabusIntent	Enum (SyllabusIntent)	Nullable	Stores the student's declared intent for missing syllabus evidence.
createdAt	DateTime	Default now(), Not Null	Stores when the course row was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time for the course row.
id	String	Primary Key	Unique identifier for each course row within an application.

Table 4.12 shows the structure of the Syllabus table, which stores syllabus evidence used for academic comparison and review. This table supports both source syllabus uploaded by students and target syllabus maintained by HOP for comparison purposes. In the workflow, it connects uploaded files, extracted text, and extraction review status into one structured evidence record.

Table 4.12: Data Dictionary for Syllabus

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each syllabus record.
type	Enum(SyllabusType)	Default SOURCE, Not Null	Distinguishes whether the syllabus is source-side or target-side.
ctaCourseId	String	Foreign Key -> CTACourse.id, Unique, Nullable	Links the syllabus directly to one CTA course when it is not shared.
sharedSyllabusCaseId	String	Foreign Key -> SharedSyllabusCase.id, Unique, Nullable	Links the syllabus to a representative shared syllabus case.
institution	String	Not Null	Stores the institution name associated with the syllabus.
courseCode	String	Indexed with type, Not Null	Stores the course code associated with the syllabus.
courseName	String	Not Null	Stores the course name associated with the syllabus.
faculty	String	Nullable	Stores the faculty name when relevant.
programmeCode	String	Nullable	Stores the programme code related to the syllabus.
programmeName	String	Nullable	Stores the programme name related to the syllabus.
creditHour	Int	Nullable	Stores the credit hour value for the syllabus course.
rawText	String	Nullable	Stores extracted raw text from the syllabus file.
worksheetName	String	Nullable	Stores the selected worksheet name when the source file is XLSX.
fileId	String	Foreign Key -> StoredFile.id, Unique, Nullable	Links the syllabus to its uploaded file when one exists.
uploadedAt	DateTime	Default now(), Not Null	Stores when the syllabus was uploaded or recorded.
extractionStatus	Enum (ExtractionStatus)	Default PENDING, Not Null	Stores the syllabus extraction and review state.

Column	Data Type	Constraints	Description
extractionReviewedById	String	Foreign Key -> User.id, Nullable	Stores the staff reviewer of extraction output.
extractionReviewedAt	DateTime	Nullable	Stores when extraction review was completed.

Table 4.13 shows the structure of the ArticulationCourse table, which stores articulation mappings between source and target courses. This table is used to manage proposed, approved, and revoked mapping rules that can later influence transcript matching and academic recommendation outcomes. It also supports RP assignment and shared review linkage when an articulation rule needs formal academic evaluation.

Table 4.13: Data Dictionary for ArticulationCourse

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each articulation mapping row.
sessionCode	String	Indexed, Not Null	Stores the academic session for the articulation rule.
targetProgrammeCode	String	Indexed, Not Null	Stores the target programme code.
targetProgrammeName	String	Nullable	Stores the target programme name.
sourceProgrammeCode	String	Nullable	Stores the source programme code.
sourceProgrammeName	String	Nullable	Stores the source programme name.
sourceCourseCode	String	Not Null	Stores the source course code.
sourceCourseName	String	Not Null	Stores the source course name.
sourceCreditHours	Int	Not Null	Stores the source course credit hours.
targetCourseCode	String	Not Null	Stores the target course code.
targetCourseName	String	Not Null	Stores the target course name.
targetCreditHours	Int	Not Null	Stores the target course credit hours.
compositeId	String	Nullable	Stores the composite group identifier for grouped mappings.
orGroup	Int	Nullable	Stores alternative group logic within a composite mapping.

Column	Data Type	Constraints	Description
status	Enum (ArticulationStatus)	Default PROPOSED, Indexed, Not Null	Stores whether the articulation rule is proposed, approved, or revoked.
revokeReason	String	Nullable	Stores the reason if an articulation rule is revoked.
proposedByHOPId	String	Nullable	Stores the HOP who proposed the articulation rule.
approvedByHOPId	String	Nullable	Stores the HOP who approved the articulation rule.
assignedRPId	String	Foreign Key -> User.id, Indexed, Nullable	Stores the RP assigned for articulation review when needed.
assignedAt	DateTime	Nullable	Stores when RP assignment was made.
rpDecision	Enum (CourseDecision)	Nullable	Stores the RP decision on the articulation row.
rpNotes	String	Nullable	Stores RP comments for the articulation review.
rpReviewedAt	DateTime	Nullable	Stores when RP completed the articulation review.
sourceSyllabusId	String	Foreign Key -> Syllabus.id, Indexed, Nullable	Links the articulation row to a selected source syllabus.
targetSyllabusId	String	Foreign Key -> Syllabus.id, Indexed, Nullable	Links the articulation row to a selected target syllabus.
sharedReviewGroupId	String	Foreign Key -> SharedReviewGroup.id, Indexed, Nullable	Links the articulation row to a shared review group.
createdAt	DateTime	Default now(), Not Null	Stores when the articulation row was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.
userId	String	Foreign Key -> User.id, Nullable	Stores an associated user link when needed by the implementation.
sourceInstitutionId	String	Foreign Key -> Institution.id, Indexed, Not Null	Stores the source institution for the articulation rule.
sourceBranchId	String	Foreign Key -> Branch.id, Indexed, Nullable	Stores the source branch for the articulation rule.

Table 4.14 shows the structure of the SharedSyllabusCase table, which stores the canonical scope for a representative source syllabus. This table exists because multiple CTA courses or articulation rows may rely on the same source syllabus evidence when the academic context is equivalent. By centralizing that scope, the system avoids duplicating review effort and supports evidence reuse.

Table 4.14: Data Dictionary for SharedSyllabusCase

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each shared syllabus case.
caseKey	String	Unique, Not Null	Stores the canonical key used to identify a reusable representative syllabus scope.
sourceInstitutionId	String	Foreign Key -> Institution.id, Indexed, Not Null	Stores the source institution for the shared case.
sourceBranchId	String	Foreign Key -> Branch.id, Indexed, Nullable	Stores the source branch for the shared case.
sourceProgrammeCode	String	Nullable	Stores the optional source programme code in the shared case signature.
sourceCourseCode	String	Not Null	Stores the source course code being represented.
sourceCourseName	String	Nullable	Stores the source course name being represented.
targetProgrammeCode	String	Indexed, Not Null	Stores the target programme code for reuse scope.
sessionCode	String	Indexed, Not Null	Stores the academic session for reuse scope.
createdAt	DateTime	Default now(), Not Null	Stores when the shared case was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.15 shows the structure of the SharedReviewGroup table, which stores the shared academic review unit assigned to RP. Instead of forcing RP to review each flagged course separately, this table groups related cases that can be evaluated through one shared syllabus comparison and one academic decision. This design reflects the actual workflow described in the system journey, where one RP decision may affect multiple linked records.

Table 4.15: Data Dictionary for SharedReviewGroup

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each shared review group.
signature	String	Unique, Not Null	Stores the canonical review-group signature used for reuse.
sessionCode	String	Indexed, Not Null	Stores the session code of the shared review scope.
targetProgrammeCode	String	Indexed, Not Null	Stores the target programme code of the shared review scope.
compositeId	String	Nullable	Stores the composite mapping identifier if the group is composite-based.
assignedRPId	String	Foreign Key -> User.id, Indexed, Nullable	Stores the RP assigned to the shared review group.
assignedAt	DateTime	Nullable	Stores when the RP assignment was made.
rpDecision	Enum (CourseDecision)	Nullable	Stores the RP decision applied at shared-group level.
rpNotes	String	Nullable	Stores RP comments for the shared review group.
rpReviewedAt	DateTime	Nullable	Stores when RP completed the evaluation.
comparisonStatus	Enum (ComparisonStatus)	Default NOT_STARTED, Not Null	Stores the comparison state for the shared group.
comparisonScore	Float	Nullable	Stores an overall comparison score when available.
createdAt	DateTime	Default now(), Not Null	Stores when the shared review group was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.16 shows the structure of the SharedReviewGroupMember table, which functions as the linking table between shared review groups and shared syllabus cases. This table allows one review group to contain one or more representative syllabus cases, including grouped or alternative combinations for composite mappings. It is therefore essential for implementing the many-to-many reuse logic behind shared RP review.

Table 4.16: Data Dictionary for SharedReviewGroupMember

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each shared review group member row.
sharedReviewGroupId	String	Foreign Key -> SharedReviewGroup.id, Unique with case and group logic, Not Null	Links the row to a shared review group.
sharedSyllabusCaseId	String	Foreign Key -> SharedSyllabusCase.id, Indexed, Not Null	Links the row to a shared syllabus case.
compositeId	String	Nullable	Stores composite grouping for the member when relevant.
orGroup	Int	Nullable	Stores OR-group logic for composite review handling.
createdAt	DateTime	Default now(), Not Null	Stores when the group member row was created.

Table 4.17 shows the structure of the StoredFile table, which stores file metadata for transcripts, syllabi, and export documents. This table does not store the file binary directly, but instead keeps the object-storage location and business classification needed to retrieve and manage uploaded evidence. In practice, it acts as the bridge between database records and the external file storage layer.

Table 4.17: Data Dictionary for StoredFile

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each stored file record.
applicationId	String	Foreign Key -> CreditTransferApplication.id, Indexed, Nullable	Links the file to an application when applicable.
ctaCourseId	String	Foreign Key -> CTACourse.id, Indexed, Nullable	Links the file to a CTA course when applicable.
articulationCourseId	String	Foreign Key -> ArticulationCourse.id, Indexed, Nullable	Links the file to an articulation row when applicable.
bucket	String	Not Null	Stores the storage bucket name.
objectKey	String	Not Null	Stores the object key used in S3 or SeaweedFS.

originalFileName	String	Not Null	Stores the original uploaded filename.
mimeType	String	Not Null	Stores the file MIME type.
sizeBytes	Int	Not Null	Stores the file size in bytes.
fileType	Enum(StoredFileType)	Indexed, Not Null	Stores the business classification of the file.
uploadedByUserId	String	Foreign Key -> User.id, Not Null	Stores the user who uploaded or generated the file.
createdAt	DateTime	Default now(), Not Null	Stores when the file record was created.

Table 4.18 shows the structure of the CreditTransferExport table, which stores the downstream export rows generated by AAS after final academic decisions are completed. This table converts finalized credit transfer outcomes into operational records that can be downloaded and processed outside the review workflow. It is therefore the last major persistence layer in the lifecycle of a completed application.

Table 4.18: Data Dictionary for CreditTransferExport

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each downstream export row.
studentMatrixNo	String	Indexed, Not Null	Stores the student matrix number included in export output.
subjectCode	String	Not Null	Stores the exported subject code.
creditHours	Int	Not Null	Stores the approved credit hours exported for the subject.
institution	String	Not Null	Stores the originating institution label in export output.
sessionCode	String	Indexed, Not Null	Stores the academic session of the export row.
semester	Int	Not Null	Stores the target semester used in the export row.
targetProgrammeCode	String	Not Null	Stores the target programme code in the export contract.
notes	String	Nullable	Stores optional export remarks.

Column	Data Type	Constraints	Description
updatedByStaffId	String	Foreign Key -> User.id, Not Null	Stores the AAS staff user who generated the export row.
applicationId	String	Foreign Key -> CreditTransferApplication.id, Not Null	Links the export row back to the source application.
updatedAt	DateTime	Default now(), Not Null	Stores when the export row was written.

Table 4.19 shows the structure of the CreditTransferExport table, which stores the downstream export rows generated by AAS after final academic decisions are completed. This table converts finalized credit transfer outcomes into operational records that can be downloaded and processed outside the review workflow. It is therefore the last major persistence layer in the lifecycle of a completed application.

Table 4.19: Data Dictionary for SharedReviewComparison

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each top-level shared comparison record.
sharedReviewGroupId	String	Foreign Key -> SharedReviewGroup.id, Unique, Not Null	Links the comparison record to one shared review group.
currentTargetSyllabusId	String	Foreign Key -> Syllabus.id, Indexed, Nullable	Stores the current target syllabus used for comparison.
currentTargetCourseCode	String	Nullable	Stores the target course code snapshot.
currentTargetCourseName	String	Nullable	Stores the target course name snapshot.
latestRunId	String	Foreign Key -> SharedReviewComparisonRun.id, Indexed, Nullable	Stores the latest run reference.
latestCompletedRunId	String	Foreign Key -> SharedReviewComparisonRun.id, Indexed, Nullable	Stores the latest completed run reference.
latestStatus	Enum (ComparisonStatus)	Default NOT_STARTED, Not Null	Stores the latest overall comparison status.

Column	Data Type	Constraints	Description
latestCoveragePercent	Float	Nullable	Stores the latest coverage percentage.
latestAverageTopicScore	Float	Nullable	Stores the latest average topic score.
latestTotalScore	Float	Nullable	Stores the latest total score.
latestTargetItemCount	Int	Nullable	Stores the number of target units evaluated in the latest run.
latestConclusion	String	Nullable	Stores the latest comparison conclusion.
latestModel	String	Nullable	Stores the model name used in the latest run.
latestProvider	Enum (SharedReview ComparisonProvider)	Nullable	Stores the provider used in the latest run.
latestRouteMode	Enum (SharedReview Comparison RouteMode)	Nullable	Stores whether the latest run used local or cloud routing.
latestHost	String	Nullable	Stores the host used for the latest run when available.
latestRunStartedAt	DateTime	Nullable	Stores when the latest run started.
latestRunCompletedAt	DateTime	Nullable	Stores when the latest run completed.
createdAt	DateTime	Default now(), Not Null	Stores when the comparison record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.20 shows the structure of the SharedReviewComparisonRun table, which stores one complete execution of the syllabus comparison pipeline. Each run records the provider, model, target syllabus snapshot, timing, scoring summary, and any error information associated with that execution. This table is useful for traceability because the system may execute the comparison process more than once for the same shared review scope.

Table 4.20: Data Dictionary for SharedReviewComparisonRun

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each comparison run.
sharedReviewComparisonId	String	Foreign Key -> SharedReviewComparison.id, Indexed, Not Null	Links the run to its top-level comparison record.
status	Enum (ComparisonStatus)	Default PROCESSING, Not Null	Stores the current run status.
routeMode	Enum (SharedReviewComparisonRouteMode)	Not Null	Stores whether the run used local or cloud routing.
provider	Enum (SharedReviewComparisonProvider)	Not Null	Stores the provider category used by the run.
model	String	Nullable	Stores the model name used in the run.
host	String	Nullable	Stores the host endpoint used in the run.
targetSyllabusId	String	Foreign Key -> Syllabus.id, Indexed, Nullable	Links the run to the target syllabus snapshot.
targetCourseCode	String	Nullable	Stores the target course code snapshot.
targetCourseName	String	Nullable	Stores the target course name snapshot.
targetFileName	String	Nullable	Stores the target syllabus filename snapshot.
summaryCoveragePercent	Float	Nullable	Stores the coverage percentage result.

Column	Data Type	Constraints	Description
summaryAverage TopicScore	Float	Nullable	Stores the average topic score result.
summaryTotalScore	Float	Nullable	Stores the total score result.
summaryMaxTopicScore	Int	Nullable	Stores the maximum topic score.
summaryTargetItemCount	Int	Nullable	Stores the number of target items evaluated.
summaryFullCount	Int	Nullable	Stores the count of fully covered target items.
summarySubstantialCount	Int	Nullable	Stores the count of substantially covered target items.
summaryPartialCount	Int	Nullable	Stores the count of partially covered target items.
summaryMinimalCount	Int	Nullable	Stores the count of minimally covered target items.
summaryNoneCount	Int	Nullable	Stores the count of non-covered target items.
summaryConclusion	String	Nullable	Stores the textual conclusion of the run.
startedAt	DateTime	Default now(), Indexed, Not Null	Stores when the run started.
completedAt	DateTime	Nullable	Stores when the run completed.
error	String	Nullable	Stores an error message if the run failed.
createdAt	DateTime	Default now(), Not Null	Stores when the run record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.21 shows the structure of the SharedReviewComparisonRunSource table, which stores the representative source syllabus inputs used in a specific comparison run. This table captures the ordered source-side evidence snapshot so that the run can later be audited even if the original source set changes. In other words, it preserves exactly which source syllabus records were compared at that time.

Table 4.21: Data Dictionary for SharedReviewComparisonRunSource

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each source-syllabus snapshot row in a comparison run.
runId	String	Foreign Key -> SharedReviewComparisonRun.id, Indexed, Not Null	Links the row to the comparison run that used the source evidence.
sourceSyllabusId	String	Foreign Key -> Syllabus.id, Indexed, Nullable	Links the row to the representative source syllabus when one exists.
sortOrder	Int	Indexed with runId, Not Null	Stores the display or processing order of source inputs in the run.
sourceCourseCode	String	Nullable	Stores the source course code snapshot used during the run.
sourceCourseName	String	Nullable	Stores the source course name snapshot used during the run.
sourceCourseTitle	String	Nullable	Stores an alternative or normalized source course title.
sourceInstitution	String	Nullable	Stores the source institution label shown in comparison evidence.
sourceFileName	String	Nullable	Stores the filename of the representative source syllabus used in the run.
createdAt	DateTime	Default now(), Not Null	Stores when the source snapshot row was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time for the row.

Table 4.22 shows the structure of the SharedReviewComparisonStage table, which stores the individual stages inside a comparison run. Because the comparison process is executed in several logical steps such as preprocessing, mapping, verification, and scoring, this table records the lifecycle of each stage separately. This makes the pipeline more transparent and easier to debug when a comparison fails or produces uncertain results.

Table 4.22: Data Dictionary for SharedReviewComparisonStage

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each stage inside a comparison run.
runId	String	Foreign Key -> SharedReviewComparisonRun.id, Indexed, Not Null	Links the stage to the parent comparison run.
stageKey	Enum(SharedReviewComparisonStageKey)	Unique with runId and stageIndex, Not Null	Stores the logical pipeline stage being executed.
stageIndex	Int	Default 0, Unique with runId and stageKey, Not Null	Stores the stage sequence index when the same stage key can repeat.
name	String	Not Null	Stores the human-readable stage name.
status	Enum(ComparisonStatus)	Default PROCESSING, Not Null	Stores the current status of the stage.
provider	Enum(SharedReviewComparisonProvider)	Nullable	Stores the provider type used for this stage.
model	String	Nullable	Stores the model used for this stage.
host	String	Nullable	Stores the host or endpoint used for this stage.
think	Boolean	Nullable	Indicates whether the stage was configured to use reasoning or thinking mode.
promptTemplate	String	Nullable	Stores the prompt template or identifier used to run the stage.
inputArtifactKeys	String[]	Default [], Not Null	Stores the artifact keys consumed by the stage.

Column	Data Type	Constraints	Description
outputArtifactKeys	String[]	Default [], Not Null	Stores the artifact keys produced by the stage.
notes	String[]	Default [], Not Null	Stores pipeline notes or processing remarks for the stage.
startedAt	DateTime	Default now(), Not Null	Stores when the stage started.
completedAt	DateTime	Nullable	Stores when the stage completed.
error	String	Nullable	Stores an error message when the stage fails.
createdAt	DateTime	Default now(), Not Null	Stores when the stage record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.23 shows the structure of the SharedReviewComparisonStageAttempt table, which stores retry-level execution data for a comparison stage. This table becomes useful when the system needs to rerun a stage because of provider failure, parsing problems, or other processing issues. It improves observability by preserving attempt counts, durations, and failure details instead of overwriting earlier execution history.

Table 4.23: Data Dictionary for SharedReviewComparisonStageAttempt

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each execution attempt of a comparison stage.
stageId	String	Foreign Key -> SharedReviewComparisonStage.id, Unique with attemptNumber, Indexed, Not Null	Links the attempt to a specific stage.
attemptNumber	Int	Unique with stageId, Not Null	Stores the retry number for the stage attempt.
status	Enum(SharedReviewComparisonAttemptStatus)	Not Null	Stores whether the attempt completed successfully or failed.
startedAt	DateTime	Not Null	Stores when the attempt started.

Column	Data Type	Constraints	Description
completedAt	DateTime	Nullable	Stores when the attempt finished.
durationMs	Int	Nullable	Stores the attempt duration in milliseconds.
think	Boolean	Not Null	Indicates whether reasoning mode was enabled for the attempt.
hadThinkingTrace	Boolean	Nullable	Indicates whether the response included a thinking trace or reasoning trace.
responseChars	Int	Nullable	Stores the size of the returned model response in characters.
repaired	Boolean	Nullable	Indicates whether the attempt output required repair or post-processing.
error	String	Nullable	Stores the failure message for an unsuccessful attempt.
createdAt	DateTime	Default now(), Not Null	Stores when the attempt record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.24 shows the structure of the SharedReviewComparisonArtifact table, which stores output and debug artifacts produced during the comparison pipeline. These artifacts may contain text, markdown, or JSON content generated by a run or one of its stages. The table is important because it preserves intermediate and final machine-produced evidence that can later support review, auditing, or troubleshooting.

Table 4.24: Data Dictionary for SharedReviewComparisonArtifact

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each comparison artifact.
runId	String	Foreign Key -> SharedReviewComparisonRun.id, Unique with artifactKey, Not Null	Links the artifact to the parent comparison run.
stageId	String	Foreign Key -> SharedReviewComparisonStage.id, Indexed, Nullable	Links the artifact to the stage that produced it when applicable.
artifactKey	String	Unique with runId, Not Null	Stores the logical name used to retrieve the artifact.
role	Enum(SharedReviewComparisonArtifactRole)	Not Null	Stores whether the artifact is a normal output or debug material.
kind	Enum(SharedReviewComparisonArtifactKind)	Not Null	Stores the artifact content type at application level.
mimeType	String	Nullable	Stores the MIME type of the artifact payload.
textContent	String	Nullable	Stores text or markdown content when the artifact is text-based.
jsonContent	Json	Nullable	Stores structured JSON output for stage or run artifacts.
createdAt	DateTime	Default now(), Not Null	Stores when the artifact record was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

Table 4.25 shows the structure of the SharedReviewComparisonUnitResult table, which stores the per-topic or per-unit comparison outcome for a target syllabus. Each row records the matched source evidence, coverage label, numeric score, and reason assigned to one target topic. This table is particularly important because it captures the detailed academic basis behind the higher-level comparison summary seen by RP and HOP.

Table 4.25: Data Dictionary for SharedReviewComparisonUnitResult

Column	Data Type	Constraints	Description
id	String	Primary Key	Unique identifier for each target-unit result inside a comparison run.
runId	String	Foreign Key -> SharedReviewComparisonRun.id, Unique with targetId, Indexed, Not Null	Links the result to the comparison run.
sortOrder	Int	Indexed with runId, Not Null	Stores the display or evaluation order of target units.
targetId	String	Unique with runId, Not Null	Stores the identifier of the target topic or unit being evaluated.
targetTopic	String	Not Null	Stores the target topic title used in the result row.
targetModuleTitle	String	Nullable	Stores the broader module or section title for the target topic.
targetSubtopics	String[]	Default [], Not Null	Stores the subtopics listed under the target topic.
targetLearningObjectives	String[]	Default [], Not Null	Stores learning objectives associated with the target topic.
matchedSourceIds	String[]	Default [], Not Null	Stores the identifiers of matched source items.
matchedSourceTopics	String[]	Default [], Not Null	Stores the matched source topic titles.
matchedSourceCourseTitles	String[]	Default [], Not Null	Stores the source course titles that supplied evidence.
matchedSourceFileNames	String[]	Default [], Not Null	Stores the source filenames referenced in the match.

Column	Data Type	Constraints	Description
matchedSourceLabels	String[]	Default [], Not Null	Stores short labels used to identify the source evidence items.
matchedSourceSummary	String	Not Null	Stores the textual summary of why the source evidence matches the target topic.
evidence	String[]	Default [], Not Null	Stores evidence bullets or excerpts supporting the match.
initialLabel	String	Nullable	Stores an initial or intermediate qualitative label before final review.
finalLabel	Enum(SharedReviewCoverageLabel)	Not Null	Stores the final coverage label for the target unit.
numericScore	Int	Not Null	Stores the numeric score assigned to the target unit.
reason	String	Not Null	Stores the explanation for the final label and score.
createdAt	DateTime	Default now(), Not Null	Stores when the unit-result row was created.
updatedAt	DateTime	Auto Updated, Not Null	Stores the latest update time.

4.3.2 Data Flow Diagram

A data flow diagram shows how the data in a system moves from one place to another. IBM (n.d.) explains that there are four main components to a data flow diagram: external entities, processes, data stores, and data flows. For the credit transfer system, these components will be used to represent the different aspects of the system, including its users, third-party services, processes, and the data that is stored between these elements. The use of a data flow diagram for the credit transfer system is appropriate due to the system's data-driven processes. When a student wishes to earn their credits from another school, they must perform a few steps to provide evidence of their syllabus and transcript. Once this data is provided to the system, the system will be able to transfer their credits to the new school by performing these processes in order. The data flow diagram will be used to represent and explain this process.

There are two levels of data flow diagrams for the credit transfer system. The first is the Level 0 diagram, which represents the system as one process (IBM, n.d.). Then, the Level 1 diagram breaks that one process into the processes that occur within the system, based on the data that they use. Using these two levels of diagrams, the reader will be able to understand not just how the system is structured, but also how the system performs its processes internally.

4.3.3 Level 0 Data Flow Diagram

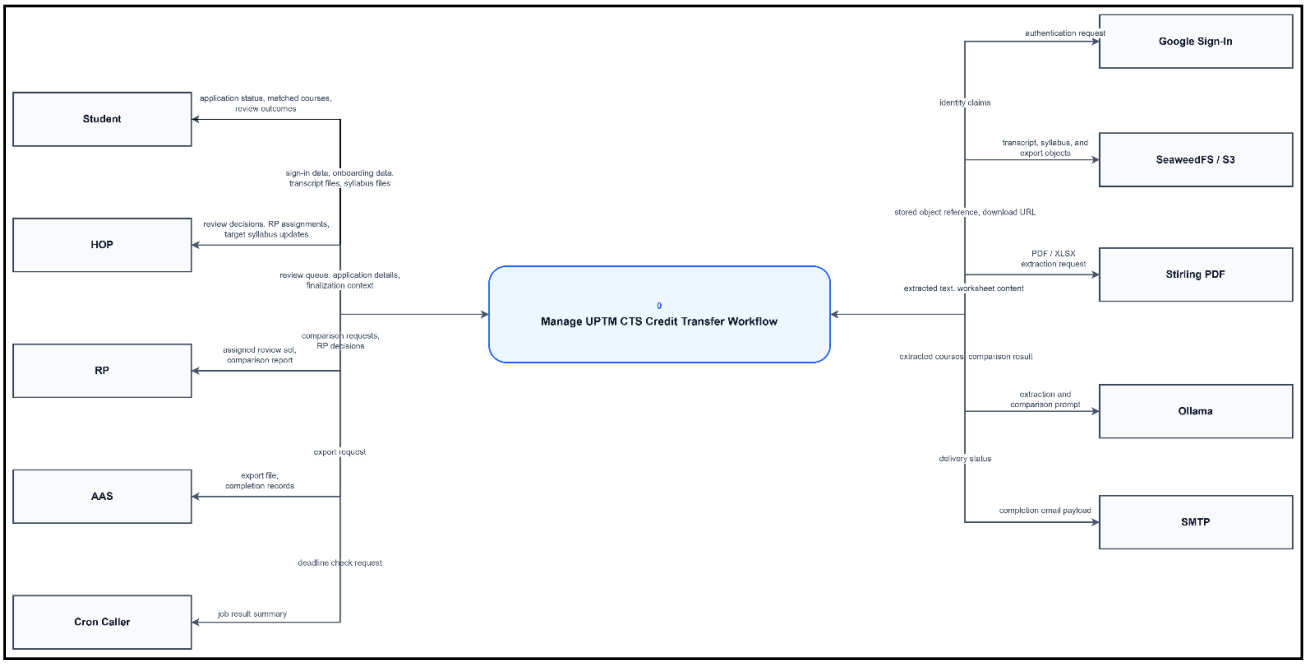


Figure 4.26: Level 0 Data Flow Diagram

Figure 4.26 shows the Level 0 DFD for the system. According to IBM, the entire system can be represented by a single central process: "Manage UPTM CTS Credit Transfer Workflow" (IBM, n.d.). This diagram presents a high-level view of the system and its interactions with other entities. The system can

be viewed in terms of its human entities (represented as a “process” and a “job” in the DFD), as well as its data entities.

On the left side of the diagram are the human entities that interact with the system. The first of these is the Student. The student can send data such as sign-in data, onboarding data, transcript files, and syllabus files to the system. The student can also receive data from the system in the form of application status, matched courses, and review outcomes. The next entity is HOP. The HOP sends data including review decisions, RP assignments, and target syllabus. It receives data including the review queue, the application, and the target syllabus. The HOP is the Head of Programme for a particular university program and reviews the application after the student has submitted their application and their transcripts have been processed. The next entity is the RP (Resource Person). This RP sends data including comparison requests and their decisions regarding the syllabi. It receives the review set and the comparison report. This RP only becomes involved in the application after the student’s syllabus has been sent to the system. The AAS entity is Academic Affairs Students and is only involved in the final stages of the application. They send an export request to the system and receive the export file. Finally, the Cron Caller is another entity that interacts with the system. This entity is used to enforce deadlines for the syllabi and sends a request to the system to check if the deadline has passed. It receives a job result summary in response to the request.

On the right side of the diagram are external services. Google Sign-In allows students to sign into the system using their existing Google accounts. The entity exchanges sign-in and authentication data with the system. SeaweedFS or S3 is used to store the transcript files, syllabus files, and export files. It returns object references for the files stored in the system. Stirling PDF allows the system to request the contents of a PDF or XLSX file. The entity returns the text for a PDF file or the worksheet content for an XLSX file. Ollama receives prompts including the extracted content from the syllabi to review and students’ applications to review. It returns the list of extracted courses or the review results. Finally, SMTP is responsible for sending an email to the student when their application is complete. This entity takes the email payload and returns an email delivery status.

Thus, data is transformed through the system. The raw data sent into the system from the student (such as transcript and syllabus files) is transformed into extracted content by the system. The courses identified for review by the student are sent to the review queue to be reviewed by the HOP, and upon approval are sent to the RP for further review. Once approved by the RP, the student’s application is complete, and the system can output the student’s application and notify the student of the outcome. Thus, the Level 0 DFD is helpful in showing what data the system receives and sends.

4.3.4 Level 1 Data Flow Diagram

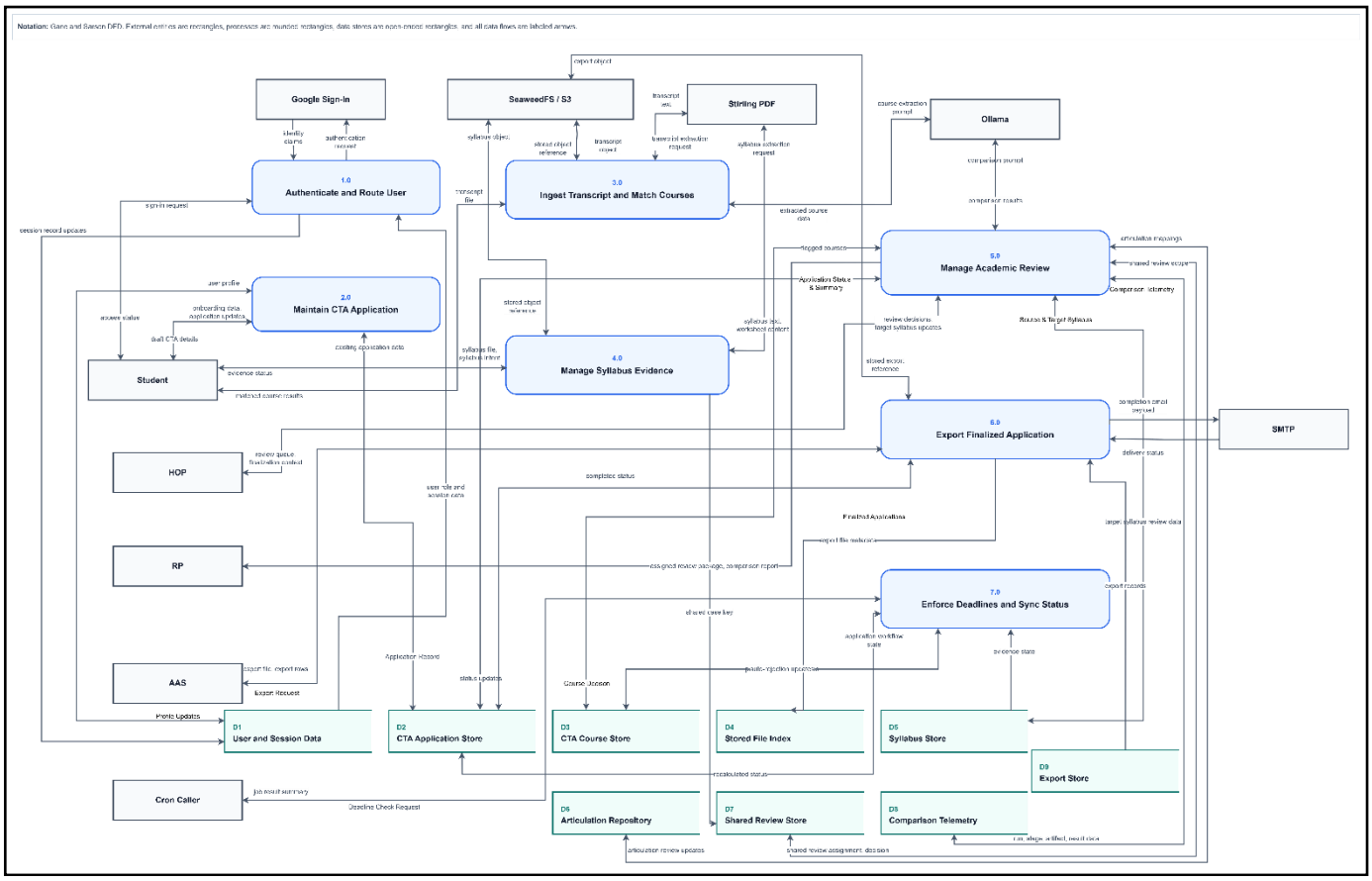


Figure 4.27: Level 1 Data Flow Diagram

Figure 4.27 expands on the Level 0 process to reveal seven major internal processes and nine supporting data stores. This process model follows the description of the Level 1 DFD that IBM developed for their systems. This diagram for the Level 1 DFD will therefore be the main analysis of the system and data from the beginning of the process until the end and the application is completed.

The first process within the system is 1.0 Authenticate and Route User. This initial process is used to receive the student's sign-in request and to interact with Google Sign-In for authentication purposes. Additionally, data from this process will read from and write to the D1 User and Session Data data store. The purpose of this process will be to authenticate the student, gain access to the system, and allow for the routing of that student to the appropriate page for the application. The second process is 2.0 Maintain CTA Application. During this stage, the student will be able to send their onboarding data or data that relates to their application, and the system will respond with a draft application for the CTA. This process will interact with the D1 and D2 data stores. This is the process during which the student's profile will be updated, and the CTA application will be created.

The third process is 3.0 Ingest Transcript and Match Courses. This is one of the most significant processes within the system. This process will take the transcript that the student uploads and store it in

the system through the D4 Stored File Index data store. Additionally, the transcript will be sent to SeaweedFS or S3, sent to Stirling PDF to extract the content of the file, and sent to Ollama for review of that content. This data will write to the D3 and D6 data stores for review. This process will transform the transcript that the student uploads into an academic record of the student's courses that can be evaluated by the system. The fourth process is 4.0 Manage Syllabus Evidence. This process handles the student's syllabus, placeholder syllabus evidence, and the shared syllabus that is representative of the student's course. This process will write to and read from the Stored File Index data store and the D5 Syllabus Store data store. Additionally, it will interact with the D7 Shared Review Store data store. Some courses will require the upload of the student's syllabus, other courses may have a placeholder for a syllabus that is yet to be uploaded, and others will use the shared syllabus as their evidence for the course approval.

The fifth process is 5.0 Manage Academic Review. This is the most academically intensive process within the Level 1 DFD. This process will interact with HOP and receive data regarding the review of the student's syllabus, the assignment of review packages to RPs, and the target syllabus for review. Additionally, this process will interact with the RPs and send requests for the comparison of the syllabus and receive responses from the RPs with the decisions for the student's syllabus. Data will flow into and out of the D2, D3, D5, D6, D7, and D8 data stores. Additionally, this process will interact with Ollama in the same manner as the transcript process. As many data stores as possible are referenced for the academic review so that the review team can have a complete understanding of the student's courses and syllabi. The sixth process is 6.0 Export Finalized Application. This process will be initiated by AAS and will result in the student's applications being exported. During this stage, the system will read the finalized applications from the D2 data store, write to SeaweedFS or S3, send an export email through SMTP, write to the D4 and D9 data stores, and initiate the exporting of the student's applications. This stage ensures that the student's applications are exported in a way that updates the various data fields of the student's completed application.

The seventh process is 7.0 Enforce Deadlines and Sync Status. This process will receive a request for a deadline check from the Cron Caller. Additionally, the system will read from the D2, D3, D5, and D7 data stores to verify the status of the applications, and write to these same data stores the new status and any auto-rejections of the students' applications. This data store will provide additional information for the system regarding the status of the students' applications and ensure that these statuses are accurate and up to date in the student information system. The nine data stores that are represented in the diagram include the D1 User and Session Data data store, the D2 CTA Application data store, the D3 CTA Course data store, the D4 Stored File Index data store, the D5 Syllabus Store data store, the D6 Articulation Repository data store, the D7 Shared Review Store data store, the D8 Comparison Telemetry data store, and the D9 Export data store. These will be explained in detail within this section. According to IBM (n.d.), the rules for data flow diagrams include that each data flow should be labeled with a brief label, each process should have a name that includes a verb phrase, each data store should have a

name that includes a noun phrase, and that the external entities should not have connections to the data stores. These rules are followed within this Level 1 DFD and are appropriate to apply to the system for such a reason. Every change that occurs within the student applications will be reflected within the system and transformed into the persistent business data.

The Level 1 DFD reveals to the readers the true organization of the information within the system. Additionally, every stage of the process and the value of the information that is transformed during each of these stages is made clear. Not only will the student be able to upload their transcript, but the information will be transformed into an academic record of their courses that can be easily evaluated and understood by the institution. Thus, the DFD for the Level 1 process reveals the true logic of the system and the justification for the consideration of the system as a workflow system for the institution.

4.3.5 Entity Relational Diagram (ERD)

An Entity-Relationship Diagram (ERD) is a visual representation of the system database. It displays the main entities within the system and the relationships between those entities. The ERD for the UPTM Credit Transfer System will help explain the database for the complete credit transfer process. Thus, rather than displaying each of the database tables separately, the ERD will help to illustrate the UPTM CTS database as a whole.

Overall, the ERD for the credit transfer system will be created with the workflow of the system at the center. Thus, the User entity can create a CreditTransferApplication. Furthermore, there can be multiple CTACourse records within a CreditTransferApplication. Each of these courses can have uploaded syllabi or stored files associated with them. Additionally, there are additional entities related to processes like reviewing the courses (ArticulationCourse), comparing syllabi from different institutions (SharedSyllabusCase), and sharing the review between the institution and the prospective college (SharedReviewGroup). These entities will help to describe the database for the UPTM CTS system.

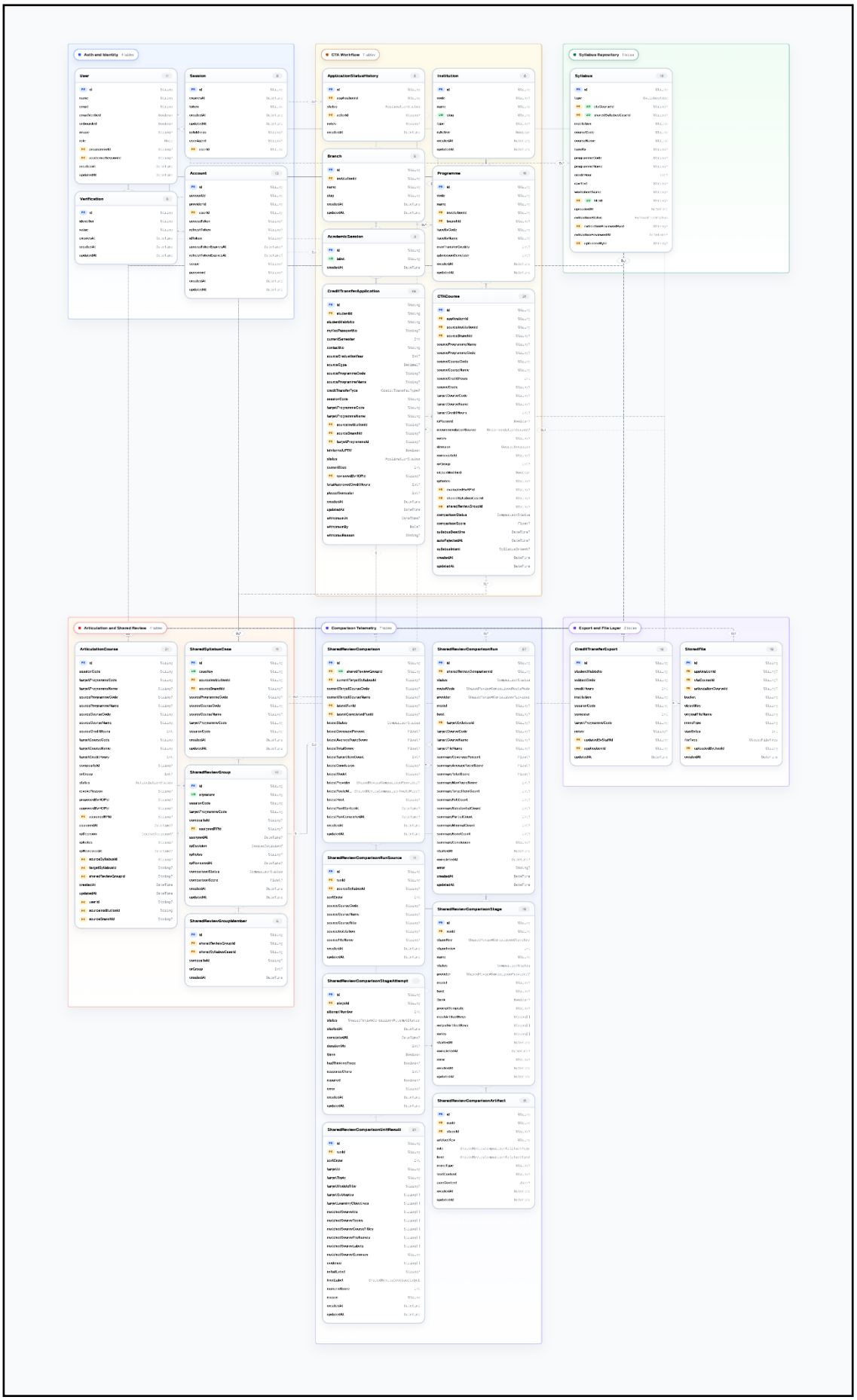


Figure 4.28: Entity-Relationship Diagram of the UPTM Credit Transfer System

Figure 4.28 presents the Entity-Relationship Diagram (ERD) of the UPTM Credit Transfer System, grouped into six functional groups to reflect the actual responsibilities of the database. The grouping of entities into these groups makes the ERD easier to understand. The system was not built around one table in the database, but rather around several groups of related tables to represent each of the system's major functions. The first group of tables is the Auth and Identity group, which contains the entities related to the system's authentication functionality: the User, Session, Account, and Verification tables. These tables manage user authentication and access to the system for students, HOP, RP, and AAS users. Additionally, the StaffAcademicScope table is also stored within this group.

The second group of tables is the CTA Workflow group. This group contains the entities related to the credit transfer workflow and functionality within the system: the Institution, Branch, Programme, AcademicSession, CreditTransferApplication, ApplicationStatusHistory, and CTACourse tables. These tables manage the application and acceptance of credit transfers into UPTM for student courses. The third group of tables is the Syllabus Repository group. This group contains the Syllabus table, which stores information about the syllabus that is uploaded by the student for review by the RP. The syllabus contains both the source syllabus and the target syllabus for the courses to be taken at UPTM, as well as content extracted from those syllabi. This table is important because the syllabi are required for the comparison between source and target courses for credit transfer acceptance. The fourth group of tables is the Articulation and Shared Review group. This group contains the ArticulationCourse, SharedSyllabusCase, SharedReviewGroup, and the group membership tables for each of these shared groups. This group is specifically important for the system's functionality to allow for the reuse of information shared between different entities. For instance, a syllabus shared with the RP by one course can also be associated with multiple CTA courses, and a review decision made for one CTA course can also be applied to multiple related CTA and articulation courses.

The fifth group of tables is the Comparison Telemetry group. This group contains the SharedReviewComparison table and related tables for the comparison's execution, such as the comparison run, stage, attempt, artifact, and unit result tables. These tables are not related to the student or user workflow of the system, but instead enable the system to trace the execution of the comparison between the uploaded syllabi. The final group of tables is the Export and File Layer group. This group contains the CreditTransferExport and StoredFile tables. The StoredFile table is used to store files uploaded by the student, such as syllabi and transcripts, while the CreditTransferExport table is used to store export records made available after the student's academic decision is completed. Thus, Figure 4.28 shows the UPTM Credit Transfer System database as a group of six functional groups with several tables within each group. Each table serves a specific function within the database and the system that uses the database.

4.4 Implementation Introduction

The implementation phase of the development of the UPTM Credit Transfer System (UPTM CTS) aims to develop the actual system based on the design phase. The implementation phase will focus on creating the actual system that will perform the required operations of the users of that system. During this phase, the technologies that will be utilized to implement the system will be discussed. Such technologies include the development environment in which the system will be created, the hosting platform on which the system will go live for end users, as well as any other software that is utilized during the implementation of the system. Through presenting this information regarding the technology of the system, users and readers will gain an understanding of the steps that were taken to implement the system.

4.5 Execution Platform

The execution platform of the system is the platform upon which the system will be developed and executed. As mentioned above, because this system exists as a web application, two environments will be utilized: a development environment and a hosting environment. The development environment will include the platform upon which the system will be coded, whereas the hosting environment will include the platform upon which the system will be published live for end users of the system. Furthermore, the selection of the execution platform is important in determining upon which platforms the system will exist and function. For instance, if the system is to be developed on a computer that utilizes Windows operating system software, then any web servers upon which the system may be hosted may also need to utilize software that is compatible with the Windows operating system.

4.5.1 Development Platform

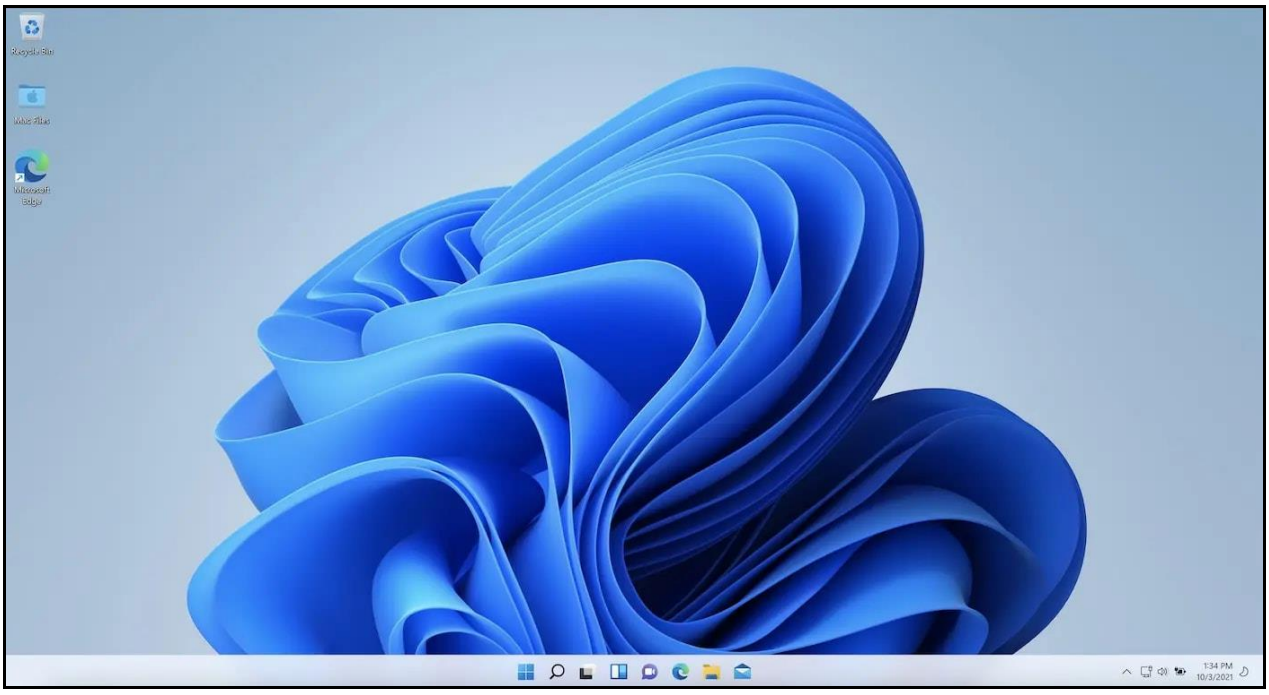


Figure 4.29: Windows 11 Pro

To develop the system, Windows 11 Pro operating system software will be utilized. This operating system was chosen for its stability in performing the tasks required to develop the software. For example, Windows 11 Pro was used in the compilation of the code for the application, the hosting of the database for the application, and in performing other tasks related to the development of the software. Furthermore, Windows 11 Pro was a suitable operating system for these tasks due to its compatibility with the computers and software that will be used to develop the system. Additionally, Windows 11 Pro was also beneficial in that it provided an environment that is already familiar to the developers of the system. Finally, Windows 11 Pro was selected due to its compatibility with the computers utilized throughout the development of the system. Thus, Windows 11 Pro was an appropriate development platform for the overall system.

4.5.2 Hosting Platform

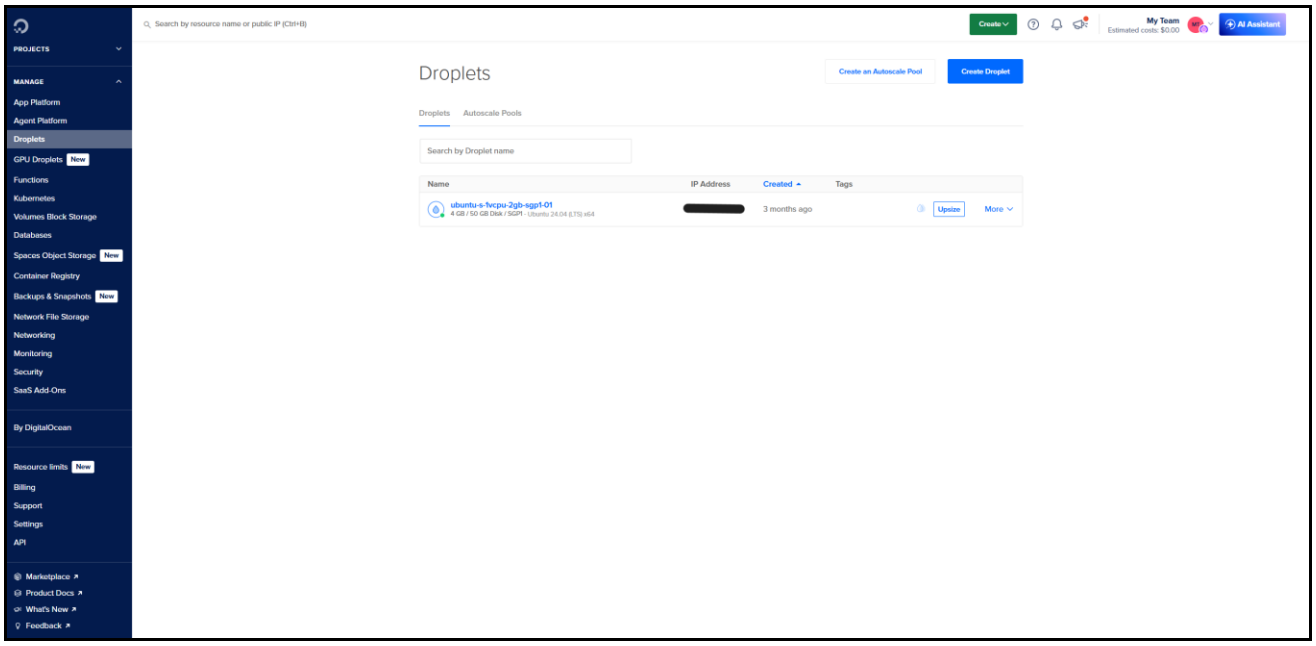


Figure 4.30: DigitalOcean Droplet Interface

DigitalOcean Droplet servers will host the UPTM Credit Transfer System (UPTM CTS) and will run the Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-90-generic x86_64) operating system software. The Droplet will contain the application software, the PostgreSQL database software, and other supporting services for the system. To deploy the software live onto the hosting platform, the software will be updated on the computers of the developers. After the developers have completed the changes that they would like to make to the software, the updated code will be committed to the software development repository on GitHub. Following the committing of the changes to the software repository, GitHub will initiate a webhook request to Coolify. Coolify will initiate the deployment of the software to the DigitalOcean Droplet by

performing an update to the code that is running on the Droplet servers. Thus, this hosting platform and its associated software are not limited to the web application software alone. For example, in addition to the main application software, the system is developed in association with a PostgreSQL database software platform. Furthermore, this server and platform is the main execution platform for the system and its functions, as all of the deployed software for the system is contained within this single Droplet server.

After the deployment, the resources deployed to the DigitalOcean Droplet may be managed through the Coolify dashboard platform.

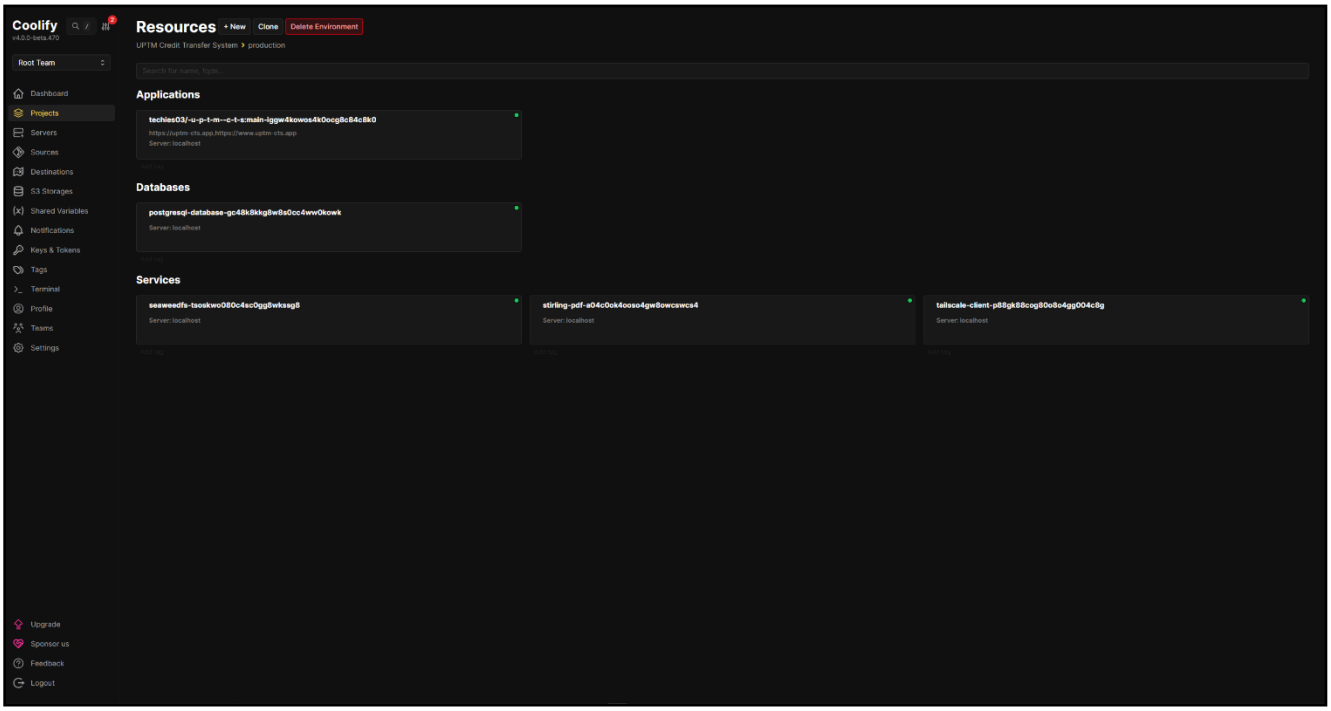


Figure 4.31: Coolify Interface

As shown in Figure 4.31, the deployed resources may be managed through the Coolify dashboard. The Coolify dashboard displays the deployed resources of the system, such as the main application software, the PostgreSQL database software, SeaweedFS software, Stirling PDF software, and Tailscale software. This Coolify dashboard is essential to managing the deployed application software of the system. For instance, after Coolify initiates the deployment of the updated software of the system, the status and deployment of the updated software and its deployed resources can be managed through the Coolify dashboard. Furthermore, the Coolify software may also assist in managing the interaction between some of the software of the system, such as the web-based application software, the database software, and other deployed services of the system.

4.6 Implementation Tools

The tools that will be used during the implementation of the system will be discussed in this section. These tools will include, but are not limited to, the coding and testing software for the application, the platform upon which the software will be hosted live for users of the system, as well as any other software that is utilized during the implementation phase of the UPTM CTS.

4.6.1 Development and Testing Software

The development and testing software of the system includes Visual Studio Code, Chrome DevTools, GitHub, and pgAdmin.

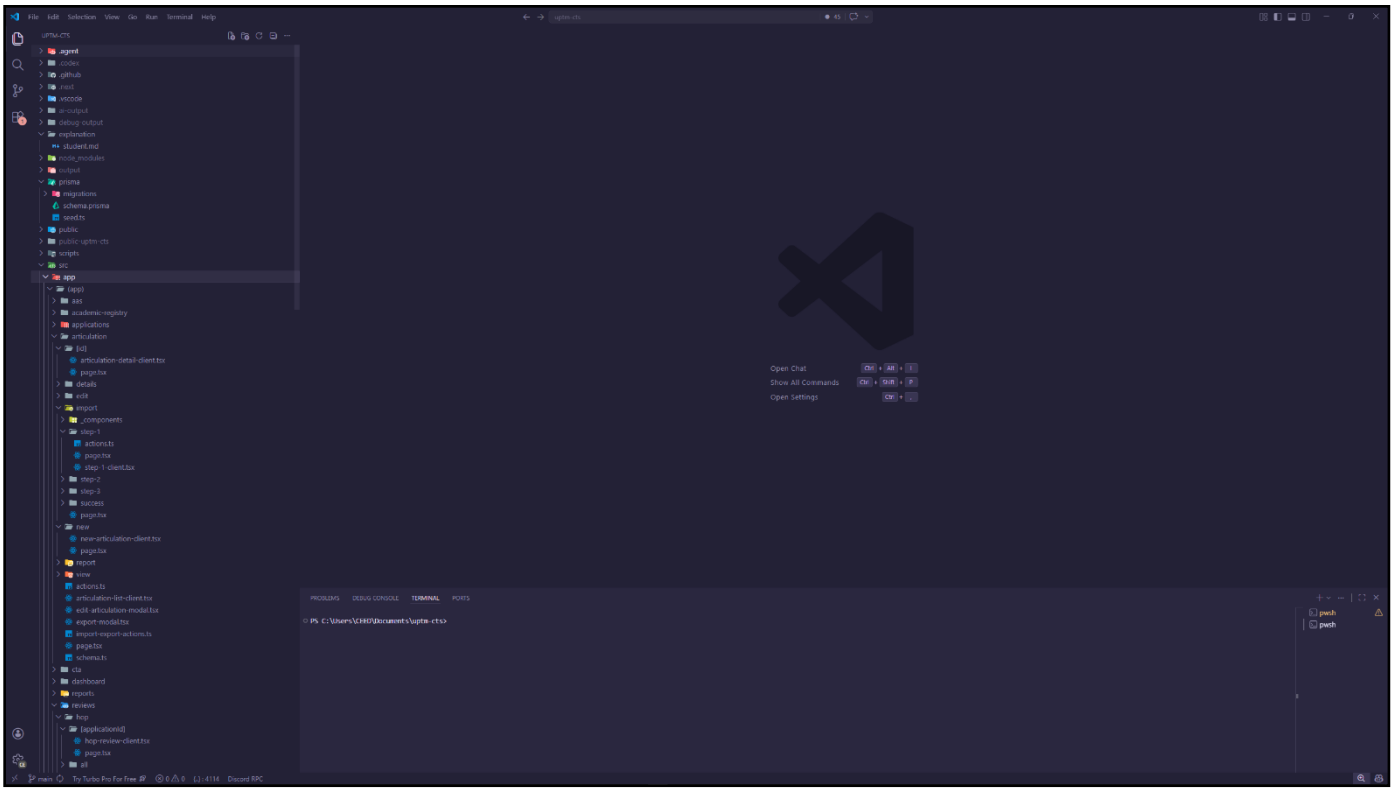


Figure 4.32: Visual Studio Code Interface

Visual Studio Code (VS Code) is an open-source code editor that was utilized in the implementation of the application software. VS Code allowed for the developers to implement the various features and components of the system, such as the Next.js pages, the React components, the Prisma schema, and the various services for the application. The VS Code software is presented in Figure 4.32. In the example of the application software as presented in Figure 4.32, the application software is based upon the App Router of the Next.js framework, and contains over 110 files of TSX software code for the various components of the implementation of the system.

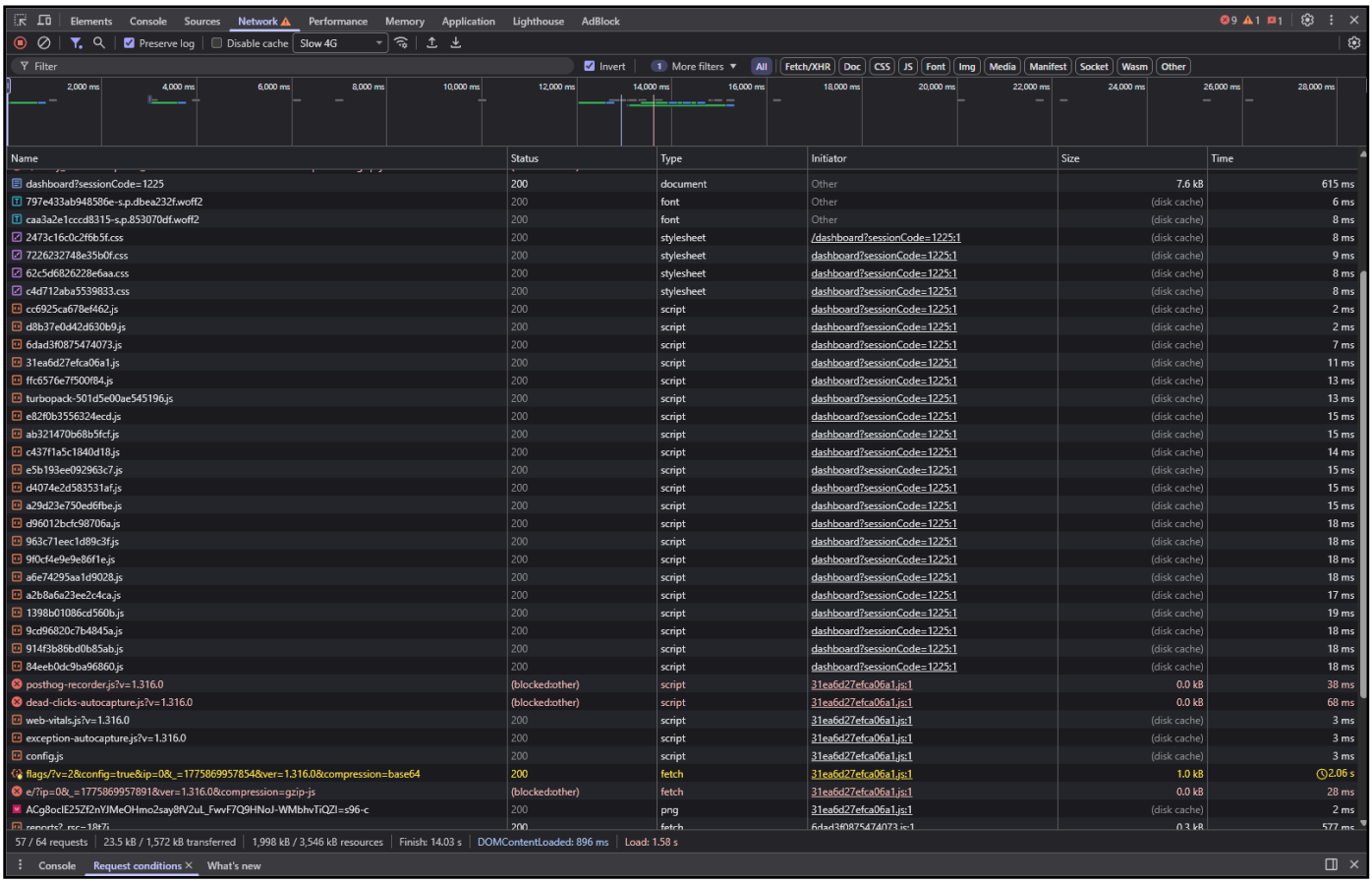


Figure 4.33: Chrome DevTools Interface

Chrome DevTools is a set of developer tools that can be accessed within the popular Chrome web browser. Within the application software of the system, DevTools was used to test and debug the application while it was being developed. As shown in Figure 4.33, the developer tools within the web browser were used to test for errors within the application, inspect the elements of the web pages, observe the network requests that were performed by the users of the system, and view the responses that were sent by the server software to those requests. Thus, Chrome DevTools helped developers to identify issues related to the client-side of the application while it was being created.

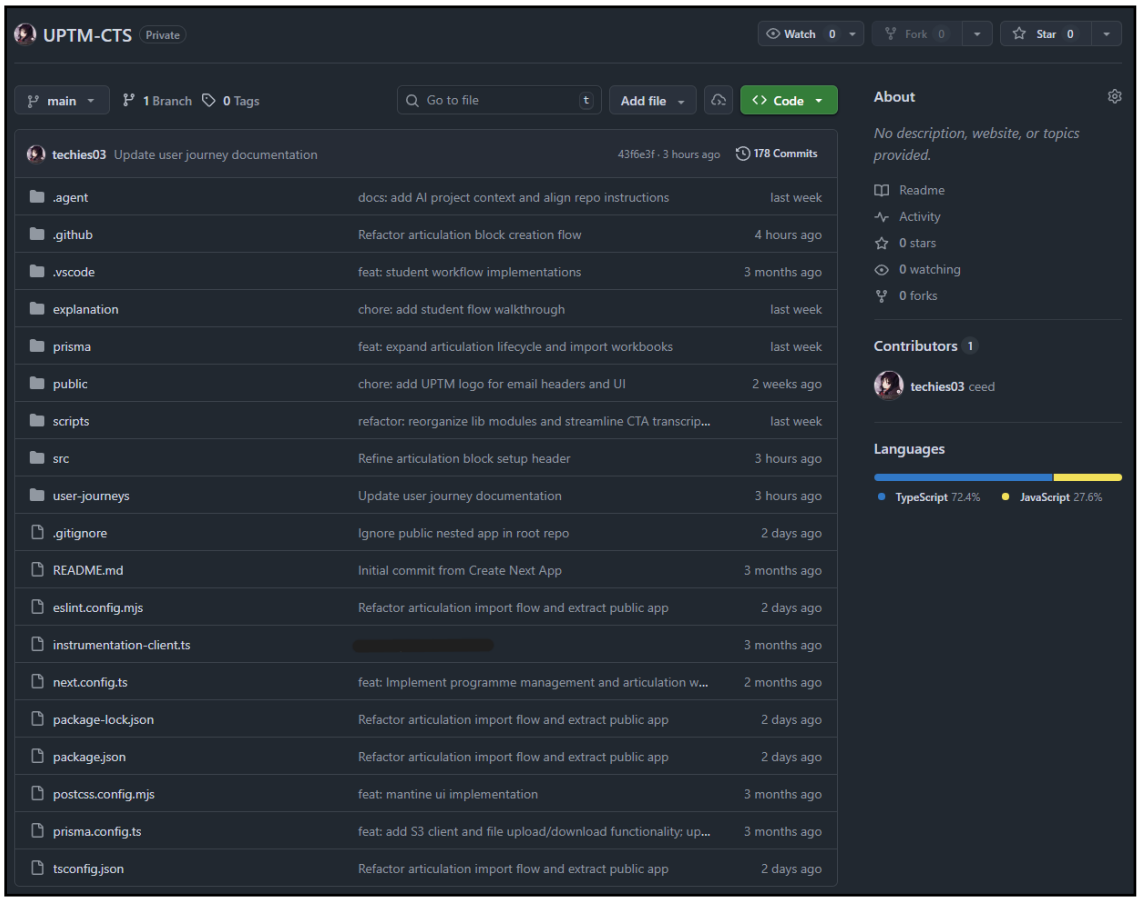


Figure 4.34: GitHub Repository Interface

GitHub is a software platform upon which the source code of the application is hosted. Furthermore, GitHub was used to manage the commits of the code that was developed during the implementation phase of the system. At the time of writing of this report, the code repository on GitHub contains over 170 commits of the codebase of the system. Furthermore, as discussed above, other software platforms, such as Coolify and DigitalOcean, use webhooks to access and pull the code from this repository upon initialization of a webhook request. Thus, the code repository on GitHub is critical to the management and publication live of the software.

4.6.2 Core Languages Used in Implementation

Although the system was built with frameworks like Next.js and React, there are several languages that are used in the implementation of the system.

Table 4.26: Core Languages in Implementation

Language	Use in UPTM CTS
TypeScript	Main language in which the application is written
SQL	SQL Used by the relational database (PostgreSQL) to store the information of the system
HTML	HTML The language of the web pages that are rendered by the browser
CSS	CSS Used to style the pages of the system

TypeScript is the main language that is used in the implementation of the system. All files of the system are written in the language. SQL is also an important language as the data of the system is stored in PostgreSQL databases, though Prisma is used to interact with the databases rather than directly writing SQL code to the databases. Finally, while the code for the web pages is written with React and Mantine components, the pages are written in HTML and styled with CSS code.

4.6.3 Frameworks and Libraries

The following tables and figures illustrate some of the frameworks and libraries that were used in the implementation of the system.

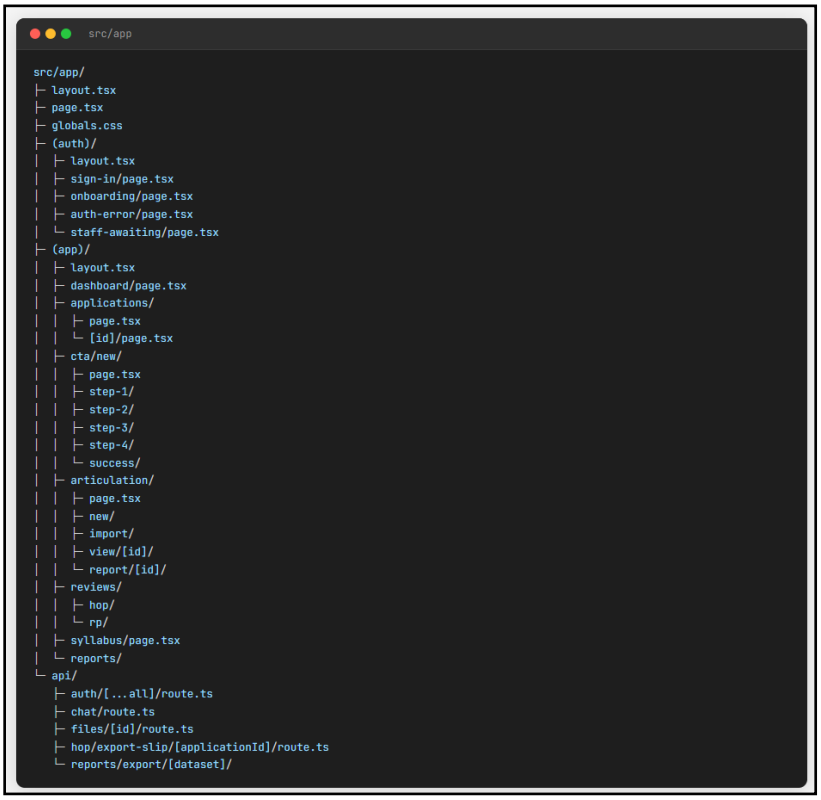
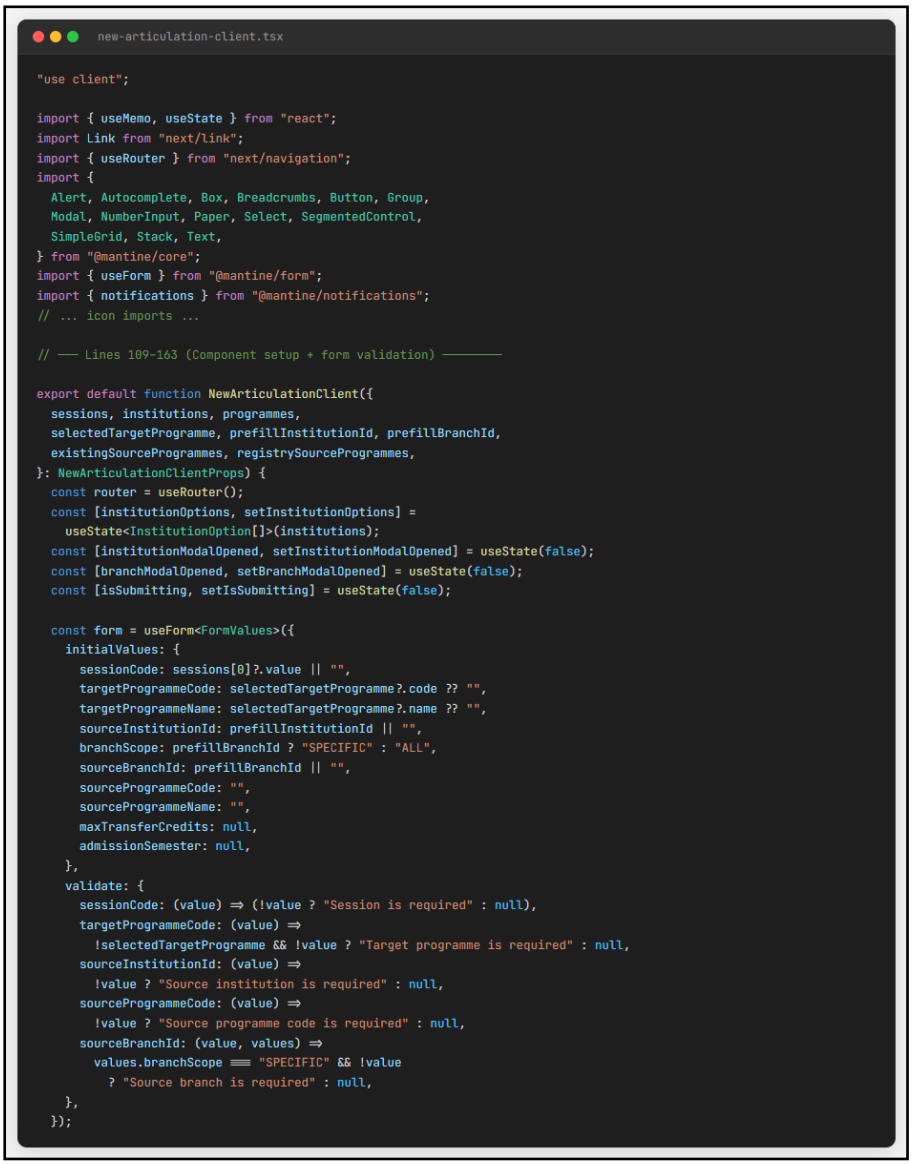


Figure 4.35: Next.js Application Structure

Figure 4.35 shows a simplified structure of the Next.js application. Next.js utilizes an App Router application structure to organize the project. The project starts at the /src/app directory, which contains the main layout of the application, the landing page, and a global styling file. Within this directory are the different route groups for the application; the (auth) route group contains the authentication routes for the system, and the (app) route group contains the application modules after login is performed. Finally, within the project is an “api” directory, which contains route handlers for features like authentication, chat, file access, and export file generation. This directory structure allows for the features of Next.js to be utilized as a full-stack framework in the application; front-end routing, layout, and backend routes are all contained within this Next.js application.



new-articulation-client.tsx

```tsx
"use client";

import { useMemo, useState } from "react";
import Link from "next/link";
import { useRouter } from "next/navigation";
import {
  Alert, Autocomplete, Box, Breadcrumbs, Button, Group,
  Modal, NumberInput, Paper, Select, SegmentedControl,
  SimpleGrid, Stack, Text,
} from "@mantine/core";
import { useForm } from "@mantine/form";
import { notifications } from "@mantine/notifications";
// ... icon imports ...

// ── Lines 109–163 (Component setup + form validation) ──────

export default function NewArticulationClient({
  sessions, institutions, programmes,
  selectedTargetProgramme, prefillInstitutionId, prefillBranchId,
  existingSourceProgrammes, registrySourceProgrammes,
}: NewArticulationClientProps) {
  const router = useRouter();
  const [institutionOptions, setInstitutionOptions] =
    useState<InstitutionOption[]>(institutions);
  const [institutionModalOpened, setInstitutionModalOpened] = useState(false);
  const [branchModalOpened, setBranchModalOpened] = useState(false);
  const [isSubmitting, setIsSubmitting] = useState(false);

  const form = useForm<FormValues>({
    initialValues: {
      sessionCode: sessions[0]?.value || "",
      targetProgrammeCode: selectedTargetProgramme?.code ?? "",
      targetProgrammeName: selectedTargetProgramme?.name ?? "",
      sourceInstitutionId: prefillInstitutionId || "",
      branchScope: prefillBranchId ? "SPECIFIC" : "ALL",
      sourceBranchId: prefillBranchId || "",
      sourceProgrammeCode: "",
      sourceProgrammeName: "",
      maxTransferCredits: null,
      admissionSemester: null,
    },
    validate: {
      sessionCode: (value) => (!value ? "Session is required" : null),
      targetProgrammeCode: (value) =>
        !selectedTargetProgramme && !value ? "Target programme is required" : null,
      sourceInstitutionId: (value) =>
        !value ? "Source institution is required" : null,
      sourceProgrammeCode: (value) =>
        !value ? "Source programme code is required" : null,
      sourceBranchId: (value, values) =>
        values.branchScope === "SPECIFIC" && !value
          ? "Source branch is required" : null,
    },
  });
```

Figure 4.36: React Component

Figure 4.36 shows an example of a React component that exists in the application. React was used as the framework to create the application's user interface components, including those within the articulation creation form. Some of the features of this form that were built with React include the use of React hooks (useState, useForm) to manage and control the form's functionality. React was also used to create all of the forms within the application, including those for syllabus creation, review group creation, comparison creation, status updates, and export file generation.

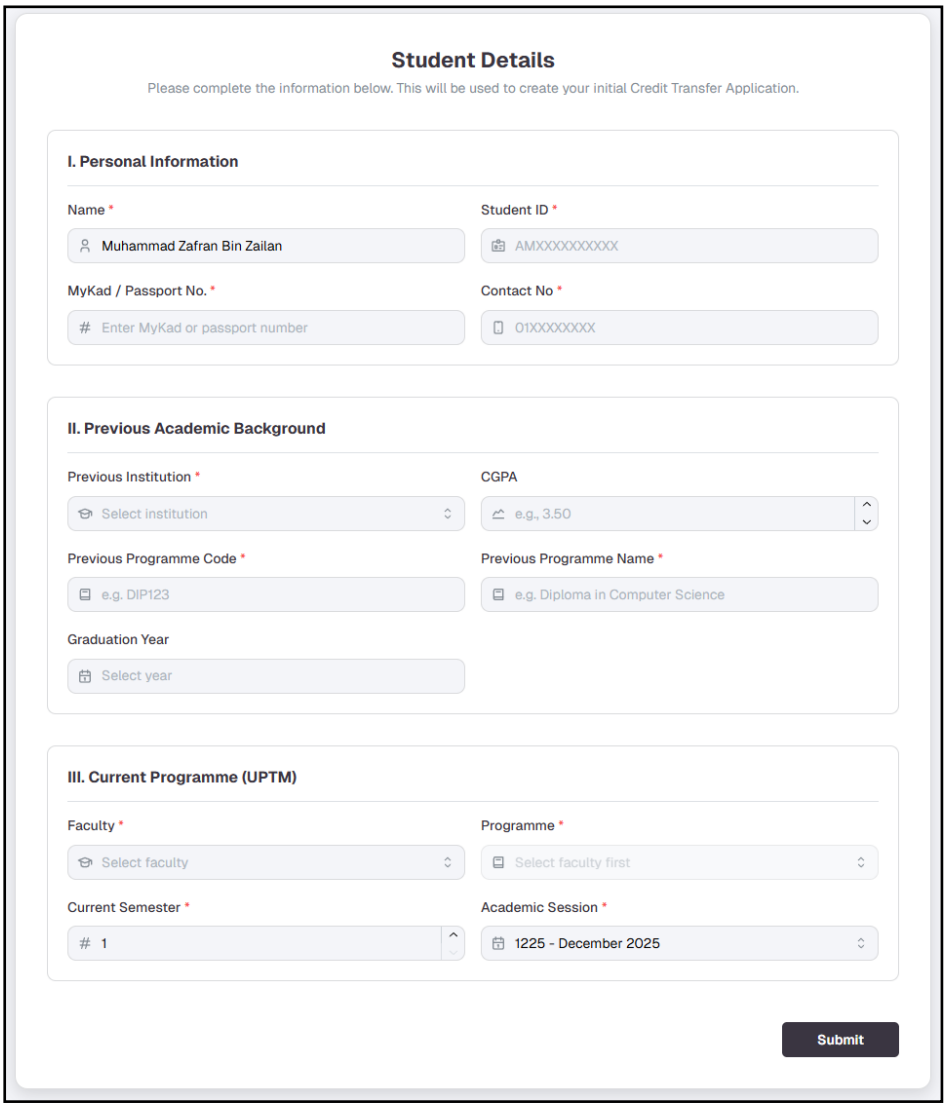


Figure 4.37: Mantine Interface

Figure 4.37 shows an interface for the application that was built with Mantine components. The Mantine UI component library was used to create the user interface components of the application; components included within the UI components are Button, Select, Modal, Autocomplete, Alert, Paper, NumberInput, SegmentedControl, and SimpleGrid. These components were used to create the sign-in screen, the articulation screen, the syllabus screen, and the review screen. Mantine was used to reduce the amount of effort required to create the user interface components from scratch, and to ensure that they are uniform throughout the application.

src/lib/auth/server.ts

```ts
export const auth = betterAuth({
  baseURL: process.env.BETTER_AUTH_URL,

  database: prismaAdapter(prisma, {
    provider: "postgresql",
  }),

  user: {
    additionalFields: {
      role: {
        type: ["STUDENT", "HOP", "RP", "AAS"],
        required: true,
        defaultValue: "STUDENT",
        input: false,
      },
      onboarded: {
        type: "boolean",
        required: true,
        defaultValue: false,
      },
    },
  },

  // Redirect any Better Auth JSON error during browser navigation
  plugins: [/* error-redirect plugin, nextCookies() */],

  socialProviders: {
    google: {
      clientId: process.env.GOOGLE_CLIENT_ID as string,
      clientSecret: process.env.GOOGLE_CLIENT_SECRET as string,

      // Validate Email Domain + Verified Email at the earliest point
      getUserInfo: async (tokens) => {
        const accessToken = (tokens as { accessToken?: string }).accessToken;
        const res = await fetch(
          "https://www.googleapis.com/oauth2/v2/userinfo",
          { headers: { Authorization: `Bearer ${accessToken}` } },
        );
        const profile = await res.json();
        if (!profile.email || !profile.verified_email || !isAllowedEmail(profile.email)) {
          throw new APIError("FORBIDDEN", {
            message: "ONLY_EDUCATION_EMAILS_ARE_ALLOWED",
          });
        }
        return { user: { id: profile.id, name: profile.name, email, ... } };
      },

      // Map Custom Role into the User record
      mapProfileToUser: async (profile) => {
        const email = normalizeAuthEmail(profile?.email ?? "");
        const bypassAssignment = getBypassedAccountAssignment(email);
        if (bypassAssignment) { return { role: bypassAssignment.role }; }
        return { role: "STUDENT" };
      },
    },
  },
});
```

Figure 4.38: Mantine Interface

Figure 4.38 shows the Better Auth configuration used within the application. Better Auth is the authentication system that is used within the application. Better Auth allows for Google OAuth to be used as the system for signing into the application, as well as allows for the domain of the users to be validated during sign-in. The Better Auth configuration allows for custom fields for users to be defined; for instance, roles for users can be either STUDENT, HOP, RP, or AAS, as well as a field to track whether each user has successfully passed onboarding. Additionally, when a user is newly created within the application, they are automatically assigned the STUDENT role within the application. Additionally, certain email

addresses can be defined within a “bypass list” so that when a user attempts to sign in with that email address, they are automatically assigned to a staff role (instead of the default STUDENT role). Finally, the Better Auth service is configured to connect to the Prisma database for storing authentication-related data.

4.6.4 Database and Data Access Tools

The database tools that are utilized by the system include the PostgreSQL database, the Prisma ORM, and the pgAdmin database inspection tool.

prisma/schema.prisma

```
generator client {
  provider   = "prisma-client"
  output     = "../src/generated/prisma"
  engineType = "client" // enable Prisma ORM without Rust
}

datasource db {
  provider = "postgresql"
}

// ==============================
// ENUMS
// ==============================

enum Role {
  STUDENT // default for all users on first login
  HOP     // Head of Program
  RP      // Resource Person
  AAS     // Academic Affairs Students
}

enum ApplicationStatus {
  DRAFT                          // student has not submitted the CTA
  AWAITING_SYLLABUS_UPLOAD       // waiting for student to upload syllabus
  PENDING_HOP_REVIEW             // waiting for HOP to review each courses
  SYLLABUS_COMPARISON_IN_PROGRESS // AI syllabus comparison running
  RP_EVALUATION_PENDING          // waiting for RP to evaluate
  EVALUATION_COMPLETED           // RP has completed the evaluation
  FINALIZED                      // HOP has approved the CTA
  COMPLETED                      // AAS has exported the CTA
  WITHDRAWN                      // student has withdrawn the CTA
}

model ApplicationStatusHistory {
  id            String                  @id @default(cuid())
  applicationId String
  application   CreditTransferApplication @relation(fields: [applicationId], references: [id], onDelete: Cascade)
  status        ApplicationStatus
  actorId       String?  // The user ID who triggered the status change
  actor         User?    @relation(fields: [actorId], references: [id])
  notes         String? // Optional context or reasoning
  createdAt     DateTime @default(now())

  @@index([applicationId])
  @@index([actorId])
}
```

Figure 4.39: Prisma Schema

Figure 4.39 shows the Prisma schema for the application. The Prisma schema defines the data structure that will be used by the database. The database contains 25 models and 18 enums (enumerated values). Examples of the models include tables for users, credit transfer applications (CTAs), articulation

mappings, syllabi, shared review groups, comparison records, and status history. An example of the enums is the ApplicationStatus enum, which contains 9 values that represent the different stages that a CTA may pass through from DRAFT to COMPLETED. Prisma was used to automatically generate a type-safe database client that would access the database according to the schema; all database queries in the application are automatically validated when compiled against the schema.

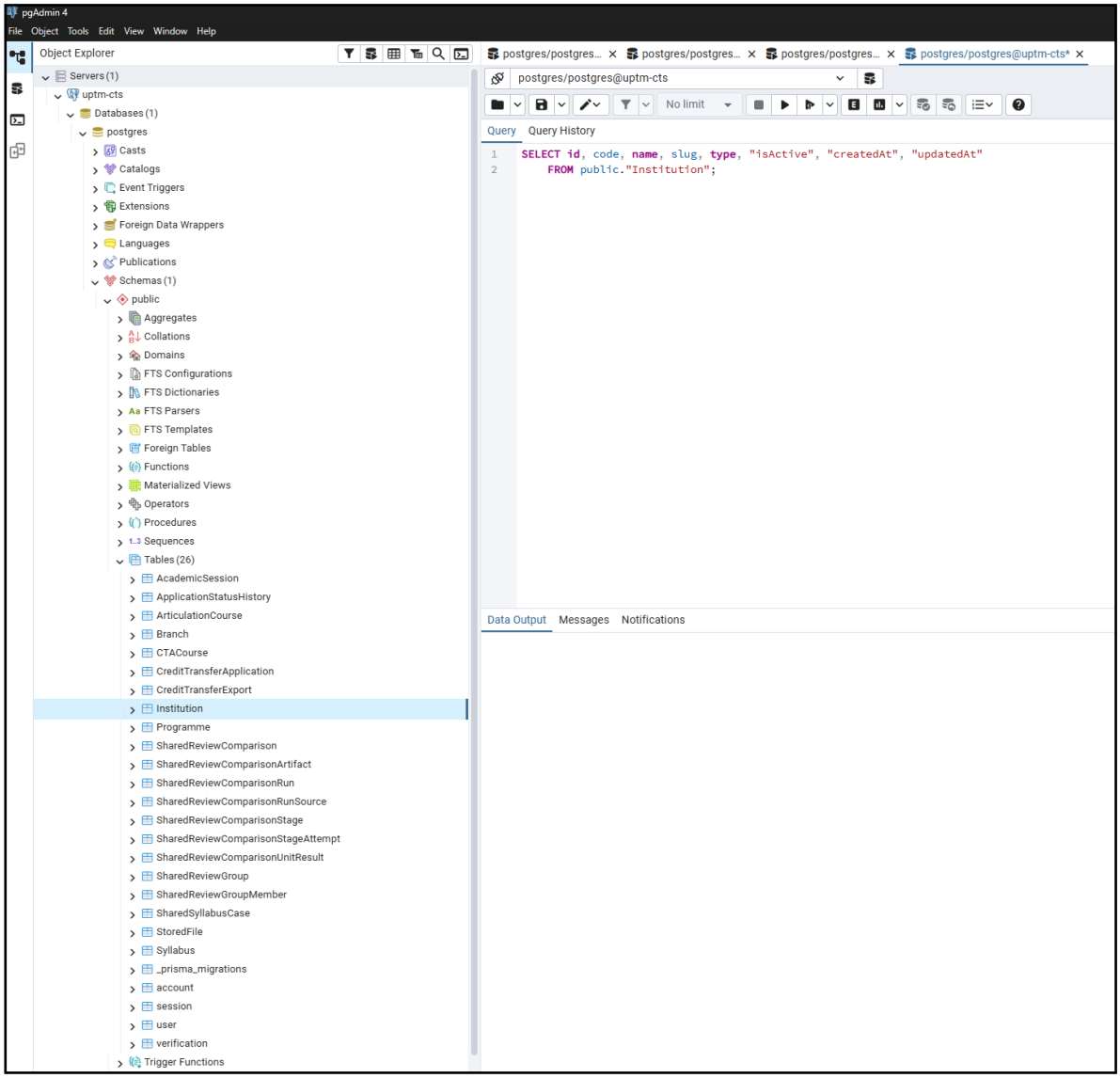


Figure 4.40: pgAdmin Schema View

Figure 4.40 shows the database as viewed through pgAdmin. pgAdmin is the tool that was used to view the PostgreSQL database while developing the application. Using pgAdmin, the tables within the database, their field types, and the records that are stored within those tables could be viewed. For example, using pgAdmin, it was possible to view the tables that were created according to the Prisma schema, view any records that were stored into those tables, and to view any issues with relationships between those tables.

4.6.5 File Storage and Document Processing Tools

In order to allow users to upload files and export files from the application, two main tools were used within the system: SeaweedFS to provide file storage, and Stirling PDF to convert the uploaded PDF files to text.

src/lib/storage/s3.ts

```
import {
  S3Client, PutObjectCommand, GetObjectCommand, DeleteObjectCommand,
} from "@aws-sdk/client-s3";
import { getSignedUrl } from "@aws-sdk/s3-request-presigner";

/** S3 Client configured for SeaweedFS */
export const s3Client = new S3Client({
  endpoint: process.env.S3_ENDPOINT,
  region: process.env.S3_REGION || "ap-southeast-5",
  credentials: {
    accessKeyId: process.env.S3_ACCESS_KEY_ID!,
    secretAccessKey: process.env.S3_SECRET_ACCESS_KEY!,
  },
  forcePathStyle: true, // Required for SeaweedFS and S3-compatible storage
});

export const S3_BUCKET = process.env.S3_BUCKET || "cta-storage";

/**
 * Storage key conventions:
 * - transcripts/{applicationId}/{fileId}.pdf
 * - syllabi/{ctaCourseId}/{fileId}.pdf
 * - exports/{applicationId}.xlsx
 * - articulation-syllabi/{articulationCourseId}/{fileId}.pdf
 */
export const StorageKeys = {
  transcript: (applicationId: string, fileId: string) =>
    `transcripts/${applicationId}/${fileId}.pdf`,
  syllabus: (ctaCourseId: string, fileId: string, ext = ".pdf") =>
    `syllabi/${ctaCourseId}/${fileId}${ext}`,
  export: (applicationId: string) =>
    `exports/${applicationId}.xlsx`,
  articulationSyllabus: (articulationCourseId: string, fileId: string, ext = ".pdf") =>
    `articulation-syllabi/${articulationCourseId}/${fileId}${ext}`,
};

/** Upload a file to S3/SeaweedFS */
export async function uploadFile({
  bucket = S3_BUCKET, key, body, contentType,
}: {
  bucket?: string; key: string; body: Buffer | Uint8Array; contentType: string;
}) {
  const command = new PutObjectCommand({ Bucket: bucket, Key: key, Body: body, ContentType: contentType });
  const result = await s3Client.send(command);
  return { success: true, etag: result.ETag ?? "" };
}
```

Figure 4.41: SeaweedFS Storage Implementation

Figure 4.41 shows the implementation of S3 storage (via SeaweedFS) within the system. Files are stored in a way that indicates the type of file that is being stored. For instance, transcript files are stored within the “transcripts/” folder, syllabi files are stored within the “syllabi/” folder, and export files are stored within the “exports/” folder. Additionally, within each of these files, the files can be accessed through presigned URLs; these URLs provide temporary access to the files without exposing the storage credentials of the system to the user. The forcePathStyle parameter is utilized in the S3 client within code for the system to communicate with SeaweedFS; SeaweedFS utilizes path-style addressing for its object storage system rather than virtual-hosted-style addressing.

src/lib/syllabus/stirling.ts

```ts
/**
 * Stirling PDF Service Utility
 * Converts PDF files to plain text using the Stirling PDF API.
 * Endpoint: POST /api/v1/convert/pdf/text
 */

const STIRLING_BASE_URL =
  process.env.SERVICE_URL_SPDF || "http://spdf-...sslip.io";
const STIRLING_API_KEY = process.env.STIRLING_API_KEY || "";

/** Minimum meaningful words — scanned PDFs produce 0-5; real ones 200+ */
const SCANNED_WORD_THRESHOLD = 20;

// ── Main conversion function ──────────────────────────

export async function convertPdfToText(
  pdfBuffer: Buffer,
  filename: string = "transcript.pdf",
): Promise<StirlingResult> {
  const formData = new FormData();
  const blob = new Blob([new Uint8Array(pdfBuffer)], { type: "application/pdf" });
  formData.append("fileInput", blob, filename);
  formData.append("outputFormat", "txt");

  const response = await fetch(
    `${STIRLING_BASE_URL}/api/v1/convert/pdf/text`,
    {
      method: "POST",
      headers: { "X-API-KEY": STIRLING_API_KEY },
      body: formData,
    },
  );

  // Detect scanned PDF by counting meaningful words
  const wordCount = countMeaningfulWords(text);
  const isScanned = wordCount < SCANNED_WORD_THRESHOLD;

  return { success: true, text, isScanned };
}
```

Figure 4.42: Stirling PDF API Implementation

Figure 4.42 shows the implementation of the Stirling PDF API within the application. The Stirling PDF API allows for uploaded PDF files to be converted into plain text through its /api/v1/convert/pdf/text endpoint. The type of text that is returned from the API is set to plain text rather than markdown text in order to minimize the number of tokens that are used for the text that is sent to Ollama. Additionally, scanned PDF files can also be uploaded to the system; when the PDF is converted to text by Stirling, the system checks the number of “meaningful” words that are returned from the PDF. If the number of meaningful words that are returned by the file is less than a threshold of 20 meaningful words, then the PDF file is considered to be a scanned PDF; in this case, it is sent to a different path using a different AI model (based on vision of the PDF) rather than the model that is used to process text-only PDFs.

4.6.6 AI and Communication Services

Two main services were utilized by the system: AI services to enable the comparison of syllabi, and the use of SMTP to allow the application to send emails to users.

actions.ts / extraction-service.ts

```ts
// Extraction pipeline in CTA Step 2
// 1. Try Stirling PDF → text pipeline (fast path)
// 2. If Stirling fails or PDF is scanned → fall back to vision model

const stirlingResult = await convertPdfToText(buffer, file.name);

if (stirlingResult.success && !stirlingResult.isScanned) {
  // Fast path: Stirling converted the PDF to plain text
  extractionResult = await extractFromText(stirlingResult.text);
} else {
  // Fallback: scanned PDF or Stirling unavailable
  extractionResult = await extractFromImages(images);
}

const MODEL_TEXT =
  process.env.OLLAMA_MODEL_TRANSCRIPT_TEXT || "qwen3:1.7b";
const MODEL_VL =
  process.env.OLLAMA_MODEL_TRANSCRIPT || "qwen3-vl:2b-instruct-q8_0";

export async function extractFromText(markdown: string) {
  return ollama.chat({
    model: MODEL_TEXT,
    messages: [{ role: "user", content: prompt }],
    format: jsonSchema,
  });
}

export async function extractFromImages(images: PageImage[]) {
  return ollama.chat({
    model: MODEL_VL,
    messages: [{
      role: "user",
      content: EXTRACTION_PROMPT,
      images: base64Images,
    }],
    format: jsonSchema,
  });
}
```

Figure 4.43: Transcript Extraction with Ollama

Figure 4.43 shows the use of Ollama to extract the text from a transcript PDF file (as used during CTA Step 2). When a user uploads a transcript PDF file, the transcript is first extracted using the convertPdfToText() function (which uses the Stirling PDF API). If the PDF file is successfully converted to text by Stirling, and the document is not scanned (as determined through the same process as Stirling), then the text from the transcript PDF is sent to Ollama using the extractFromText() function using the “text” model in Ollama. If, however, the PDF file is unsuccessfully converted by Stirling, or if it is scanned, then the same process is followed but with the extractFromImages() function and the “vision” model in Ollama.

src/lib/staged-syllabus-comparison/pipeline.ts

```
const PROMPTS_DIR = path.resolve(
  process.cwd(),
  "scripts",
  "syllabus-staged",
  "prompts",
);

type PipelineStageKey =
  | "PREPROCESS_SOURCE"
  | "PREPROCESS_TARGET"
  | "STANDARDIZE_SOURCE"
  | "STANDARDIZE_TARGET"
  | "COMBINE_SOURCES"
  | "MAPPING"
  | "VERIFICATION"
  | "SCORING"
  | "SUMMARY";

function buildMappingPrompt(promptTemplate, targetItem, sourceItems) {
  return [promptTemplate, "# Target Item To Evaluate", "# Source Items Available As Coverage Evidence"].join("\\n");
}

function buildVerificationBatchPrompt(promptTemplate, cases) {
  return [promptTemplate, "# Cases", "Return a JSON object with a verified array."].join("\\n");
}

function buildSummaryPrompt(promptTemplate, verified, score) {
  return [promptTemplate, "# Verified Results JSON", "# Score Summary JSON"].join("\\n");
}

tracker.startStage({
  stageKey: "STANDARDIZE_TARGET",
  promptTemplate: promptPath("standardize.md"),
  outputArtifactKeys: ["02-target.structured.json"],
});

tracker.startStage({
  stageKey: "MAPPING",
  promptTemplate: promptPath("map.md"),
  outputArtifactKeys: ["03-mapping.json"],
});

tracker.startStage({
  stageKey: "VERIFICATION",
  promptTemplate: promptPath("verify.md"),
  outputArtifactKeys: ["04-verified.json"],
});

tracker.startStage({
  stageKey: "SCORING",
  provider: "CODE",
  outputArtifactKeys: ["05-score.json"],
});

tracker.startStage({
  stageKey: "SUMMARY",
  promptTemplate: promptPath("summarize.md"),
  outputArtifactKeys: ["06-report.md"],
});
```

Figure 4.44: Staged Syllabus Comparison

Figure 4.44 shows the staged comparison of syllabi within the system using the decomposed (decomp) prompting that is used by the application. Rather than creating a long prompt that describes the entire process of comparing the syllabi from two different colleges, the process is decomposed into several separate stages; these stages include preprocessing, standardization, combining the syllabi from the source college, creating a mapping between the syllabi from the source and target colleges, verification of the syllabi mapping, scoring of each of the syllabi, and generating a summary report of the syllabi comparison. Separate prompt templates are used within the system for each of these separate stages; templates like standardize.md, map.md, verify.md, and summarize.md contain the prompts that are used by the LLMs when performing each of these separate stages. These separate stages are used so that the comparison of syllabi can be performed in a manner that allows each stage to focus on only that portion of the syllabus comparison process; each stage also returns its results as a separate JSON file (03-mapping.json, 04-verified.json, 05-score.json, and 06-report.md). Using this separate-stage structure for syllabus comparison, therefore, not only can the syllabi comparison process be automated, but the individual results of each of the separate stages of the process can be individually reviewed.

Figure 4.45: SMTP Configuration

Figure 4.45 shows the SMTP configuration that is used by the application to send emails to users. The system utilizes SMTP to, for instance, request syllabi from the students that submitted the CTAs, to notify resource persons of their assigned syllabi reviews, and to notify the users of status changes to their CTAs. The SMTP server is configured through a series of environment variables; these variables contain information like the host and port of the mail server, the username and password to access the mail server, and the return email address for the application. Additionally, the transporter is created once when the application is started (and then is re-used throughout the application) using Nodemailer's connection pooling feature. Finally, a verifySmtpConnection function is used to verify the SMTP connection to the mail server when the application is started.

4.7 System Interface

The system interface refers to the actual pages that have been developed and implemented into the UPTM Credit Transfer System (UPTM CTS). Unlike the wireframe design, these are the actual pages that were developed for the system. Because there are four different types of users within the system, the interface is divided according to each of those role types in order to more easily provide explanations of each of those pages. Each of the authentication pages are presented first, since each of the user types will interact with those pages when initially using the application.

4.7.1 Authentication Interface

The authentication interface includes the login page, the onboarding page for students, the staff awaiting page, and the sign in page for users with staff roles (HOP, RP, AAS). Each of these pages are shared by the different types of users.

4.7.1.1 Sign-In Page

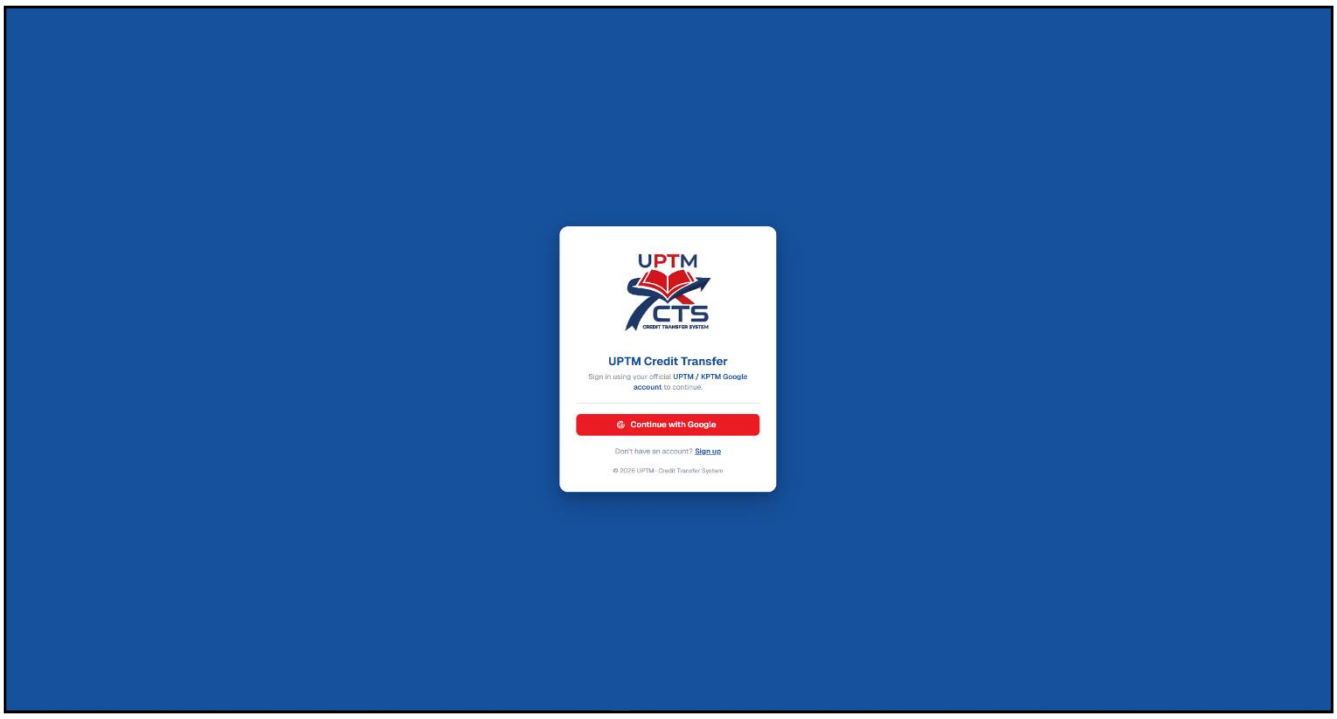


Figure 4.46: Sign-In Page

The sign-in page is the first page that all users encounter. When a user accesses this page, they are presented with a “Continue with Google” button. When a user signs in with their Google account, the backend server checks a few different features of their account. First, it checks if they are a returning student or staff member. If they are a returning student or staff member with a role within the system, they will be redirected to their dashboard. Additionally, first-time student users are redirected to the onboarding page for students to complete their onboarding process. Finally, users with staff roles at UPTM will be redirected to the staff awaiting page. If the sign-in is unsuccessful, the user will be presented with an error message on the sign-in page.

4.7.1.2 Student Onboarding Page

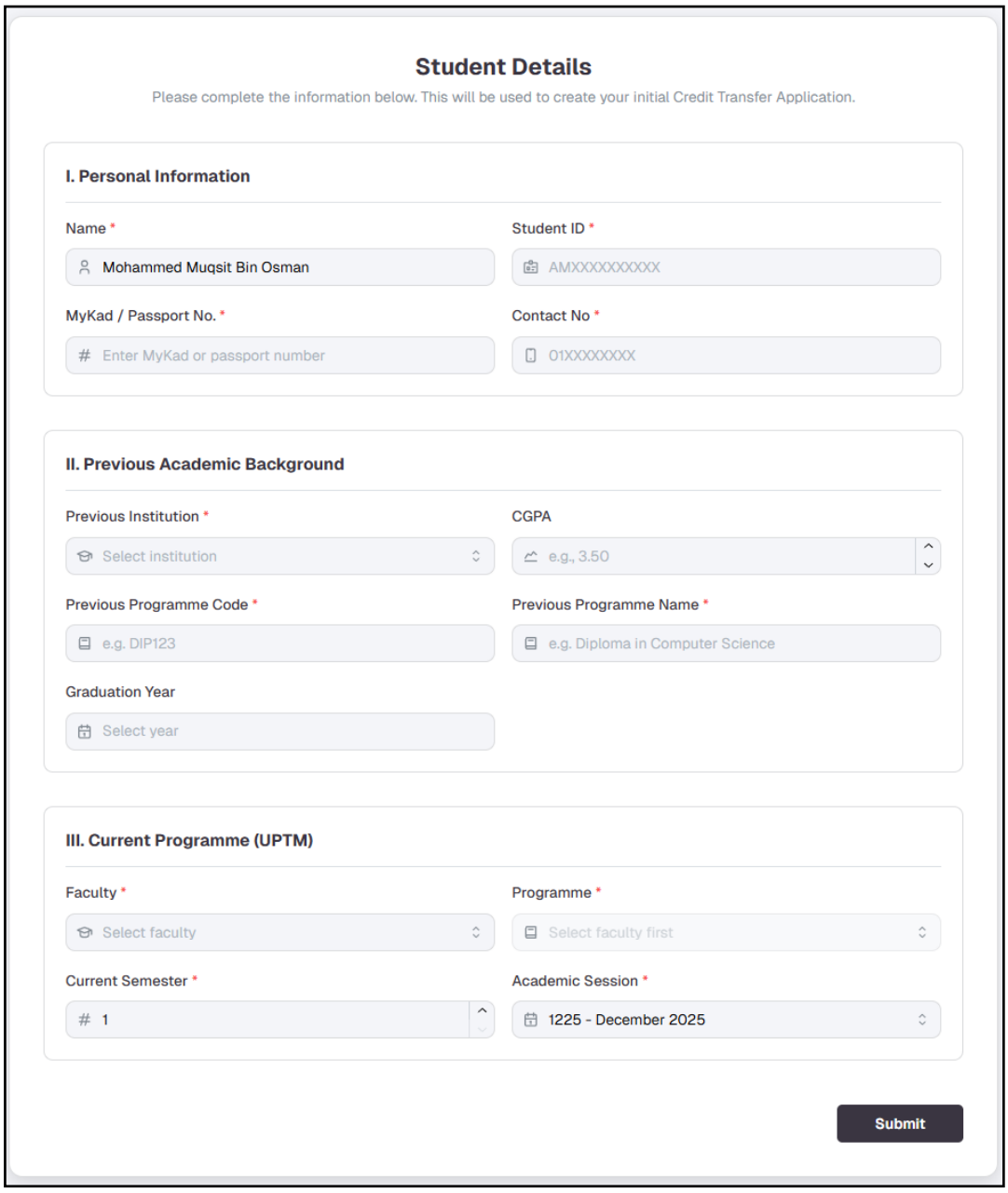


Figure 4.47: Student Onboarding Page

The sign-in page is the first page that all users encounter. When a user accesses this page, they are presented with a “Continue with Google” button. When a user signs in with their Google account, the backend server checks a few different features of their account. First, it checks if they are a returning student or staff member. If they are a returning student or staff member with a role within the system, they will be redirected to their dashboard. Additionally, first-time student users are redirected to the onboarding page for students to complete their onboarding process. Finally, users with staff roles at UPTM will be redirected to the staff awaiting page. If the sign-in is unsuccessful, the user will be presented with an error message on the sign-in page.

4.7.1.3 Staff Awaiting Page

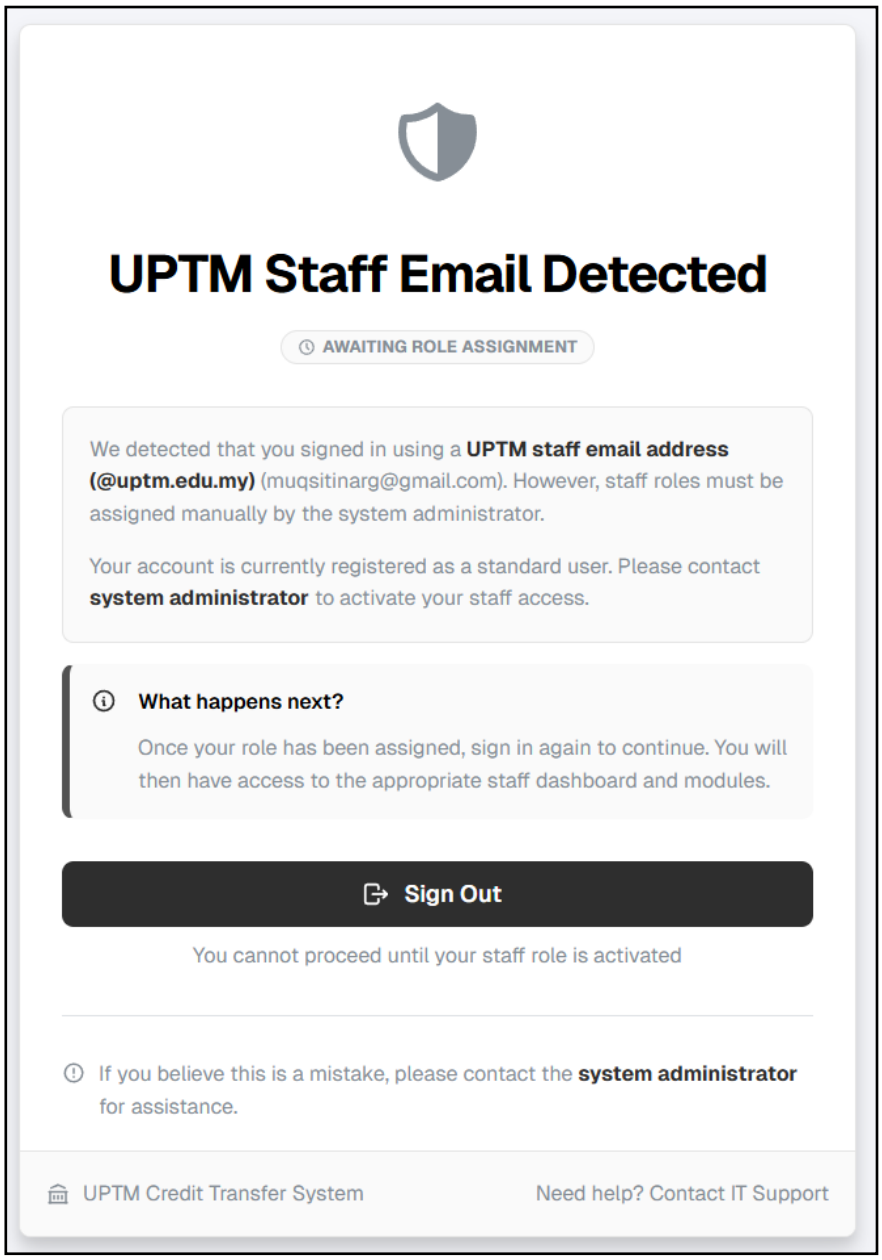


Figure 4.48: Staff Waiting Page

The staff awaiting page is presented to users that attempt to access the system with an email address that belongs to a staff member at UPTM but who have not yet been assigned a staff role within the system. When the staff member signs in for the first time, they will be redirected to this page to inform them that their staff role must be activated by the system administrator. After they successfully sign in for the second time after their role has been activated, they will be automatically onboarded into the system and redirected to their main dashboard.

4.7.2 Student Interface

The student interface includes the main student dashboard and four steps for creating a credit transfer application (CTA). The student interface includes two separate possible flows for the students according to the system. Flow 1 will occur for students whose courses have no syllabus that would need to be uploaded to support the CTA application. Flow 2 will occur for students whose courses may have a syllabus that would need to be uploaded to support their CTA application.

4.7.2.1 Student Dashboard

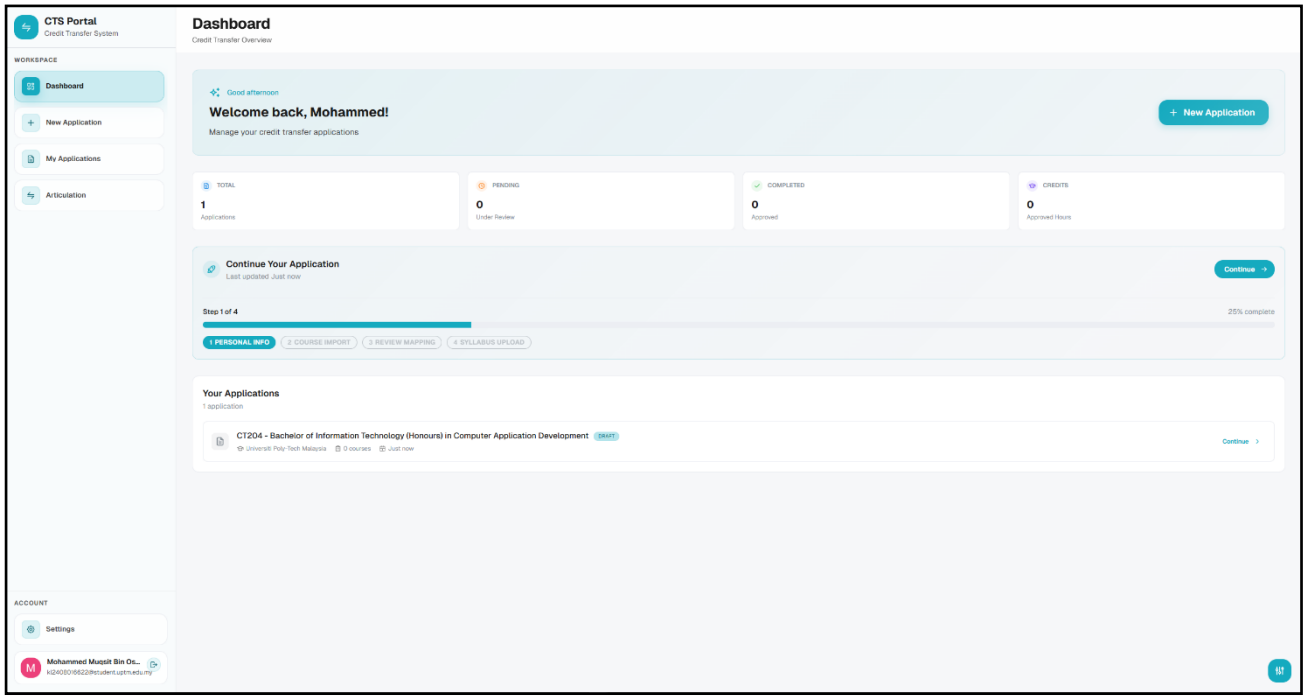


Figure 4.49: Sign-In Page

The student dashboard is the main page for students after they have successfully created their CTA and completed their onboarding process. The dashboard displays information regarding the status of their CTA application, the total number of courses that are included within their application, and any other information regarding their transfer application. Additionally, if their CTA is still in draft status, they will be able to click a button that will allow them to continue along their draft CTA process. If they have submitted their CTA application, the dashboard will change from a data entry page to a monitoring page for that CTA application.

4.7.2.2 CTA Step 1: Application Profile

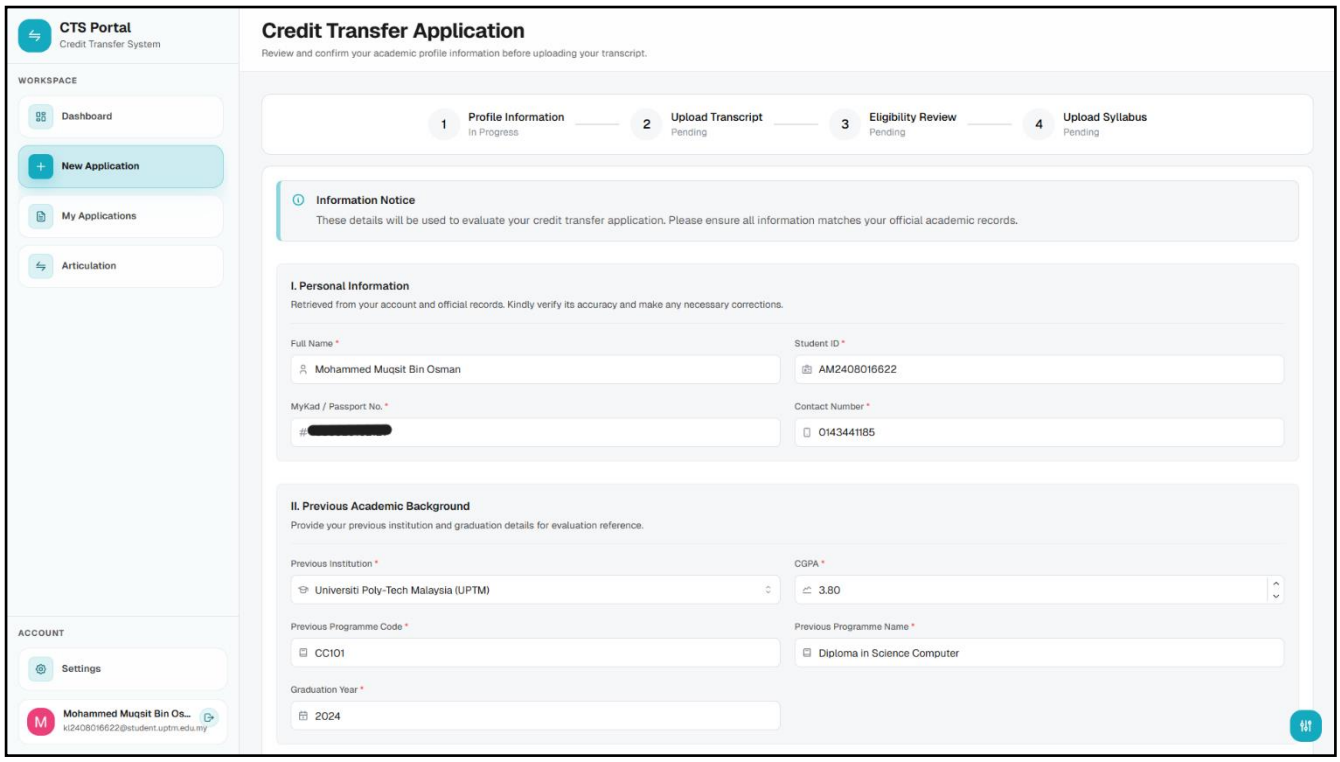


Figure 4.50: CTA Step 1 (Application Profile)

The first step in creating a CTA is creating an application profile for the student. The application profile includes information about the student (such as name, student ID, MyKad number, and contact number), their previous education (such as institution, campus, CGPA, programme code, programme name, and year of graduation), and their current programme within UPTM (such as faculty, programme, academic session, and semester). Additionally, the student will be prompted to save this draft application, or they can continue to the next step in creating their CTA. Any information that is not yet completed will be indicated for the student on this page.

4.7.2.3 CTA Step 2: Transcript Upload and Extraction

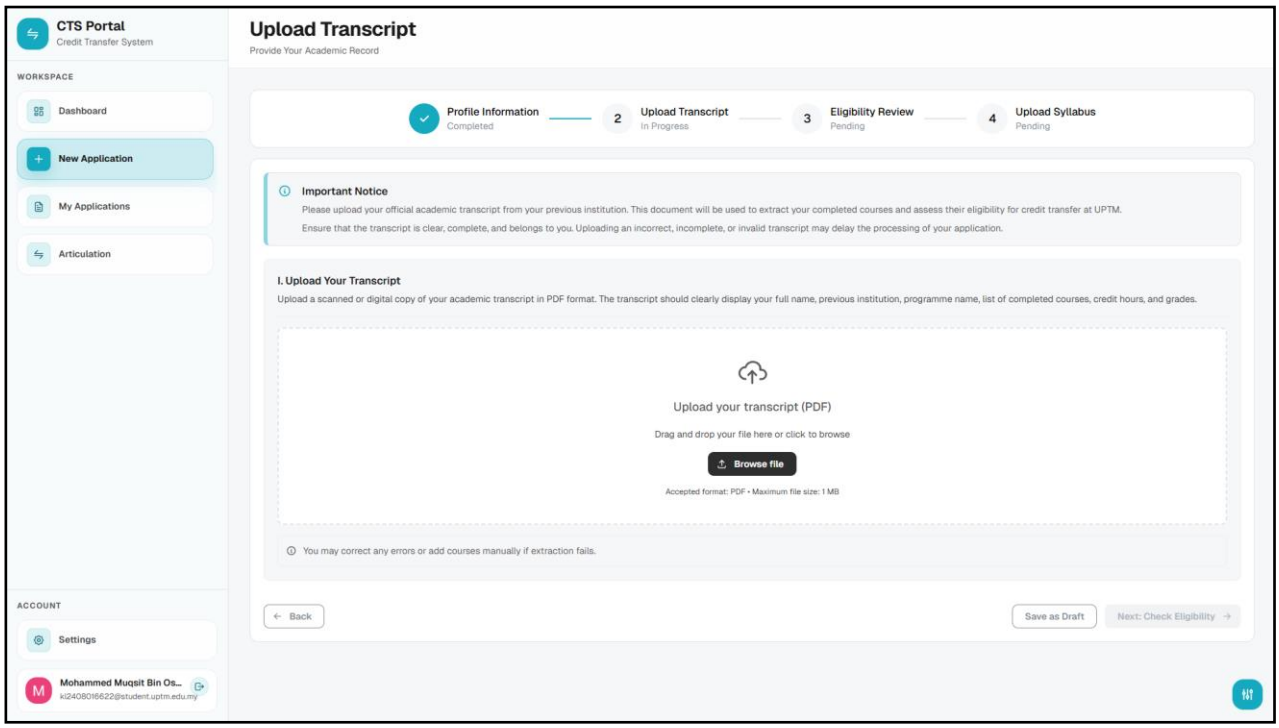


Figure 4.51: CTA Step 2 (Transcript Upload)

The second step in creating a CTA is for the student to upload their transcript (PDF file under 1 MB) into the system.

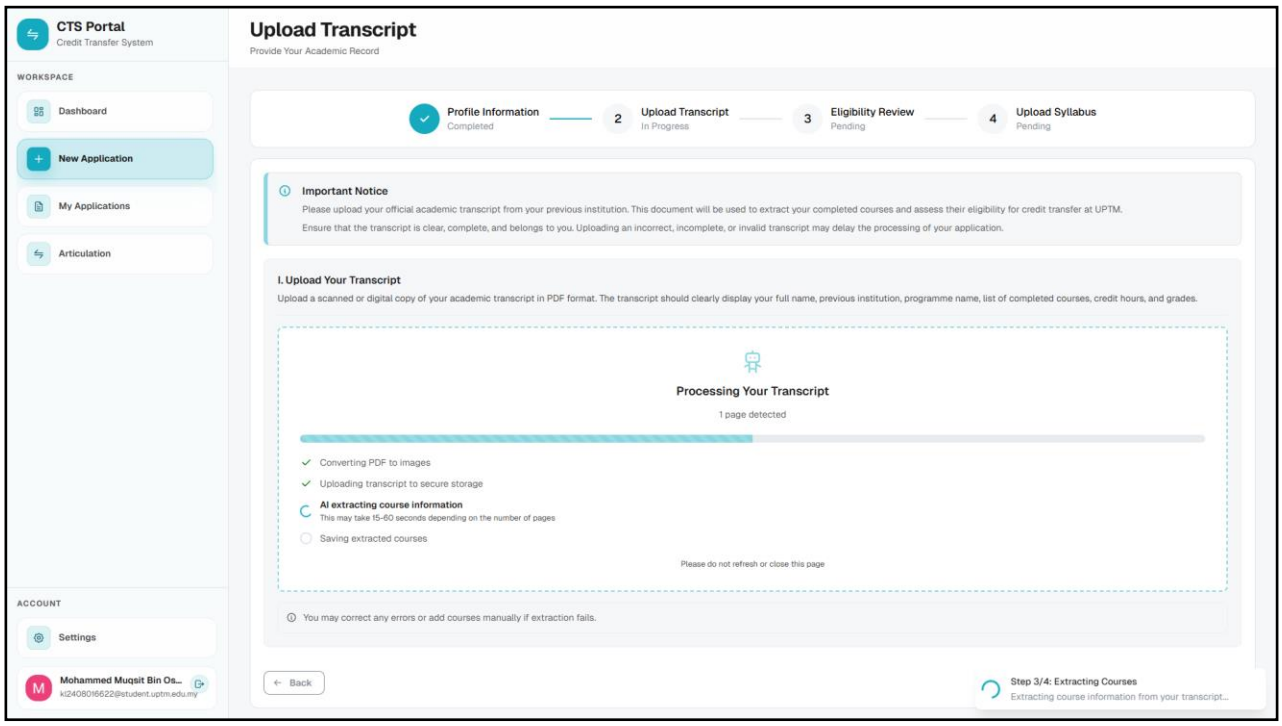


Figure 4.52: CTA Step 2 (Extraction Progress)

Once the transcript has been uploaded, the system will automatically begin extracting the courses from the transcript. The backend utilizes the same four step process to extract the courses from the transcript by utilizing Stirling PDF and Ollama servers to extract the course information. Additionally, if the PDF file that was uploaded by the student is a scanned document, the system will switch to the fallback PDF file process to extract their courses. In this case, the student will be notified that they may want to review the extracted courses to ensure that they are accurate.

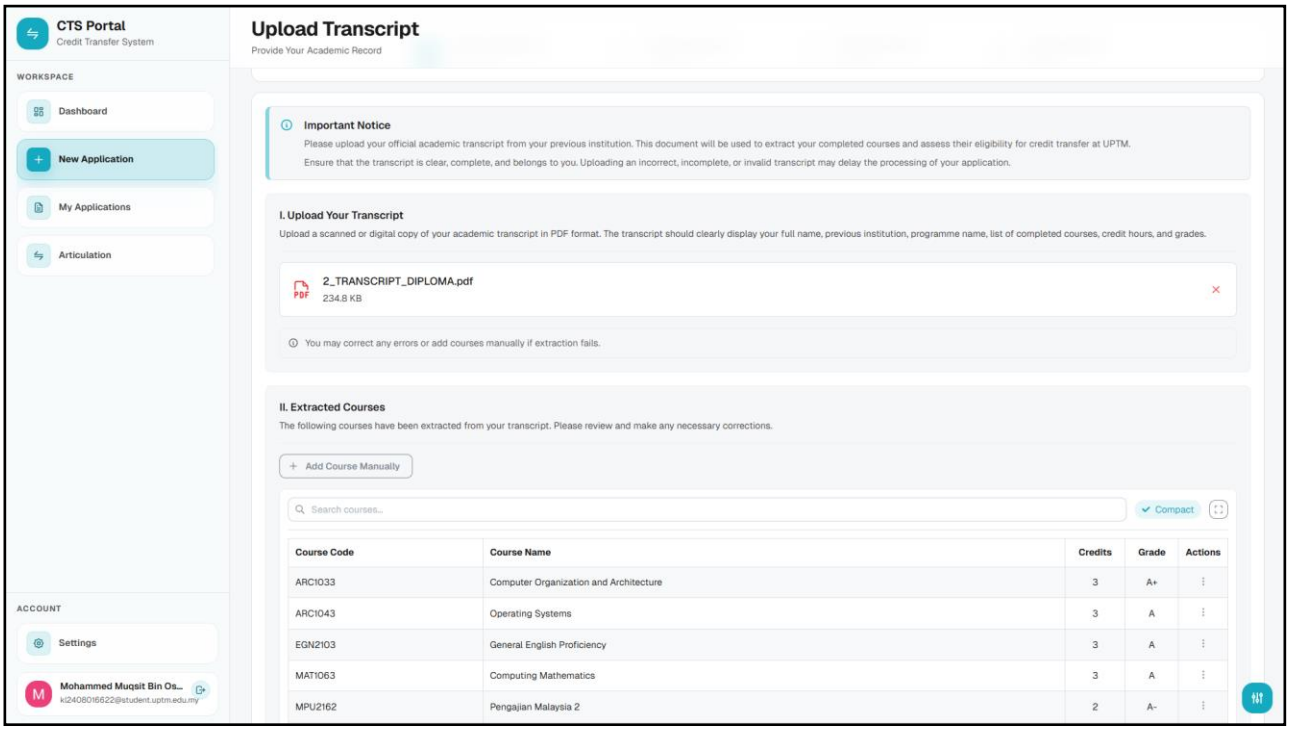


Figure 4.53: CTA Step 2 (Extracted Course Table)

After the courses have been extracted from the transcript, they will be displayed in a table on the screen. The table displays each of the courses that were extracted from the student's transcript. Additionally, there are features within this table that allow the student more control over the courses in their CTA application. For example, the student will be able to search for specific courses by the course code or course name. Additionally, the student will have a button to "Add Course" to allow for the addition of a course that was not extracted from the transcript. Each row within the extracted course table will have an action menu that allows for each course to be edited or deleted. Additionally, the student has the ability to adjust the density of the rows in the table or view the extracted courses in fullscreen viewing mode.

Additionally, since the transcript will be uploaded into the system during this step, it is also possible for that transcript to be removed from the CTA being created. Additionally, even if the courses were automatically extracted from the transcript, there is still the ability for the student to

continue with creating their CTA without uploading their transcript, as they may choose to manually add courses to their CTA application.

4.7.2.4 CTA Step 3: Course Review and Eligibility

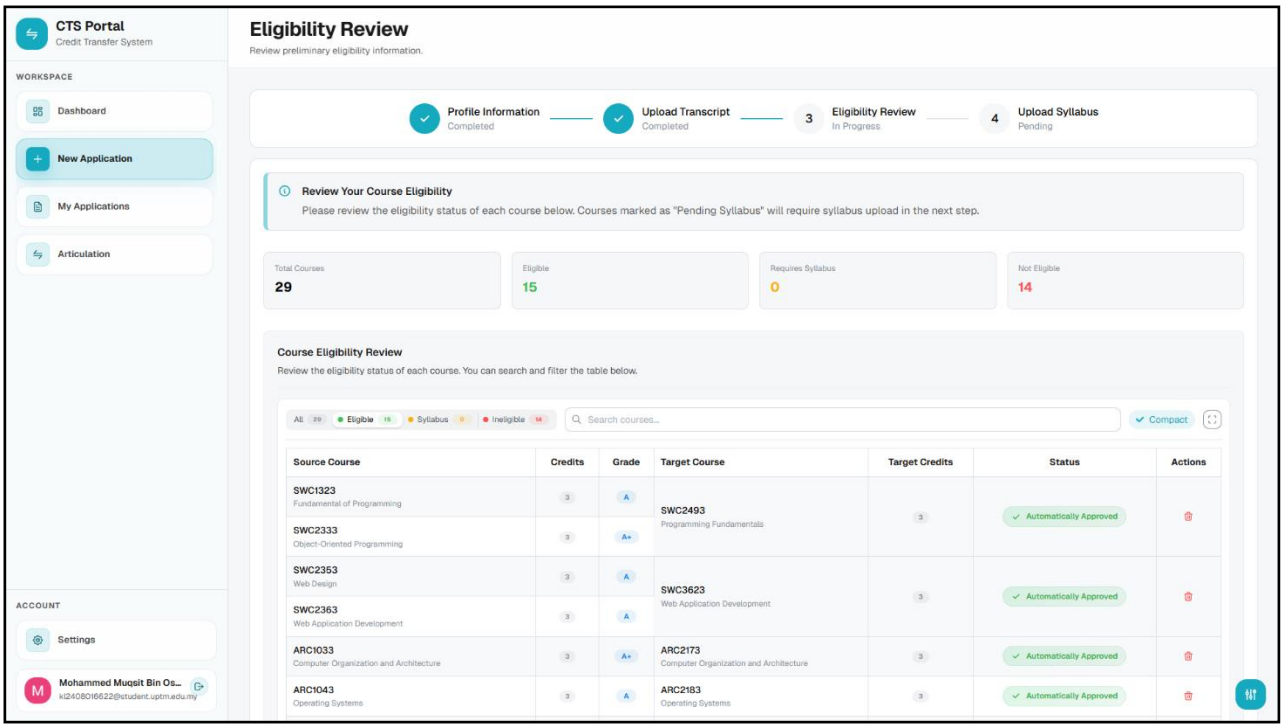


Figure 4.54: CTA Step 3 (Course Review - Flow 1: No Syllabus Required)

The third step in creating a CTA is reviewing the courses that are to be included within the student's CTA application to ensure that they meet the eligibility requirements to be transferred to their desired programs at UPTM. The courses will be automatically grouped into one of the following categories: courses that have been automatically approved (they passed all grade and credit hour checks), courses with a grade below a "C" (they were rejected automatically due to a grade below a "C"), courses with below the required credits (they were rejected automatically due to the credits from their transcript were lower than the credits required for the programs that they wish to attend at UPTM), and courses that did not meet any of the criteria for inclusion within the CTA application. Additionally, any courses that are automatically approved or accepted by the system will be eligible to be submitted for review by the HOP. In this case, after clicking the "Submit" button, the CTA will have a status of "PENDING_HOP_REVIEW".

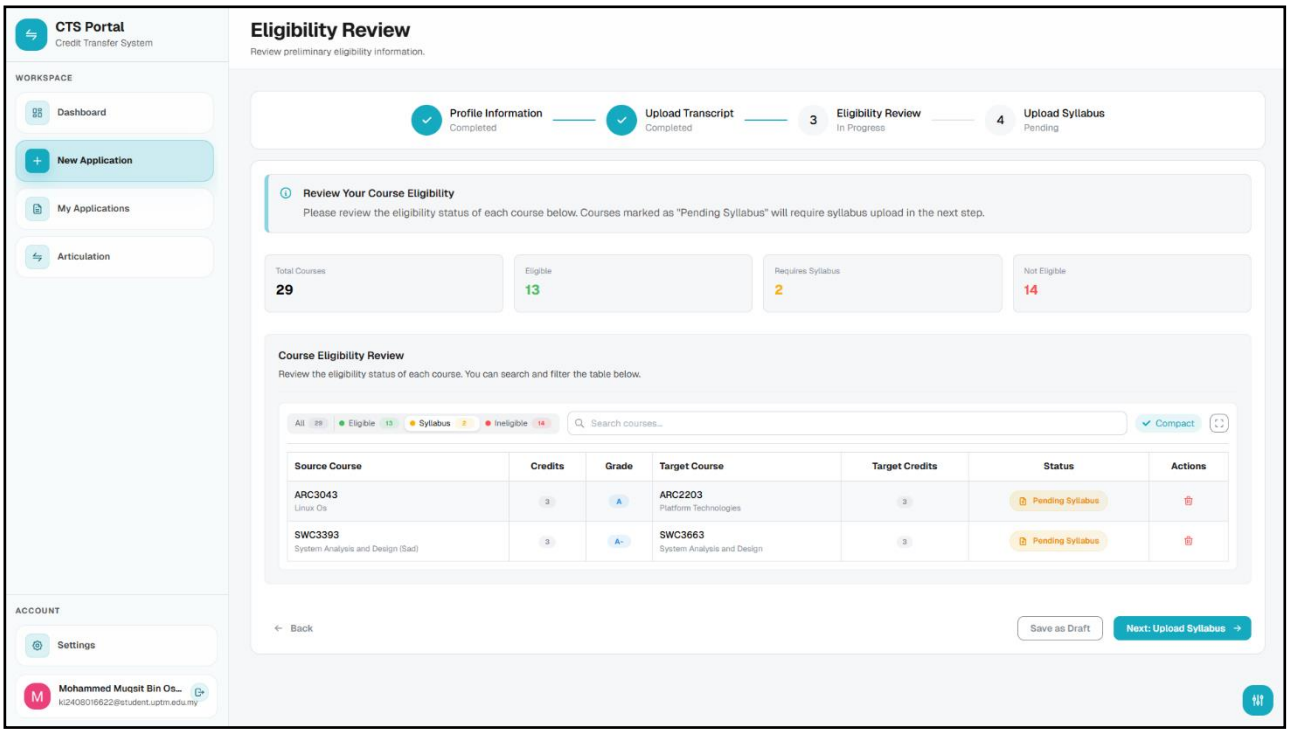


Figure 4.55: CTA Step 3 (Course Review - Flow 2: Syllabus Required)

Similar to the first flow, the courses will be automatically grouped into one of the following categories: courses that have been automatically approved (they passed all grade and credit hour checks), courses with a grade below a “C” (they were rejected automatically due to a grade below a “C”), courses with below the required credits (they were rejected automatically due to the credits from their transcript were lower than the credits required for the programs that they wish to attend at UPTM), and courses that did not meet any of the criteria for inclusion within the CTA application. Additionally, any courses that are automatically approved or accepted by the system will be eligible to be submitted for review by the HOP. In this case, however, it is also possible for some courses to fall into the “Pending Syllabus” category in addition to the other categories. For instance, the courses that are in the “Pending Syllabus” category are those that may require proposed or manual articulation records for those courses to be eligible for review by the RP; in which case, these courses will have to be uploaded during the next step in the process. Thus, a student whose courses fall into this category will not be able to click the “Submit” button during this step within the process, but will be sent to Step 4 to upload these syllabi.

4.7.2.5 CTA Step 4: Syllabus Upload

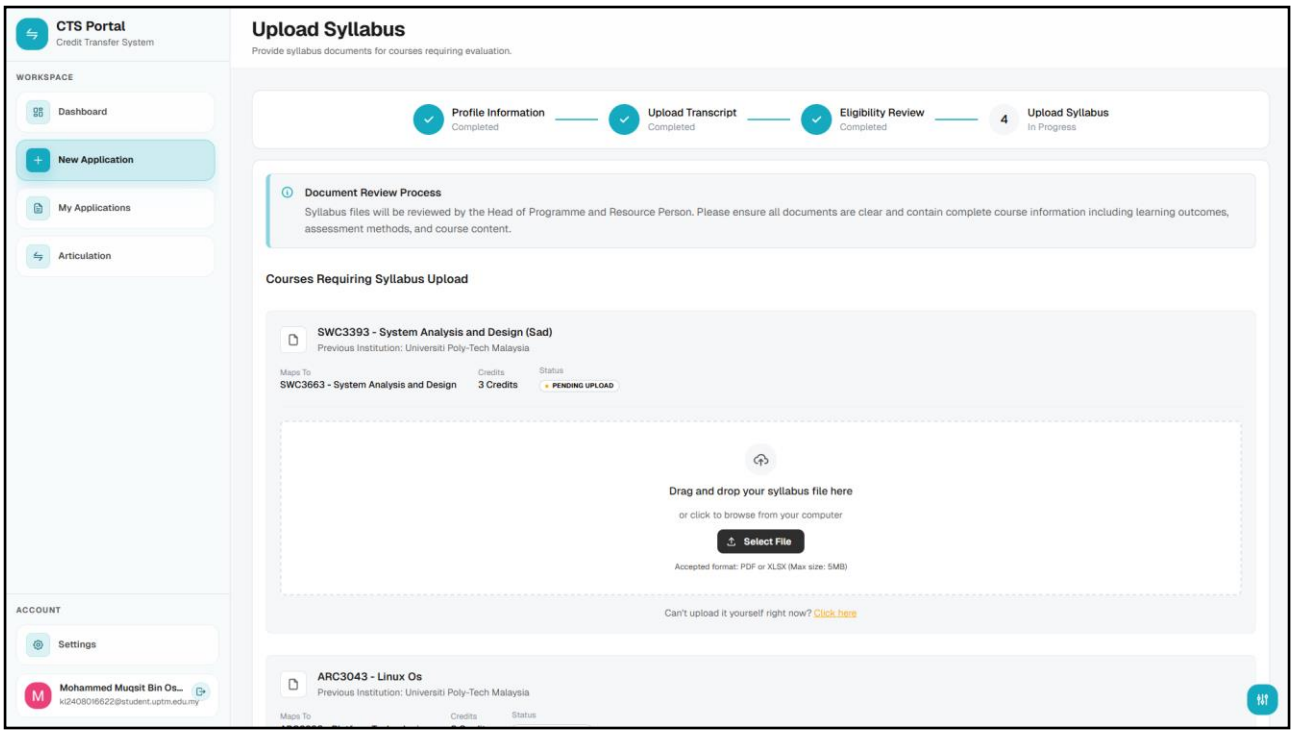


Figure 4.56: CTA Step 4 (Syllabus Upload)

Figure 4.56 shows Step 4 of the CTA creation process. This page appears only when the application contains flagged courses that still require syllabus evidence. The page lists each affected course separately and provides upload or status controls for that course. The implemented interface accepts PDF or XLSX syllabus files, with a maximum file size of 5 MB.

For each course, the student has the following options:

1. Upload Syllabus - Upload a PDF or XLSX source syllabus for that course. If the uploaded workbook contains multiple worksheets, the student must first choose the worksheet that should be used.
2. Upload Later - Mark the course as a future upload. The system creates a placeholder and assigns a 14-day deadline. The overall application can remain in AWAITING_SYLLABUS_UPLOAD.
3. Not Available - Record that the syllabus is currently unavailable. This option also creates a placeholder and applies the same deadline-based follow-up logic.
4. Shared Upload Pending - Link the course to a shared syllabus case so that another student's representative upload may satisfy the syllabus requirement for multiple related courses.

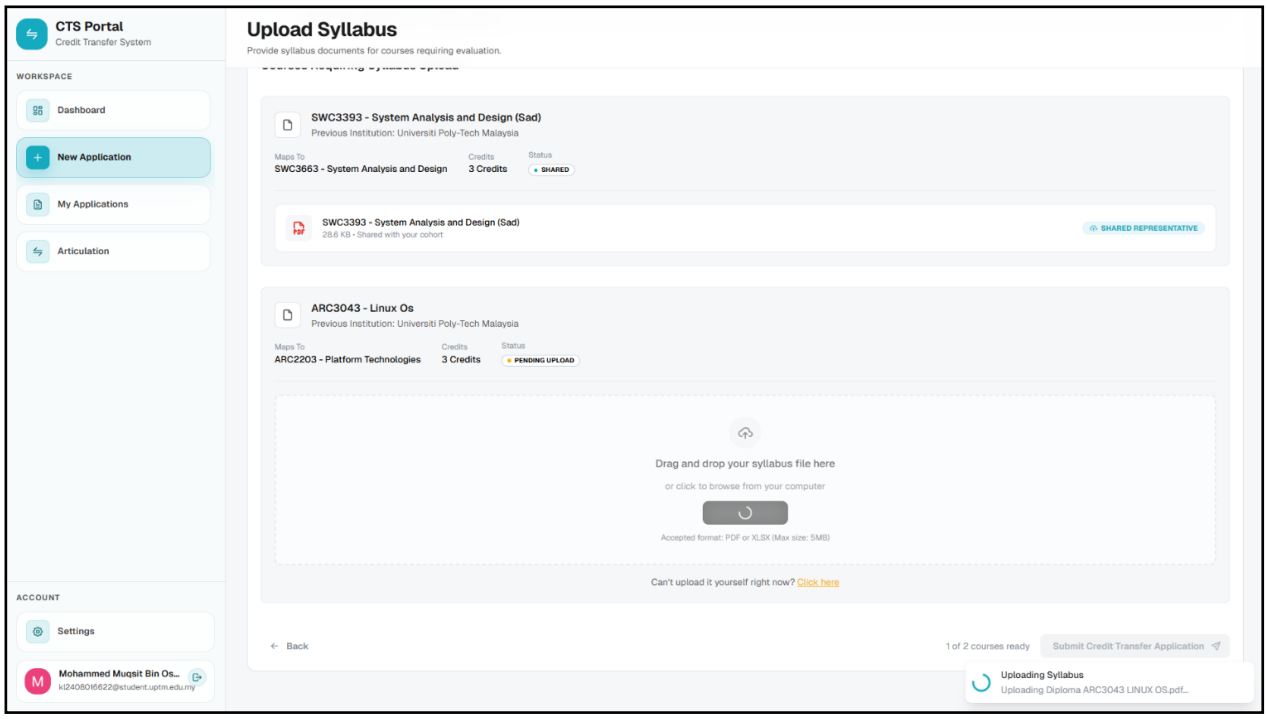


Figure 4.57: CTA Step 4 (Syllabus Upload Progress)

Figure 4.57 shows the upload-progress state for an individual course in Step 4. When the student uploads a syllabus file, the system stores the file in SeaweedFS and creates a syllabus record linked to that course. The page then updates the course state in place. Courses with a completed upload display a completed status and file information, while unresolved courses continue to show the upload and selection controls.

Once every flagged course has been handled through one of the available options, the student can submit the application. The resulting application status depends on the overall state of those courses. If all required syllabi have been uploaded, the application moves to PENDING_HOP_REVIEW. If some courses are still pending, the application remains in AWAITING_SYLLABUS_UPLOAD.

4.7.2.6 Submission Success Page

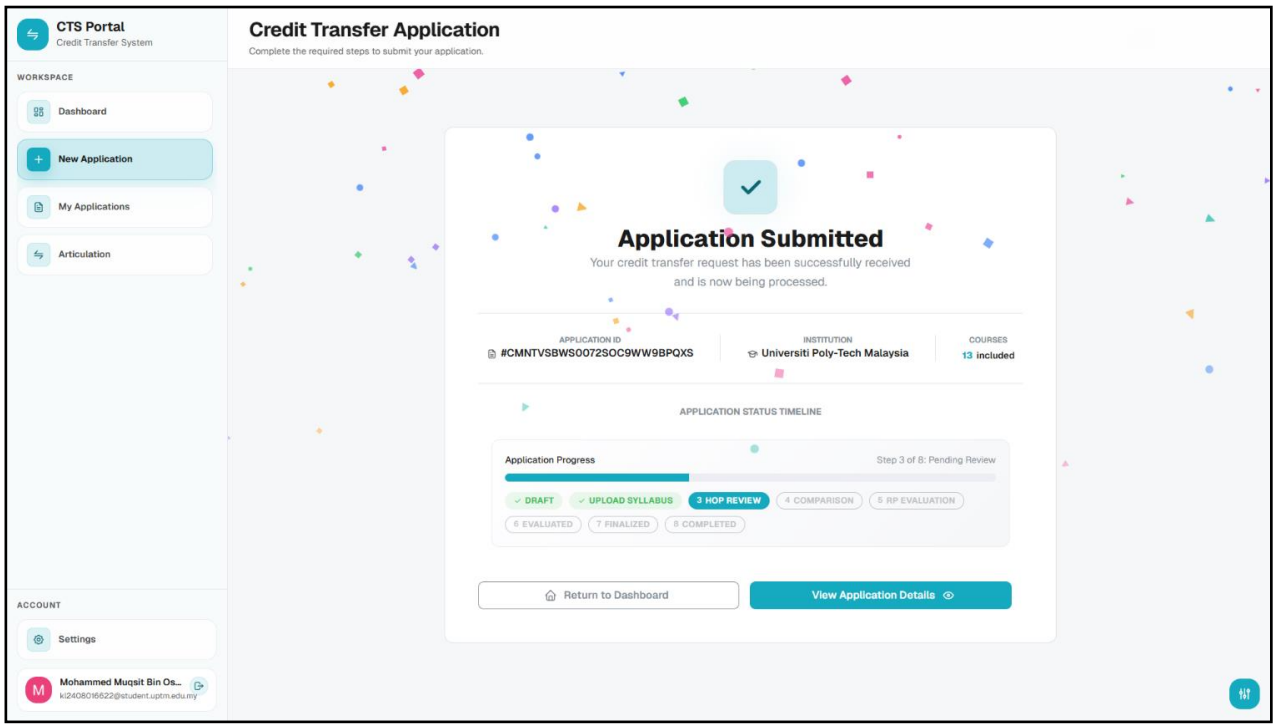


Figure 4.58: Submission Success Page

The flowchart in Figure 4.58 represents the success page after the student submits their application. The success page contains messages and animations to congratulate the student, followed by a compact application summary that includes the application ID, the institution from which the student is transferring, and the number of courses to be transferred. Following the application summary is a timeline that demonstrates the status of the student's application. For Flow 1 (applications without syllabus requirements), the timeline includes the following stages: Draft > HOP Review > Finalized > Completed. For Flow 2 (applications that require syllabi to be uploaded), the timeline includes the following stages: Draft > Upload Syllabus > HOP Review > Comparison > RP Evaluation > Evaluated > Finalized > Completed. Finally, the success page includes two action buttons: "Return to Dashboard" and "View Application Details".

4.7.2.7 Application List and Detail

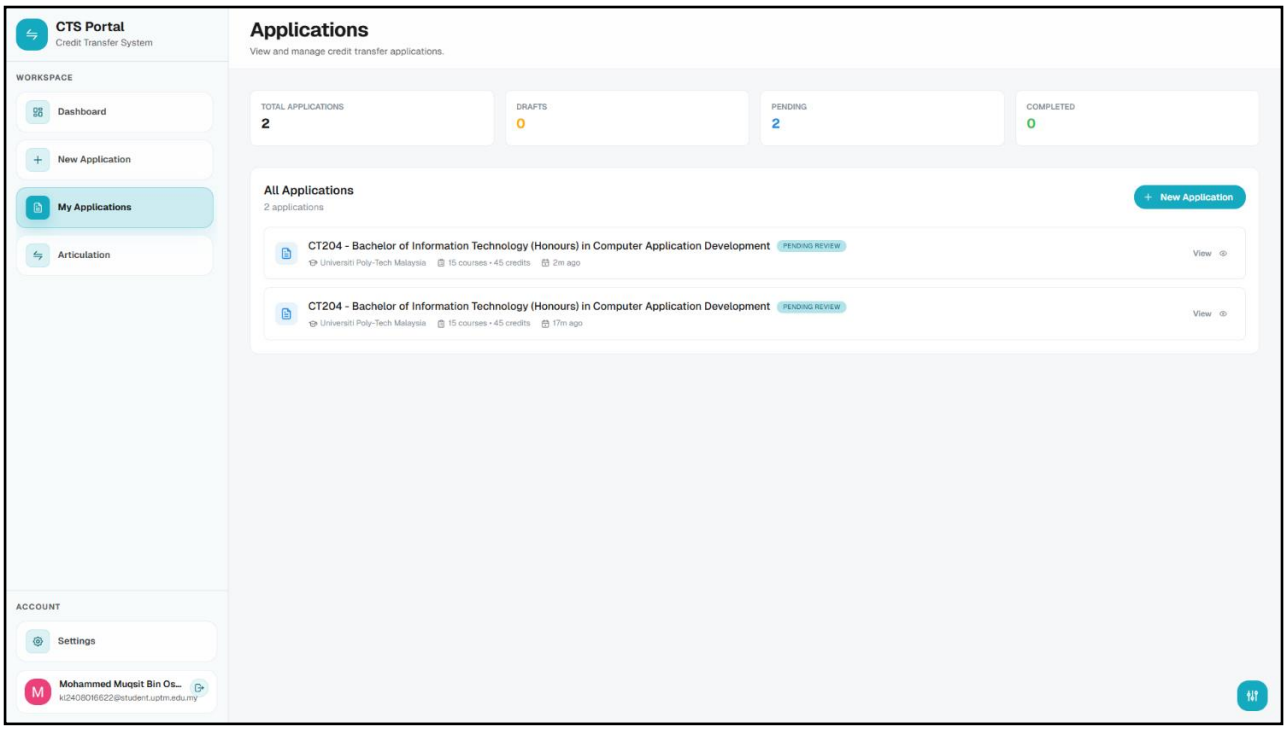


Figure 4.59: Application List Page

Figure 4.59 shows the application list page. The application list page displays an overview of the student's application, including its status, its source institution, the date of its submission, and the number of courses within the application. Because each student can only have one active application at a time, the application list page only includes one record of their submitted application.

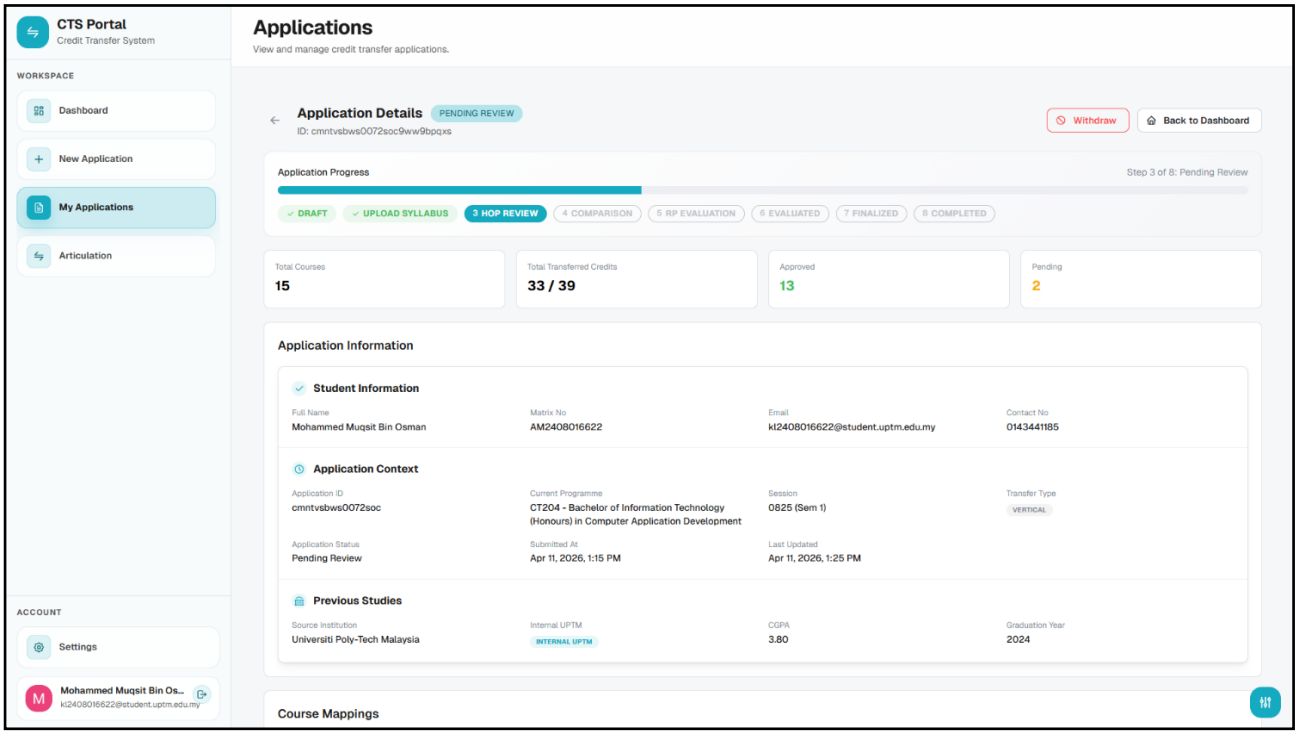


Figure 4.60: Application Detail Page

Figure 4.60 represents the detail page for the student's application. This page includes several separate sections that contain the details of the student's application:

1. Application Summary - includes information about the application, such as the application ID, the status of the application, the student's source institution, the target programme from which the student intends to transfer into, and the date on which the application was submitted;
2. Course List - lists the courses that the student intends to transfer, including information about each of the student's source courses, the target courses within the students' target programme, the credit values of the courses, the student's grade within each course, and the eligibility status of each of the student's courses to be transferred;
3. Uploaded Documents - displays the transcript that the student uploaded along with their application, and any syllabus files that were uploaded;
4. Effective Syllabus - for applications that include the syllabus requirement (Flow 2), displays the syllabus that was used to support the student's application;
5. Status History - displays a timeline of each status through which the application has passed during its review process;
6. Export Slip - once the student's application is in the COMPLETED status within the application, a button becomes available for the HOP to click in order to generate a slip that details the student's transferred credits; the slip can be previewed prior to being downloaded by the HOP;

In instances in which the student wishes to withdraw their application prior to completion, a "withdrawal" button is displayed on the application detail page. When clicked, the system will open a dialog box that requires the student to type a reason for the withdrawal of their application. Additionally, the student is required to input the application ID within the dialog box. Upon submitting the withdrawal request, the application will change to the WITHDRAWN status within the system.

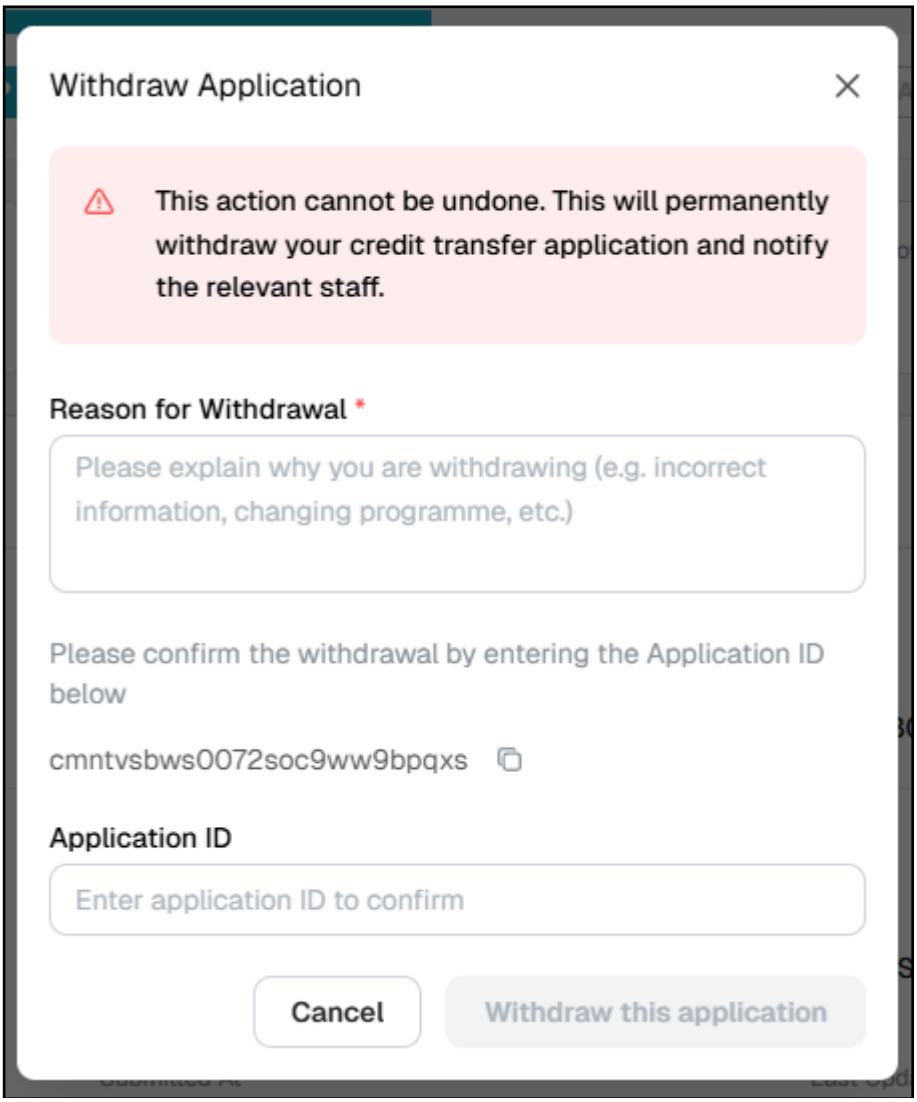


Figure 4.61: Withdrawal Confirmation

Figure 4.61 depicts the withdrawal confirmation dialog box. When the student attempts to click the “withdraw” button on the application detail page, this dialog box will appear. Within the dialog box, the student is required to type in the reason for the withdrawal of the student’s application. Additionally, the student will be required to type the application ID of the application to be withdrawn. Upon entering this information, the student will have to click the “withdraw” button in order to confirm the withdrawal of their application.

4.7.3 HOP Interface

The HOP interface is the main workspace for the HOP within the system. HOPs are responsible for reviewing the student applications that arrive in their inbox, approving the courses to be transferred by the student, assigning an RP to evaluate the syllabi of the student submitted courses (if required for the student's application), and approving the HOP review of the application by determining whether the courses can be transferred to the student's target programme within UPTM. There are two main workflows within the system for HOPs to follow:

1. CTA-driven workflow: HOPs are primarily responsible for reviewing the applications that are submitted by the student. As such, HOPs must review the courses that are to be transferred, approve or reject those courses, assign an RP to evaluate the syllabi of those approved courses, edit the student applications if necessary, and finally, approve or reject the application of the student as a whole.
2. Articulation-initiated workflow: In addition to reviewing applications submitted by students, HOPs are also responsible for the management of the articulation repository within the system independently of the applications that are submitted by students. The HOP can create, import, edit, and export articulation rules within the system; they can also initiate a comparison of the syllabi from courses that are in a student's application.

The following subsections describe each of the modules that is available to the HOP user within the system.

4.7.3.1 HOP Dashboard

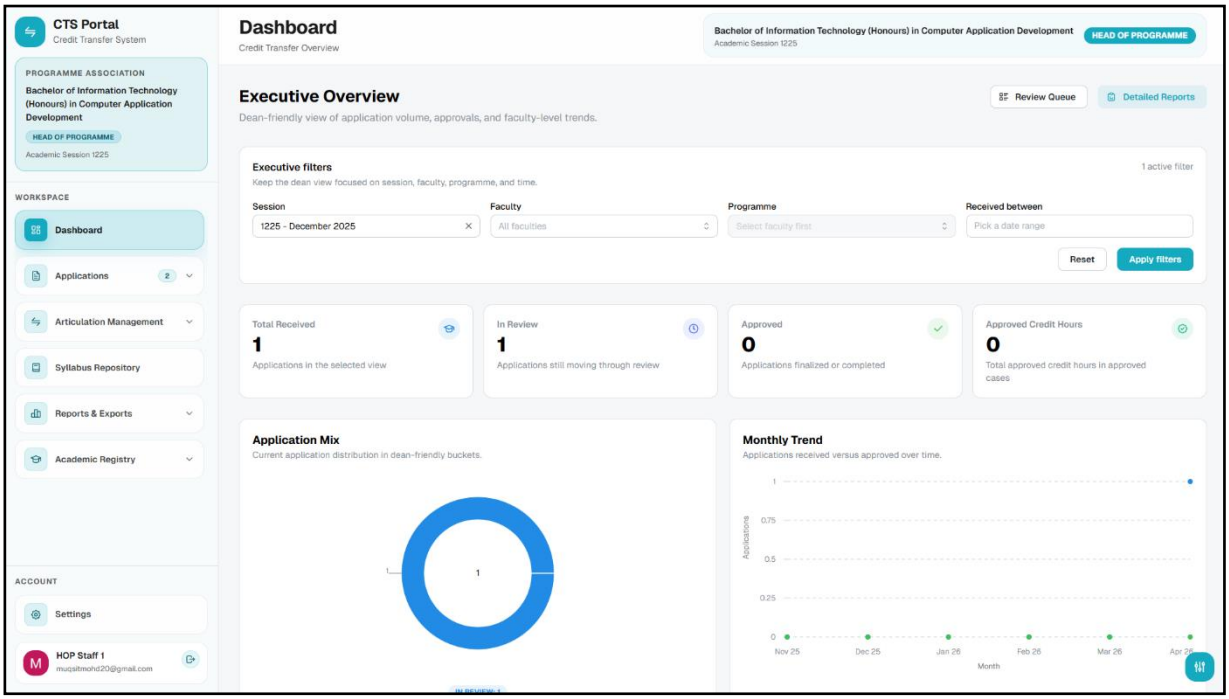


Figure 4.62: HOP Dashboard

Figure 4.62 represents the HOP dashboard. The HOP dashboard is displayed following the user's successful sign-in into the system. The HOP dashboard displays summary information about the HOP's current workload. Specifically, the dashboard will display the total number of active applications, the number of applications that are waiting to be reviewed by the HOP, the number of applications that have been sent to another stage within the application review process (after being approved by the HOP), and a summary report of the HOP's system.

The HOP dashboard also includes links to the following main modules within the system: the review queue, the articulation list, the syllabus repository, the academic registry, and the reports module.

4.7.3.2 Review Queue

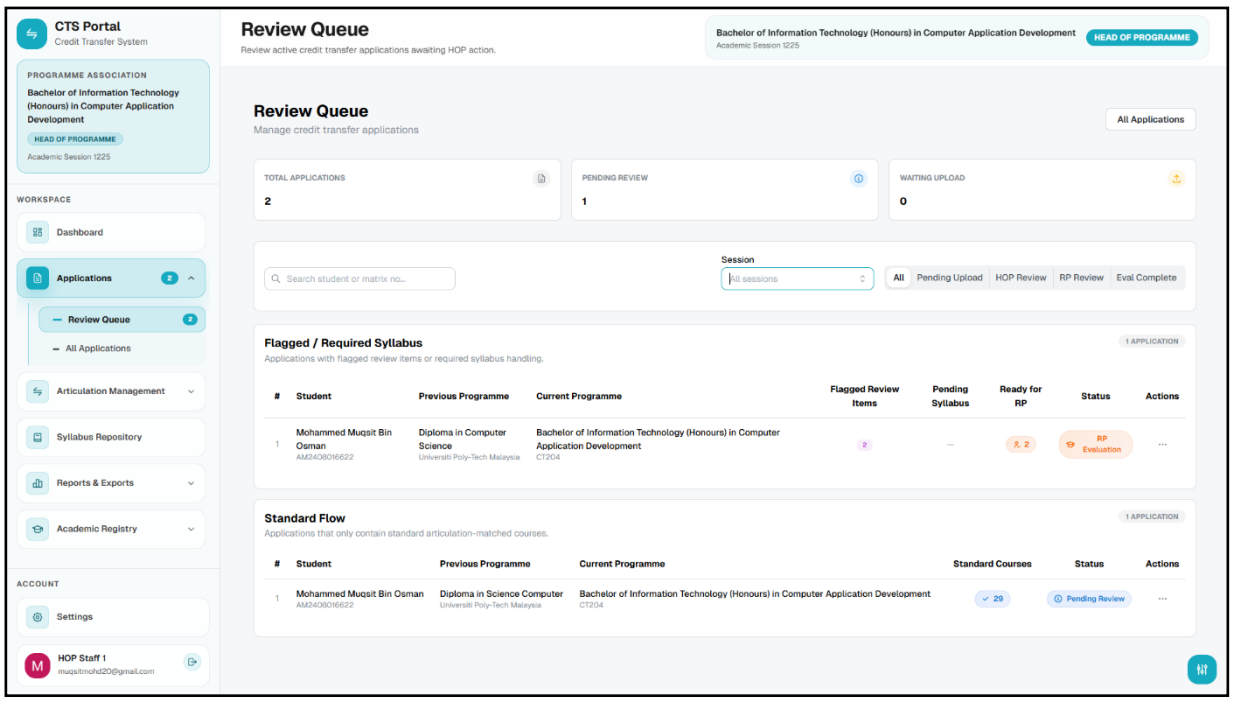


Figure 4.63: HOP Review Queue

Figure 4.63 represents the HOP review queue page. This is the main page where the HOP reviews submitted applications from students. The page includes three summary cards that describe the total number of applications, the number of applications waiting for review by the HOP, and the number of applications that have been sent to the next stage within the application review process (after the syllabi are uploaded by the student).

Below these summary cards are a toolbar that includes the following controls:

1. A search bar for HOPs to search for any specific student within the system;
2. A dropdown menu to select the academic session that the student applied to;
3. A segmented control to filter the applications according to their status (All applications, Pending Upload, HOP Review, RP Review, Evaluated Complete).

Within the review queue page are two separate tables:

1. A table that includes any applications that have flagged items within their review and require the student to upload syllabi for those courses;
2. A table that includes any applications that contain only courses that have been matched between their student's previous programme and their current programme at UPTM, and do not have any flagged items in their application.

Each row within these tables displays information about the student's application, and clicking on any row will take the HOP to the full review page for that student's application.

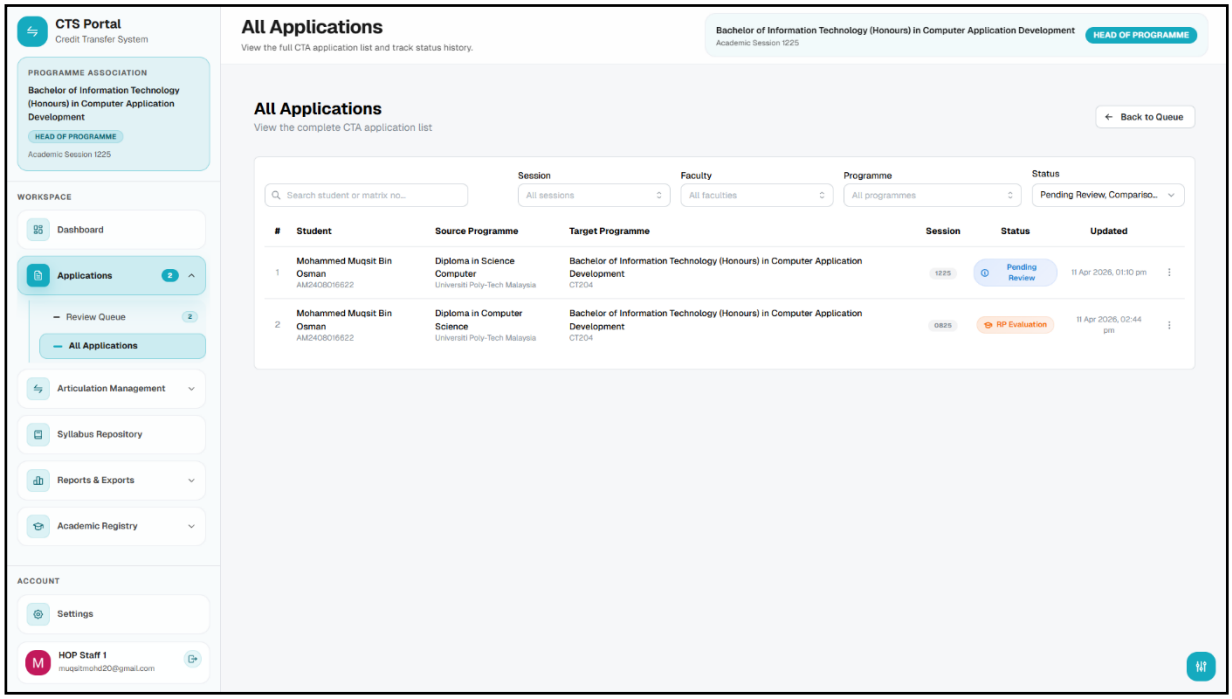


Figure 4.64: All Applications Page

Figure 4.64 represents the "All Applications" page for the HOP. This page is accessed from the review queue page after the HOP clicks on the "All Applications" button on the review queue page. This page displays all of the applications that have been submitted by students by the HOP, and all applications can be viewed in either a standard or finalized status. Thus, instead of filtering according to the status of the application, the "All Applications" page for the HOP includes all statuses of

applications: Pending Upload, HOP Review, RP Review, Evaluated Complete, Finalized, and Completed.

4.7.3.3 Application Review

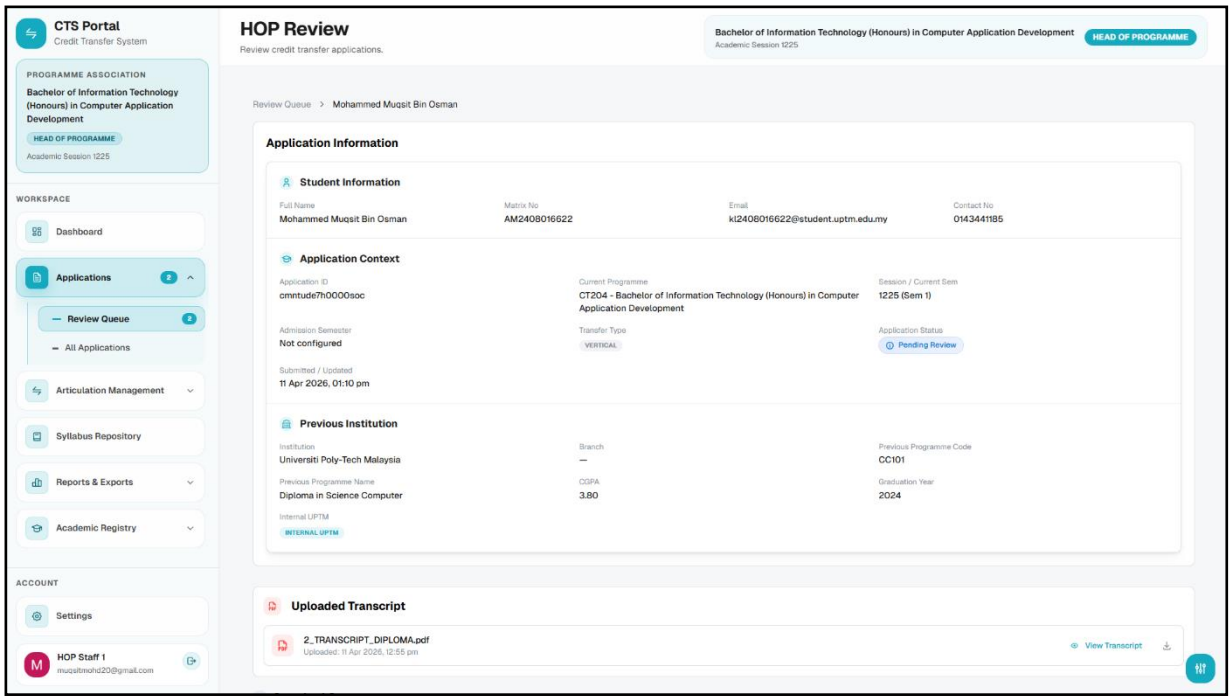


Figure 4.65: HOP Application Review (Header and Application Summary)

Figure 4.65 depicts the HOP application review page. The header of this page includes the breadcrumb navigation for the HOP to navigate back to the review queue, the application ID that can be copied, a status badge for the application, and action buttons for either exporting the student's slip or viewing the student's application. The action button to export the slip becomes visible only when the application reaches the FINALIZED or COMPLETED status. The application summary includes the personal information of the student, their previous programme, their current programme at UPTM, and the academic session.

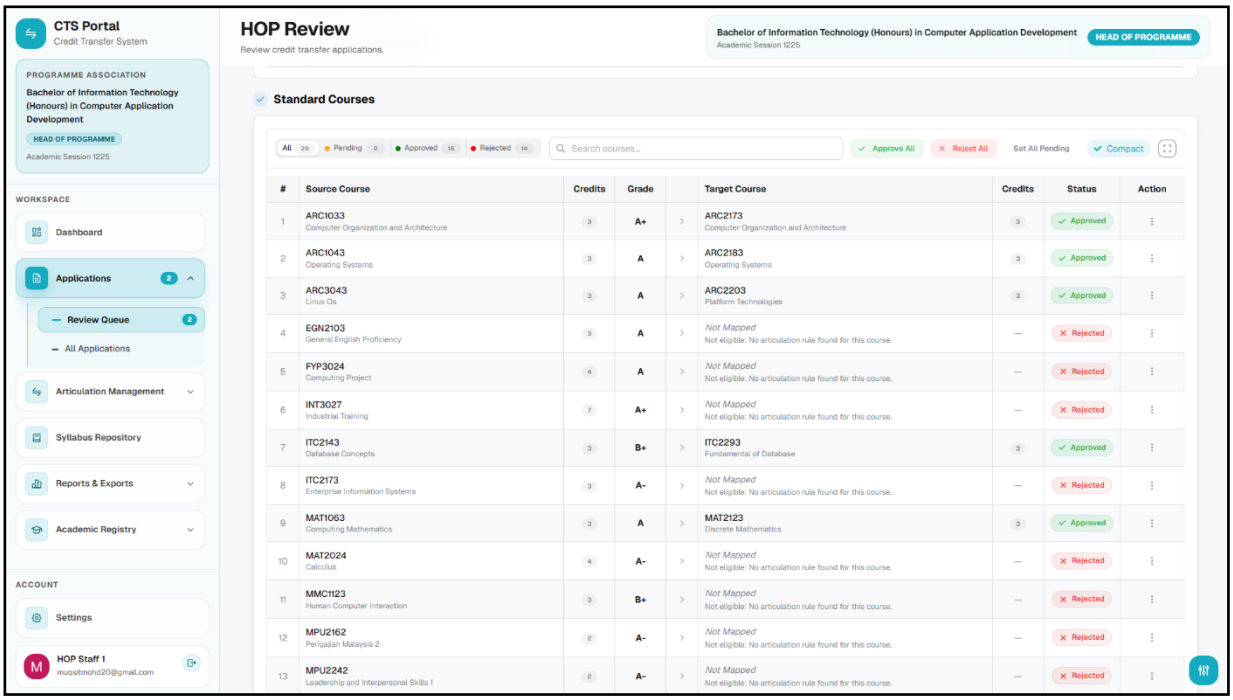


Figure 4.66: HOP Application Review (Standard Courses Table)

Figure 4.66 depicts the “Standard Courses” table that is included within the application review page for the HOP. This table includes the details of the courses that were automatically matched between the student’s previous institution’s courses and the student’s target courses within their target programme. Each row within the application review page includes information about the source course (such as the course code, name, credit hours, and grade), the target course (such as the course code, name, and credit hours), from which the recommendation for the course was made (Articulation or Manual), the decision regarding the course being approved or rejected, and actions for the HOP to edit the course details or make a decision about the course that is to be transferred.

Additionally, any courses that are rejected by the student’s eligibility requirements will also appear in this table. Such courses will have the reason for their rejection within the notes portion of the row for that course. These courses will not have any approval or rejection actions next to them.

HOP can make course decision selections either individually or through the bulk decision controls. For each of the reviewable courses, HOP can choose either to Approve or Reject the course, with optional notes to be entered with the decision.

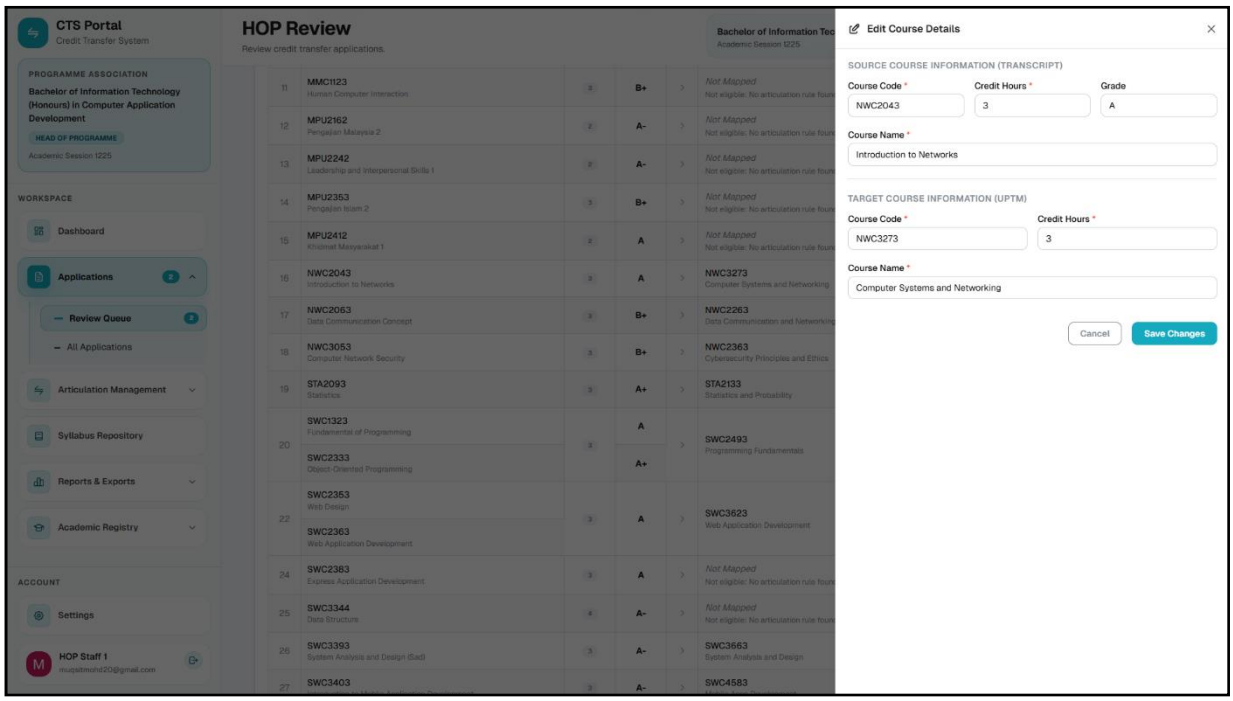


Figure 4.67: HOP Application Review (Edit Course Details

As shown in Figure 4.67, the Edit Course Details drawer is shown. Within this drawer, there are two main sections:

1. Source Course Information (Transcript) - This section allows for editing of the information for the source course being considered for articulation; source course code, name, credit hours, and the grade received in the course. For composite courses with many-to-one mappings, the individual source courses are listed individually within this section.
2. Target Course Information (UPTM) - This section allows for editing of the target course information, including the target course code, name, and credit hours for the course.

Upon saving the changes to either the source or target course, the eligibility for the student to take the course will be recalculated. For example, changing the grade in a course from a D to a B may result in the course being automatically eligible for HOP review, rather than being rejected automatically due to the low grade.

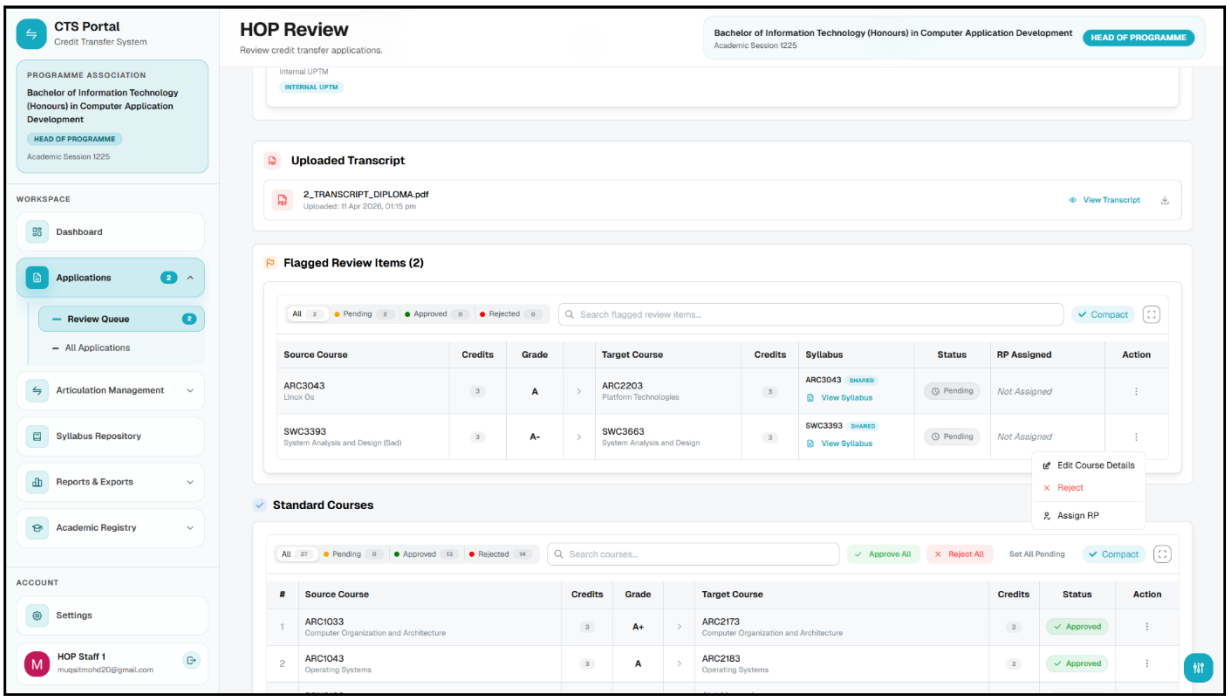


Figure 4.68: HOP Application Review (Flagged Review Units)

The section of the HOP review page that is shown in Figure 4.68 is the section dedicated to reviewing the courses that require syllabus evidence to be uploaded and reviewed by an RP (Resource Person). Each of these review units is represented in the system by a card that displays:

1. Source course(s) and target course mapping
2. Syllabus availability status (Uploaded, Pending Upload, Not Available, Shared)
3. Current assigned RP (if any)
4. Comparison status and score (if comparison has been run)
5. RP evaluation decision (if evaluation is complete)

For each of the flagged review units, the HOP administrator can take the following actions:

1. View Syllabus - Allows for the HOP administrator to download the syllabus for the course.
2. View Application - Opens the student's application in a new window.
3. Assign RP - Assigns an RP to the review unit for the syllabus to be reviewed.
4. Edit Course Details - Allows for edits to the details of the source and target courses.
5. Set Decision - Allows for the setting of the decision for the review units; either Approve or Reject.

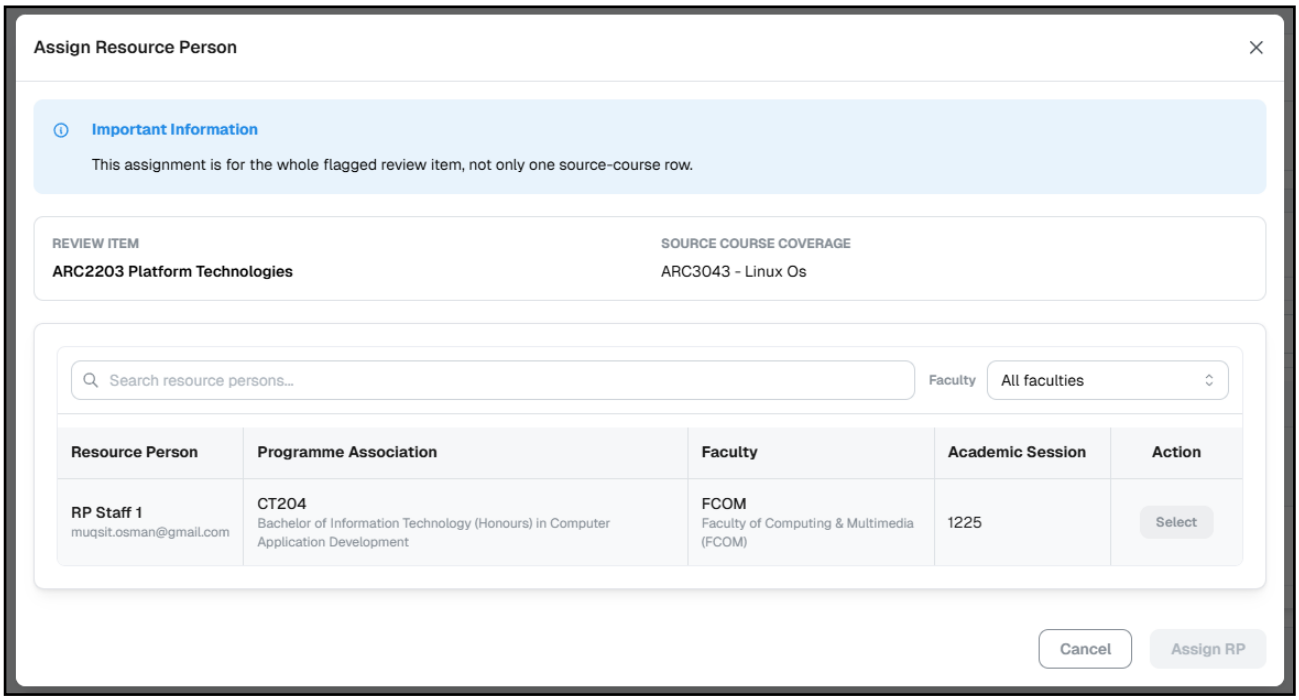


Figure 4.69: HOP Application Review (Assign RP Modal)

As shown in Figure 4.69, clicking on the Assign RP action will open a modal with an alert that explains that the RP will be assigned to the entire review unit. Additionally, the review item is displayed in this modal. Below this section is a table of available RPs that can be assigned. This table can be filtered by faculty member and searched by name, email address, or programme. Each RP has a select button associated with their details in the table. Upon selecting an RP and confirming the assignment, the RP will be assigned to the shared review group, as well as to all of the individual CTA courses that are associated with the shared review group.

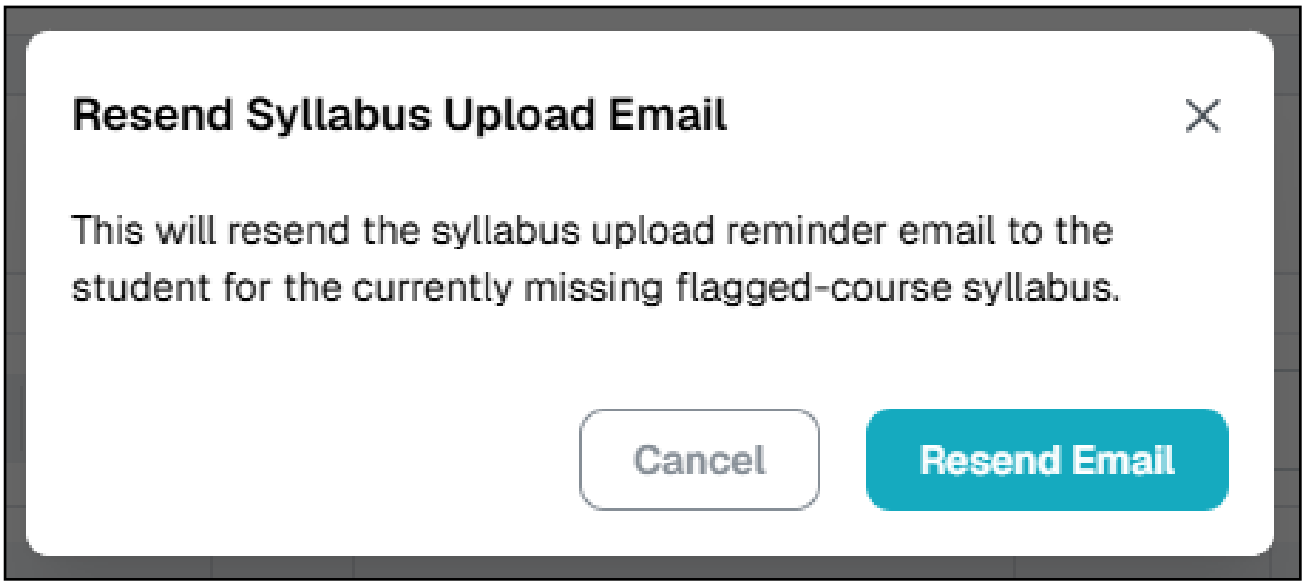


Figure 4.70: HOP Application Review (Request Syllabus Upload)

The action depicted in Figure 4.70 is for requesting that the student upload their syllabus. When clicked, this action will change the status of the application to AWAITING_SYLLABUS_UPLOAD. Additionally, an email will be sent to the student when the syllabus is successfully requested. Furthermore, this request will be visible in the application status history. When the student returns to the application in their next visit, it will be automatically sent to the syllabus upload workflow.

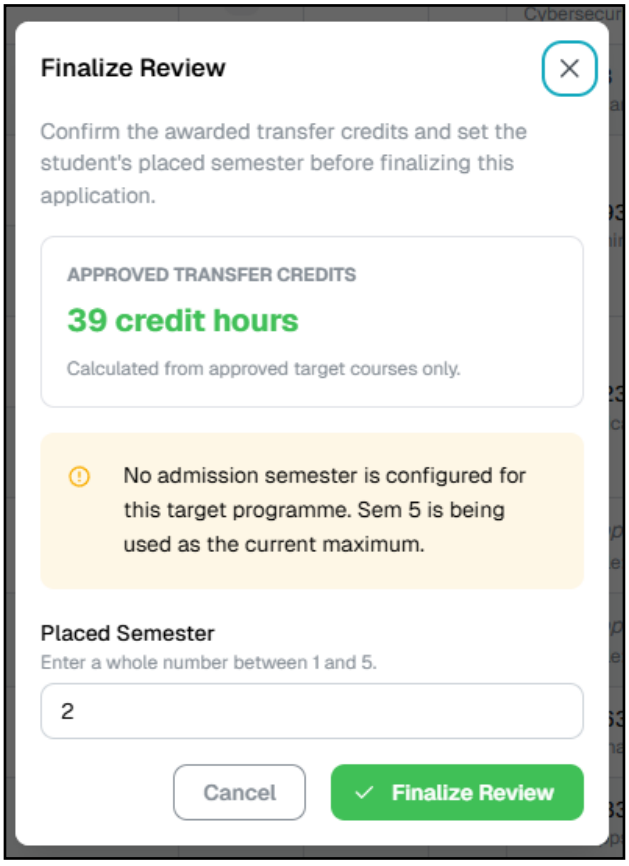


Figure 4.71: HOP Application Review (Finalization)

Figure 4.71 depicts the finalization section of the application review. This section will only become visible to the HOP administrator once all of the review units for the student have been resolved. The action required to finalize the application is to place the student into a semester. The semester that the student will be placed into will be visible in the placed semester field, and is validated against the admission semester for the student's programme. The finalization section will display a checklist to ensure that all requirements have been fulfilled for the placement and finalization of the student's application. The checklist includes items such as the total number of approved credit hours, ensuring that all courses have been reviewed, and that the placed semester is valid for the student's degree and programme. Should any of the prerequisites within this checklist be not fulfilled, the HOP administrator will not be able to click the finalize button for the application. When the requirements have all been fulfilled, the application will be finalized; its status will change to FINALIZED, the reviewByHOPId will be stored, and the total number of approved credit hours and the placed semester will be recorded for the student.

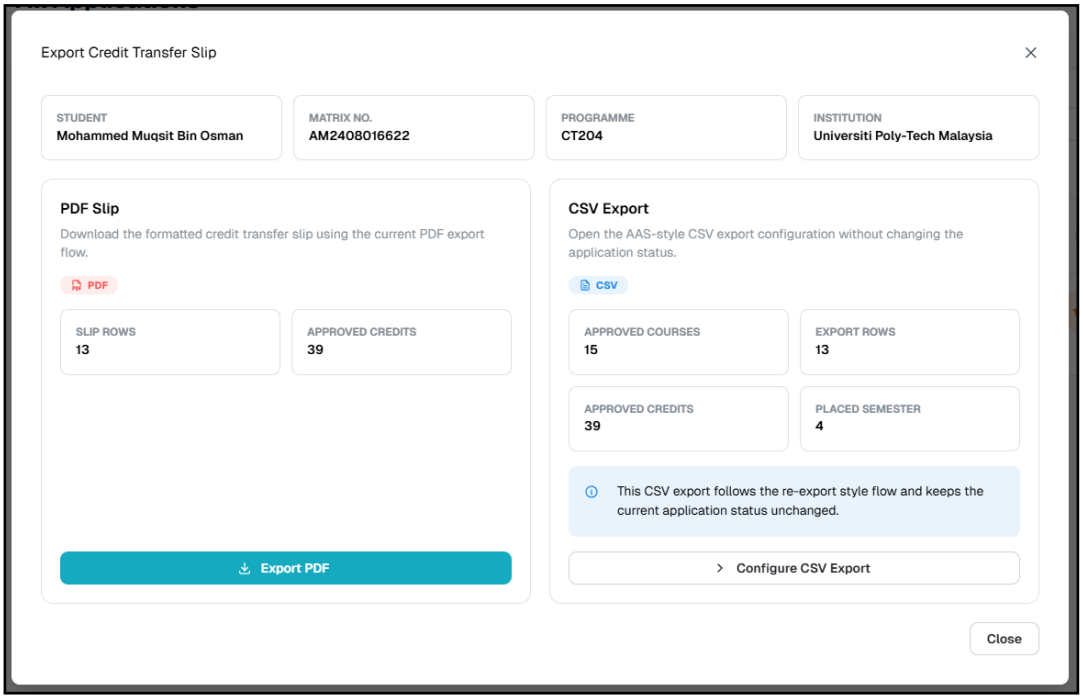


Figure 4.72: HOP Application Review (Export Slip)

As depicted in Figure 4.72, the Export Slip action is available to HOP administrators only for applications that have been finalized and completed. This action will generate a slip that details the credits that will be transferred to the student, as well as allow the HOP administrator to export the list of approved courses in a CSV file. The slip will be previewed in the modal prior to being downloaded as a PDF file. Additionally, the export to CSV will only be visible to HOP administrators; students can only download the PDF version of the slip.

4.7.3.4 Articulation Management

The articulation rules are stored within the system as a “library” of courses that can be approved for transfer into the student’s credits. These approved articulation rules are those that will be utilized by the system to automatically match the student’s courses to those in the target UPTM programme (Step 2 of the student’s workflow). Additionally, proposed articulation rules will be flagged for review.

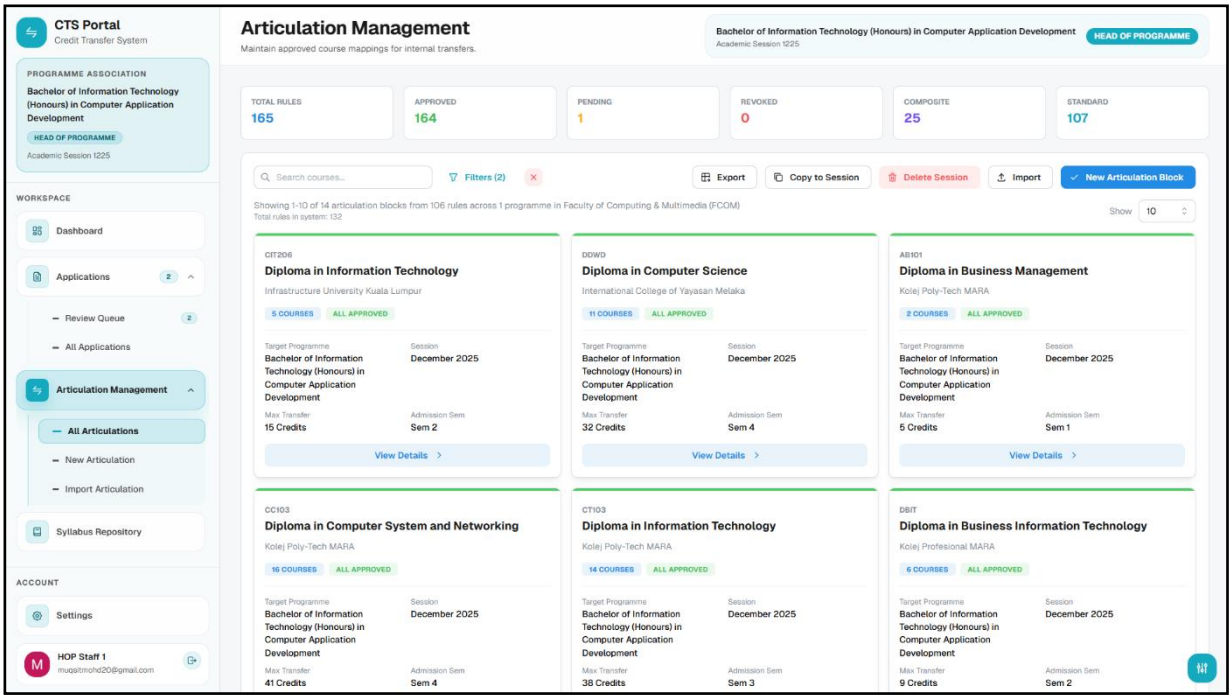


Figure 4.73: Articulation List Page

As shown in Figure 4.73, the articulation list page displays various statistics regarding the articulation rules: the total number of articulation rules, the number of approved rules, the number of proposed rules, the number of revoked rules, the number of composite course rules, and the number of standard one-to-one mappings. Following these statistics are various filtering options for the rules, including:

1. Search bar for filtering by course code, course name, or institution
2. Session filter
3. Programme filter (visible only to faculty members within the HOP)
4. Institution filter
5. Status filter
6. The rules are displayed within groups according to session, programme, institution, and branch. Additionally, individual courses can be viewed as either standard one-to-one mappings or as many-to-one mappings with “or” group notations for the various courses within the composite rule.

From this page, the HOP administrator can perform the following actions:

1. New Articulation - Creates a new articulation rule block.
2. Import - Allows the HOP administrator to import articulation rules from an Excel file.
3. Export - Opens a modal that allows the HOP administrator to export the articulation rules as either a CSV or XLSX file.

4. Clicking on any of the blocks of articulations will take the HOP administrator to the detail page for that group.

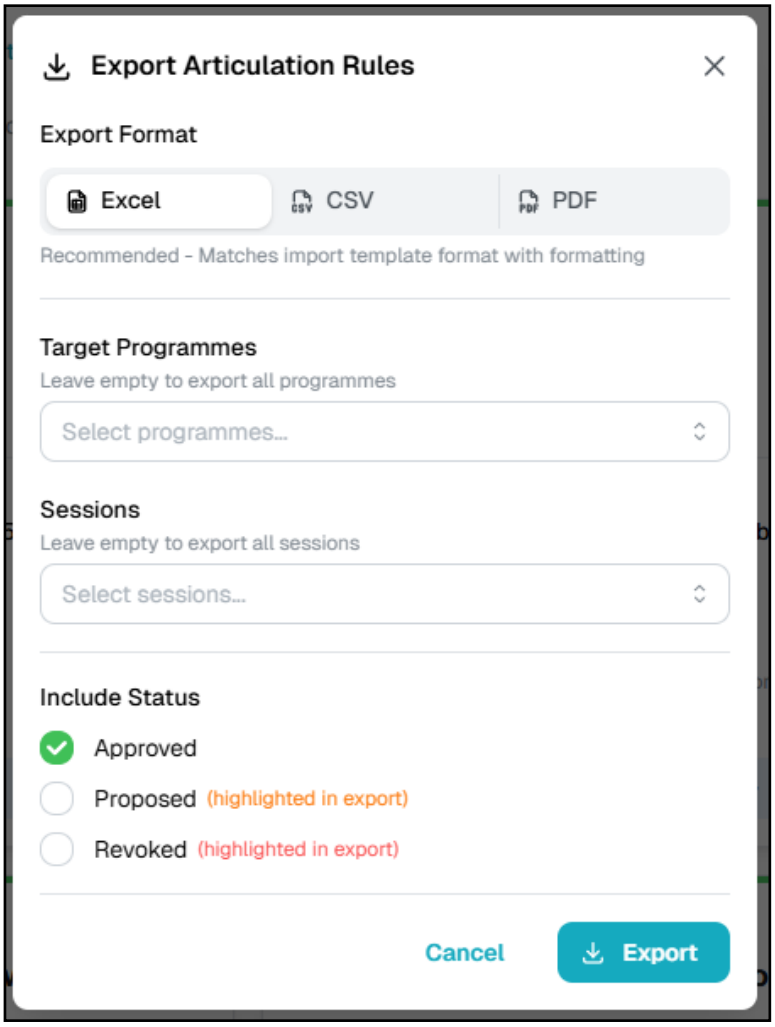


Figure 4.74: Articulation Export Modal

Figure 4.74 shows the export modal for the articulation module. When HOP clicks the Export button, a modal opens with format options for CSV and XLSX. A PDF option is also shown in the selector, but the interface marks it as not yet available. The export uses the currently filtered articulation data. The XLSX export format is aligned with the import template, which means exported files can be re-imported after modification.

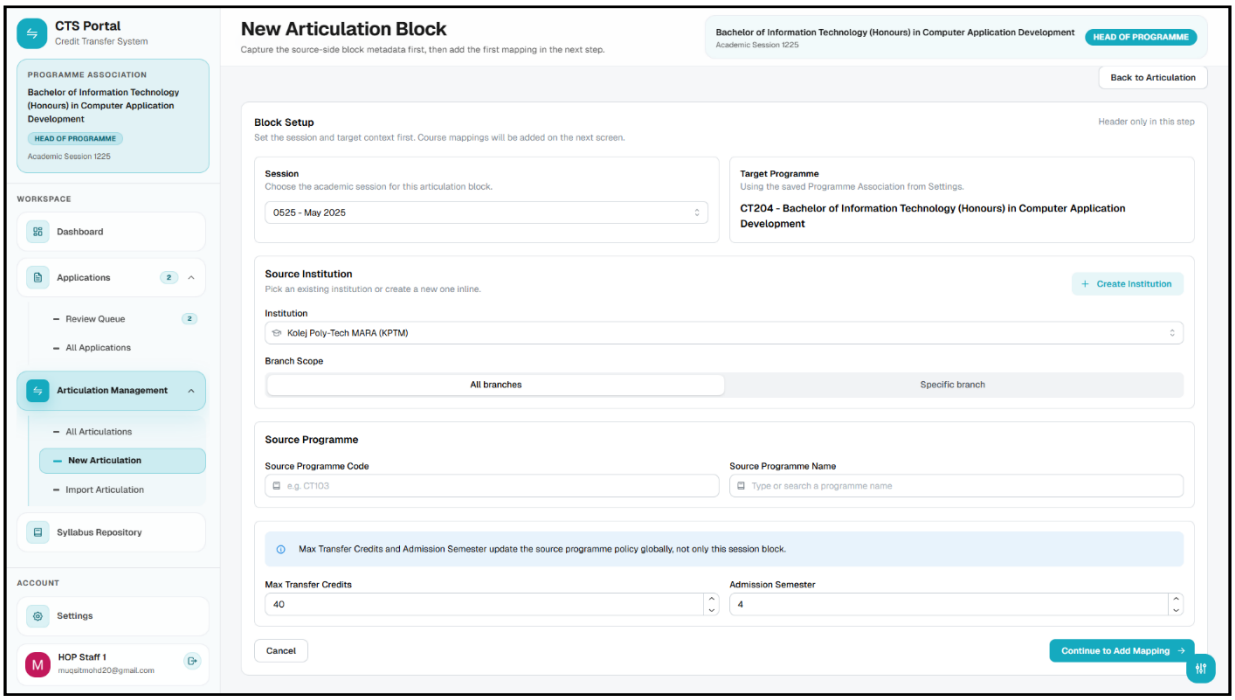


Figure 4.75: New Articulation Block (Step 1)

Figure 4.75 shows the new articulation block creation page. This page captures the block-level context that will apply to all mappings within the block:

1. Academic session (dropdown from available sessions)
2. Target UPTM programme (filtered by HOP's faculty scope)
3. Source institution (searchable, with an option to create a new institution inline)
4. Source branch (dependent on selected institution, with an option to create a new branch inline)
5. Source programme code and name (optional)
6. Programme policy settings: maximum transfer credits and admission semester

If the source institution or branch does not exist in the database, HOP can create it directly from this form without leaving the page. After saving, the page navigates to the block detail page where HOP adds the first course mapping.

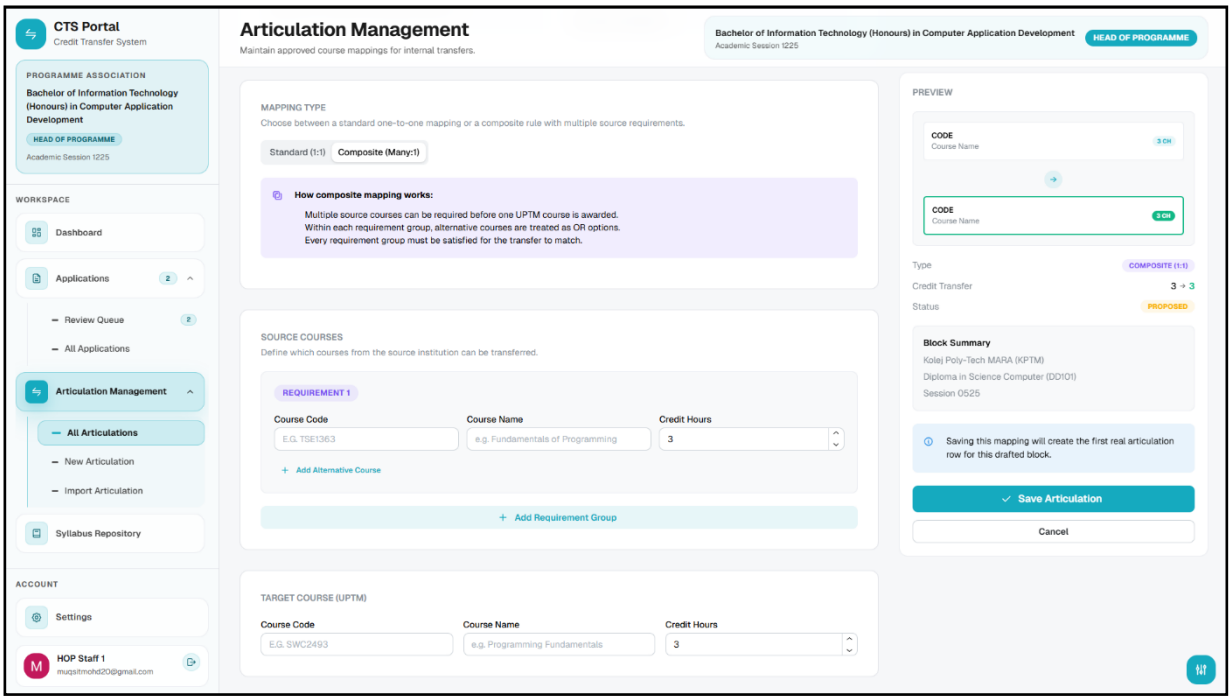


Figure 4.76: Add First Articulation Mapping

Figure 4.76 shows the page for adding the first articulation mapping within a new block. This is the step where the first articulation rule is actually created. The form contains:

1. Source course code, name, and credit hours
2. Target course code, name, and credit hours
3. Composite grouping fields (composite ID and or-group number) for many-to-one mappings
4. Initial status selection (Proposed or Approved)

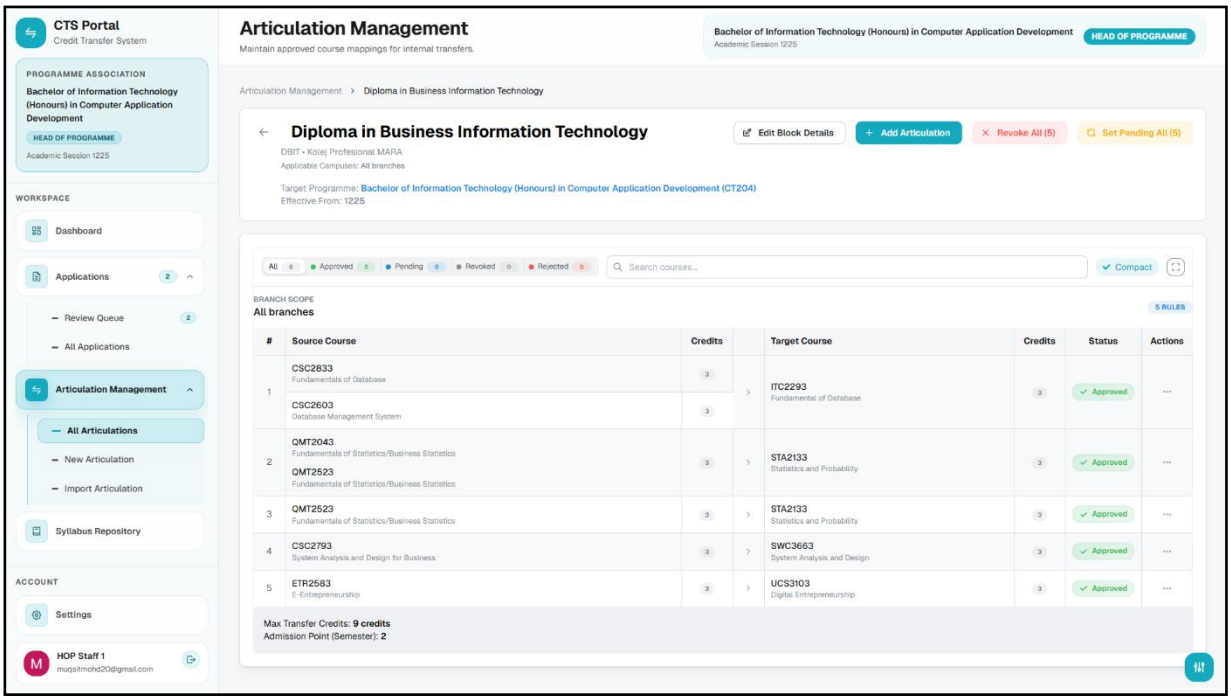


Figure 4.77: Articulation Block Detail Page

Figure 4.77 shows the articulation block detail page. This is the main workspace for managing all mappings within a block. The page header displays the block context (session, target programme, source institution, branch, and programme policy). Below the header, the page provides:

1. Add Articulation button to add more course mappings to the block
2. Edit Block button to modify the block-level source context and policy fields

The mapping table displays each articulation rule with:

1. Source course code, name, and credit hours
2. Target course code, name, and credit hours
3. Composite ID and or-group (if applicable)
4. Status badge (Proposed, Approved, Revoked)
5. Assigned RP (if any)
6. Action menu per row

The action menu for each mapping row provides:

1. Edit to modify the course details
2. Approve to change a proposed rule to approved status
3. Revoke to revoke an active rule (requires a revocation reason)
4. Assign RP to assign a Resource Person for syllabus-based evaluation
5. Upload Syllabus to upload supporting source or target syllabus files
6. Delete to remove the mapping (guarded when dependent records exist)

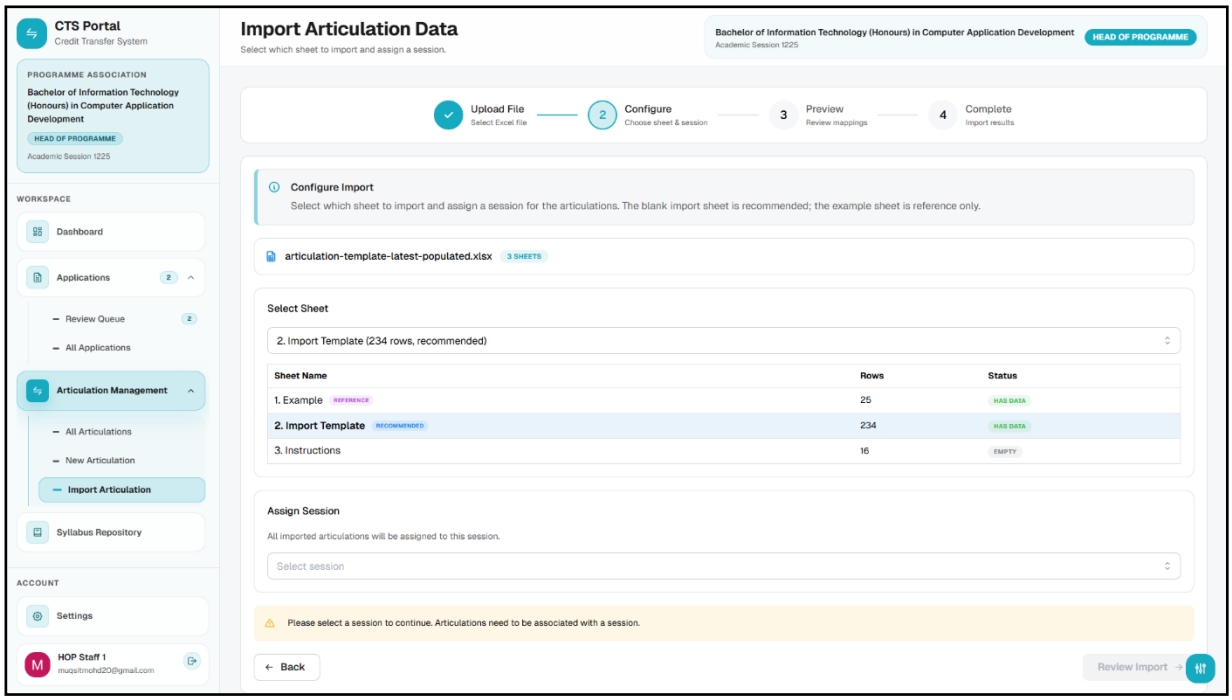


Figure 4.78: Articulation Import (Step 1: File Upload)

Figure 4.78 shows the first step of the articulation import flow. HOP uploads a structured Excel workbook containing articulation data. The import page accepts XLSX files that follow the articulation import template.

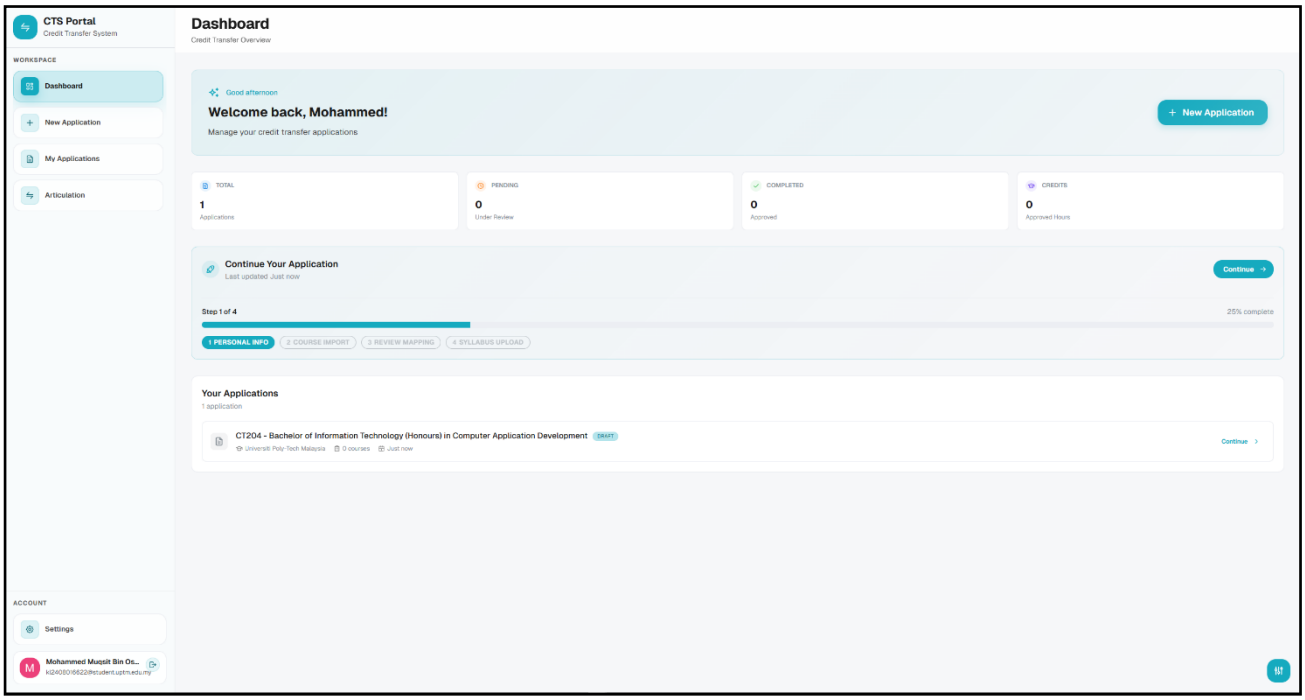


Figure 4.79: Articulation Import (Step 2: Configure Import)

Figure 4.79 shows the second step of the import flow, labelled Configure Import. After the workbook is uploaded, the page displays the file name and the list of available sheets with their row counts and

data status. HOP selects which sheet should be imported and assigns the academic session that will be applied to all imported articulations. When the workbook contains the standard template sheets, the blank import sheet is recommended, while sheets without articulation data cannot be selected for import.

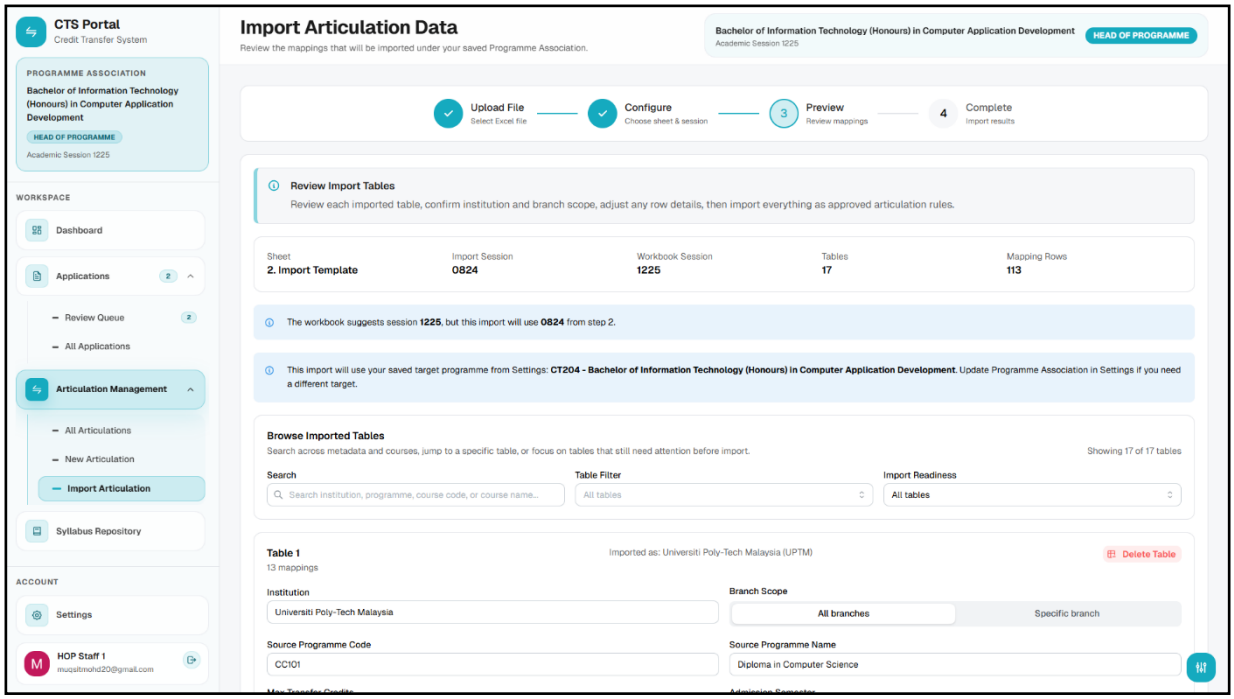


Figure 4.80: Articulation Import (Step 3: Review Import Tables)

Figure 4.80 shows the third step of the import flow, titled Review Import Tables. After the selected sheet and session are carried forward from Step 2, the system parses the sheet into table-based articulation blocks and displays a summary header showing the selected sheet, import session, workbook session, total tables, and total mapping rows. Any workbook parsing warnings, session mismatch notices, and target-programme scope alerts are shown above the review area. The main review section lets HOP search and filter the imported tables, then revise each table before submission. For every table, HOP can confirm or correct the institution, enter a short name if a new institution must be created, choose whether the articulation applies to all branches or a specific branch, update the source and target programme details, and adjust programme policy fields such as maximum transfer credits and admission semester. The mapping rows are displayed in a table showing the original row number, grouped source courses, and target course details. Through the action menu, HOP can edit or delete individual mapping rows, while the whole table can also be removed from the import draft.

The page blocks submission until required fields are resolved. When HOP clicks the "Import as Approved" button, the server validates the edited draft, resolves or creates the required institution, branch, and source programme records, enforces the HOP's allowed target-programme scope, and

imports each remaining mapping as an approved articulation. If an equivalent articulation already exists for the same source, target, institution, and session context, the system promotes the existing record to approved instead of inserting a duplicate.

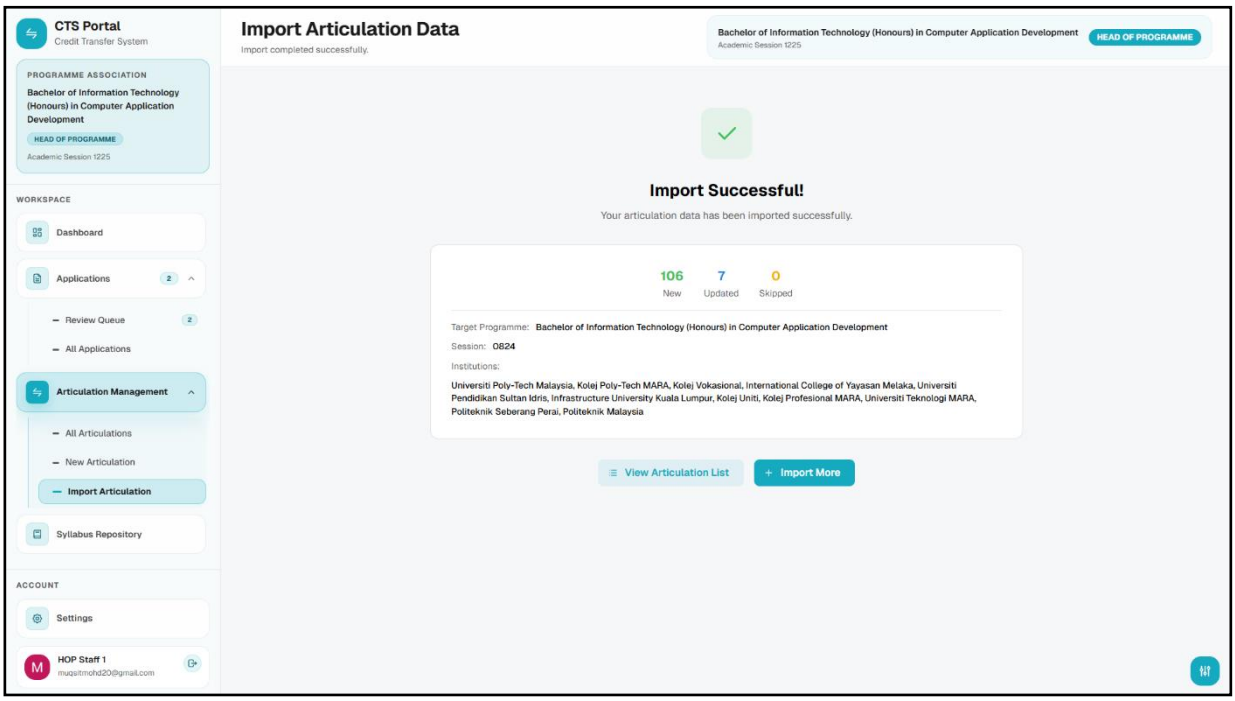


Figure 4.81: Articulation Import (Success)

Figure 4.81 shows the interface after the importation of the reviewed tables is completed successfully. The page displays information regarding the number of mappings that were imported as new articulations, the number of existing articulations that were updated due to the promotion importation, the number of mappings that were skipped during the importation process, and whether there were any errors to rows of articulations during importation. Finally, the page displays the target programme, session, and list of institutions of the imported articulations. In the case of any errors in the importation of articulations, those errors can be viewed prior to HOP's return to the articulation list or the beginning of a new importation process.

4.7.3.5 Syllabus Repository

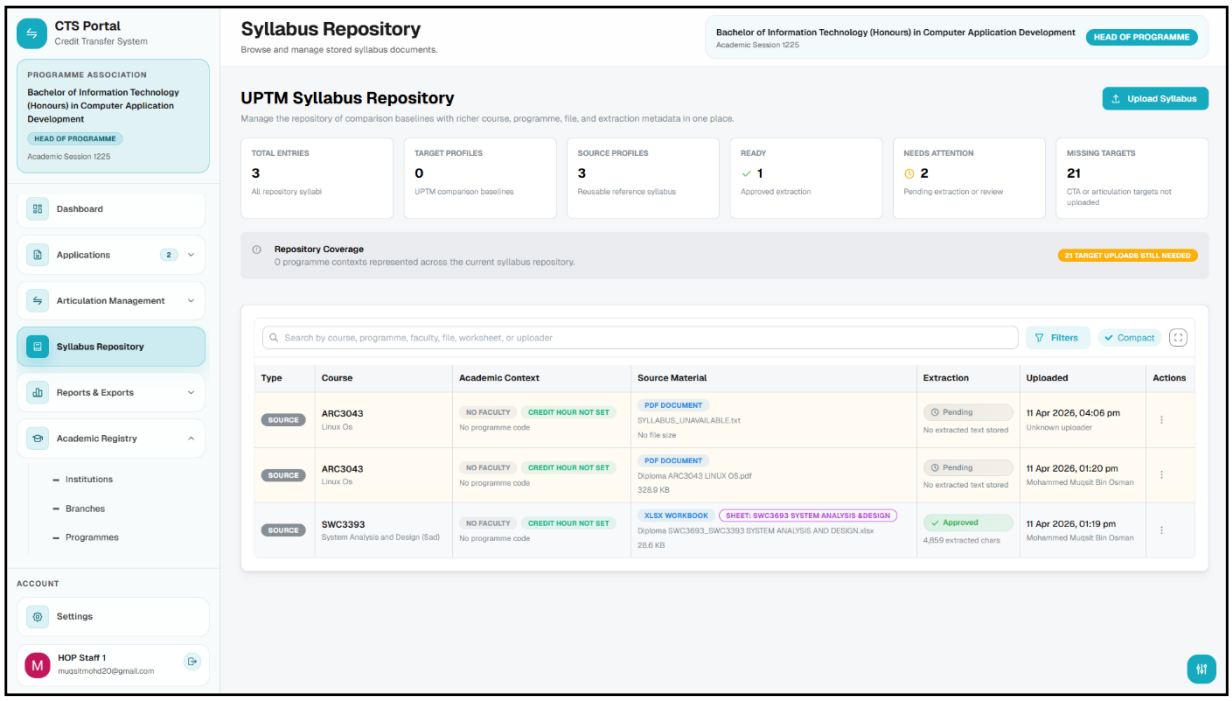


Figure 4.82: Syllabus Repository

Figure 4.82 displays the syllabus repository interface. The syllabus repository allows for the management of the syllabi that will be used in the process of comparing the credits of the students with the requirements of the target universities (UPTM). Within this interface are six summary cards that display information regarding the total number of syllabi within the repository, the number of UPTM target syllabi, the number of reusable reference syllabi, the number of syllabi that are ready for extraction, the number of syllabi that are ready for extraction but pending by the staff member, and the number of syllabi that are discovered target courses but without an uploaded syllabus. Following these summary cards is the main table of the syllabi within the repository, which includes the following information for each syllabus: the type of syllabus (Target or Source), the course code and name, the academic context of the course (who teaches the course, how many credits the course earns, and the student programs that offer the course), the information regarding the source of the syllabus (type of file, worksheet within a workbook, the file name, and the size of the syllabus file), information regarding the extraction status of the syllabus, the date upon which the syllabus was uploaded, and the name of the user that uploaded the syllabus.

Additionally, each syllabus also displays an actions menu that allows the user to view the syllabus, edit the details regarding the course, re-extract the text from the syllabus, and delete the syllabus file from the repository. To upload a new syllabus to the repository, HOP must click the “Upload Syllabus” button that displays on the page. A modal will open that allows the user to select the course that will be uploaded (either a CTA or articulation target course, or to enter a custom course code), to edit the details regarding the course that will be uploaded (faculty that teaches the

course, the student program, and the number of credits the course earns), and to upload the syllabus as a PDF or XLSX file. If an XLSX file is selected, another modal will appear that allows the user to select the worksheet within the workbook to be uploaded. Additionally, to the right of the syllabus list is a set of collapsible filters that allow the user to filter the syllabi according to type (Target or Source), the faculty that teaches the selected course, and the status of extraction of the syllabus.

4.7.3.6 Academic Registry

The academic registry allows for the management of the information that relates to the students that are applying to the programs within the UPTM target universities. The academic registry includes three different tabs: Institutions, Branches, and Programmes.

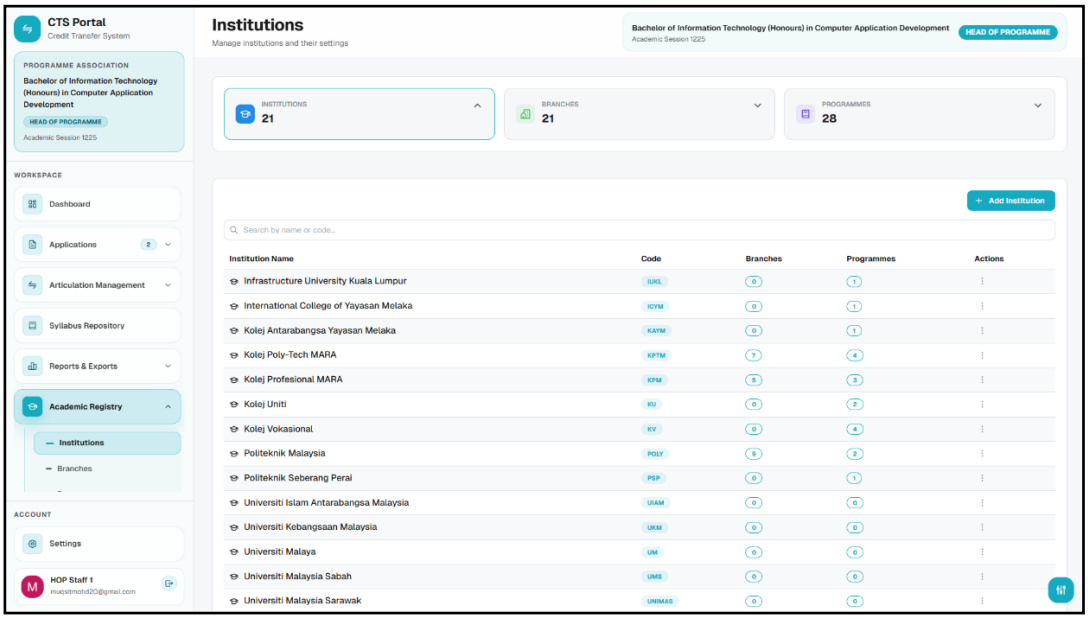


Figure 4.83: Academic Registry (Institutions)

Figure 4.83 displays the Institutions tab within the academic registry interface. This tab displays each institution that is registered to the system, including its institution code, institution name, institution slug, institution type, and the active status of the institution. Each institution can be individually edited or deleted; however, attempts to delete an institution will result in a message that displays the number of programmes within the institution; if there are any programmes within the institution, that institution cannot be deleted.

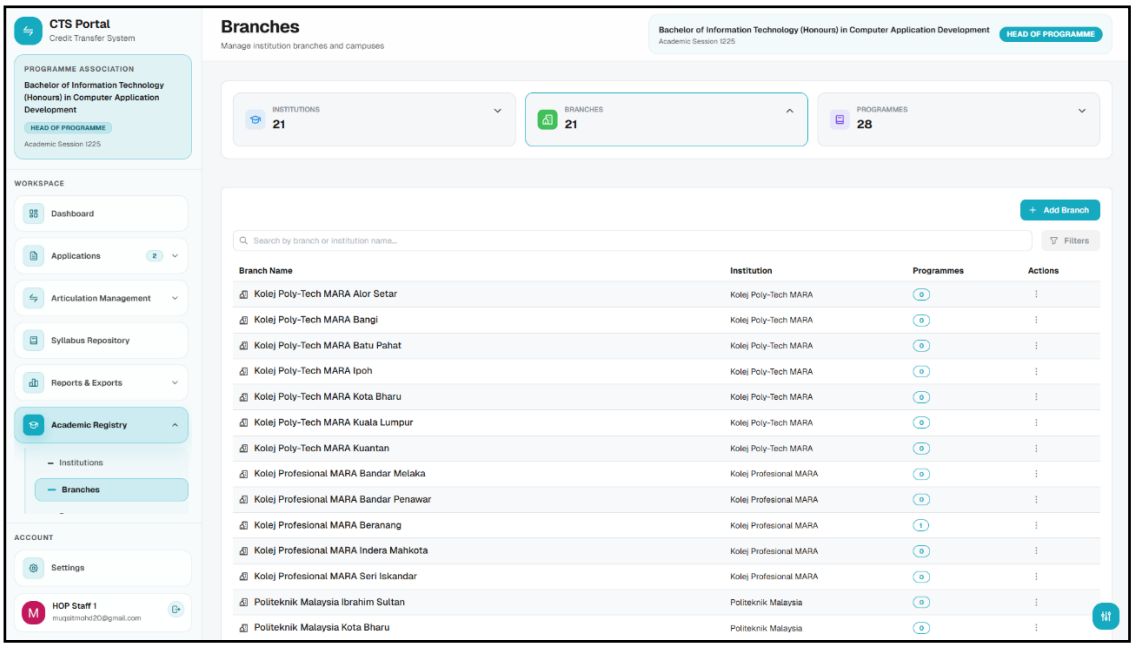


Figure 4.84: Academic Registry (Branches)

Figure 4.84 displays the Branches tab within the academic registry interface. Each branch within the system is listed on this tab, and each branch will have a name for that branch, a slug, and the institution to which it belongs. Each branch can be individually edited or deleted; however, attempts to delete a branch will result in a message that displays the number of programmes within the branch; if there are any programmes within the branch, that branch cannot be deleted.

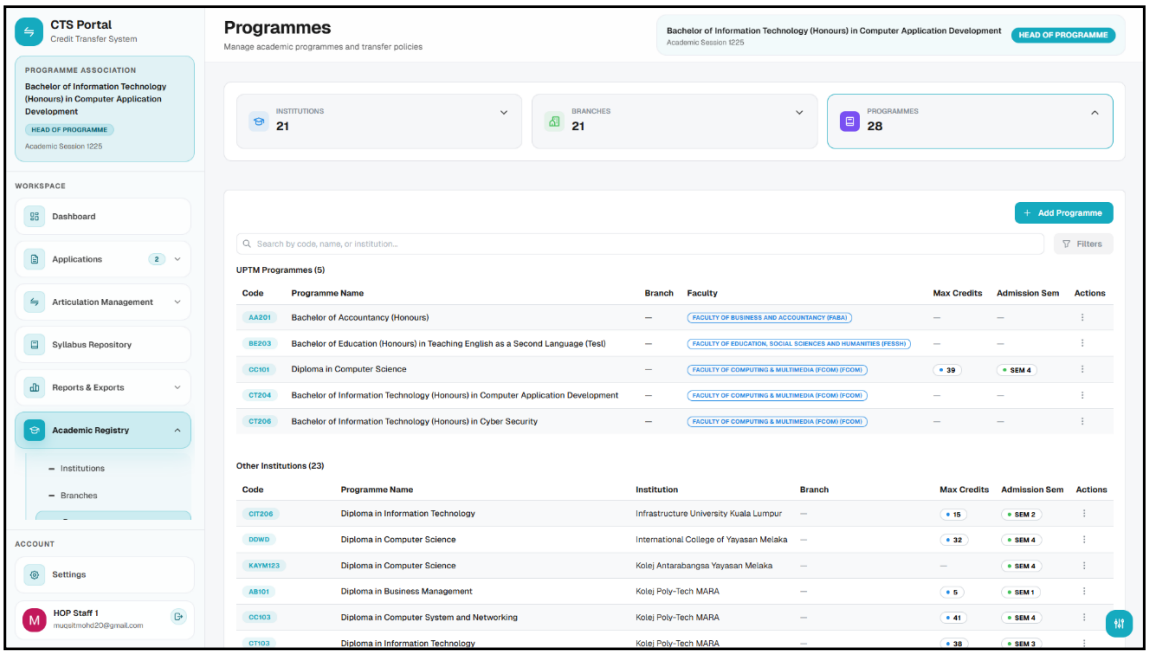


Figure 4.85: Academic Registry (Programmes)

Figure 4.85 displays the Programmes tab within the academic registry interface. Each programme within the system is listed on this tab, and each listing will have fields for the programme code,

programme name, institution to which the programme belongs, the branch within that institution that offers the programmes, the faculty that teaches the courses within the programme, the number of credits that can be transferred into the student's degree from another institution, and the semester during which the student can apply and begin their transfer into the programme. Each of these fields can be edited for each programme, but attempts to delete a programme will result in a message that displays the number of students that are currently enrolled in that programme; if there are students within the programme, that programme cannot be deleted.

4.7.3.7 Reports and Analytics

The reports and analytics interface for the application allows the administrators of the application and the Resource Persons (RP) to gain insight into the application and the process of transferring credits between students' former and target universities. The reports page includes a toolbar that allows filters to be applied to the reports that will be displayed on the page regarding the institution, programme, session, and application status. The reports include three main sections that display different types of reports regarding the application process.

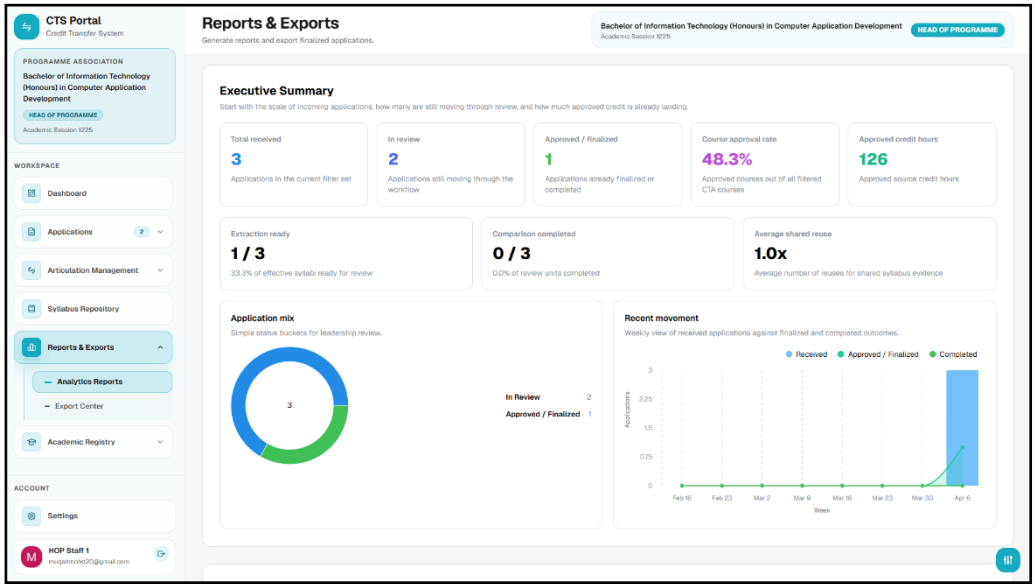


Figure 4.86: Reports (Executive Summary)

Figure 4.86 displays the Executive Summary section of the reports application. The Executive Summary section includes five main metrics regarding the applications submitted from CTA and articulation students; the total number of received applications, the total number of applications that are currently being reviewed by the staff members, the total number of applications that were approved and finalized, the percentage of the courses within those applications that were approved, and the total number of approved credits. Additionally, there are three summary cards for applications that display information regarding the number of applications that have been submitted, extracted,

and compared, as well as the percentage of shared syllabi within the applications. Additionally, there are two charts within the Executive Summary portion of the reports: a donut chart that displays the number of applications within each status, and a composite chart that displays the number of received and finalized applications over time.

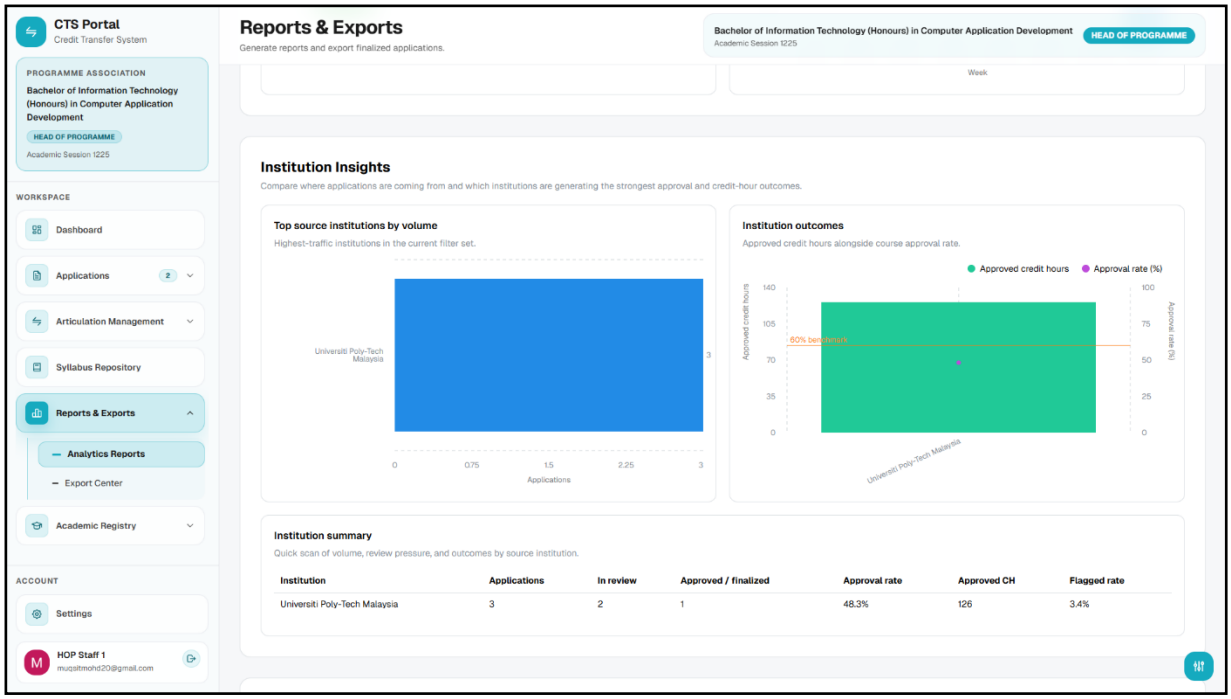


Figure 4.87: Reports (Institution Insights)

Figure 4.87 displays the Institution Insights section of the reports application. This section includes two charts: a horizontal bar chart that displays each institution according to the number of applications that they have received from students, and a composite chart that displays the number of approved credits from each institution, as well as the percentage of courses within each application that are approved from each institution (with a 60% approval rate indicated as a reference line on the chart). Below the charts is a table that includes the same information as the charts: the application count from each institution, the number of applications that are in review from each institution, the number of applications that were approved from each institution, the percentage of applications that were approved from each institution, the number of approved credits from each institution, and the percentage of courses within each institution's approved applications that were flagged for review.

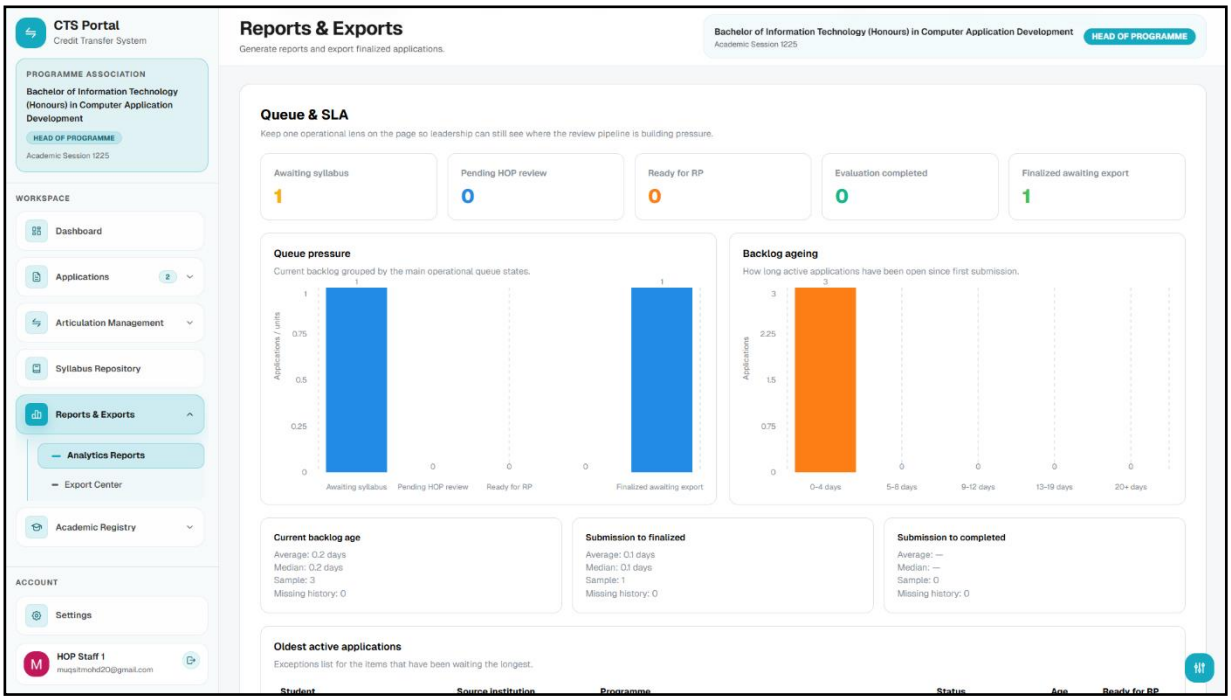


Figure 4.88: Reports (Queue and SLA)

Figure 4.88 displays the Queue and SLA section of the reports application. This section displays a series of cards that describe the number of applications within each stage of the application and review process (backlog, in review, approved, completed), bar charts that display the number of applications within each stage of the process (by status), the number of applications that have been in each stage of the process over time, the average and median number of days that it takes to complete specific stages of the process (from received until approved, from approved until completed, from submitted until approved), and a table of the applications that have been active the longest in the system (with their age and the number of students within those applications that may be sent to the RP for review).

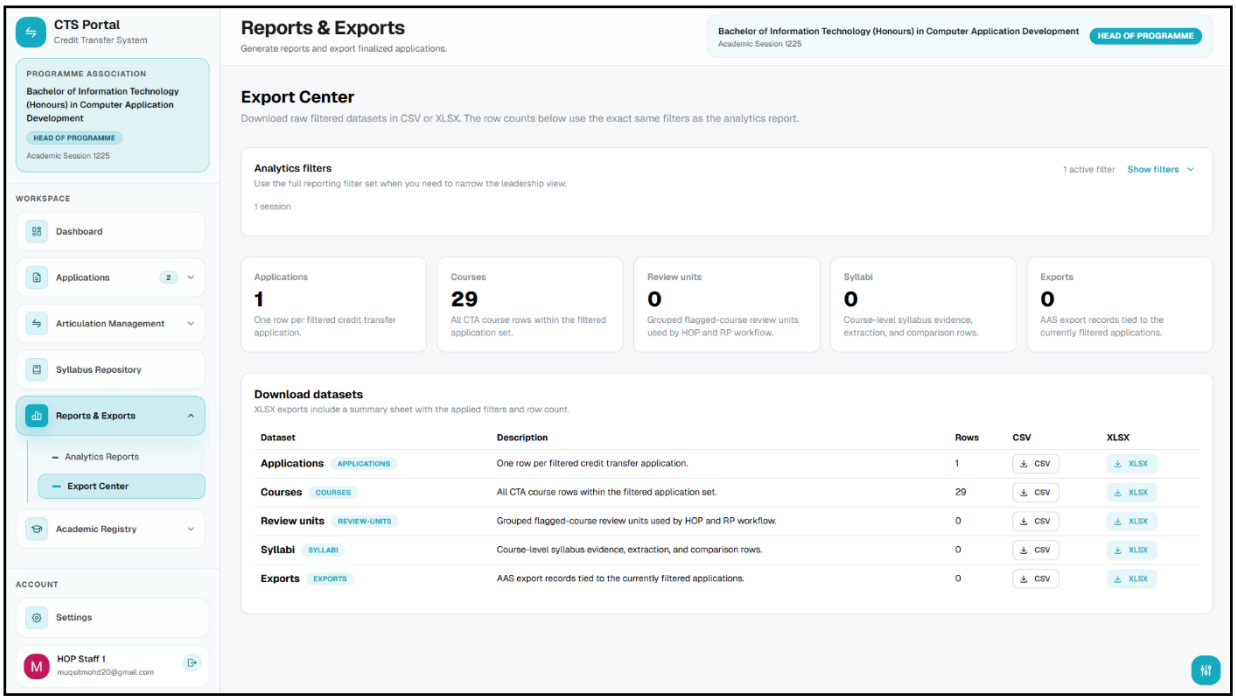


Figure 4.89: Report Export Center

Figure 4.89 displays the Report Export Center interface. The Report Export Center is accessible from the Reports interface for the application (by clicking the “Open export center” button that is displayed on the page). The Report Export Center allows HOP to download the reports in both CSV and XLSX file formats. There are export options for applications, courses, institutions, and the various metrics that relate to the application process.

4.7.4 RP Interface

The RP interface within the UPTM application will be used by the academic Resource Persons that are assigned to each student’s application. The RP interface has two main areas within the website: the dashboard at /dashboard, and the review area at /reviews/rp. The RP interface is different from the HOP interface in that the RP is to review syllabi and course credits for applications by the students to the CTA and articulation universities, rather than managing the application process as a whole. Thus, each RP will be assigned specific applications to review; upon making a decision regarding the credits within each student’s application, that decision will be applied to any other applications that have used that same syllabus of the student to be reviewed.

Thus, there are two types of applications that may be assigned to an RP: those that were initiated by the student (known as CTA applications), and those that were initiated through the articulation process of HOP. Each of these applications will have a detailed review page, which is explained in the following sections of this chapter.

4.7.4.1 RP Dashboard

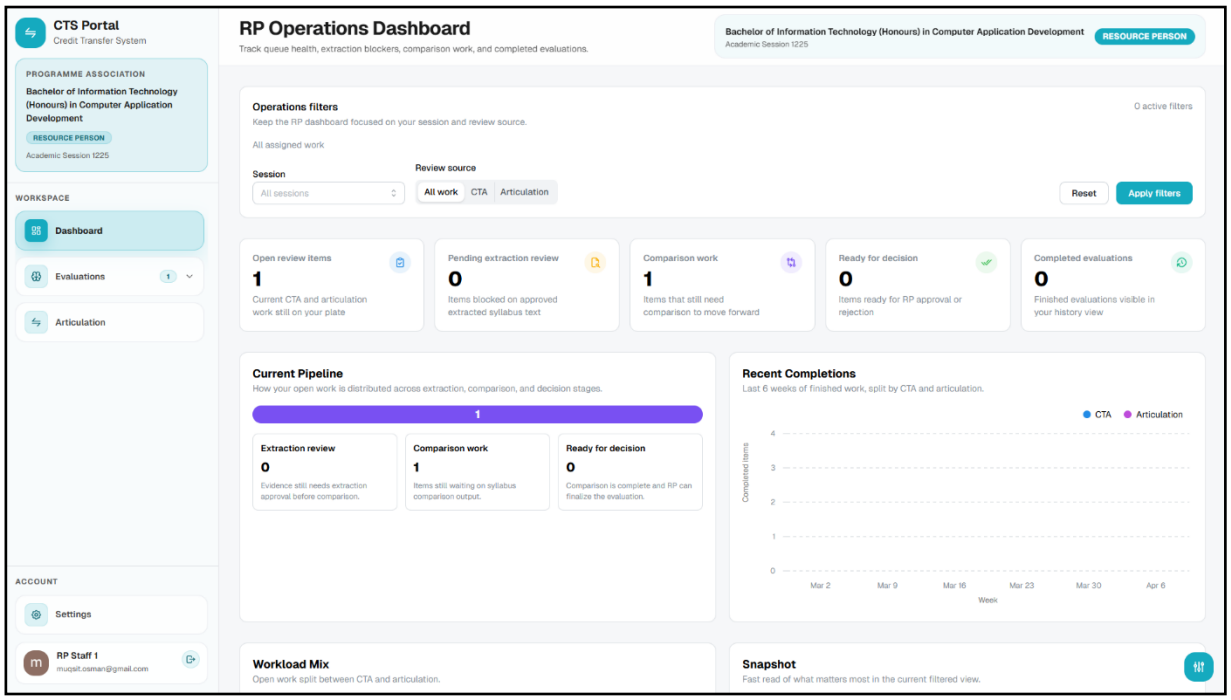


Figure 4.90: RP Dashboard

Figure 4.90 displays the RP dashboard interface. The RP dashboard allows for the RP user to have an overview of the workload of the RP, as well as a history of the evaluations that the RP has completed. Along the top of the page is a filter that allows the RP to scope the data on the RP dashboard to only their review source (All, CTA, or Articulation applications), or to a specific academic session. Following the filter are five KPI cards that display the following information about the RP: the total number of review items assigned to the RP, the number of items pending extraction, the number of items that have been compared to the requirements of the target university, the number of items ready for the RP to make a decision, and the total number of evaluations that the RP has completed. Additionally, there are two main analytical sections of the RP dashboard. The first is the Current Pipeline information, which displays a progress bar of the number of applications within each stage of the process (review, extraction, comparison, decision), as well as three separate cards that display the number of applications within each of those stages. The second main analytical section is Recent Completions, which displays a bar chart that represents the number of evaluations of both CTA and articulation students that were completed within the last six weeks.

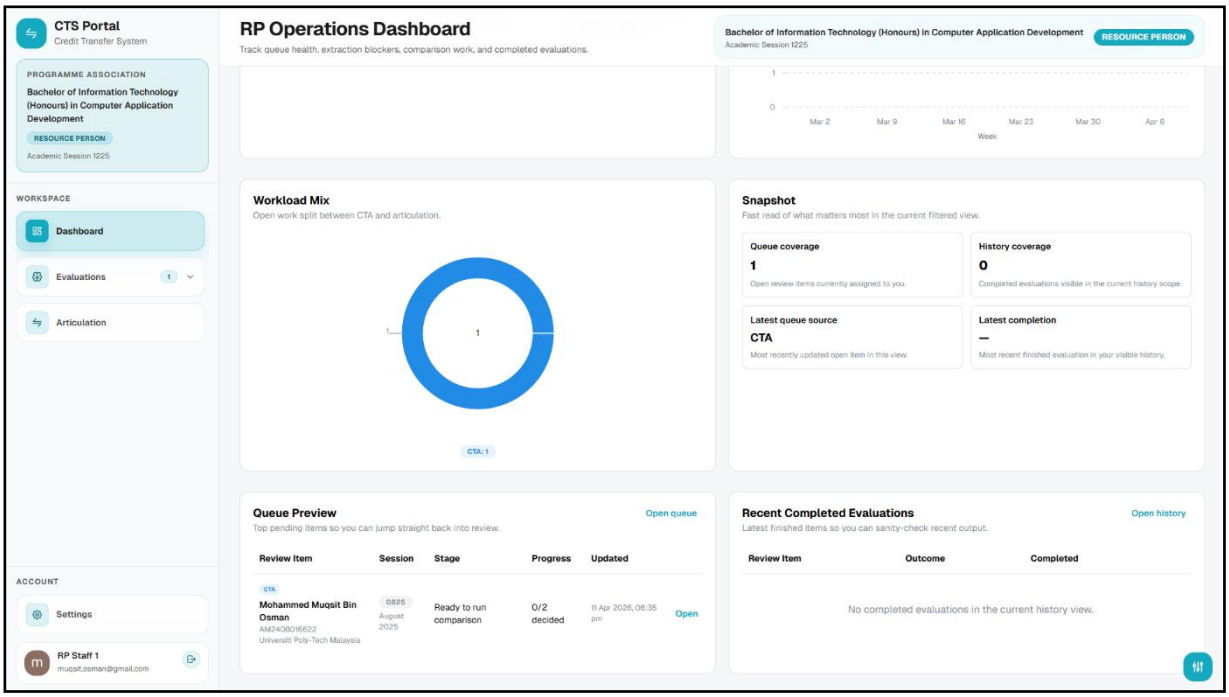


Figure 4.91: RP Dashboard

Figure 4.91 displays the lower half of the RP dashboard. Within this area is information that displays the number of applications within each of the CTA and articulation applications for the RP; if the RP has filtered their view of the applications to only one type of application, then this area will change from a donut diagram to a list of the number of applications within each type. Additionally, within this area are four indicators of the coverage of the RP's workload: the number of applications that are still pending review, the number of applications that have been reviewed in the past seven days, the institution from which the newest applications within the queue originated, and the date and time of the newest completed applications. Finally, at the bottom of the RP dashboard are two tables of information about the workload of the RP. The first table is the Queue Preview table, which includes a list of the top applications that are pending review by the RP; the table includes information such as the type of application (from the badge), the title of the application, the subtitle of the application, the institution from which the application originated, the session during which the application was submitted, the stage during which the application is currently being processed by the RP, the status of the application's review, and the date upon which the application was last changed. The second table of information is the Recent Completed Evaluations table, which also includes information about the top applications that have been reviewed by the RP during the most recent time period; however, the table additionally includes information regarding the outcome of the review (approved or rejected), the target programme for the student, and a link to the completed review of that application.

4.7.4.2 Evaluation Queue

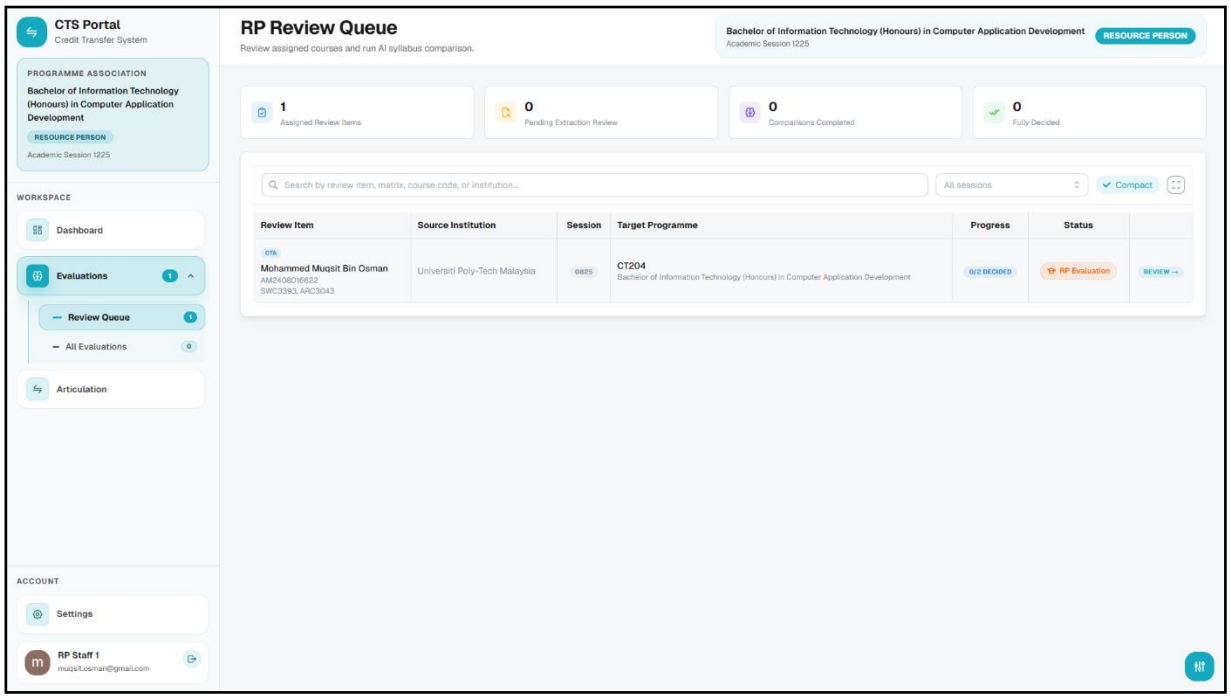


Figure 4.92: RP Evaluation Queue

Figure 4.92 displays the RP evaluation queue at /reviews/rp. This page displays a list of the applications that are assigned to the RP (and that are still being managed by the RP). At the top of the list are four cards that display the following information: the total number of assigned review items, the total number of items pending review by the staff members, the total number of items that have been compared with the requirements for the target university, and the total number of items that have been decided upon by the RP. Following these cards is a table that displays each of the review items that are assigned to the RP. The table includes fields for the review item (including type of review item, the title of the review item, the subtitle of the review item, and the reference string to that item), the institution from which the student applied, the academic session of the application, the target programme to which the student is applying, the number of units of the application that have been decided upon by the RP out of the total number of units in that application, the status of the application, and a “Review” link button that allows the RP to jump to the detailed review page for that application. To the right of the table is a search bar and a filter that allows for the RP to search for specific review items by the review item’s fields (title, subtitle, reference, institution), as well as to filter according to the academic session of the students; if there are no applications within the system that meet the search and session filters, the page will indicate that there are no review items assigned to the RP at this time.

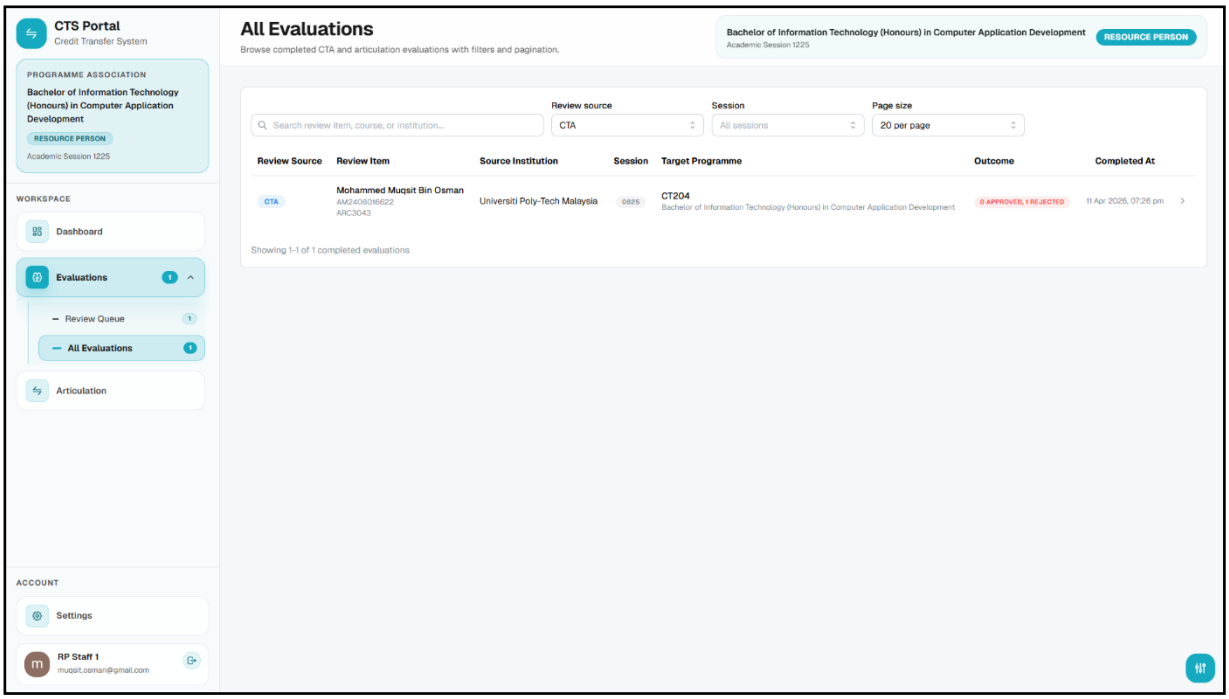


Figure 4.93: All Evaluations Page

Figure 4.93 displays the All Evaluations page for the RP at /reviews/rp/all. This page allows for the RP to review each of the applications that have been evaluated by the RP and to view the details of each evaluation. The toolbar at the top of the page includes fields for a search field for review items, courses, and institutions, a field for filtering according to the type of review item (CTA, Articulation, All), the academic session of the students, and the number of records to be displayed on the screen (10, 20, or 50). Following these fields is a table that displays each of the evaluations that have been completed by the RP. Each table row includes information regarding the type of review (CTA or Articulation), the review item, the institution of the student, the academic session of the student, the target programme of the student, the outcome of the review, the timestamp of when the review was completed, and a chevron that allows the RP to jump to the review of that evaluation. For CTA applications, the outcome will be represented as colored badges that indicate, for instance, the number of approved versus rejected courses within the student's application. For articulations, the outcome will be represented as the status badge for the articulation. Within the table is information regarding each evaluation that can be clicked in order to open the review of that application in read-only mode. Finally, at the bottom of the screen is information regarding the current page of the list of evaluations (for instance, Showing 1-20 of 45 completed evaluations). Additionally, there are controls at the bottom of the screen that allow for pagination of the results.

4.7.4.3 Course Equivalency Review

The course equivalency review page is the main RP workspace. This page can be accessed directly from the RP queue or from the “completed evaluations” history. The page follows a fixed sequence from top to bottom: review header → review groups → course overview → evaluation evidence → staged comparison → academic decision.

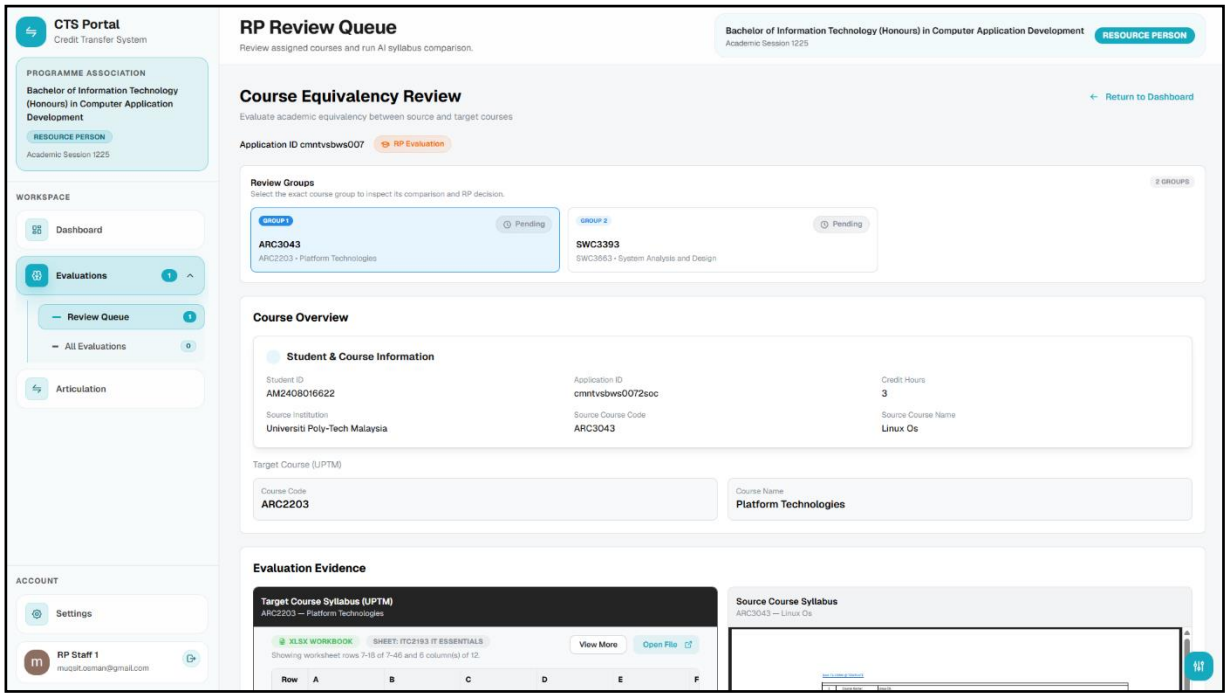


Figure 4.94: RP Review (Header and Review Groups)

Figure 4.94 displays the top section of the RP review page. The page header displays the title of the page (Course Equivalency Review) and a short subtitle. Depending on how the RP accessed the review page, the navigation button may read Back to All Evaluations or Return to Dashboard. The navigation will return to the main RP review queue page ("/reviews/rp"). Following the header is the application ID and the application status badge. If the item within the shared review group includes more than one group of courses to be reviewed, a panel will display the review groups. Each review group includes the group number, the number of courses within the group, the course codes within the group, the target course to which the source course(s) is/are mapped, and the decision badge for that group of courses. Clicking on any of the group summaries will cause the remainder of the RP review page to be refreshed so that the RP can review the specific group of courses.

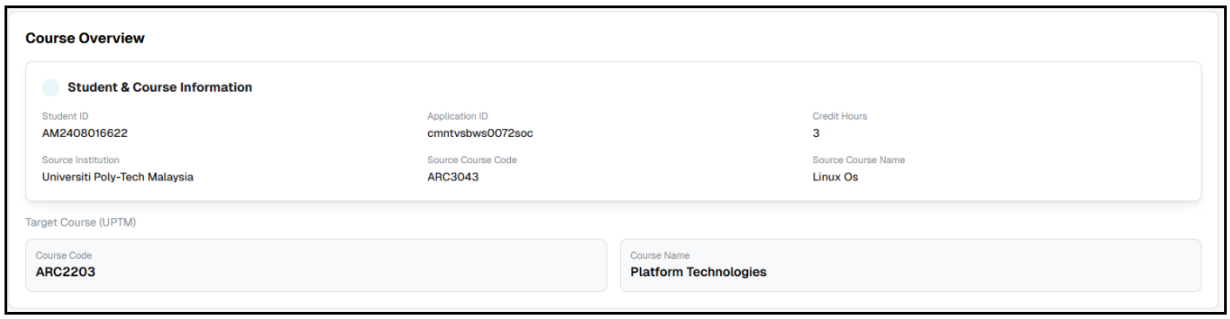


Figure 4.95: RP Review (Course Overview)

Figure 4.95 displays the course overview section of the RP review page. Within this section is a layout that displays information regarding the student, the application, and the source course. The information includes the following fields:

1. Student ID
2. Application ID
3. Credit Hours
4. Source Institution
5. Source Course Code
6. Source Course Name

If the shared review group includes more than one source course, the credit hours, course codes, and course names will be displayed together and joined by the “+” symbol. Following the information layout is a block displaying the target course (UPTM). Within this block are two cards that include the target course code and the target course name.

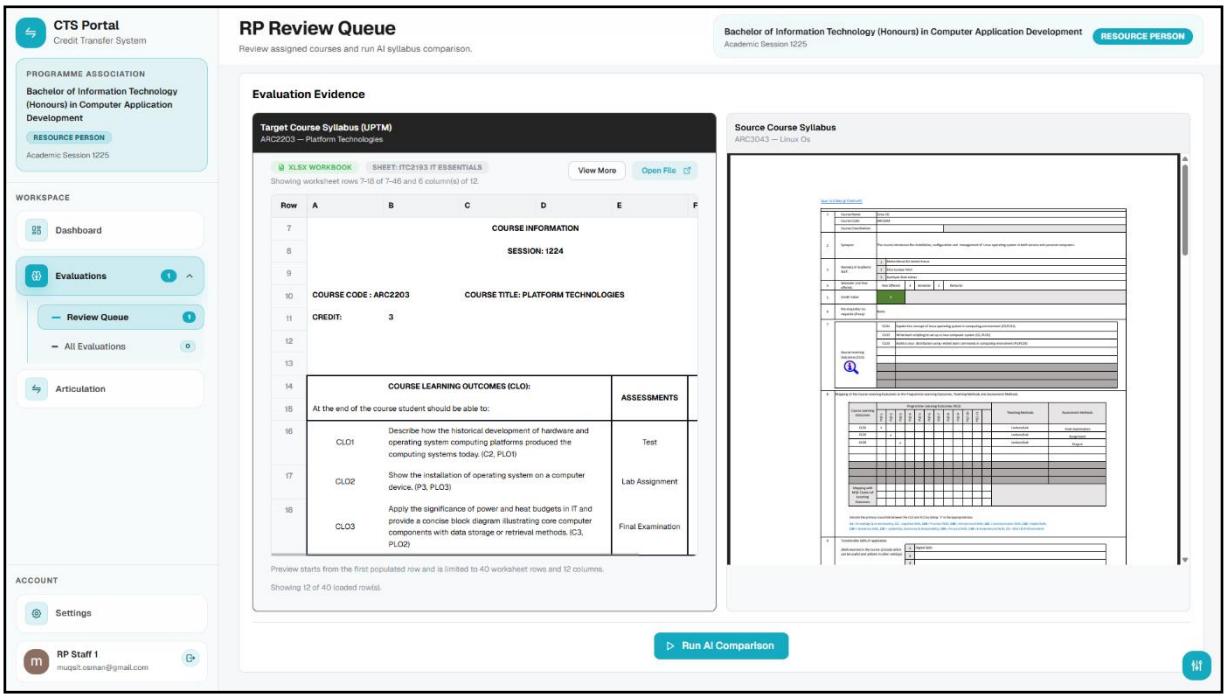


Figure 4.96: RP Review (Evaluation Evidence)

Figure 4.96 displays the evaluation evidence section of the RP review page. Prior to the evaluation evidence section is a check for whether the syllabus text for the source course has been approved for use in the comparison process. If the approved syllabus text is still missing from the system, the banner will read Preparing Source Syllabus. If the preparation of the syllabus is successful, the banner will change to Evaluation Evidence. Within this section, the RP may click on the Check Target Syllabus Availability button to access the target syllabus in the syllabus repository (if it exists). If there is no target syllabus available for the target course, an alert will display on the screen informing the RP of the missing syllabus; in this instance, there will be a button that will open the syllabus upload page in the browser ("/syllabus"). Following the messages and buttons is a two-column display of the syllabi for the courses to be compared. The left column will feature the target course syllabus (UPTM) and the right column will feature the syllabus for the source course. PDF files will be displayed within the web browser. For XLSX files, the syllabi will be displayed as previews that display the worksheet name, row numbers, content of the cells, formatting of those cells, and buttons to view more rows of the syllabus that may exist within the uploaded XLSX file. An Open File button will be visible to allow the RP to examine the syllabus file as it was uploaded. For shared review groups that include more than one source course, the source syllabus will be displayed as a series of tabs with each tab representing one of the source course codes included within the shared review group.

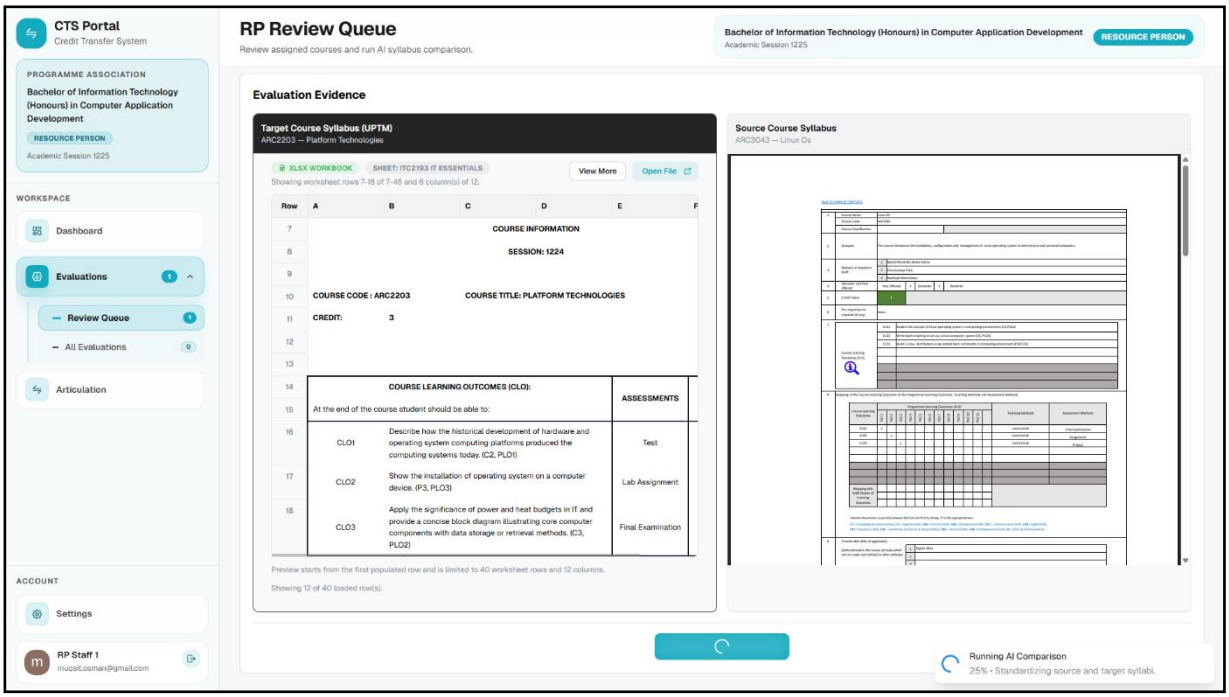


Figure 4.97: RP Review (Run AI Comparison)

Figure 4.97 displays the Run AI Comparison button. If the syllabus text for the source course has been approved, if the target syllabus exists in the syllabus repository, and if there is no comparison that has already been completed for the shared review group, an AI comparison between the two syllabi will be initiated by clicking the Run AI Comparison button. The AI comparison will go through the following stages within the comparison pipeline:

- PREPROCESS_SOURCE
- PREPROCESS_TARGET
- STANDARDIZE_SOURCE
- STANDARDIZE_TARGET
- COMBINE_SOURCES
- MAPPING
- VERIFICATION
- SCORING
- SUMMARY

While the AI comparison is processing, the system will display a notification on the screen stating that the AI comparison will take around one to two minutes to complete. If the AI comparison is already in progress, the Run AI Comparison button will change to the following label: Comparison in Progress...

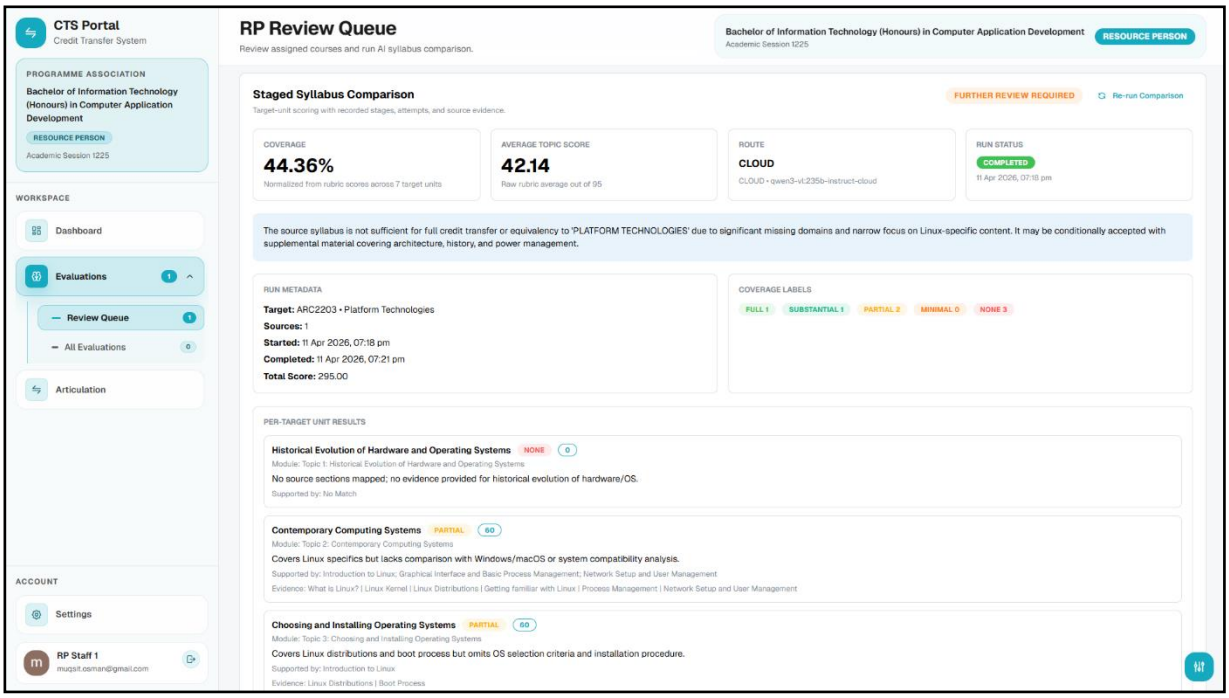


Figure 4.98: RP Review (Staged Comparison Results)

Figure 4.98 displays the results of the staged syllabus comparison. Following the comparison is a recommendation badge that will read Recommended if the coverage score from the comparison is 80% or higher, or it will read Further Review Required if the score is 80% or below. Additionally, the recommendation header will also feature a button that will initiate another round of the syllabus comparison. The comparison results will display four score cards presenting the comparison of the two syllabi:

1. Coverage
2. Average Topic Score
3. Route
4. Run Status

Following the score cards are three additional sections of information that can be reviewed by the RP. The AI-generated conclusion will be visible in an alert. The Run Metadata card will display the metadata regarding the comparison. The Coverage Labels card will display how many target learning units have been assessed with each label: Full, Substantial, Partial, Minimal, or None. Following this card is a list of the comparison results for each target learning unit. Each of these results will include the target learning unit, the label for the unit's coverage, a score for that unit, the title of the unit (if one exists), the reason that the AI determined the coverage label and score for the learning unit, a summary from the source course syllabus that justifies its match to the target learning unit, and the references to the syllabus content that supported the justification. At the very bottom of the page is

a panel that will feature the markdown report that was generated by the AI system regarding the comparison of the two syllabi.

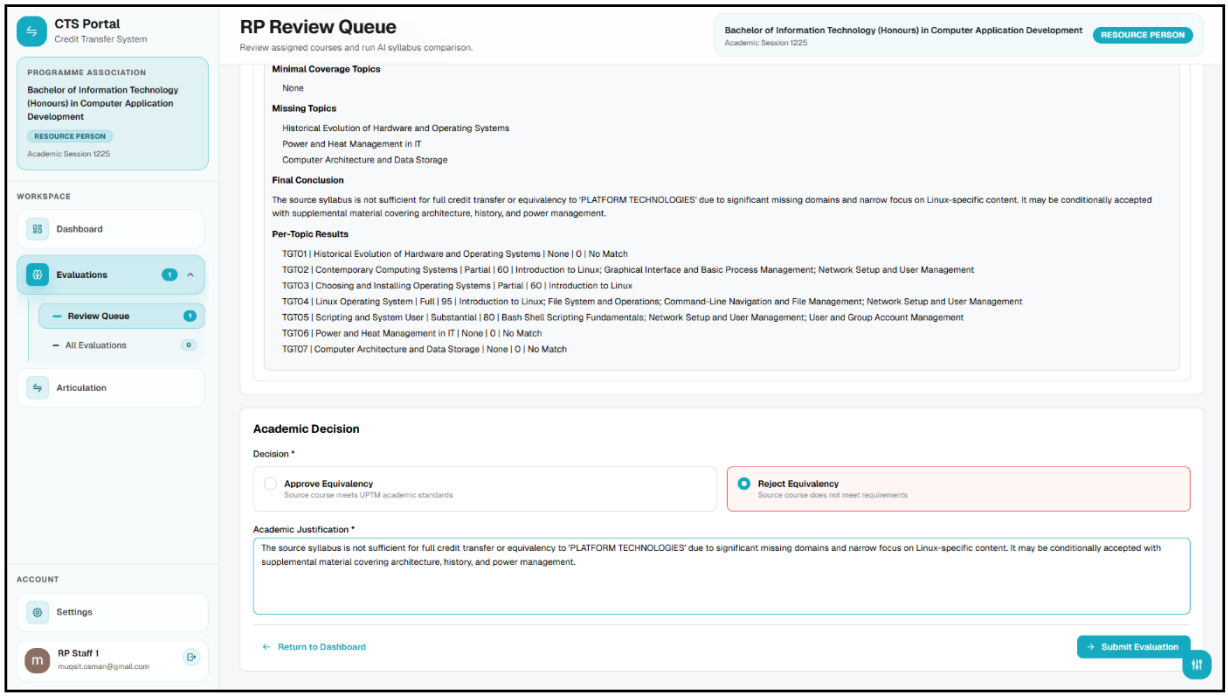


Figure 4.99: RP Review (Academic Decision)

Figure 4.99 displays the academic decision area of the RP review page. This section will not be visible until there is a comparison result for the current review group. For this area, the RP must select either one of two decision cards:

- Approve Equivalency
- Reject Equivalency

Below the two decision cards is the Academic Justification field for the RP to input their justification for the academic decision. The placeholder text in the field requests the RP to describe any differences in the academic content of the two courses or any significant differences between the courses. To complete the review, the RP will click either the return button to the RP review queue or the Submit Evaluation button. When the RP clicks the Submit Evaluation button, the decision will be written into the shared review group. As a result, any pending CTA courses that are linked to the same shared review group will also receive that decision and notes regarding the decision. Additionally, all articulation rows that are linked to the same review group will receive that decision from the RP; they will also have their articulation status changed to either APPROVED or REVOKED. The shared review group will also have its shared comparison status changed to “completed”. The applications that have been impacted by this decision will also be revalidated. Any application that has all of the courses that were assigned to RP now decided by the RP will be moved to the

EVALUATION_COMPLETED status. If the application is moved to EVALUATION_COMPLETED the RP will immediately see the HOP notification modal.

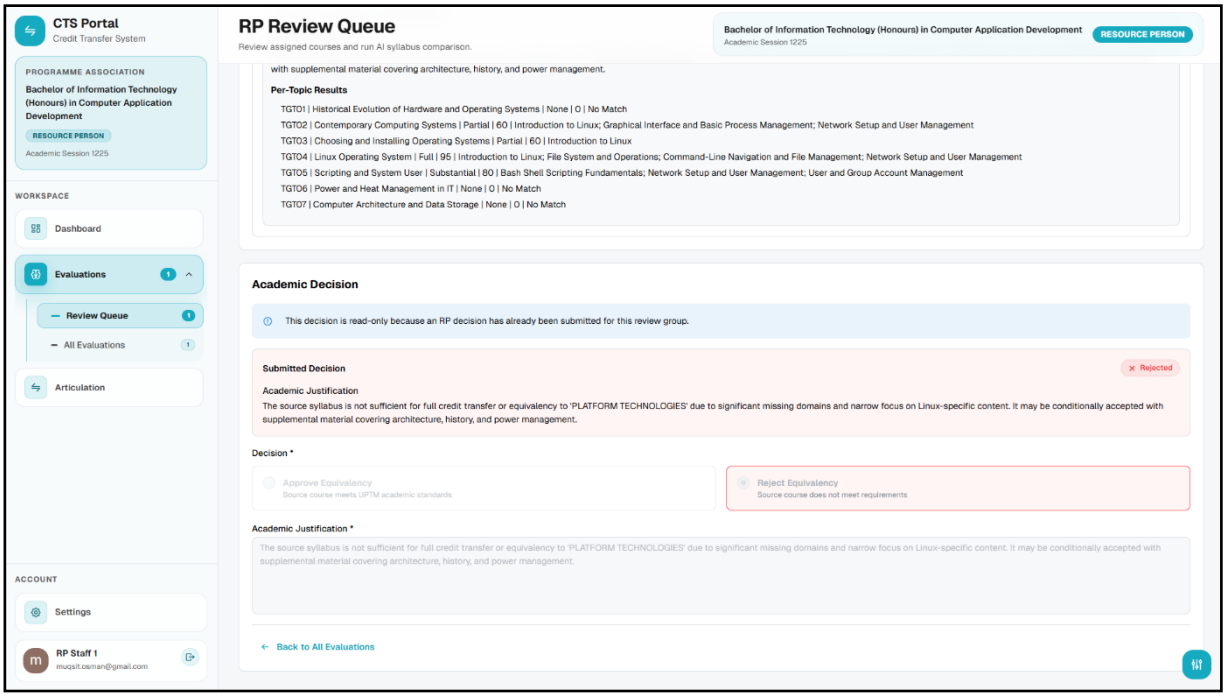


Figure 4.100: RP Review (Submitted Decision, Read-Only)

Figure 4.100 displays the academic decision area of the RP review page after the RP has submitted their decision. A blue alert will be visible on the screen stating that the RP review is now in read-only mode due to the RP having submitted a decision within the review group. Under this alert is a colored summary panel that displays the decision that was submitted by the RP and the academic justification for that decision. The decision cards and the field for entering academic justification will still be visible but will be disabled so that the RP can review the decision and justification without being able to change them.

4.7.4.4 HOP Completion Notification

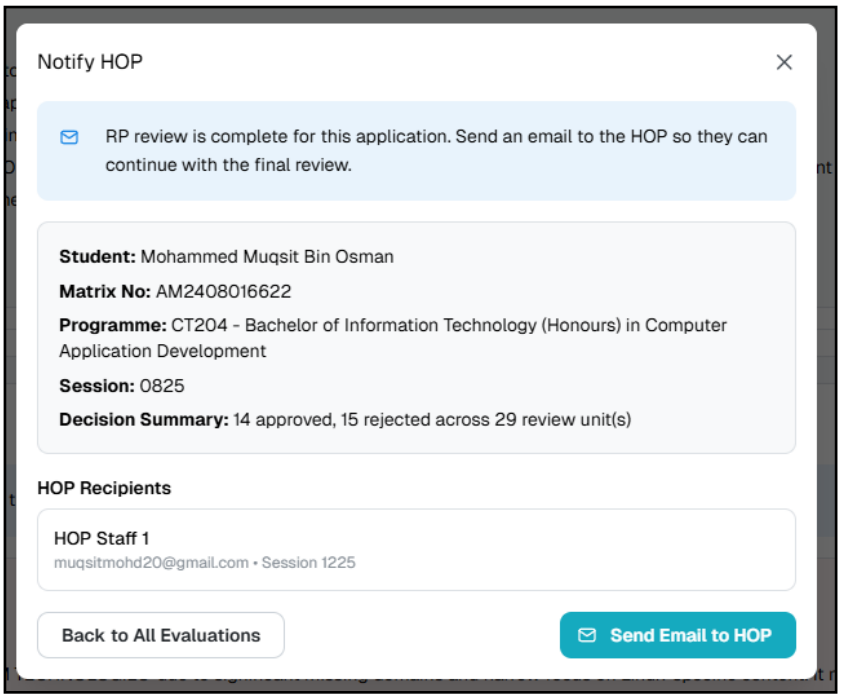


Figure 4.101: Notify HOP Modal

Figure 4.101 displays the Notify HOP modal. This modal will appear for the RP after the RP completes reviewing the remaining units for an application. Within the modal is a blue alert that explains to the RP and the HOP that the RP review is now complete and that the HOP should be able to continue with the review of the courses for the student. Following the alert is a summary panel with the following information:

1. student name
2. matrix number
3. target programme code and name
4. session code
5. Decision Summary - Number of approved courses, number of rejected courses, and total number of review units reviewed

The HOP Recipients section will feature the students that will receive an email when the HOP is notified of the completion of the review by the RP. The system will first attempt to find an HOP member that is assigned to the same academic session as the student. If no HOP is found for that session, it will attempt to find the HOP for the student's programme. Each of these recipients will feature the recipient's name, email address, and session code (if assigned to the RP). There are two buttons at the bottom of the modal. The left button will return the RP to their review queue. The rightmost button will initiate the sending of an email to the HOP members who will review the remaining courses for the student. If the system does not find an HOP member with the student's

courses, the modal will display a yellow alert and disable the Send Email to HOP button. When the RP clicks on the Send Email to HOP button, the RP will see a loading notification. Upon the completion of the sending of the email to the HOP, a success or failure alert will appear on the screen with the system's result of the attempted sending of the email to the HOP.

4.7.5 AAS Interface

The AAS interface is used by individuals who will receive the applications following the decision from HOP regarding the applications. AAS does not make decisions regarding the acceptance of the students' applications. Instead, AAS is responsible for converting the applications to exportable records, ensuring that the applications are completed by AAS, and maintaining a record of the applications that have been exported from the system. Following the step in which HOP has made their decision regarding an application, the application will appear in the AAS AAS finalized queue. AAS is able to view those applications, and to export them as XLSX or CSV files. AAS can perform the export operation to change the application's status from FINALIZED to COMPLETED and attempt to send emails to the students that indicate their completion of the application. Additionally, however, AAS is able to choose to generate another export file for those same applications without changing their status. Thus, the interface is dedicated to both exporting the applications that have been FINALIZED by HOP and REEXPORTING those same applications or applications that have been COMPLETED by AAS should another file by AAS be required.

4.7.5.1 AAS Dashboard

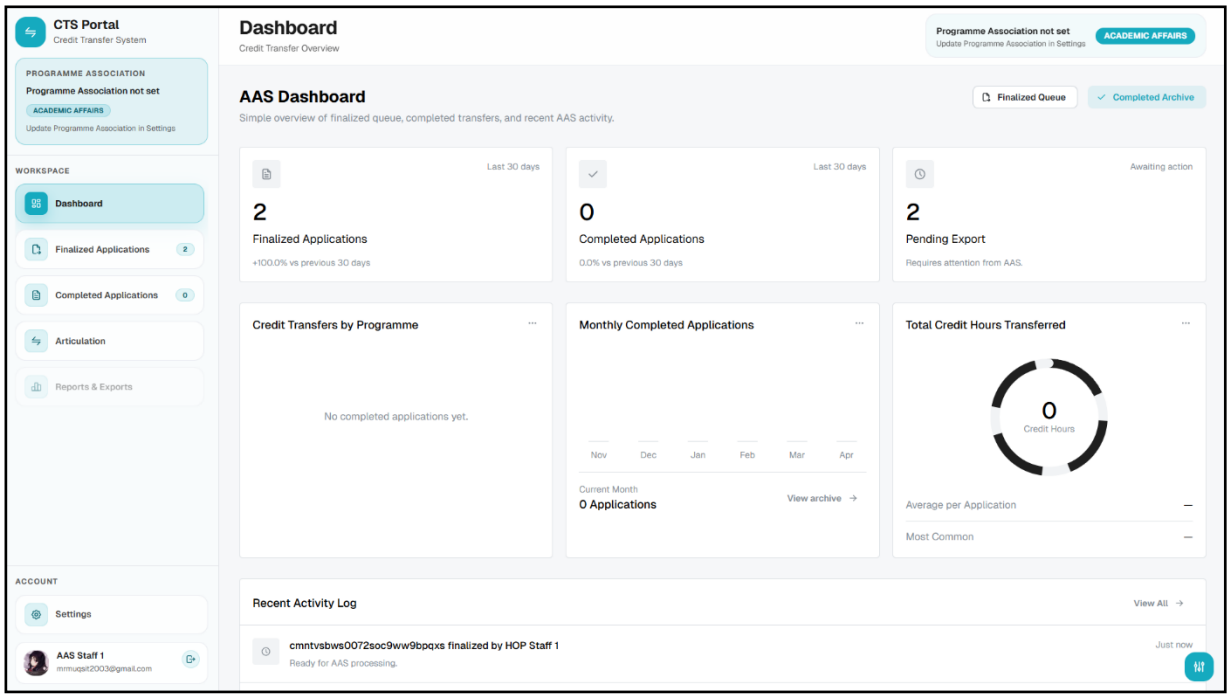


Figure 4.102: AAS Dashboard (Metric Cards and Insights)

Figure 4.102 displays the AAS dashboard. The AAS dashboard is the main screen for AAS users after they have performed sign in. The title for the screen is “AAS Dashboard,” with a shorter subtitle that describes the dashboard. Two buttons on the right side of the screen allow for users to navigate to either the “Finalized Queue” or the “Completed Archive”. On the screen are three metric cards:

1. The first metric card displays the number of applications that have been FINALIZED by HOP in the past 30 days, as well as the percentage change from the same period of time in the previous 30 days;
2. The second metric card displays the number of applications that have been COMPLETED by AAS in the past 30 days, as well as the percentage change from the same period of time in the previous 30 days;
3. The third metric card displays the number of applications that have been FINALIZED by HOP but which AAS has not yet COMPLETED. Should the count of these applications be 0, the card will read “Up to date”.

Following these three cards are three insight sections:

1. The first insight section displays a bar chart of the number of CREDIT TRANSFER applications that have been COMPLETED by AAS for each of the TOP 5 PROGRAMMES in the institution;
2. The second insight section displays a bar chart of the number of applications that have been COMPLETED in the last 6 months by AAS, with an additional line below the bar chart that displays the number of applications COMPLETED this current month for AAS, with a link to the “Completed Archive” page;
3. The third insight section displays a ring chart that displays the TOTAL CREDIT HOURS that have been TRANSFERRED to students via these applications to the institution, with two lines below the ring chart that display the average credit hours per application for AAS, as well as the most common number of credit hours for applications submitted to AAS.

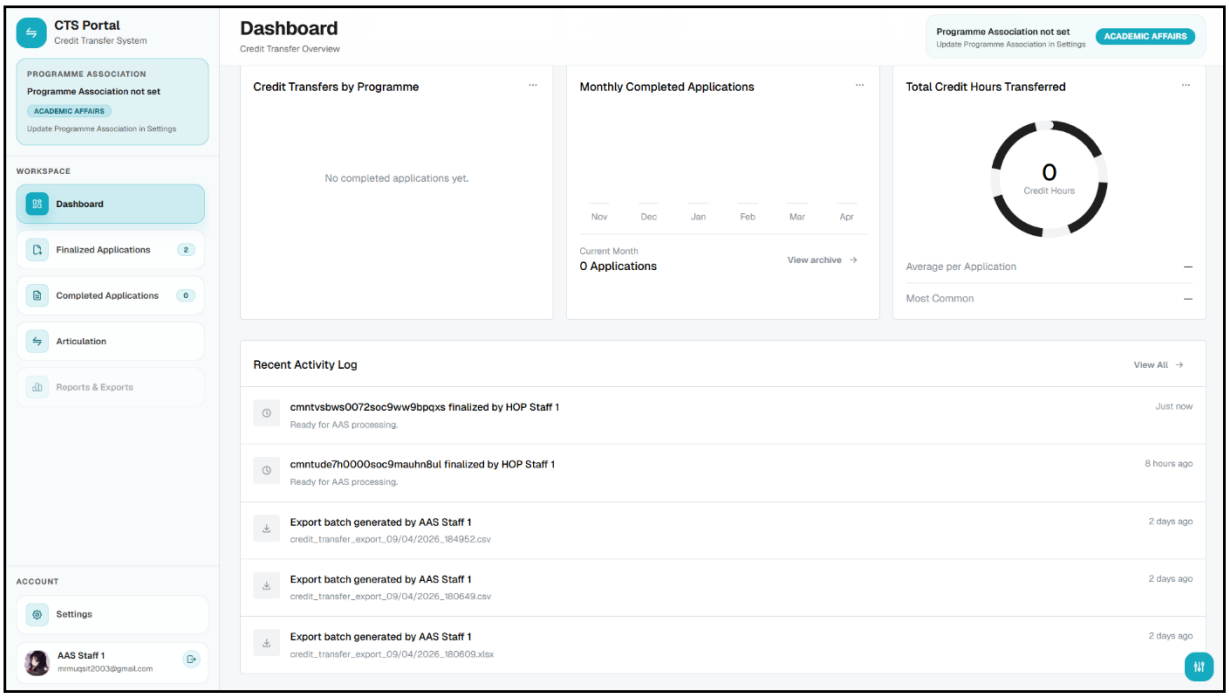


Figure 4.103: AAS Dashboard (Recent Activity Log)

Figure 4.103 displays the Recent Activity Log for AAS. This section of the AAS dashboard displays activity regarding the applications in three different categories:

1. COMPLETED applications, with their application IDs and the number of approved CREDIT HOURS that were transferred to the students;
2. FINALIZED applications, with their application IDs and the name of the individual at HOP that FINALIZED the applications for AAS;
3. EXPORTED applications, with their stored files and the name of the ORIGINAL file from which the information was exported.

Each activity has an icon that relates to that activity, a title for that activity, a description of that activity, and how many minutes or days ago that activity took place. The “View All” button linked to this list allows AAS to navigate to the “Completed Archive” page.

4.7.5.2 Finalized Queue

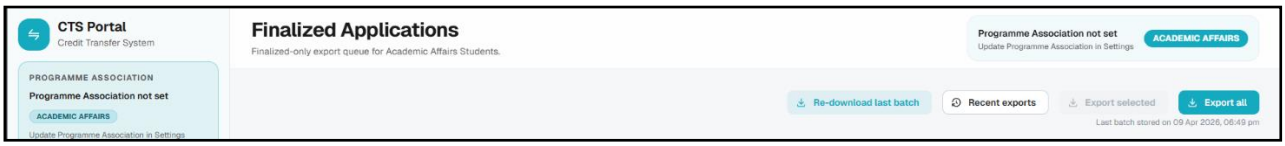


Figure 4.104: AAS Finalized Queue (Action Bar)

Figure 4.106 displays the action buttons at the top of the AAS “Finalized Queue.” This screen displays all applications that have been FINALIZED by HOP but which AAS has not yet COMPLETED. Actions that may be performed include:

1. Re-downloading the batch of applications that were exported by AAS in the last batch export;
2. Accessing the dropdown box that displays all of the applications that have been EXPORTED by AAS in the past few exports;
3. EXPORTING the applications that are selected by AAS (if there are any checked in the list);
4. EXPORTING ALL applications that meet the current filter criteria for AAS;

Additionally, below the action buttons is a timestamp for when the LAST BATCH of applications was exported by AAS.

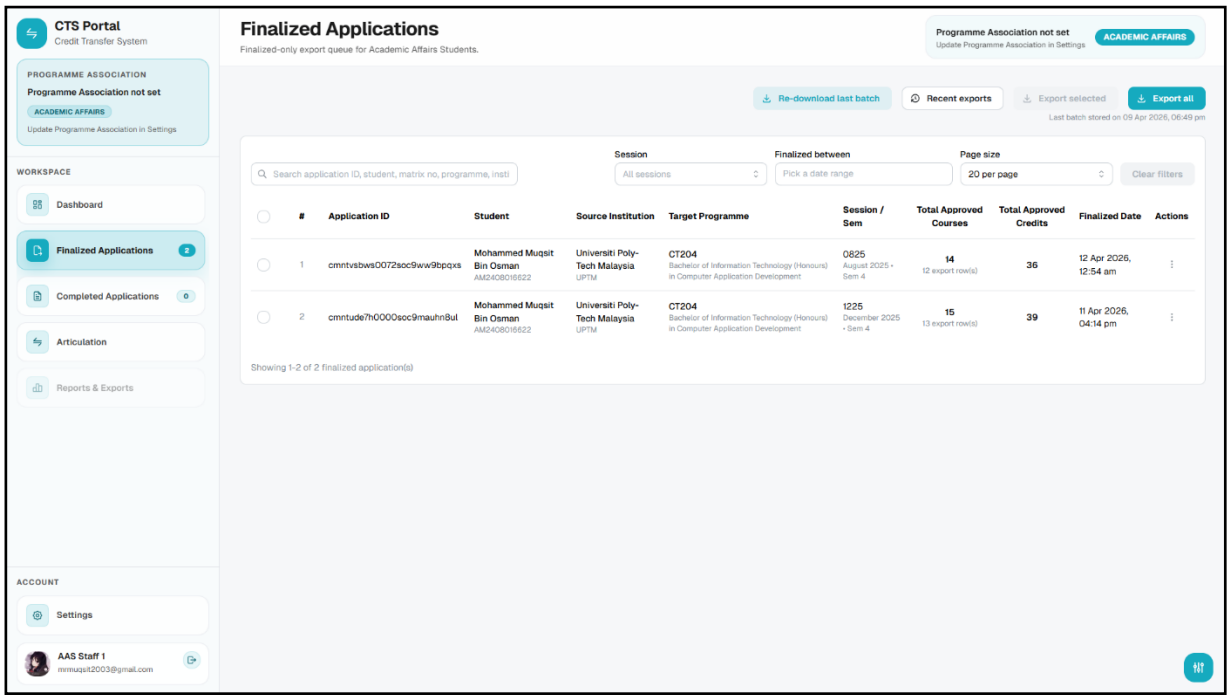


Figure 4.105: AAS Finalized Queue (Filter Bar and Data Table)

Figure 4.105 shows the filter toolbar and data table on the Finalized Queue page. The filter toolbar provides:

1. Search bar for searching across application ID, student name, matrix number, programme, or institution
2. Session filter dropdown to scope by academic session
3. Finalized between date range picker to filter by finalized date
4. Page size selector (10, 20, 50, or 100 per page)

Clear filters button to reset all active filters

The data table displays each finalized application with the following columns:

1. Checkbox for row selection (header checkbox supports select-all and indeterminate states)
2. Row number (#)
3. Application ID
4. Student name and matrix number
5. Source Institution name and code
6. Target Programme code and name
7. Session / Sem (session code, formatted session label, and semester number)
8. Total Approved Courses with export row count
9. Total Approved Credits (credit hours)
10. Finalized Date (formatted as "DD Mon YYYY, HH:MM")
11. Actions menu with three options: View (navigates to application detail), Export (opens the export modal for this single application), and History (opens the application status history modal)

Pagination controls at the bottom show the range (for example, "Showing 1-20 of 43 finalized application(s)") and page navigation.

4.7.5.3 Export Credit Transfer Application

The export modal is central to the AAS workflow. It handles scope preview, export configuration, identity confirmation, and the actual file generation with download.

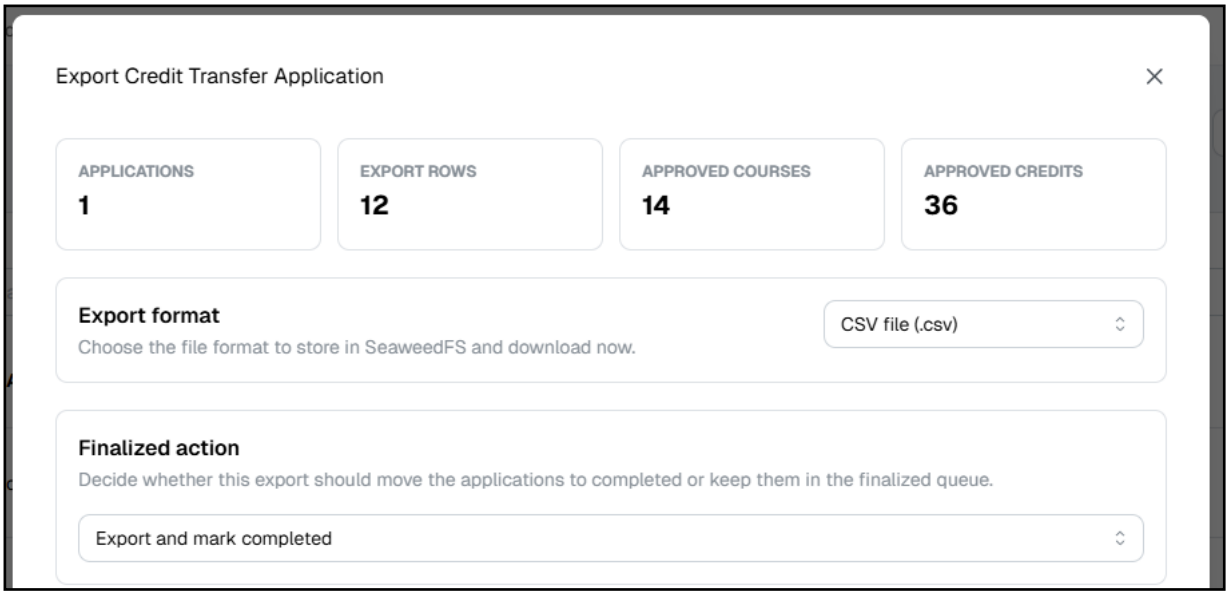


Figure 4.106: Export Modal (Preview and Format Selection)

Figure 4.106 shows the top section of the Export Credit Transfer Application modal. When AAS opens the export modal (from single application, selected batch, or filtered set), the modal first loads a server-side preview. Four preview cards at the top show:

1. Applications count
2. Export Rows count
3. Approved Courses count
4. Approved Credits (total credit hours)

Below the preview cards, the Export format section allows AAS to choose between Excel (.xlsx) and CSV (.csv) formats. The Finalized action section (only visible in finalized mode, not in completed re-export mode) lets AAS choose between:

1. "Export and mark completed" (the default, which moves applications from FINALIZED to COMPLETED and then attempts student completion emails)
2. "Re-export only and keep finalized" (generates the file without changing application status)

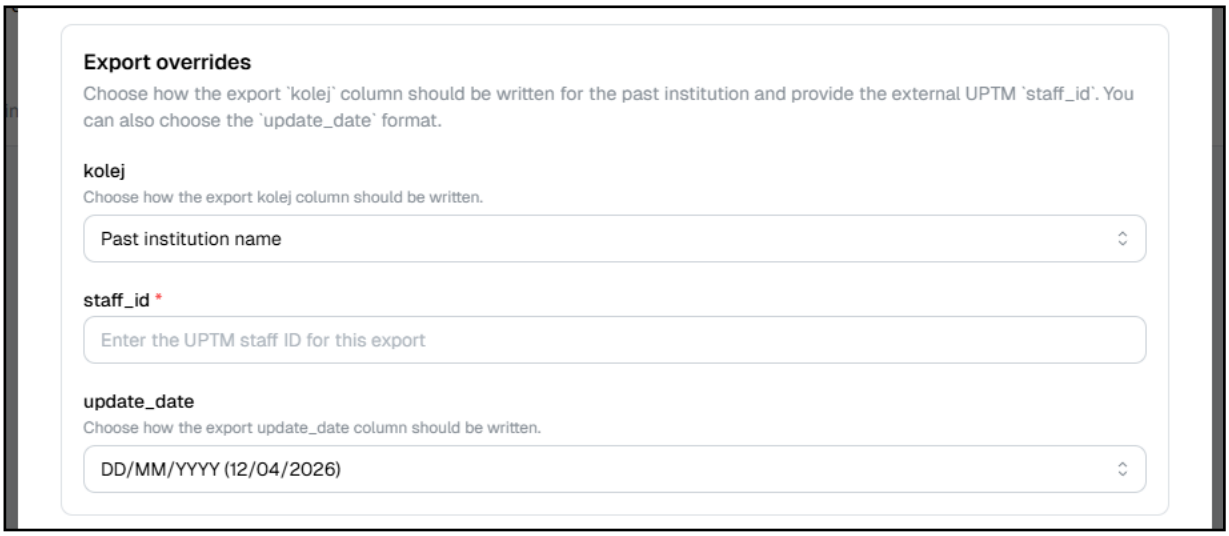


Figure 4.107: Export Modal (Export Overrides)

Figure 4.107 shows the Export Overrides section. This section provides three configuration options that control the format of the generated export file:

1. kolej dropdown to choose how the past institution column is written in the export. Options are "Past institution name" (default) or "Past institution code (shortname)"
2. staff_id text input where AAS must enter the external UPTM staff ID associated with this export. This field is required
3. update_date dropdown to choose the date format for the export. Options include YYYY-MM-DD, DD/MM/YYYY, DD-MM-YYYY, and DD-MMM-YYYY, each showing a live preview of the formatted date

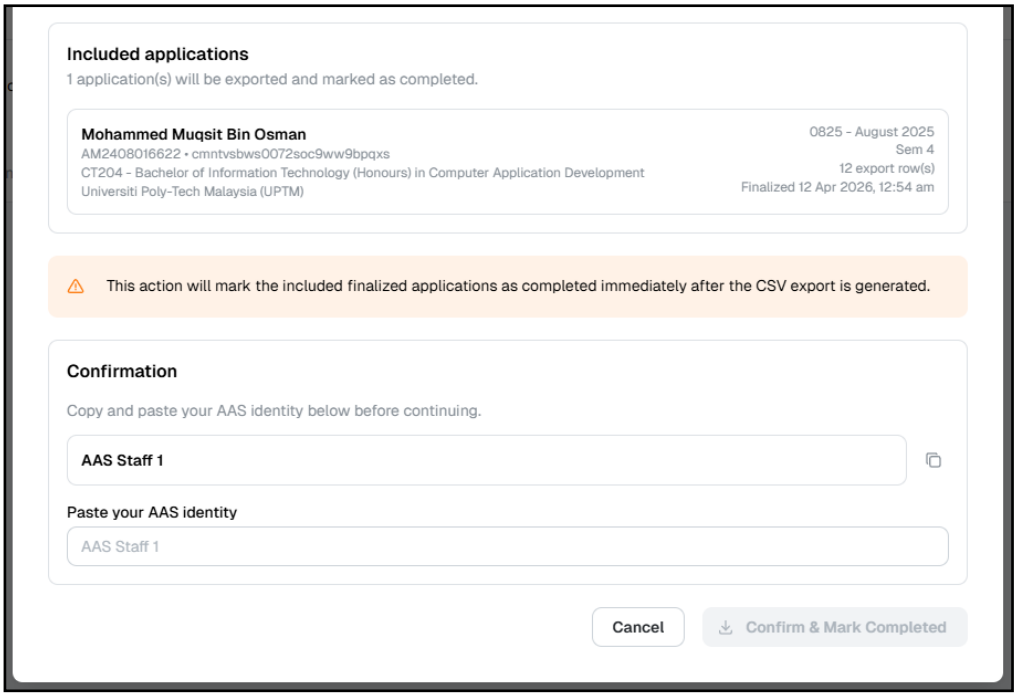


Figure 4.108: Export Modal (Included Applications and Confirmation)

Figure 4.108 shows the Export Modal with the Included Applications listing and the Confirmation section. The Included Applications listing is displayed as a scrollable panel (up to 240 pixels tall) with the following displayed for each application:

1. Student name, matrix number, and application ID
2. Target programme code and name
3. Source institution name and code
4. Session code, formatted session label, and semester number
5. Export row count
6. Finalized Date (appears during the export of finalized applications) or Completed Date (appears during the export of completed applications)

Below the application list is an orange alert that displays the following information:

1. During the export of finalized applications will be marked as “completed” after the export
2. During the re-export of completed applications will generate a new export file without changing the status of the applications

The confirmation section requires the AAS staff member to copy and paste their identity into the text field to confirm the export of the selected applications. The identity string is displayed in a bordered card with a button to copy the string. The string must be pasted into the confirmation text field for the confirm button to be enabled. If the string that is pasted into the confirmation text field is not the same as the identity string displayed in the confirmation card, the text field will display “Value does not match”. At the bottom of the modal are the “Cancel” button and either a “Confirm and Mark Completed” or “Confirm and Re-export” button based on the mode in which the export is occurring. When the confirm button is clicked, the following actions are performed by the AAS application server:

1. Resolve the approved transfer rows for the applications that were exported
2. Create the export file based on the selected format (XLSX or CSV)
3. Upload the file to object storage to create a stored export record
4. Create the CreditTransferExport table row for the transferred credits
5. Change the status of the exported applications from FINALIZED to COMPLETED (only during the export of finalized applications)
6. Create an ApplicationStatusHistory record of the action that occurred during the export

7. Attempt to send emails to the students to confirm the completion of the applications (only if changing status to COMPLETED)
8. Return the stored path to the downloaded file and automatically download the file to the browser

A notification will display to confirm the success of the export. During the export of completed applications, the notification will include information about the success of the delivery of the completion emails to the students. During the re-export of applications, the notification will only confirm the success of the generation of the export file.

4.7.5.4 Completed Archive

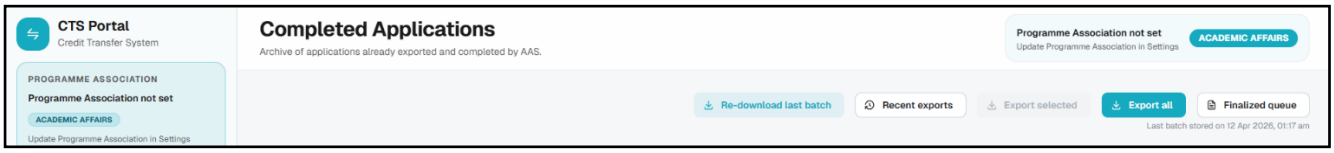


Figure 4.109: AAS Completed Archive (Action Bar)

Figure 4.109 shows the action bar that is displayed on the AAS Completed Archive page. The AAS Completed Archive page displays the following action buttons in the action bar:

1. Button to re-download the last batch of exported applications
2. Dropdown menu of the recently exported applications
3. Button to export the selected applications
4. Button to export all applications
5. Button to navigate to the Finalized Queue page

AAS Completed Archive (Filter Bar and Data Table)

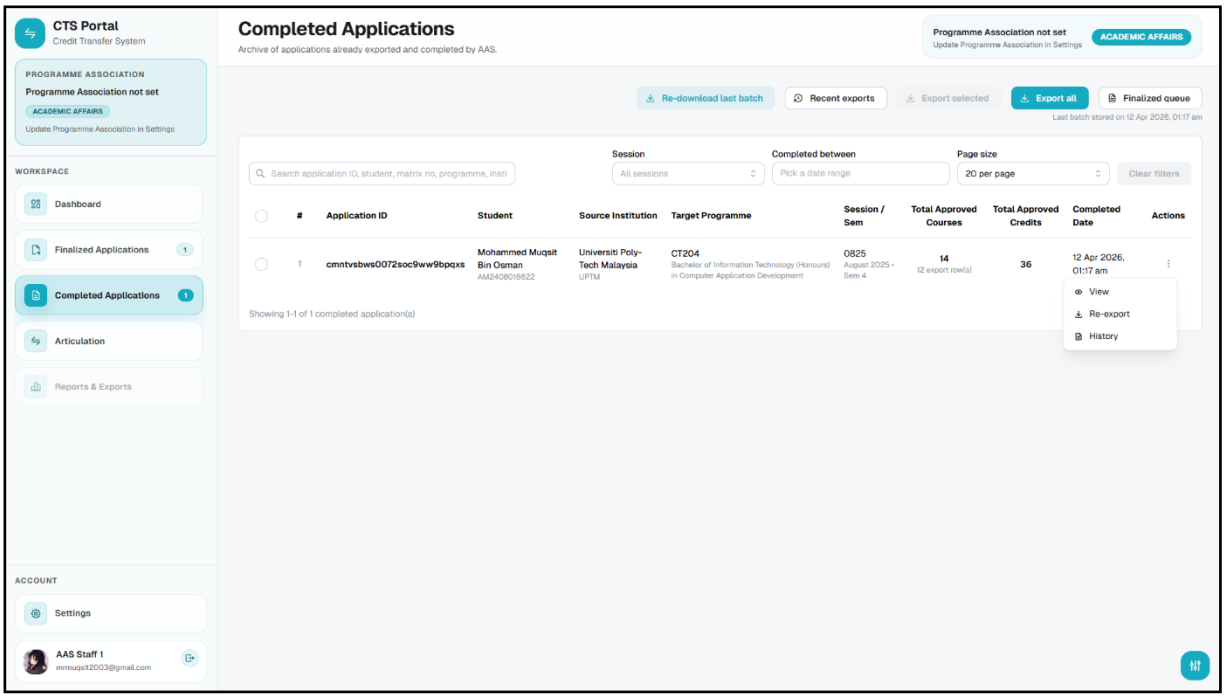


Figure 4.110: AAS Completed Archive (Filter Bar and Data Table)

Figure 4.110 shows the filter toolbar and data table on the Completed Archive page. The filter toolbar follows the same pattern as the Finalized Queue but uses:

1. Completed between date range picker (instead of "Finalized between")
2. Same search bar, session filter, page size selector, and clear filters button

The data table displays each completed application with the same column structure as the Finalized Queue, with one difference: the date column shows "Completed Date" instead of "Finalized Date." The per-row action menu provides three options:

1. View navigates to the application detail page
2. Re-export opens the export modal in completed mode and generates a new file without changing status
3. History opens the application status history modal

Pagination controls at the bottom show the range and page navigation.

4.7.5.5 Application Status History Modal

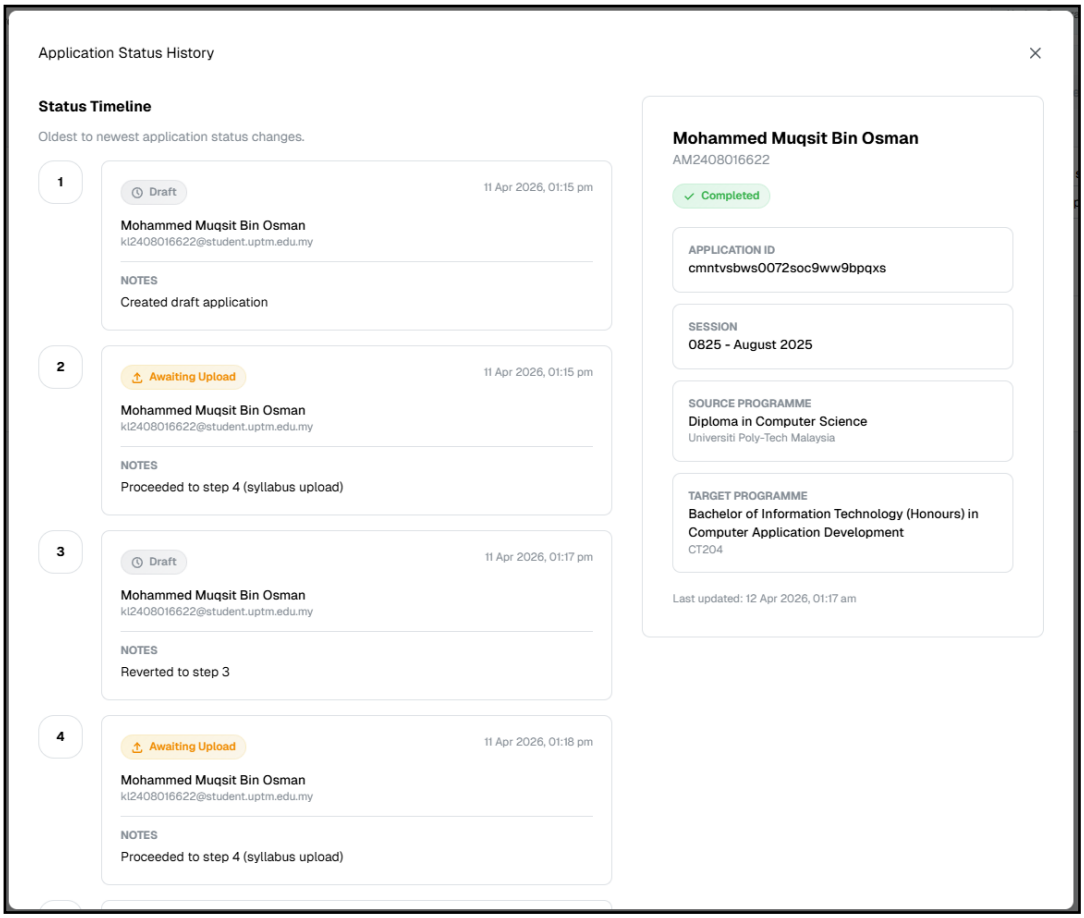


Figure 4.111: Export Modal (Preview and Format Selection)

Figure 4.111 shows the Application Status History modal, accessible from both the Finalized Queue and Completed Archive via the "History" action menu option. This modal displays a chronological timeline of all status changes for the selected application, showing:

1. Status label
2. Actor name (or "System / automated" for automated transitions)
3. Notes or reason for the status change
4. Timestamp of the transition

This modal is shared across multiple staff interfaces (HOP, AAS) and provides traceability of the application's lifecycle, including export actions and completion-email notes recorded during the AAS stage.

4.8 Conclusion

The design and implementation discussion in this chapter established how the UPTM Credit Transfer System was translated from proposal into a working web application. It brought together the interface structure, shared components, database design, implementation approach, hosting environment, and supporting tools used to run the system in practice. Taken together, these elements show that the system was not designed only at a conceptual level, but was built on a clear technical foundation that supports data management, document handling, automated processing, and role-based operation within the credit transfer workflow. The interface discussion demonstrated how that technical foundation is experienced by actual users across the full process. From authentication and student submission to HOP review, RP evaluation, and AAS export handling, each interface was designed to support a specific responsibility while remaining connected to the same application lifecycle. This confirms that the system is able to manage the credit transfer process in an orderly and traceable manner, with each stage supporting the next through consistent interface behaviour, structured decisions, and complete workflow continuity.

5 FINDINGS

5.1 Introduction

The testing of the system was performed to ensure that the developed system fulfilled the requirements stated in the tables mentioned above. There were three main categories of testing that were to be performed: non-functional testing, functional testing, and acceptance testing. Non-functional testing would help to ensure that the system was free from any issues related to its quality attributes. Functional testing would ensure that the system fulfilled the needs of each role within the system. Finally, acceptance testing would allow for the feedback of the system from those who will actually use the system.

The testing that was performed was predominantly manual testing of the system within the development environment. The system was tested on localhost utilizing the PostgreSQL database, object storage, the Stirling PDF library for PDF analysis, and the Ollama AI system for natural language analysis. Additionally, some of the helper modules for the system were also automatically tested to ensure that they were also fulfilling their purpose. Such automated testing included tests for e-mail policies, internal transcript analysis, RP review summary creation, SMTP access, syllabus comparison, and HOP slip PDF export.

5.2 Testing

Table 5.1 provides an overview of the testing that was performed for the system, as well as some general information regarding the acceptance testing that will be performed with the stakeholders of the system when the system is ready to be deployed.

Table 5.1: Testing Overview

Testing Category	Testing Type	Scope	Method
Non-Functional Testing	Performance Testing	Evaluate system response time, page loading speed, and processing speed of key modules such as transcript extraction and syllabus comparison	Manual testing and observation
Non-Functional Testing	Security Testing	Evaluate login control, role-based access, input validation, file upload restrictions, and protection against unauthorized actions	Manual testing and black-box testing

Testing Category	Testing Type	Scope	Method
Non-Functional Testing	Usability Testing	Evaluate ease of use, interface clarity, navigation consistency, and user friendliness across all four user roles	User observation and guided walkthrough
Non-Functional Testing	Reliability Testing	Evaluate system stability, data consistency, repeated execution of key workflows, and error handling behavior	Repeated manual testing
Non-Functional Testing	Maintainability Testing	Evaluate modular code structure, separation of concerns, and ability to introduce updates with minimal disruption	Code review and structural analysis
Non-Functional Testing	Traceability Testing	Evaluate record linkage, workflow status visibility, audit trail completeness, and decision provenance	Manual testing and data verification
Functional Testing	Role-Based Functional Testing (Student)	Evaluate individual student modules including onboarding, CTA submission, transcript upload, course review, syllabus upload, status tracking, and withdrawal	Manual testing and black-box testing
Functional Testing	Role-Based Functional Testing (HOP)	Evaluate HOP modules including application review, course match adjustment, articulation management, syllabus repository, RP assignment, and finalization	Manual testing and black-box testing
Functional Testing	Role-Based Functional Testing (RP)	Evaluate RP modules including review queue, syllabus comparison, evaluation evidence, AI comparison analysis, and decision submission	Manual testing and black-box testing
Functional Testing	Role-Based Functional Testing (AAS)	Evaluate AAS modules including finalized queue, export configuration, file generation, completed archive, and re-export behavior	Manual testing and black-box testing

Testing Category	Testing Type	Scope	Method
Functional Testing	Integration Testing	Evaluate the interaction between connected modules including transcript extraction to course matching, syllabus upload to comparison pipeline, RP decision to application status, and export to completion flow	Manual testing and scenario-based testing
Functional Testing	System Testing	Evaluate the complete credit transfer workflow from student submission through HOP review, RP evaluation, and AAS export	End-to-end testing
Acceptance Testing	Client Acceptance Testing	Evaluate whether the system meets the project objectives and stakeholder expectations for the UPTM credit transfer process	Planned client review and feedback
Acceptance Testing	User Acceptance Testing	Evaluate user satisfaction, usefulness, and suitability for actual use across Student, HOP, RP, and AAS roles	Planned questionnaire and user feedback

In addition to the scenario-based manual test cases listed above, there are also automated test scenarios for the codebase that verify some internal logic. Such automated testing includes the use of node:test for verifying email policy, transcript parsing, and review summaries, as well as a variety of scripts that test the SMTP server, text extraction, syllabus comparison, PDF comparison, and HOP slip export generation. These automated tests do not replace the manual testing of the system's functional requirements, but they strengthen confidence that the internal logic used by the main workflows is working as expected.

5.3 Non-Functional Testing

Non-functional testing was conducted to evaluate the quality attributes specified in Table 3.6 of Chapter 3. The test cases in this section cover performance, security, usability, reliability, maintainability, and traceability.

5.3.1 Performance Testing

Performance testing focused on the routine operations that most directly affect user experience, especially page loading, extraction time, comparison processing, and export generation. Table 5.2 summarizes the observed response times for these key activities.

Table 5.2: Performance Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-P01	Load the sign-in page from a cold start	Page loads within 3 seconds	Page loaded in approximately 1.8 seconds	Pass
NFT-P02	Load the student dashboard after sign-in	Dashboard renders with application summary within 3 seconds	Dashboard loaded in approximately 1.5 seconds	Pass
NFT-P03	Load the HOP dashboard with metric cards, charts, and queue tables	Dashboard renders all sections within 4 seconds	Dashboard loaded in approximately 2.3 seconds with all charts rendered	Pass
NFT-P04	Upload a 5-page transcript PDF and complete extraction	Extraction completes within 60 seconds	Extraction completed in approximately 35 seconds with all courses extracted	Pass
NFT-P05	Run AI syllabus comparison between source and target syllabi	Comparison completes within 120 seconds	Comparison completed in approximately 90 seconds with all stages processed	Pass
NFT-P06	Load the AAS finalized queue with 50 applications	Page loads and renders the data table within 3 seconds	Page loaded in approximately 2.1 seconds with pagination functioning	Pass
NFT-P07	Generate an XLSX export file for 20 applications	Export file generates and download begins within 10 seconds	Export generated in approximately 6 seconds and download triggered automatically	Pass
NFT-P08	Load the RP evaluation evidence page with side-by-side PDF viewers	Both PDF panels render within 5 seconds	Both PDFs rendered in approximately 3.5 seconds	Pass
NFT-P09	Navigate between CTA wizard steps (step 1 to step 4)	Each step transition completes within 2 seconds	Step transitions completed in approximately 0.8 seconds on average	Pass
NFT-P10	Load the HOP review page with 15 course rows and grouped sections	Page renders all course groups and action buttons within 3 seconds	Page loaded in approximately 2.0 seconds	Pass

5.3.2 Security Testing

Security testing examined whether authentication rules, role-based access control, file restrictions, and protected server actions behaved correctly when exposed to unauthorized or invalid requests. Table 5.3 presents the security-related test cases and their outcomes.

Table 5.3: Security Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-S01	Access the dashboard page without signing in	System redirects to the sign-in page	User was redirected to the sign-in page as expected	Pass
NFT-S02	A student account attempts to access the HOP review queue at /reviews/hop	System returns a 403 forbidden or redirects to the dashboard	System denied access and redirected the student to their own dashboard	Pass
NFT-S03	A student account attempts to access the AAS finalized page at /aas/finalized	System blocks access based on role check	System denied access with role-based guard	Pass
NFT-S04	An allowed non-staff email (for example, @kptm.edu.my or @student.uptm.edu.my) signs in and attempts to access staff routes	System allows sign-in but does not assign staff access or auto-onboard the user as staff	System correctly treated the account as non-staff and did not trigger staff auto-onboarding or staff-route access	Pass
NFT-S05	Upload a non-PDF file in the transcript upload step	System rejects the file and displays a validation error	System displayed "Only PDF files are accepted" and prevented upload	Pass
NFT-S06	Upload a non-PDF/XLSX file in the syllabus upload step	System rejects the file and displays a validation error	System displayed a file type validation error and prevented upload	Pass
NFT-S07	Attempt to access another student's application detail page	System blocks access and returns an error or redirects	System returned a 404 page, preventing cross-user access	Pass
NFT-S08	Attempt to download an AAS export file using a non-AAS account	System blocks the download request	System returned 403 for unauthorized download attempt	Pass
NFT-S09	Submit a CTA course decision without the HOP role	Server action rejects the request with role validation	Server returned an authentication error and did not process the action	Pass
NFT-S10	Access the settings page and verify session management	System displays the current session and supports sign-out	Session information displayed correctly and sign-out functioned as expected	Pass

5.3.3 Usability Testing

Usability testing focused on whether each role could understand and complete the main tasks through the provided interface labels, page structure, and feedback messages without additional technical guidance. Table 5.4 summarizes the observations recorded during these checks.

Table 5.4: Usability Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-U01	Navigate the student CTA wizard from step 1 to step 4 without guidance	User can complete all steps using interface labels and instructions alone	Users completed the wizard without external assistance, guided by step labels and field placeholders	Pass
NFT-U02	Identify the current application status from the student dashboard	Status badges and labels are clear and unambiguous	Users correctly identified their application status from the colored status badges	Pass
NFT-U03	Use the HOP review page to locate flagged courses	Flagged courses are visually distinct and grouped separately	Users located flagged courses through the separate "Flagged Review Items" section with flag icons	Pass
NFT-U04	Use the RP side-by-side evidence viewer to compare syllabi	Both panels are visible simultaneously with clear labeling	Users confirmed the split-panel layout was clear and the target/source labels were easy to distinguish	Pass
NFT-U05	Use the AAS export modal to configure and confirm an export	All configuration options are understandable and the confirmation step is clear	Users completed the export configuration without confusion, noting the identity confirmation as a helpful safety measure	Pass
NFT-U06	Use the search and filter controls on the HOP review queue	Filters respond immediately and results update in real time	Filters applied correctly with debounced search and the clear filters button reset all fields	Pass
NFT-U07	Access the system on a tablet-sized viewport (768px width)	Layout adapts without horizontal scrolling on main content areas	Interface adapted to smaller widths with responsive grid layouts and scroll containers on data tables	Pass
NFT-U08	Understand notification messages after submitting a decision	Toast notifications provide clear confirmation of the action taken	Users confirmed that success and error notifications were descriptive and correctly colored	Pass

5.3.4 Reliability Testing

Reliability testing evaluated whether repeated actions, interrupted navigation, re-runs, and status updates could be handled without data loss, duplication, or workflow inconsistency. Table 5.5 shows the reliability checks carried out for the main operational scenarios.

Table 5.5: Usability Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-R01	Submit the same CTA application form twice by double-clicking the submit button	System prevents duplicate submission and processes only one request	System disabled the submit button after the first click and processed only one submission	Pass
NFT-R02	Upload a transcript, navigate away, and return to step 2	Previously uploaded transcript data persists correctly	Extracted courses were preserved and displayed correctly on return	Pass
NFT-R03	Run the AI comparison twice on the same review group	System handles re-run without data corruption and updates results	Second comparison replaced the first with updated results and no duplicate records were created	Pass
NFT-R04	Export the same finalized application twice using the re-export option	System generates a new file without changing application status	New export file was generated and stored separately while the application remained in its current status	Pass
NFT-R05	Withdraw an application and verify that it cannot be modified further	Application enters WITHDRAWN status and all edit actions are disabled	Application was marked WITHDRAWN and the system prevented further edits or status changes	Pass
NFT-R06	Verify that the application status history records all transitions accurately	Each status change creates an ApplicationStatusHistory entry with actor and timestamp	All transitions from DRAFT through COMPLETED were recorded with correct actors, timestamps, and notes	Pass
NFT-R07	Upload a scanned PDF transcript with no selectable text	System falls back to vision-model extraction path	System detected the scanned PDF and used the vision extraction fallback, producing extracted courses	Pass

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-R08	Finalize an application and verify that the student view updates	Student application detail shows FINALIZED status with no edit options	Student view correctly reflected the FINALIZED status with all action buttons disabled	Pass

5.3.5 Maintainability Testing

Maintainability testing reviewed the implementation structure behind the system, with attention to modular organization, shared helper logic, schema management, and the availability of lightweight regression checks. Table 5.6 summarizes these maintainability observations.

Table 5.6: Maintainability Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-M01	Verify that the system uses modular file structure with separated components	Source code follows a structured directory layout with separated concerns	The codebase uses Next.js App Router conventions with separated routes, components, server actions, and library modules	Pass
NFT-M02	Verify that the database schema uses Prisma with migration support	Schema changes can be applied through migration commands	Prisma schema was up to date and migrations could be generated and applied without issues	Pass
NFT-M03	Verify that shared logic is centralized in reusable library modules	Common functions (auth helpers, storage utilities, email templates) are not duplicated across routes	Shared logic was centralized under src/lib/ with auth helpers, storage utilities, database access, and email functions reused across all role modules	Pass
NFT-M04	Verify that critical helper modules support lightweight regression checks	Repository includes focused automated or script-based verification for selected internal logic	The codebase contains targeted node:test coverage and script-based checks for email policy, transcript parsing, RP summary helpers, SMTP, text extraction, staged syllabus comparison, and HOP slip export generation	Pass

5.3.6 Traceability Testing

Traceability testing verified whether the system preserves a clear record of application progress, review decisions, and export activity so that actions can be traced back to the responsible stage and actor. Table 5.7 presents the traceability results.

Table 5.7: Traceability Testing Results

Test ID	Test Description	Expected Result	Actual Result	Status
NFT-T01	View the full status history of a completed application from draft to completion	All status transitions are recorded and viewable in the status history modal	Status History modal displayed all transitions from DRAFT, PENDING_HOP_REVIEW, FINALIZED, to COMPLETED with correct actors and timestamps	Pass
NFT-T02	Trace an RP decision back to its shared review group and affected applications	The decision outcome is visible on all linked CTA courses and articulation records	RP decision propagated correctly across all linked courses and the review group displayed the applied outcome	Pass
NFT-T03	Verify that export files are stored with metadata linking them to the AAS actor	StoredFile records include the uploader reference and file type	Export files were stored with correct uploader ID, EXPORT file type, and accessible through the Recent Exports menu	Pass

5.4 Functional Testing

The purpose of functional testing is to ensure that the system fulfills the functional requirements as listed in the tables in Chapter 3. The test cases for each user role correspond to the functional requirements for each of those roles.

5.4.1 Student Functional Testing Results

Student testing was performed to assess whether the software fulfills the requirements of students from start to finish (sign in, onboarding, making a CTA, extracting transcripts, uploading syllabi, and applying to programs). Table 5.8 presents the results of testing the system from the perspective of students against the requirements listed in Table 3.2.

Table 5.8: Student Functional Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Google OAuth Login	Auth Module -> Student Module	1. Navigate to the sign-in page. 2. Click "Sign in with Google". 3. Complete Google OAuth. 4. Verify redirect to dashboard.	Student is authenticated and redirected to the student dashboard.	Student was authenticated via Google OAuth and landed on the student dashboard with correct user details displayed.	Pass
Student Onboarding	Auth Module -> Onboarding Module -> CTA Module	1. Sign in as a new student. 2. Fill in all onboarding fields (name, matrix number, MyKad, contact, semester, institution, branch, programme, session). 3. Submit the onboarding form.	System creates a draft CTA and redirects the student to the CTA wizard.	Onboarding form submitted successfully. A draft CreditTransferApplication was created with DRAFT status and the student was redirected to step 1.	Pass
CTA Metadata Entry	CTA Step 1 Module	1. Navigate to CTA step 1. 2. Fill in application metadata (source graduation year, CGPA, programme code, programme name, credit transfer type). 3. Click "Save & Continue".	Application metadata is saved and the student proceeds to step 2.	All metadata fields were saved correctly and the student advanced to the transcript upload step.	Pass
Transcript Upload and Extraction	CTA Step 2 Module -> Transcript Extraction Module	1. Navigate to CTA step 2. 2. Upload a valid PDF transcript file. 3. Wait for extraction to complete.	System extracts courses from the transcript and displays them in the course table.	Transcript was uploaded to object storage, and Stirling PDF extraction produced structured course rows displayed in the table.	Pass

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Extracted Course Review	CTA Step 3 Module -> Eligibility Module	1. Complete step 2 with extracted courses. 2. Navigate to step 3. 3. Review the course eligibility buckets (Automatically Approved, Pending Syllabus, Grade Below C, etc.).	Courses are grouped by eligibility status and displayed correctly.	Courses were categorized into the correct eligibility buckets with visual badges and counts matching the extraction results.	Pass
PDF Syllabus Upload	CTA Step 4 Module -> Syllabus Module	1. Proceed to step 4 with flagged courses requiring syllabus. 2. Upload a PDF syllabus for a flagged course. 3. Verify the upload status indicator.	Syllabus file is stored and linked to the course with extraction initiated.	Syllabus PDF was uploaded to object storage, and a Syllabus record was created with SOURCE type. The course status updated to reflect the upload.	Pass
XLSX Syllabus Upload	CTA Step 4 Module -> Syllabus Module	1. Proceed to step 4. 2. Upload an XLSX workbook file. 3. Select a worksheet from the sheet selector. 4. Confirm the upload.	System stores the XLSX file and records the selected worksheet name.	XLSX file was uploaded and the worksheet selector appeared. After selecting a sheet, the Syllabus record was created with the correct worksheet name stored.	Pass
Upload Later Request	CTA Step 4 Module -> Syllabus Placeholder Module	1. Proceed to step 4. 2. Select "Upload Later" for a flagged course. 3. Submit the step.	System creates a placeholder syllabus, sets a two-week deadline, and the application enters AWAITING_ SYLLABUS_UPLOAD.	Placeholder file (SYLLABUS_ UNAVAILABLE.txt) was created and the course received a syllabusDeadline. Application status became AWAITING_ SYLLABUS_UPLOAD.	Pass
Application Status Tracking	Student Dashboard -> Application Detail Module	1. Sign in as a student with an existing application. 2. Navigate to the applications list. 3. Open the application detail page.	Application list shows status badges and the detail page shows full application metadata with course-level decisions.	Applications list displayed correct status badges. Detail page showed all metadata, course decisions, effective syllabus information, and progress summary.	Pass
CTA Slip Download	Application Detail Module -> Slip Export Module	1. Sign in as a student with a COMPLETED application. 2. Navigate to the application detail page. 3. Click the export/download CTA slip button.	System generates and downloads the credit transfer slip document.	The CTA slip was generated and downloaded as a formatted document containing the approved course transfer details.	Pass

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Withdraw Application	Application Detail Module -> Status Module	1. Sign in as a student with a PENDING_HOP_REVIEW application. 2. Navigate to the application detail page. 3. Click "Withdraw Application". 4. Confirm the withdrawal with a reason.	Application status changes to WITHDRAWN and all edit actions are disabled.	Application was marked WITHDRAWN with the reason recorded. The withdrawal timestamp and actor were stored in the status history.	Pass
Withdrawal Restriction	Application Detail Module -> Status Guard	1. Sign in as a student with a FINALIZED application. 2. Navigate to the application detail page. 3. Verify that the withdraw button is not available.	Withdraw option is hidden or disabled for FINALIZED and COMPLETED applications.	The withdraw button was not displayed for the FINALIZED application, preventing withdrawal at this stage.	Pass
Add Course Manually	CTA Step 2 Module -> Course Editor Module	1. Navigate to CTA step 2 after extraction. 2. Click "Add Course Manually". 3. Fill in course details. 4. Save the new course.	The manually added course appears in the course table.	Manually added course appeared in the table and was processed through the matching and eligibility pipeline.	Pass
Edit Extracted Course	CTA Step 2 Module -> Course Editor Module	1. Navigate to CTA step 2. 2. Click edit on an extracted course. 3. Modify the course code or name. 4. Save the changes.	Course details are updated and re-matching is triggered.	Course details were updated and the system re-evaluated matching and eligibility rules for the modified course.	Pass
Student Notification Email	Notification Module -> Student Module	1. Complete a workflow action that triggers a notification (for example, HOP finalization). 2. Check the student email.	Student receives an email notification about the status change.	Completion email was sent to the student's email address with the correct application details and status information.	Pass

5.4.2 HOP Functional Testing

HOP functional testing focused on the academic review workflow, including application checking, course decision adjustment, articulation management, syllabus handling, RP assignment, and finalization. Table 5.9 summarizes the HOP-side functional checks against the requirements in Table 3.3.

Table 5.9: HOP Functional Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Staff Sign-In and Access	Auth Module -> Staff Awaiting Module -> HOP Dashboard	1. Sign in with a UPTM staff email (uptm.edu.my). 2. Verify that the system assigns the staff awaiting role. 3. After role assignment, verify HOP dashboard access.	HOP is authenticated and can access the HOP dashboard with role-based menu items.	Staff email was recognized, user was routed through staff awaiting, and after role assignment the HOP dashboard loaded with correct menu items.	Pass
Application Review	HOP Queue Module -> HOP Review Module	1. Navigate to the HOP review queue. 2. Open a PENDING_HOP_REVIEW application. 3. Review application details, transcript, course mappings, and documents.	HOP can view all application data including student info, course list, and attached files.	Review page displayed all application sections: overview, course groups (auto-approved, direct-approved, flagged, pending), evidence files, and status timeline.	Pass
Course Decision Adjustment	HOP Review Module -> Course Decision Module	1. Open an application with matched courses. 2. Review the auto-approved and pending course lists. 3. Override a course decision by selecting a different outcome.	HOP can see system-suggested matches and adjust individual course decisions.	Course match suggestions were displayed with recommendation sources. HOP successfully overrode a pending course to approved with justification notes.	Pass
Articulation Management	Articulation Module -> Import Module	1. Navigate to the Articulation Management page. 2. Create a new articulation mapping. 3. Edit an existing mapping. 4. Import articulation rules from a spreadsheet.	HOP can create, edit, and import articulation records.	New articulation mapping was created with source and target course details. Existing mapping was edited. Import from spreadsheet created multiple articulation rows.	Pass
Syllabus Repository Management	Syllabus Repository Module	1. Navigate to the Syllabus Repository. 2. Upload a new target-profile syllabus PDF. 3. View the uploaded syllabus. 4. Delete a syllabus.	HOP can upload, view, and delete syllabus records in the repository.	Target-profile syllabus was uploaded and stored with the target course association. The syllabus appeared in the repository list, and deletion removed both the record and the stored file.	Pass

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Request Missing Syllabus	HOP Review Module -> Notification Module	1. Open an application with flagged courses missing syllabus. 2. Click "Request Syllabus Upload". 3. Verify that the application status changes and the student is notified.	Application moves to AWAITING_ SYLLABUS_ UPLOAD and the student receives an email notification to upload the missing syllabus.	Application status changed to AWAITING_ SYLLABUS_ UPLOAD. Student received an email notification with details about the required syllabus upload.	Pass
RP Assignment	HOP Review Module -> RP Assignment Module	1. Open an application with flagged courses that have uploaded syllabi. 2. Click "Assign RP". 3. Select an RP from the staff list. 4. Confirm the assignment.	Flagged courses are assigned to the selected RP for evaluation.	RP assignment was recorded. The RP's queue updated to show the new review item. Application status reflected the pending RP evaluation.	Pass
Application Finalization	HOP Review Module -> Finalization Module	1. Open an application where all courses have decisions. 2. Click "Finalize Application". 3. Confirm the finalization.	Application status changes to FINALIZED and all course decisions are locked.	Application was finalized with FINALIZED status. All course decisions were locked and the finalization timestamp was recorded in the status history.	Pass
Reports and Analytics Review	Reports Module -> Export Module	1. Navigate to the Reports and Analytics section. 2. View the Executive Summary, Institution Insights, and Queue/SLA tabs. 3. Export a report dataset.	Reports display computed metrics and charts. Export generates a downloadable file.	Executive Summary showed overall metrics. Institution Insights displayed per-institution breakdowns. Queue/SLA showed processing time data. Reports exported as CSV/XLSX.	Pass
Academic Registry Management	Institution Module -> Branch Module -> Programme Module	1. Navigate to Institutions, Branches, and Programmes pages. 2. Add a new institution. 3. Edit a branch. 4. View programme details.	HOP can manage academic registry data across all three entity types.	New institution was added. Branch details were edited. Programme information was viewable with associated articulation counts.	Pass

5.4.3 RP Functional Testing

RP functional testing focused on the evaluation workflow after assignment, including evidence review, staged syllabus comparison, academic justification, shared decision submission, and post-completion notification. Table 5.10 summarizes the RP-side functional checks against the requirements in Table 3.4.

Table 5.10: RP Functional Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
RP Sign-In and Access	Auth Module -> RP Dashboard -> RP Queue Module	1. Sign in with an RP-assigned staff account. 2. Verify that the RP dashboard loads. 3. Confirm that RP-specific menu items appear.	RP is authenticated and can access the RP dashboard and evaluation queue.	RP dashboard loaded with KPI cards, pipeline progress, and queue preview table. RP-specific navigation items were displayed.	Pass
Assigned Evaluation Queue	RP Queue Module	1. Navigate to the RP evaluation queue. 2. Verify that only assigned review items are shown. 3. Check that metric cards display correct counts.	Queue shows only items assigned to this RP with accurate workload metrics.	Queue displayed only assigned review items. Metric cards showed correct counts for Assigned, Pending Extraction, Comparisons Completed, and Fully Decided.	Pass
Run AI Syllabus Comparison	RP Review Module -> Shared Comparison Module	1. Open a review item with available source and target syllabi. 2. View the side-by-side evidence panel. 3. Click "Run AI Comparison". 4. Wait for the staged comparison to complete.	AI comparison processes all stages and displays coverage results.	Comparison ran through all stages (Preprocess, Standardize, Combine, Mapping, Verification, Scoring, Summary). Coverage percentage, topic scores, and per-unit results were displayed.	Pass
Comparison Summary Review	RP Review Module -> Comparison Summary Module	1. Complete an AI comparison. 2. Review the comparison results section. 3. Inspect per-target unit results, coverage labels, and the generated report.	Summary displays coverage percentage, recommendation badge, unit-level results, and a downloadable report.	Coverage percentage and recommendation badge (Recommended/Further Review Required) were displayed. Per-unit results showed topic labels, scores, and evidence. Report was downloadable as markdown.	Pass
Academic Justification Entry	RP Review Module -> Decision Form Module	1. After reviewing comparison results, enter academic justification text in the justification textarea. 2. Verify that the text is preserved.	RP can enter detailed academic remarks supporting their evaluation decision.	Justification text was entered and preserved in the textarea. The placeholder prompted for analysis of learning outcomes and content coverage.	Pass

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Approve Equivalency Decision	RP Review Module -> Shared Decision Module	1. Select "Approve Equivalency" in the Academic Decision section. 2. Enter justification text. 3. Click "Submit Evaluation".	Decision is recorded as APPROVED and propagated to all linked CTA courses.	Decision was submitted as APPROVED. All linked CTA courses and articulation records received the approved outcome. Status history was updated.	Pass
Reject Equivalency Decision	RP Review Module -> Shared Decision Module	1. Select "Reject Equivalency" in the Academic Decision section. 2. Enter justification text. 3. Click "Submit Evaluation".	Decision is recorded as REJECTED and propagated to all linked records.	Decision was submitted as REJECTED. Linked records were updated accordingly. The decision became read-only after submission.	Pass
Target Syllabus Availability Check	RP Review Module -> Syllabus Repository Module	1. Open the evaluation evidence section. 2. Click "Check Target Syllabus Availability". 3. Verify that the mapped target syllabus is either loaded or reported as unavailable.	RP can verify whether the mapped UPTM target syllabus exists before running comparison.	The target syllabus check resolved the mapped target syllabus. When it existed, the preview loaded correctly; when it was unavailable, the page displayed a warning with a link to the Syllabus Repository.	Pass
Notify HOP After Completion	RP Review Module -> Notification Module	1. Submit the final decision that completes all review units for an application. 2. Verify the Notify HOP modal appears. 3. Click "Send Email to HOP".	HOP notification modal shows summary and sends email to the assigned HOP.	Notify HOP modal appeared with student details, decision summary, and recipient list. The system attempted delivery to the configured HOP recipient and confirmed the result through the notification flow.	Pass
Completed Evaluation History	All Evaluations Module -> RP Review Module	1. Navigate to the All Evaluations page. 2. Filter by review source (CTA or Articulation). 3. Search for a specific evaluation. 4. Navigate to a completed evaluation detail.	All completed evaluations are listed with filtering, pagination, and navigation to read-only detail pages.	All Evaluations page displayed completed items with working filters, pagination, and clickable rows leading to read-only review pages.	Pass

5.4.4 AAS Functional Testing

AAS functional testing focused on the final administrative stage, including finalized queue handling, export configuration, archive access, re-export behavior, and completion updates. Table 5.11 summarizes the AAS-side functional checks against the requirements in Table 3.5.

Table 5.11: AAS Functional Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
AAS Sign-In and Access	Auth Module -> AAS Dashboard	1. Sign in with an AAS-assigned staff account. 2. Verify that the AAS dashboard loads. 3. Confirm that AAS-specific menu items appear.	AAS is authenticated and can access the AAS dashboard, finalized queue, and completed archive.	AAS dashboard loaded with metric cards, programme breakdown, monthly chart, and activity log. AAS menu items were displayed.	Pass
Finalized Queue Review	AAS Finalized Queue Module	1. Navigate to the AAS Finalized Queue. 2. Verify that only FINALIZED applications are shown. 3. Use search and session filter to narrow results.	Queue shows finalized applications with filtering, pagination, and selection controls.	Finalized queue displayed only FINALIZED applications. Search, session filter, date range picker, and pagination worked correctly.	Pass
Single Export and Completion	Finalized Queue Module -> Export Modal -> Completion Module	1. Open the action menu on a finalized application. 2. Click "Export". 3. Review the default CSV format, then configure kolej mode, staff_id, and update_date format. 4. Paste identity confirmation. 5. Click "Confirm & Mark Completed".	System generates the export file, downloads it, and marks the application as COMPLETED.	Export modal loaded preview cards with CSV selected by default in finalized mode. After configuration and confirmation, the CSV file was generated, downloaded automatically, and the application status changed to COMPLETED.	Pass
Batch Export	Finalized Queue Module -> Export Modal	1. Select multiple applications using checkboxes. 2. Click "Export selected". 3. Complete the export configuration and confirmation.	System generates a batch export file covering all selected applications.	Batch export included all selected applications. Preview showed correct aggregate counts. File was generated and downloaded with all selected records.	Pass
Filtered Export	Finalized Queue Module -> Export Modal	1. Apply session and date range filters. 2. Click "Export all". 3. Complete the export configuration.	System exports all applications matching the current filter criteria.	Export covered all applications matching the active filters. Preview showed the correct count and the file was generated accordingly.	Pass

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Recent Export Download	Recent Exports Module -> File Download Module	1. Click "Recent exports" dropdown. 2. Verify that previous export files are listed with timestamps. 3. Click a previous export entry. 4. Verify the file downloads.	AAS can view and re-download previously generated export files.	Recent exports dropdown listed previous exports with creation timestamps and filenames. Clicking an entry triggered the download of the stored file from object storage.	Pass
Mark Application Completed	Finalized Queue Module -> Completed Archive Module	1. Export a finalized application using "Export and mark completed". 2. Verify the application no longer appears in the Finalized Queue. 3. Navigate to the Completed Archive. 4. Verify the application appears there.	Application transitions from FINALIZED to COMPLETED and moves from the finalized queue to the completed archive.	Application was removed from the Finalized Queue after export. The Completed Archive showed the application with the correct completed date and student received a completion email.	Pass
Re-export Completed Application	Completed Archive Module -> Export Modal	1. Navigate to the Completed Archive. 2. Open the action menu on a completed application. 3. Click "Re-export". 4. Complete the export in completed mode.	System generates a new export file without changing the application status.	Re-export generated a new XLSX file. Application status remained COMPLETED. The new file appeared in the Recent exports menu.	Pass
Re-export Without Status Change	Finalized Queue Module -> Export Modal	1. Open the export modal from the Finalized Queue. 2. Change the finalized action to "Re-export only and keep finalized". 3. Complete the export.	Export file is generated but application remains FINALIZED.	Export file was generated and downloaded. Application status remained FINALIZED in the queue. No completion emails were sent.	Pass
XLSX Export Selection	Export Modal -> File Generation Module	1. Open the export modal from the Finalized Queue. 2. Change the file format from the default CSV selection to "Excel (.xlsx)". 3. Complete the export.	Export generates an XLSX file instead of CSV.	XLSX file was generated successfully with the correct exported rows. The downloaded workbook opened correctly and retained the configured export metadata.	Pass

5.4.5 Integration Testing

Integration testing focused on the handoff between connected modules so that data created in one stage could be correctly consumed by the next stage of the workflow. Table 5.12 summarizes the main cross-module integration checks.

Table 5.12: Integration Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
Transcript to Course Matching	Transcript Extraction Module -> Matching Module -> Eligibility Module	1. Upload a transcript in step 2. 2. Verify that extracted courses are automatically matched against articulation records. 3. Check eligibility assignment.	Extraction, matching, and eligibility evaluation operate as a connected pipeline.	Transcript was extracted, courses were matched against existing articulation rules (with branch-specific priority), and eligibility labels were assigned automatically.	Pass
Syllabus Upload to RP Evidence	Student Syllabus Module -> Extraction Module -> RP Review Module	1. Student uploads a source syllabus. 2. Text extraction runs on the uploaded file. 3. RP opens the review page and sees the syllabus ready for comparison.	Syllabus flows from student upload through extraction to RP evidence display.	Uploaded syllabus was stored, text extraction completed, and the RP evidence panel displayed the PDF inline with extraction status marked as APPROVED.	Pass
RP Decision to Status Sync	RP Decision Module -> Application Status Module	1. RP submits a decision for a review group. 2. Verify that all linked CTA courses receive the decision. 3. Check that the application status re-syncs.	RP shared decision propagates to all linked CTA courses and updates the overall application status.	RP decision was applied to all linked courses across multiple applications. Application status re-synced to EVALUATION_COMPLETED after all flagged courses had decisions.	Pass
AAS Export to Student Email	AAS Export Module -> Notification Module	1. AAS exports a finalized application with "Export and mark completed". 2. Verify that the student receives a completion email.	Export triggers both file generation and student notification.	Export file was generated and stored. Application status changed to COMPLETED. Completion email was sent to the student with transfer details.	Pass

5.4.6 System Testing

System testing examined the complete credit transfer workflow as a single end-to-end process, from the student's first submission steps to the final AAS export outcome. Table 5.13 presents the overall system-level test result.

Table 5.13: System Testing Results

Test Case	Modules Involved	Test Steps	Expected Result	Actual Result	Status
End-to-End Credit Transfer Workflow	Student Module -> HOP Module -> RP Module -> AAS Module	1. Student signs in and completes onboarding. 2. Student fills step 1 metadata. 3. Student uploads transcript in step 2 and courses are extracted. 4. Student reviews courses in step 3. 5. Student uploads syllabus in step 4 and submits. 6. HOP reviews the application, assigns RP for flagged courses. 7. RP runs comparison, reviews results, and submits evaluation. 8. HOP reviews RP results and finalizes. 9. AAS exports the finalized application and marks completed. 10. Student views the completed application and downloads CTA slip.	The application flows from DRAFT to COMPLETED through the required review, decision, and export stages with data consistency maintained at every stage.	The full workflow executed successfully. Each status transition was recorded in ApplicationStatusHistory. Course-level decisions were consistent. The final CTA slip contained correct transfer details.	Pass

5.5 Acceptance Testing

The final stage of the testing process was acceptance testing to determine whether the system was suitable for use in the real world. Acceptance testing was performed from two perspectives to assess whether the system fulfilled the requirements of the project:

1. Client Acceptance Testing (CAT): Interviewing the Head of Programme (HOP), the Resource Person (RP), and the Academic Affairs Students (AAS) officer, the main stakeholders and clients of the project, to assess whether the system fulfilled the objectives of the project.
2. User Acceptance Testing (UAT): Distributing a questionnaire to student users of the system to determine their level of satisfaction with the system.

5.5.1 Client Acceptance Testing

Client Acceptance Testing (CAT) was performed by interviewing the Head of Programme (HOP), the Resource Person (RP), and the Academic Affairs Students (AAS) officer. These three individuals were interviewed during the requirements analysis phase of the project (Chapter 3). During CAT, these three individuals were provided access to the system and were interviewed using the same questions as those asked during the requirements analysis phase to determine whether the system met the requirements of the project and whether it would be appropriate to utilize such a system within the university.

5.5.1.1 Head of Programme (HOP) Interview Response Summary

The interview with the Head of Programme (HOP) was conducted in the same way as the interview with the HOP during the requirements analysis phase of the project (Table 3.10). The HOP was provided with access to the system and was asked to perform each of the typical tasks for the HOP to review and prepare the students' applications for submission to other universities. Following the performance of these tasks, the HOP was interviewed to determine his satisfaction with the system and whether it would be appropriate to implement it within the university.

Table 5.14: Head of Programme (HOP) Interview Response Summary

Question	Answer	Analysis
Does the system achieve the objective of automating the credit transfer application process?	Yes. Students can submit everything online and the system automatically extracts courses from transcripts. There is no need to collect physical forms or manually write course details anymore.	The system successfully replaces the manual paper-based submission and data entry process that was identified as a major bottleneck in the Chapter 3 interview.
Does the system achieve the objective of supporting academic evaluation with AI-assisted comparison?	Yes. The AI comparison gives a percentage score and breaks down topics, which helps when deciding equivalency. It does not replace the academic judgment, but it provides a clear starting point.	The AI comparison feature directly addresses the challenge of "slow and time-consuming manual syllabus comparison" raised by the HOP in the requirements interview.
Does the system achieve the objective of providing centralized tracking and record management?	Yes. All application progress can be seen in one place. There is no need to ask the RP or student for updates because the dashboard already shows everything.	The centralized tracking resolves the manual progress tracking issue where the HOP previously had to "ask RP, students, or AAS" for status updates.
Are the functional requirements for HOP (Table 3.3) adequately supported by the system?	Yes. I can review applications, see the matched courses, manage the articulation list, upload target syllabi, request missing documents from students, and assign RPs. The finalization process also locks decisions properly.	All seven HOP functional requirements from Table 3.3 are confirmed as supported by the system.
Does the articulation management feature improve how you maintain course mappings?	Yes. Having the articulation list in the system is much better than using spreadsheets. I can import existing records and the system automatically uses them during matching.	Articulation management replaces the spreadsheet-based workflow and integrates directly with the transcript matching pipeline.

Question	Answer	Analysis
Does the system provide acceptable performance during normal operations?	Yes. Pages load quickly and I did not experience delays when reviewing applications or using the dashboard.	Performance requirement from Table 3.6 is satisfied based on stakeholder experience.
Is the system interface clear and easy to use for your tasks?	Yes. The layout is organized and I can find what I need. The review queue makes it clear which applications need attention.	Usability requirement from Table 3.6 is satisfied. The review queue addresses the need for prioritized task visibility.
Does the system enforce proper access control and data security?	Yes. I can only see applications related to my programme. Staff roles are separated, so students cannot access review functions.	Security requirement from Table 3.6 is satisfied with role-based access control.
Would you recommend using this system for the actual credit transfer process at UPTM?	Yes. It would save a lot of time and reduce the paperwork. The reports section is also useful for tracking statistics, which is something I requested during the first interview.	The HOP confirms that the system addresses the "desired features" identified in Table 3.10, specifically report generation and digital storage.

5.5.1.2 Resource Person (RP) Interview Response Summary

This interview was conducted with the Resource Person who was previously interviewed during the requirements analysis phase (Table 3.11). The RP was given access to the system and asked to perform typical evaluation tasks including viewing assigned review cases, running an AI syllabus comparison, reviewing comparison results, and submitting an evaluation decision. The feedback was then collected through the structured interview questions.

Table 5.15: Resource Person (RP) Interview Response Summary

Question	Answer	Analysis
Does the system achieve the objective of automating the credit transfer application process?	From my side, yes. I no longer need to wait for the HOP to physically hand me the documents. Everything is already in the system when I open my queue.	The system eliminates the dependency on HOP for document delivery, which was identified as a bottleneck in Table 3.11 ("all syllabus documents are provided through the Head of Program").
Does the system achieve the objective of supporting academic evaluation with AI-assisted comparison?	Yes. The AI comparison is helpful because it shows the percentage and breaks it down by topic. I still review the actual content, but having the comparison summary saves a lot of time compared to doing it manually.	The AI comparison directly addresses the RP's previous challenge of "unclear or overly simple syllabus content" and the need to "review additional references" by providing structured, topic-level analysis.
Does the system achieve the objective of providing centralized tracking and record management?	Yes. I can see all my completed evaluations and past decisions in one place. I do not have to search through physical files or redo work for repeated evaluations from the same institution.	The evaluation history resolves the previous issue of "having to refill documents multiple times for repeated IPT evaluations" from Table 3.11.
Are the functional requirements for RP (Table 3.4) adequately supported by the system?	Yes. I can see my assigned cases, view both syllabi side by side, run the comparison, and submit my decision with justification. The system also lets me change the target syllabus if it was mapped incorrectly.	All six RP functional requirements from Table 3.4 are confirmed as supported by the system.

Question	Answer	Analysis
Does the AI comparison meet the expectation you expressed in the earlier interview about wanting automated similarity percentages?	Yes. In the earlier interview I said I would prefer a cloud system that can generate similarity percentages. This system does that and also breaks it down by topic, which is even better than what I expected.	The system exceeds the RP's original expectation from Table 3.11, where the RP expressed a preference for "cloud system that can generate similarity percentages."
Does the system provide acceptable performance during the comparison process?	The comparison takes some time to process, but it is still faster than doing it manually. The progress indicator shows which stage it is on, so I know it is working.	Performance requirement from Table 3.6 is satisfied. The staged progress indicator addresses user experience during longer processing times.
Is the system interface clear and easy to use for your evaluation tasks?	Yes. The side-by-side view is very useful and the decision section is straightforward. I appreciated that I could see the AI recommendation before making my own decision.	Usability requirement from Table 3.6 is satisfied. The evidence viewer layout supports the RP's evaluation workflow.j
Does the system maintain proper records of your evaluation decisions?	Yes. Every decision I make is recorded with my justification and timestamp. I can also see the full history of the application.	Traceability requirement from Table 3.6 is satisfied with recorded decisions and status history.
Would you recommend using this system for the actual credit transfer evaluation process?	Yes. It simplifies the process significantly. As I mentioned before, I agree with AI automation because it simplifies the process, and this system delivers that well.	The RP reaffirms the strong support for AI automation expressed in Table 3.11 and confirms the system meets practical expectations.

5.5.1.3 Academic Affairs Students (AAS) Interview Response Summary

This interview was conducted with the AAS officer who was previously interviewed during the requirements analysis phase (Table 3.12). The AAS officer was given access to the system and asked to perform typical administrative tasks including viewing finalized applications, configuring and executing an export, downloading previously generated export files, and reviewing completed application records. The feedback was then collected through the structured interview questions.

Table 5.16: AAS Officer Interview Response Summary

Question	Answer	Analysis
Does the system achieve the objective of automating the credit transfer application process?	From the AAS side, yes. The export process is much faster now. I do not have to manually type student information from handwritten forms into CMS. The system already has everything structured and I just configure the export and download.	The system eliminates the data entry bottleneck identified in Table 3.12 where "unclear handwriting slows down data entry" and "sometimes causes incorrect CMS entries."
Does the system achieve the objective of providing centralized tracking and record management?	Yes. All completed records are stored digitally in the system. I can search for any past application and re-download the export file. There is no risk of losing physical forms anymore.	The digital archiving directly resolves the AAS officer's concern about forms being "misplaced during busy intake periods" and the reliance on "physical storage cabinets" from Table 3.12.
Does the system reduce the issues you experienced with physical form handling?	Yes, completely. There is no more problem with unclear handwriting, missing forms, or having to contact students for corrections. Everything is already validated before it reaches AAS.	The system addresses all three major AAS challenges from Table 3.12: unclear handwriting, misplaced forms, and incorrect data entry.
Are the functional requirements for AAS (Table 3.5) adequately supported by the system?	Yes. I can view all finalized applications, export them in different formats, manage the export history, and mark applications as completed. The batch export feature is especially useful during peak intake periods.	All five AAS functional requirements from Table 3.5 are confirmed as supported by the system.

Question	Answer	Analysis
Does the export feature meet your operational needs for updating CMS?	Yes. The export generates a structured file with all the fields I need. I can choose the format and configure how certain columns like the institution name and date should appear. The staff ID field also helps for record keeping.	The configurable export options (kolej mode, update_date format, staff_id, file format) address the AAS need for CMS-compatible output.
Does the system provide acceptable performance when processing exports?	Yes. The export generates quickly even with many applications. The download starts automatically after confirmation.	Performance requirement from Table 3.6 is satisfied for the AAS export workflow.
Is the system interface clear and easy to use for your administrative tasks?	Yes. The finalized queue is clear and the export modal guides me through the steps. The identity confirmation may seem like an extra step, but I understand it is there for safety.	Usability requirement from Table 3.6 is satisfied. The confirmation step provides operational safety that the AAS officer recognized as appropriate.
Does the system preserve proper records of export activities?	Yes. Every export file is stored permanently and I can re-download any previous batch from the Recent Exports menu. The completed archive also shows when each application was processed.	Traceability requirement from Table 3.6 is satisfied with permanent export storage and timestamped completion records.
Would you recommend using this system for the actual credit transfer administrative process?	Yes. As I mentioned in the earlier interview, I strongly support digitalization. This system delivers exactly what I was hoping for. It reduces paperwork, is faster to process, and makes tracking much easier.	The AAS officer reaffirms the strong support for digitalization expressed in Table 3.12 and confirms the system meets practical administrative needs.

5.5.2 User Acceptance Testing

User acceptance testing was performed through a questionnaire distributed on Google Forms to students who had used the system to effect the credit transfer process within the university. Fifty-five students responded to the questionnaire. The questionnaire contained fifteen questions, divided into four different main categories. Each of the questions utilized a four-point Likert scale to avoid the use of “neutral” answer options:

- 1 = Strongly Disagree
- 2 = Disagree
- 3 = Agree
- 4 = Strongly Agree

The results of each of the questions are represented in the following charts, exported directly from Google Forms.

The first two questions were used to collect background information regarding the students who utilized the system to request credit transfers.

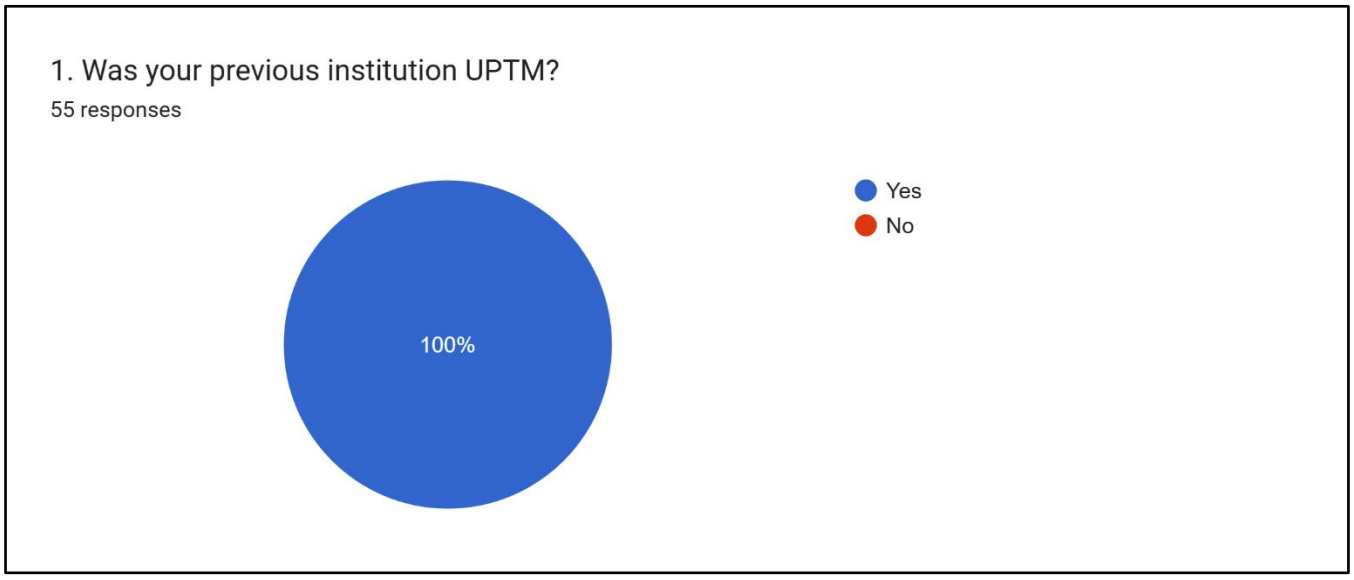


Figure 5.1: Respondent Previous Institution

All of the students who responded to the questionnaire (100%) to the question of their previous institution within the university were from the UPTM college itself. Thus, all of the students that were utilized to test the system were internal students within the college, who were to request credit transfers from their diploma status to their degree status within the college.

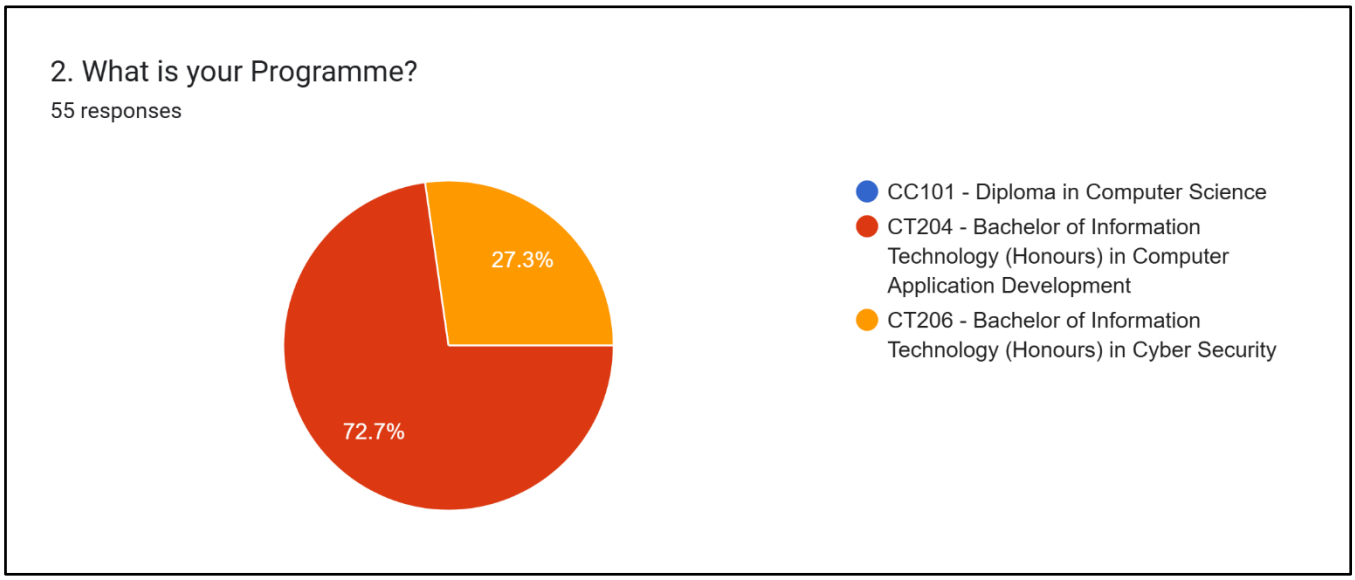


Figure 5.2: Respondent Target Programme Distribution

Seventy-two point seven percent of the students (40 students) were enrolled in the CT204 programme (Bachelor of Information Technology [Honours] in Computer Application Development). The remaining 27.3% of students within the questionnaire (15 students) were enrolled in the CT206 programme (Bachelor of Information Technology [Honours] in Cyber Security). Both of these programmes are actively targeted within the university for the implementation of this system.

Questions three to eight assessed whether the system fulfilled the three project objectives that were established for this project.

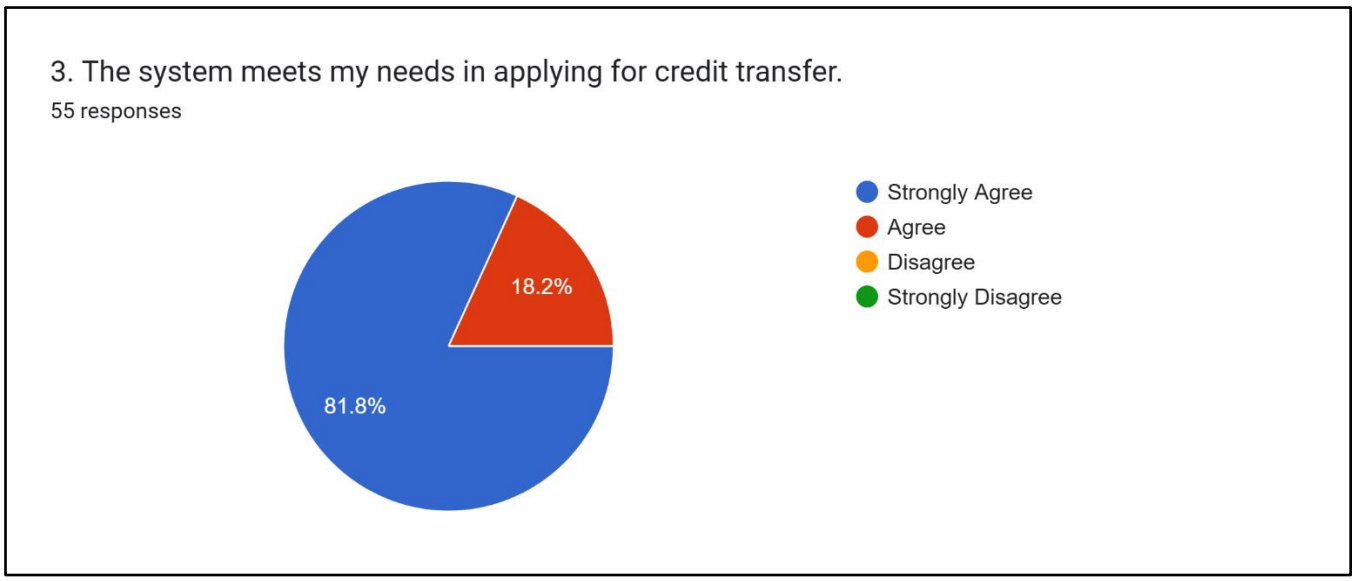


Figure 5.3: Credit Transfer Application Needs Fulfillment

Eighty-one point eight percent of students (45 students) indicated that the system fulfilled the needs for credit transfer applications. The remaining students (10 students) were of the opinion that the system did

fulfill these needs, but to a slightly lesser extent. Thus, the system was generally recognized as fulfilling its main purpose and objective for credit transfers (objective one defined within chapter one).

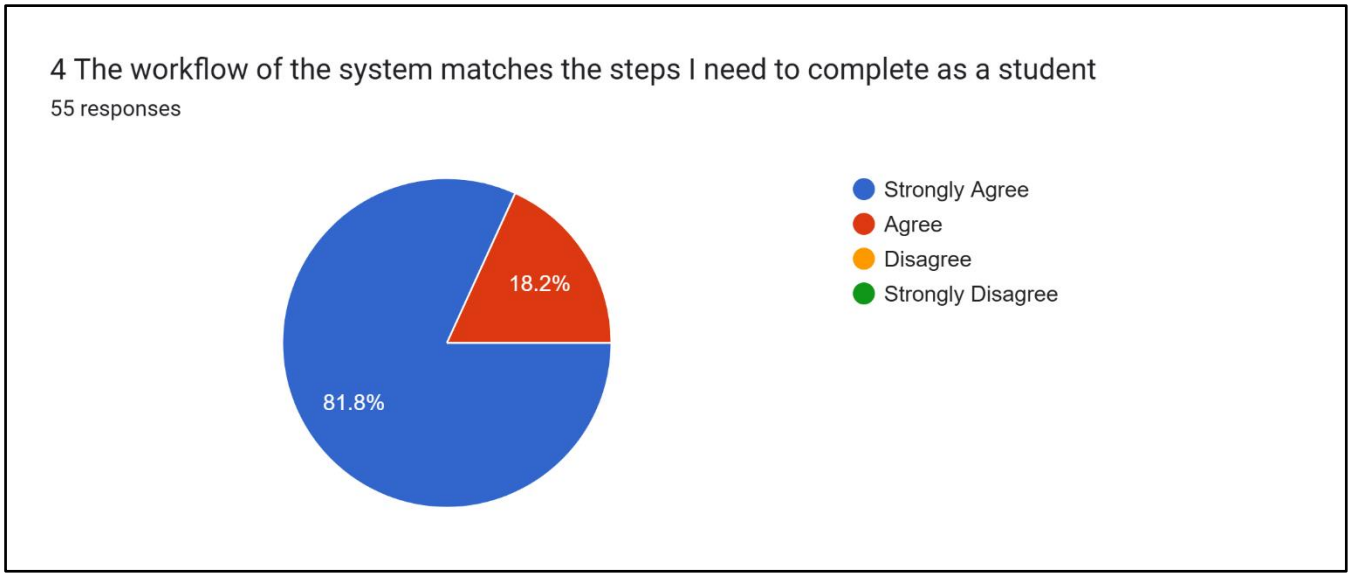


Figure 5.4: System Workflow Alignment with Student Steps

Similar to the result regarding fulfillment of credit transfer applications, 81.8% of students (45 students) indicated that the workflow for the system corresponded to the steps that a student would require to successfully request a credit transfer. The remainder of the students (10 students) were of the same opinion but with a slightly lesser degree of agreement.

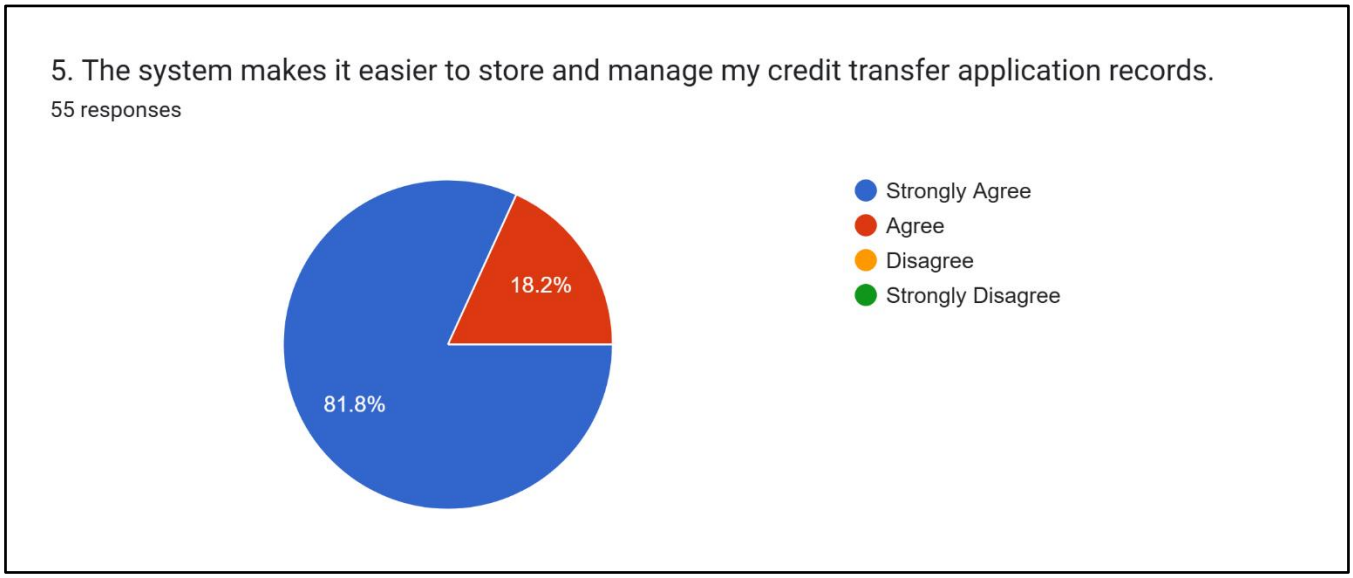


Figure 5.5: Ease of Record Storage and Management

As with the previous two questions, 81.8% of students (45 students) indicated that records within the system were easy to store and manage. This supports the third project objective related to the system (objective three).

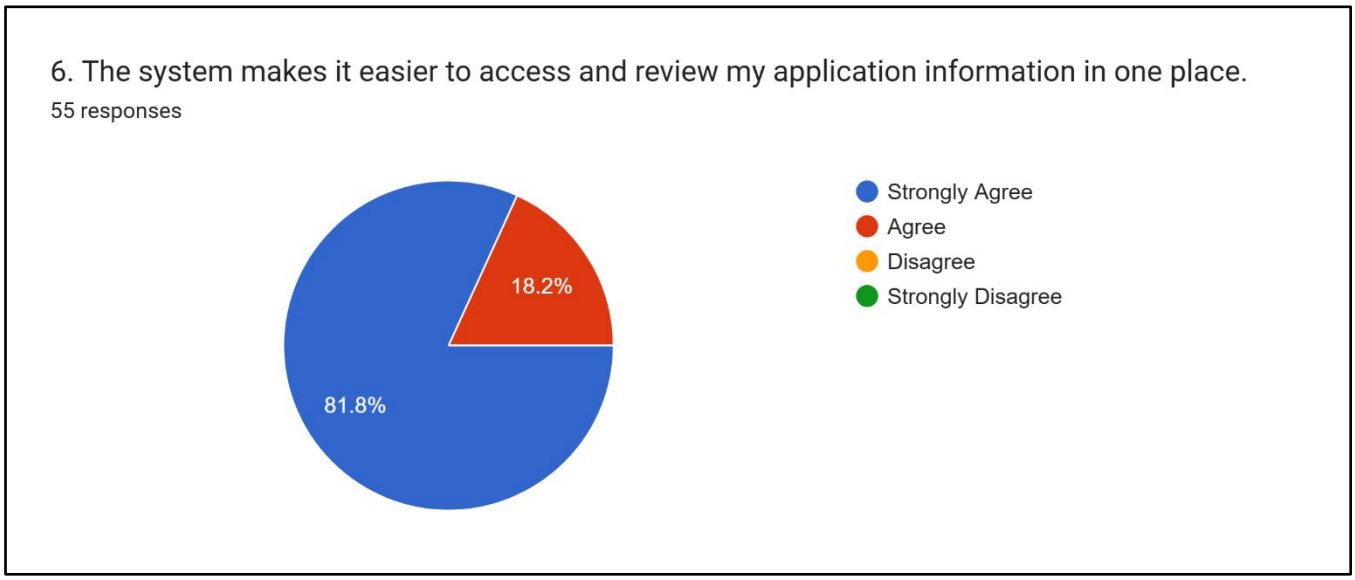


Figure 5.6: Centralized Application Information Access

Again, the same answers as for question five and six were recorded. Thus, the students found the centralized application page to be useful in providing access to the information regarding the credit transfer applications that they had submitted.

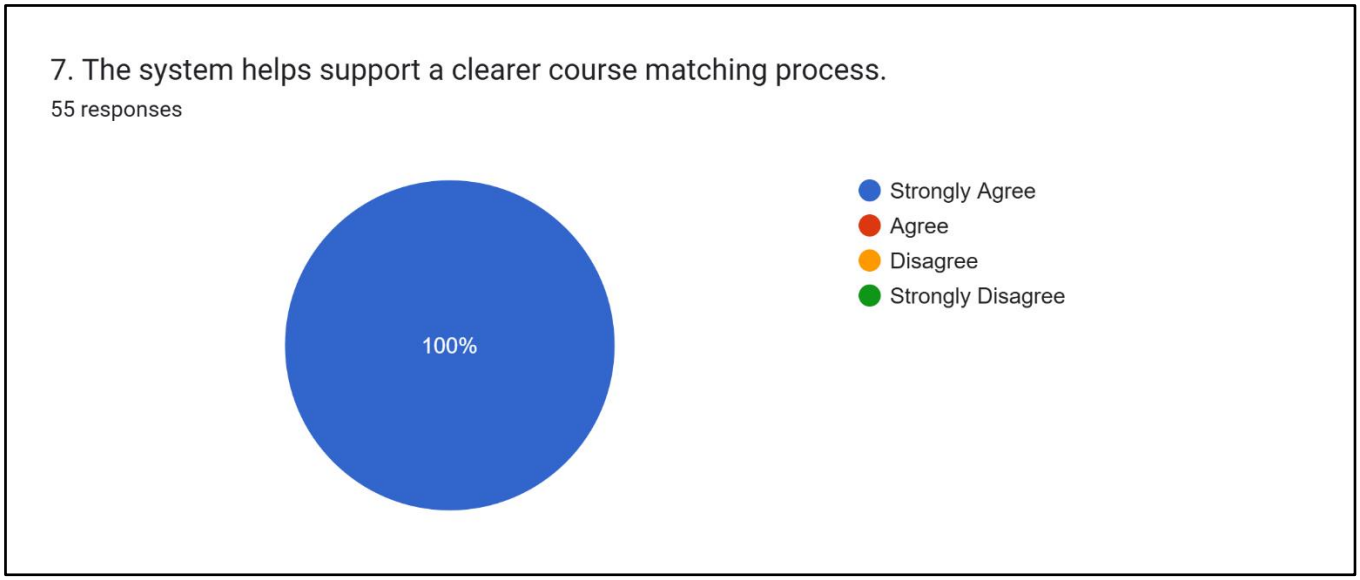


Figure 5.7: Clarity of Course Matching Process

All students (100%) indicated that the course matching process within step three of the system was clear to the students. This result is the highest of the four categories of questions within the questionnaire. Thus, this result indicates that the course matching and grouping of students according to their eligibility requirements was well-understood by the students.

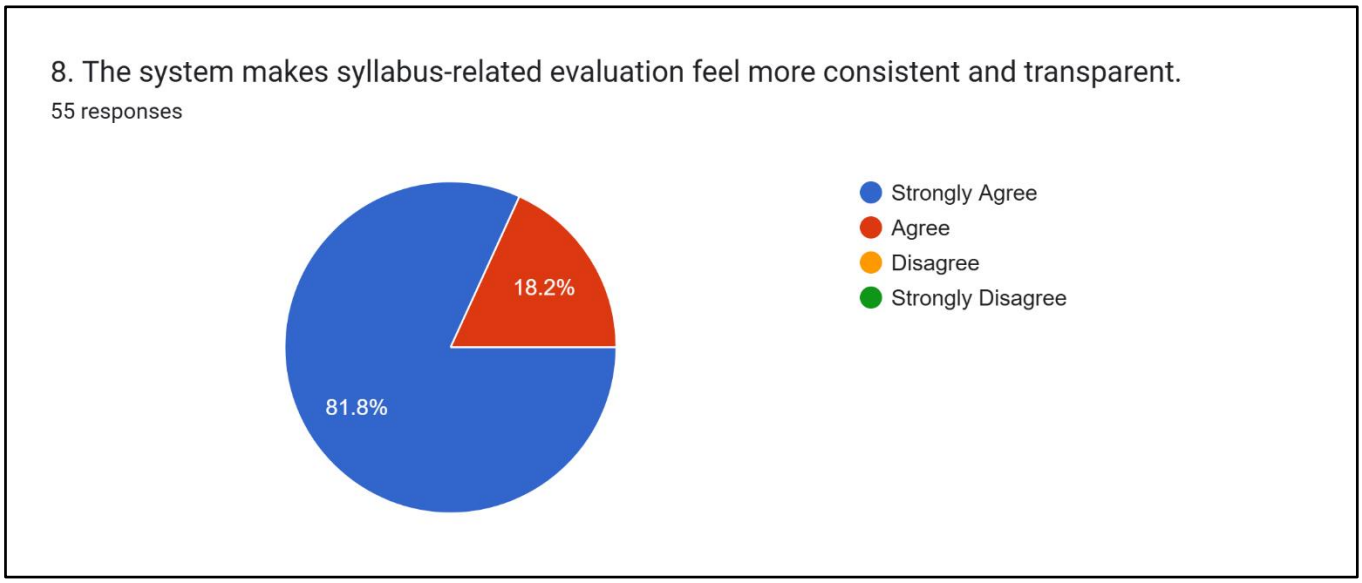


Figure 5.8: Consistency and Transparency of Syllabus Evaluation

As with the previous three questions, 81.8% of the students (45 students) indicated that the process for evaluating syllabi was consistent and transparent for the students. Thus, the process of evaluating syllabi using the system was well-understood by the students, as defined within project objective two for the system.

Questions nine to twelve assessed the functional requirements for the system as defined within Table 3.2.

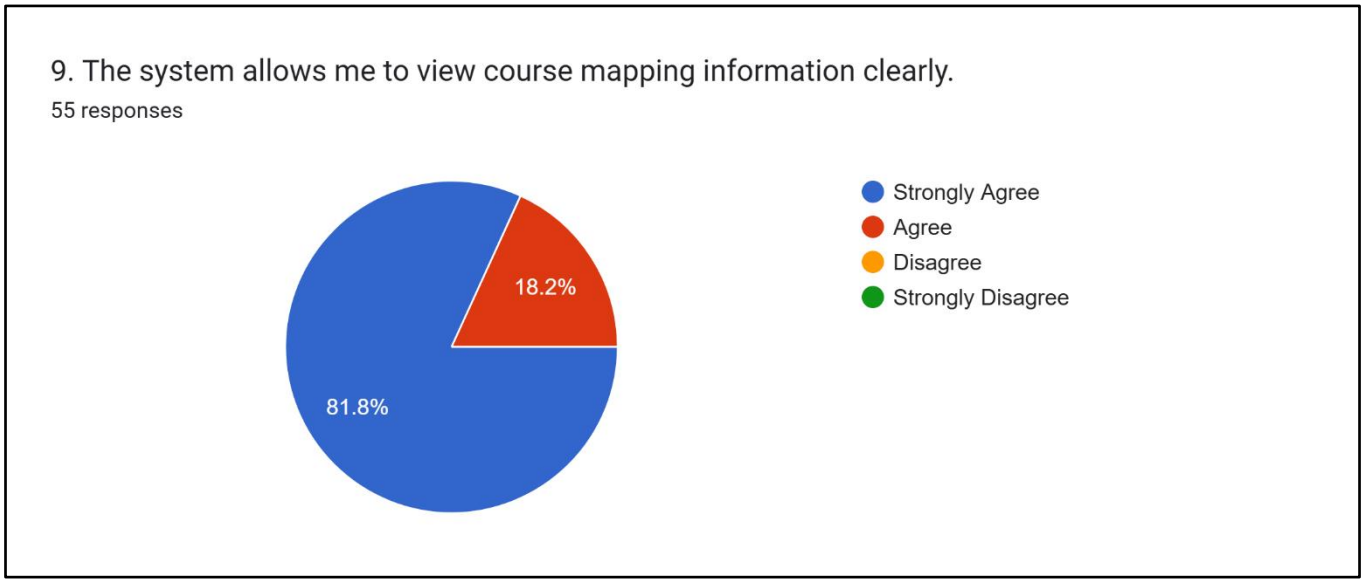


Figure 5.9: Clarity of Course Mapping Information Display

Eighty-one point eight percent of the students (45 students) indicated that the course mapping information was clear for the student. Thus, the information is well-understood by the users of the system, specifically

within step three in which the students are provided with course mapping information as to their eligibility for credit transfer.

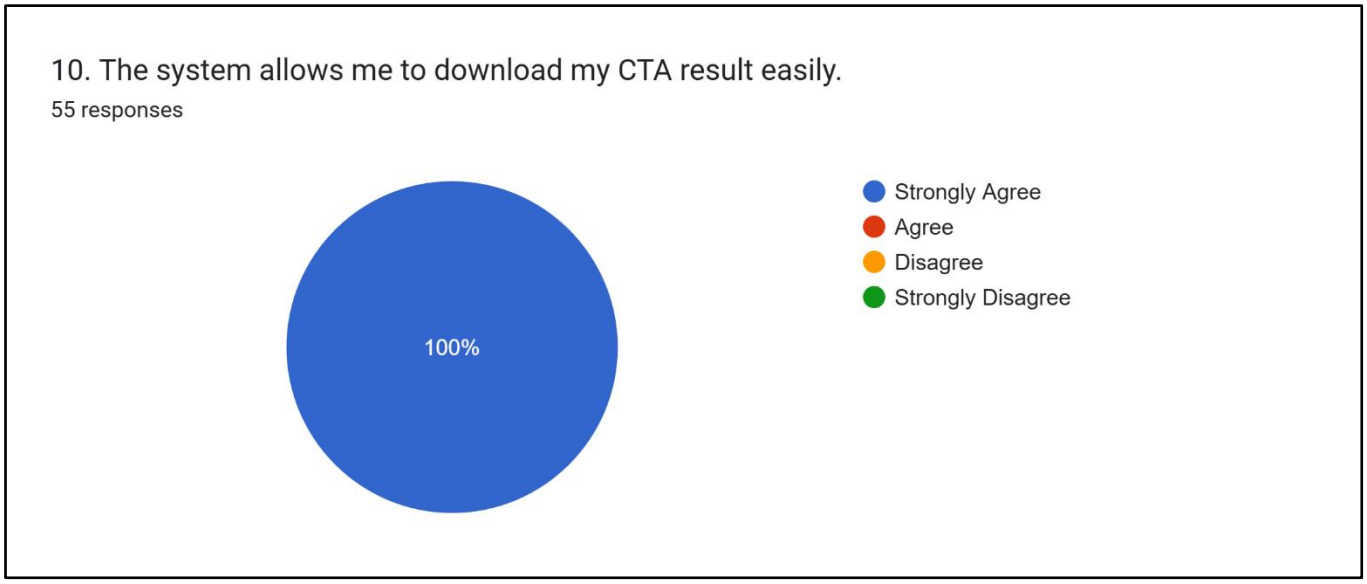


Figure 5.10: Ease of CTA Result Download

One hundred percent of the students indicated that results of the CTA were easy to download from the system. Thus, the functionality within the system allows students for easy access to their results of the credit transfer application.

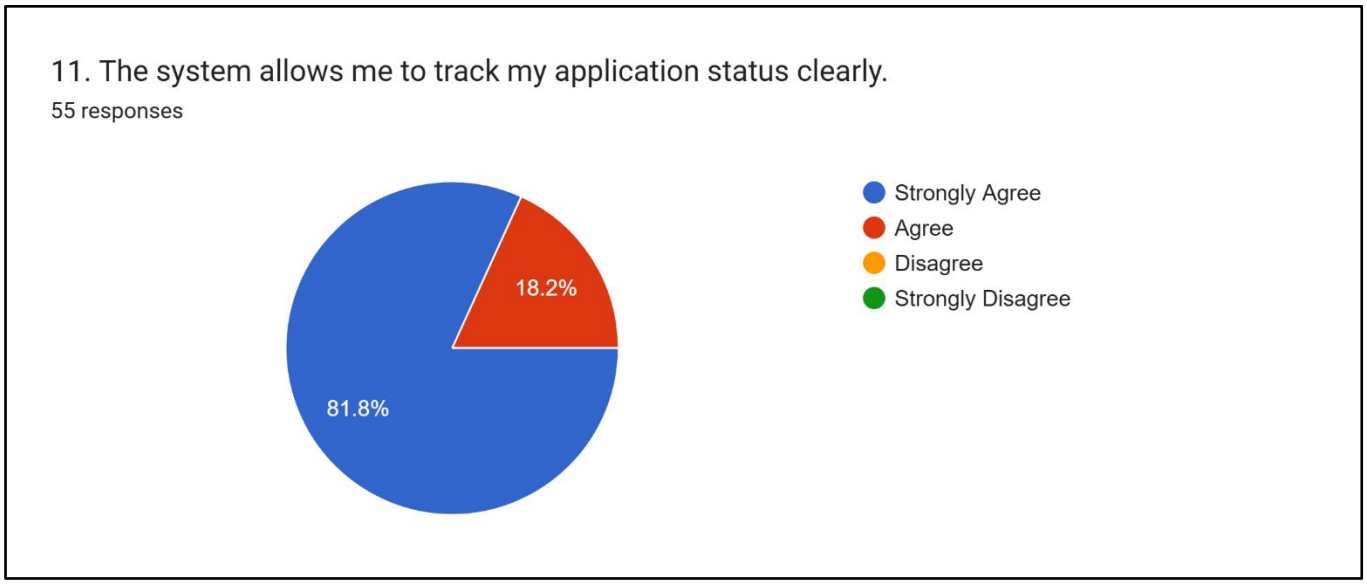


Figure 5.11: Clarity of Application Status Tracking

Similar to the other system features, 81.8% of the students (45 students) indicated that the system made it easy for students to track the status of their applications. Thus, the status badges within the system of the student are easily understood and tracked.

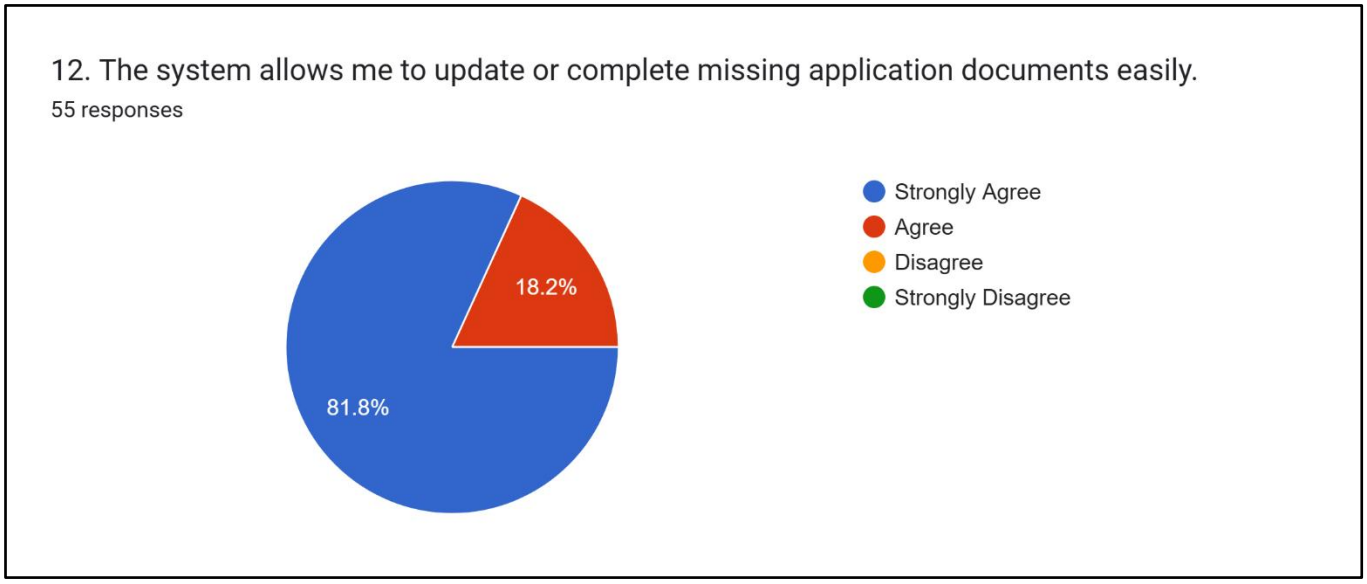


Figure 5.12: Ease of Updating Missing Documents

As with the previous four questions, the same answers were provided for this question as well. Thus, the system made it easy for students to update the documents that were required for the credit transfer applications.

Questions thirteen to fifteen assessed whether the system met the non-functional requirements of the system as defined within Table 3.6.

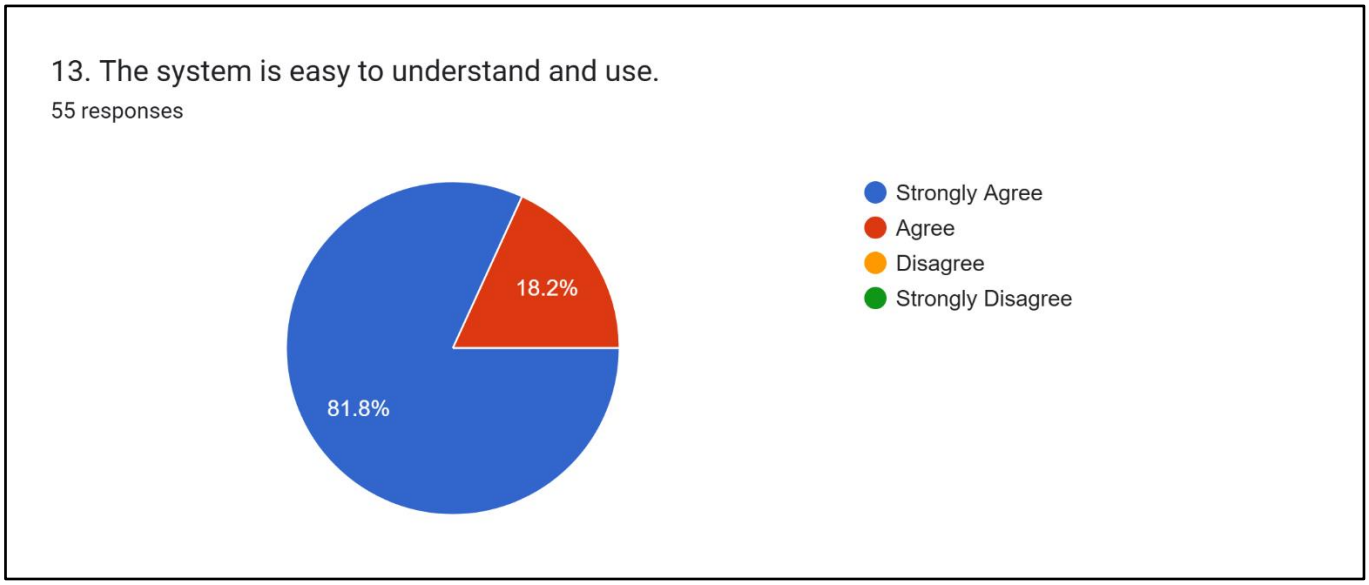


Figure 5.13: System Usability and Ease of Use

Eighty-one point eight percent of the students (45 students) indicated that the system was easy to use by the student. Thus, the usability of the system as determined by the students was within the required parameters within the table.

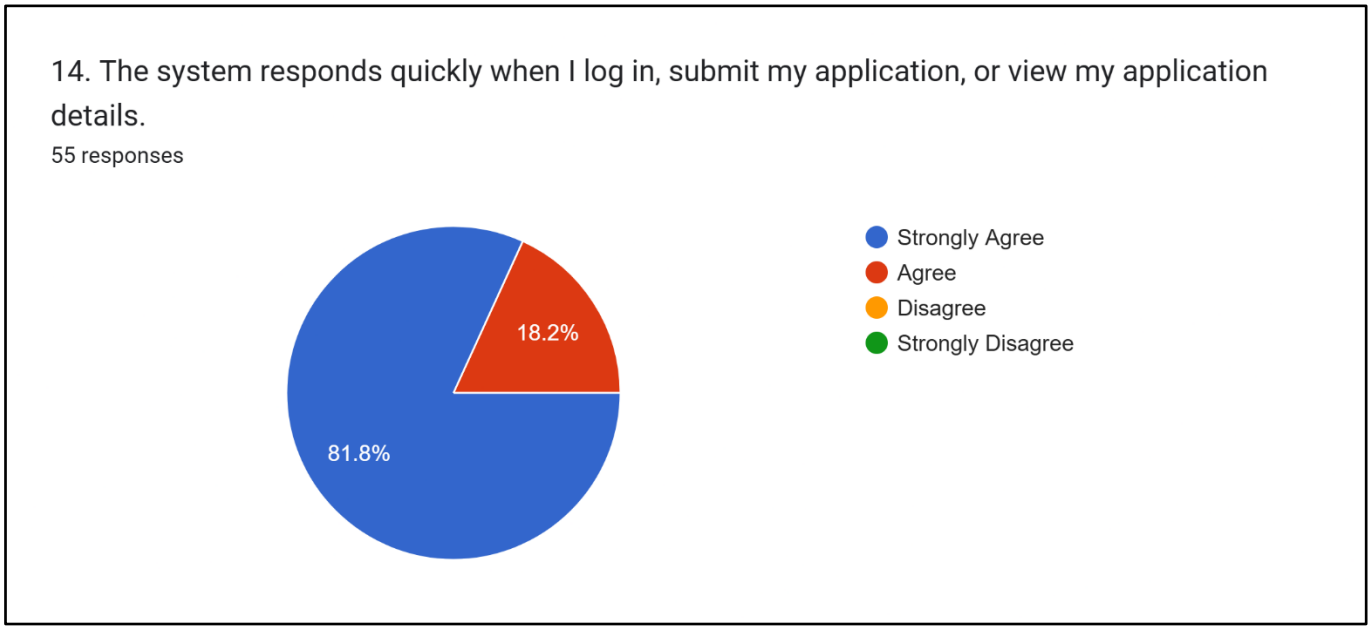


Figure 5.14: System Response Speed and Performance

The same answer was provided for system response times and performance for the system. Thus, the system was able to respond quickly to the students during the application process.

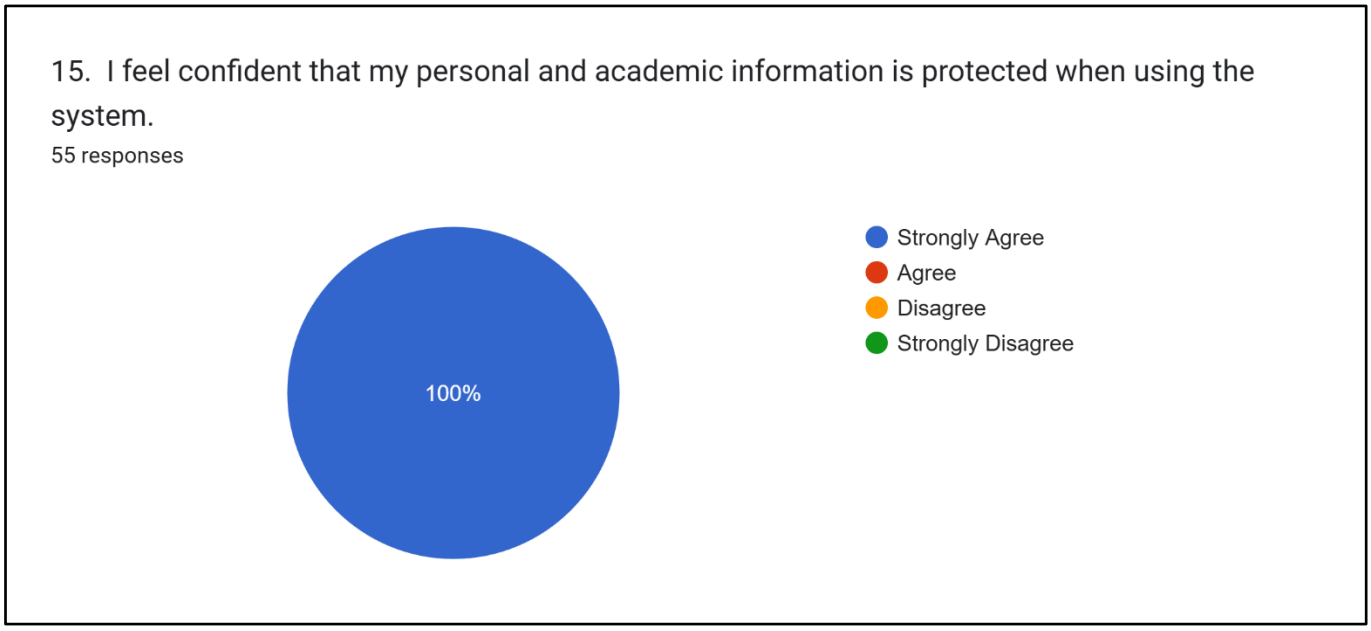


Figure 5.15: Confidence in Data Protection and Security

All of the students (100%) indicated that they had confidence in the system's ability to protect their data. Thus, the security of their data within the system is well-protected as defined by the non-functional requirements within the system.

Table 5.17 presents a summary of the results of the acceptance testing for the system.

Table 5.17: UAT Overall Summary

Category	Questions	Total Responses	Strongly Agree	Agree	Disagree
Project Objectives	Q3–Q8	330 (55 × 6)	290 (87.9%)	40 (12.1%)	0 (0%)
Functional Requirements	Q9–Q12	220 (55 × 4)	200 (90.9%)	20 (9.1%)	0 (0%)
Non-Functional Requirements	Q13–Q15	165 (55 × 3)	145 (87.9%)	20 (12.1%)	0 (0%)
Overall	Q3–Q15	715 (55 × 13)	635 (88.8%)	80 (11.2%)	0 (0%)

Based on the results of these different questions, the following overall observations can be made regarding the users' acceptance of the system:

1. The system received zero negative responses to the questions posed to the students. Thus, students generally had positive opinions of the system as a whole.
2. Three of the questions received answers of "Strongly Agree" from 100% of the students: Questions 7, 10, and 15. Each of these questions pertained to aspects of the system that were received as especially favorable from the students.
3. The remaining questions received answers of "Strongly Agree" from 81.8% of the students (45 students), with the remaining 18.2% of students (10 students) indicating that they also found the aspect of the system to be easy to understand or use.
4. The overall average score for the users of the system was calculated as being 3.89 out of a possible 4.00.

Based on these results, it can be stated that the system was accepted by the users as whole. Thus, the system fulfills the functional and non-functional requirements of the system, and meets the acceptance criteria of the users.

5.6 Conclusion

In this chapter, testing of the Credit Transfer System was performed generally in the same ways as non-functional, functional, and acceptance testing was performed. Non-functional testing of the system ensured that the system fulfilled the non-functional requirements of the system as specified in Table 3.6. All 43 of the non-functional test cases were passed. Functional testing of the system ensured that the system fulfilled the functional requirements of the system as specified in Tables 3.2 to 3.5. All 15 student test cases, 10 HOP test cases, 10 RP test cases, and 10 AAS test cases were passed. Integration testing determined whether the system passed through the individual modules correctly to the next stage in the process. System testing ensured that the entire system functioned correctly, from start to finish.

Acceptance testing of the system was performed for both the stakeholders and the users of the system. Client Acceptance Testing indicated to the HOP, RP, and AAS officers that the system fulfills each of the three objectives of the project, and would be recommended to the university for their use. User Acceptance Testing indicated that users of the system provided an average score of 3.89 out of 4.00 for the system, with zero negative responses to the questions asked of them. Based on these tests and acceptance criteria, the system has fulfilled its functional and non-functional requirements, and has been accepted as positive by the stakeholders and users of the system.

6 CONCLUSION

6.1 Introduction

The purpose of this chapter is to conclude the development of the UPTM Credit Transfer System. Mainly, this chapter will review the project schedule that was used to develop the system, the risk analysis and risk mitigation measures that were performed during its development, the extent to which each of the project objectives were achieved, the various constraints and limitations to the development of the system, and finally, recommendations for future development of the system.

The main purpose of the development of the Credit Transfer System was to address the challenges that existed within the current process of transferring academic credits between the different degree programmes within the Universiti Poly-Tech Malaysia (UPTM). The System successfully created an automated solution to each of the challenges that existed within the current process, whether in the steps of the process of comparing syllabi, transferring transcripts, or performing other actions for each of the four defined roles within the system (Student, Head of Programme, Resource Person, and Academic Affairs Staff).

6.2 Project Schedule

The project schedule for the development of the Credit Transfer System was essential in ensuring that the development of the system was systematically completed within the allotted timeframe. Such a schedule was created using the Agile software development methodology, and reflected in two separate project management tools: a Work Breakdown Structure (WBS) and a Gantt chart.

The WBS is a management tool that assists in the breakdown of the project into its individual components or tasks. Each component of the project has a specific name, description, and an estimation of the amount of effort that is required to complete that component of the project. The Gantt chart, in contrast, provides a visual timeline of the project schedule. The components of the project can be mapped to a visual timeline, indicating the timeframe during which each component of the project will be completed.

6.2.1 Work Breakdown Structure

The Work Breakdown Structure for the Credit Transfer System scheduled for development within this project is presented in Figure 6.1.

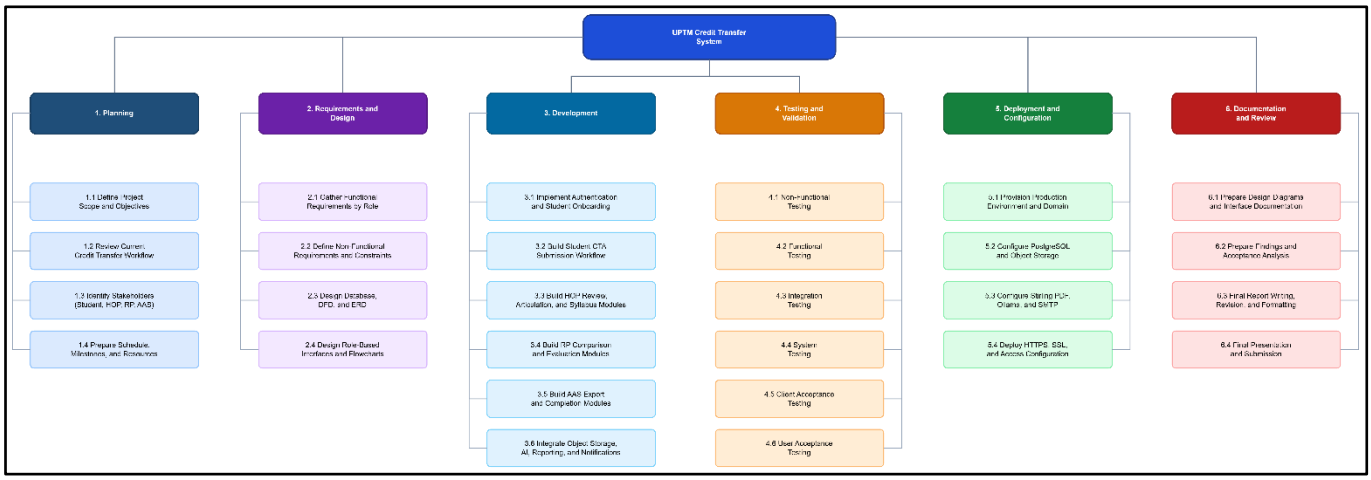


Figure 6.1: Work Breakdown Structure of the UPTM Credit Transfer System Project

The Work Breakdown Structure is utilised to break down the entire project into its individual components and tasks. For instance, the main stages of the project can be broken down into five major components: Planning, Requirements and Design, Development, Testing and Validation, Deployment and Configuration, and Documentation and Review.

During the Planning component of the project, the team will plan each aspect of the project, determine the extent of each aspect, and create a schedule for the project. During the Requirements and Design component of the project, the system will be designed according to the requirements that are determined during the Planning component. During the Development component, the actual development of the system will occur. Testing and Validation will occur after the system is developed and will focus upon testing each component of the system to ensure that they are functioning as they should. Finally, the Documentation and Review component will occur at the completion of the project, and will focus upon creating documentation for each aspect of the system (such as the design diagrams, interface documentation, reports, and presentations), and reviewing the entire process of the creation of the system.

6.2.2 Gantt Chart

The Gantt chart for the Credit Transfer System is represented in Figure 6.2.

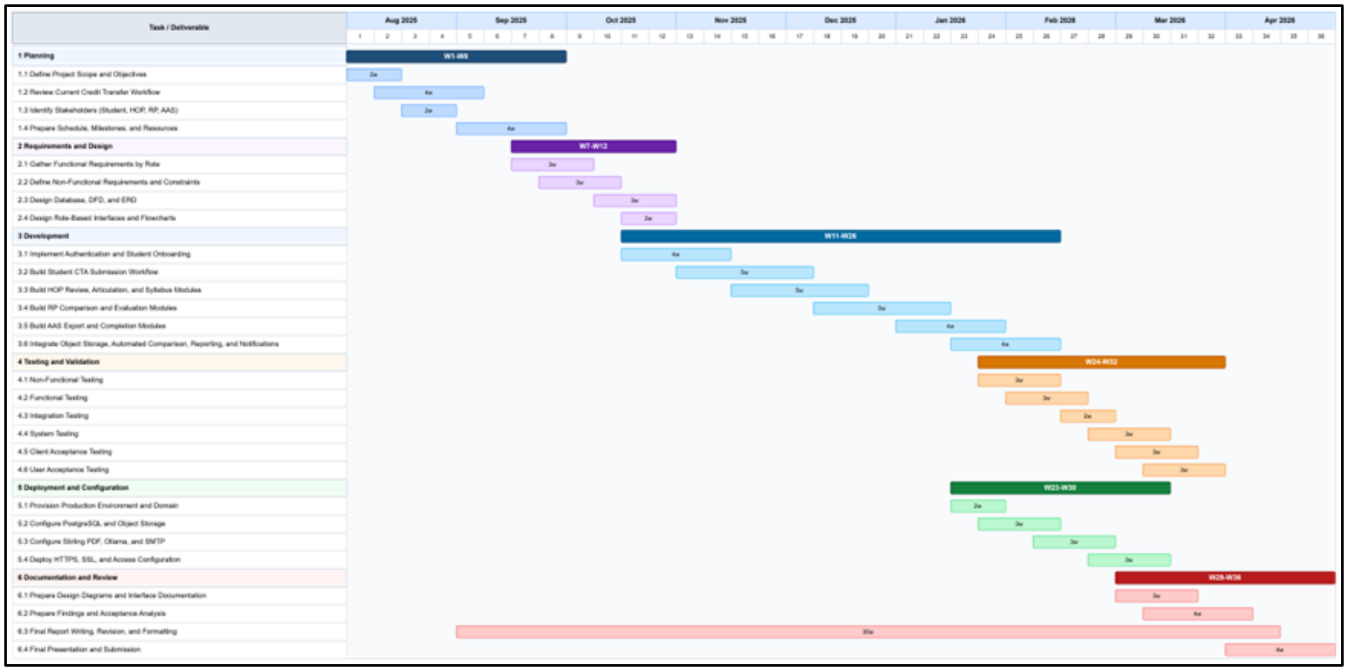


Figure 6.2: Gantt Chart of the UPTM Credit Transfer System Project Timeline

The Gantt chart for the project represents each of the components of the project as they will appear within the visual timeline of the project. For example, the Planning component of the project will occur between August 2025 and January 2026. During the Requirements and Design component, the system will be designed between January 2026 and March 2026. During the Development component, the actual development of the system will occur between March 2026 and January 2027. Testing and Validation will occur between April 2027 and July 2027. Finally, the Deployment and Configuration and Documentation and Review components will occur between August 2027 and January 2028. Table 6.1 contains a detailed breakdown of the components of the project according to the schedule that is represented within Figure 6.2.

Table 6.1: Project Task Timeline and Duration

ID	Task Name	Start	Finish	Duration
1	Planning	August 2025 (Week 1)	September 2025 (Week 8)	8 Weeks
1.1	Define Project Scope and Objectives	August 2025 (Week 1)	August 2025 (Week 2)	2 Weeks
1.2	Review Current Credit Transfer Workflow	August 2025 (Week 2)	September 2025 (Week 5)	4 Weeks
1.3	Identify Stakeholders (Student, HOP, RP, AAS)	August 2025 (Week 3)	August 2025 (Week 4)	2 Weeks
1.4	Prepare Schedule, Milestones, and Resources	September 2025 (Week 5)	September 2025 (Week 8)	4 Weeks
2	Requirements and Design	September 2025 (Week 7)	October 2025 (Week 12)	6 Weeks
2.1	Gather Functional Requirements by Role	September 2025 (Week 7)	October 2025 (Week 9)	3 Weeks
2.2	Define Non-Functional Requirements and Constraints	September 2025 (Week 8)	October 2025 (Week 10)	3 Weeks
2.3	Design Database, DFD, and ERD	October 2025 (Week 10)	October 2025 (Week 12)	3 Weeks
2.4	Design Role-Based Interfaces and Flowcharts	October 2025 (Week 11)	October 2025 (Week 12)	2 Weeks
3	Development	October 2025 (Week 11)	February 2026 (Week 26)	16 Weeks
3.1	Implement Authentication and Student Onboarding	October 2025 (Week 11)	November 2025 (Week 14)	4 Weeks
3.2	Build Student CTA Submission Workflow	November 2025 (Week 13)	December 2025 (Week 17)	5 Weeks
3.3	Build HOP Review, Articulation, and Syllabus Modules	November 2025 (Week 15)	December 2025 (Week 19)	5 Weeks
3.4	Build RP Comparison and Evaluation Modules	December 2025 (Week 18)	January 2026 (Week 22)	5 Weeks
3.5	Build AAS Export and Completion Modules	January 2026 (Week 21)	February 2026 (Week 24)	4 Weeks
3.6	Integrate Object Storage, Automated Comparison, Reporting, and Notifications	January 2026 (Week 23)	February 2026 (Week 26)	4 Weeks
4	Testing and Validation	February 2026 (Week 24)	March 2026 (Week 32)	9 Weeks

ID	Task Name	Start	Finish	Duration
4.1	Non-Functional Testing	February 2026 (Week 24)	February 2026 (Week 26)	3 Weeks
4.2	Functional Testing	February 2026 (Week 25)	March 2026 (Week 27)	3 Weeks
4.3	Integration Testing	March 2026 (Week 27)	March 2026 (Week 28)	2 Weeks
4.4	System Testing	March 2026 (Week 28)	March 2026 (Week 30)	3 Weeks
4.5	Client Acceptance Testing	March 2026 (Week 29)	March 2026 (Week 31)	3 Weeks
4.6	User Acceptance Testing	March 2026 (Week 30)	March 2026 (Week 32)	3 Weeks
5	Deployment and Configuration	January 2026 (Week 23)	March 2026 (Week 30)	8 Weeks
5.1	Provision Production Environment and Domain	January 2026 (Week 23)	February 2026 (Week 24)	2 Weeks
5.2	Configure PostgreSQL and Object Storage	February 2026 (Week 24)	February 2026 (Week 26)	3 Weeks
5.3	Configure Stirling PDF, Ollama, and SMTP	February 2026 (Week 26)	March 2026 (Week 28)	3 Weeks
5.4	Deploy HTTPS, SSL, and Access Configuration	March 2026 (Week 28)	March 2026 (Week 30)	3 Weeks
6	Documentation and Review	March 2026 (Week 29)	April 2026 (Week 36)	8 Weeks
6.1	Prepare Design Diagrams and Interface Documentation	March 2026 (Week 29)	March 2026 (Week 31)	3 Weeks
6.2	Prepare Findings and Acceptance Analysis	March 2026 (Week 30)	April 2026 (Week 33)	4 Weeks
6.3	Final Report Writing, Revision, and Formatting	September 2025 (Week 5)	April 2026 (Week 34)	30 Weeks
6.4	Final Presentation and Submission	April 2026 (Week 33)	April 2026 (Week 36)	4 Weeks

6.3 Risk Management

Risk management was also carried out throughout the project in order to identify the various issues that could potentially impact the development of the system. The main risks relating to the various components of the project were identified, and presented within Table 6.2, in which the risks to the project are listed along with the methods that were to be applied to mitigate each of those risks.

Table 6.2: Risk Management Summary

No.	Risk	Likelihood	Impact	Risk Analysis	Mitigation Strategy
1	Database schema changes causing migration conflicts or inconsistent relationships	Medium	High	Prisma schema changes could affect linked workflow records and create inconsistent data.	Schema changes were introduced incrementally, reviewed before use, and kept under version control.
2	Ollama service unavailability affecting automated syllabus comparison	Medium	Medium	If Ollama is slow or unreachable, RP comparison results may fail or be delayed.	The comparison workflow uses staged processing with clear error reporting, while staff can still rely on the uploaded evidence.
3	Document storage failure affecting transcript, syllabus, or export files	Low	High	Missing or inaccessible stored files would interrupt the credit transfer workflow.	File references are stored in the database and uploads are verified before the workflow continues. Missing files can be resolved through re-upload or regeneration.
4	Transcript or syllabus extraction producing incomplete text	Medium	High	Inaccurate extraction can affect course matching and syllabus comparison quality.	Students can review transcript extraction results before submission, and syllabus extraction can be repeated while staff still refer to the original file.
5	Scope creep from stakeholder requests during development	Medium	Medium	Additional requests beyond the agreed scope could delay completion.	Development was prioritised around the project objectives and only necessary changes were included.

No.	Risk	Likelihood	Impact	Risk Analysis	Mitigation Strategy
6	Authentication or role errors leading to unauthorized access	Low	High	Weak role enforcement could expose student, evaluation, or export data to the wrong user.	Better Auth was used for session handling, while role checks were enforced in server-side helpers, protected pages, and server actions.
7	Production environment misconfiguration disrupting core services	Medium	High	Incorrect configuration of PostgreSQL, object storage, Stirling PDF, Ollama, SMTP, domain, or SSL could prevent core functions from working.	Deployment was handled as a dedicated phase, with environment settings and service connectivity checked during setup.
8	Email notification delivery failures affecting workflow communication	Medium	Low	SMTP issues may prevent students or staff from receiving workflow emails on time.	Email is treated as a supporting feature, while the main workflow status remains visible inside the system.

6.4 Achievement

The main measure of the success of the project is the extent to which the system achieved the objectives that were established at the beginning of the project. Each of the three objectives will be evaluated in this section to determine whether the system has achieved its purpose. Such an evaluation will be based upon the implementation of the system, as well as the testing that was performed in evaluating the system (as discussed in Chapter 5). Through this discussion, it will be possible to determine whether the system effectively fulfills its purpose.

6.4.1 First Objective

The first objective of the project was to analyze the requirement of an automated credit transfer system. This objective was achieved through the identification of the requirements of the system, as well as through the translation of those requirements into the system itself. Such identification of the requirements of the system occurred through interview and questionnaire-based surveys of the students, the HOP, the RP, and the AAS departments. These interviews helped to determine the various requirements of each of these departments and their relationship to the credit transfer system.

Within the system that is developed, there are separate modules for each of the various departments of the university. For the students, for instance, there are separate modules for the completion of the CTA application, uploading of their transcripts and syllabi, and for reviewing the applications that have been submitted. For the HOP, there are separate modules for reviewing applications, managing articulation and syllabi, and for referencing and reporting on the students' academic records. For the RP, there are separate modules for managing an evaluation queue for submitted syllabi, reviewing the comparison of those syllabi with other syllabi in the university, and submitting decisions for each of those reviewed syllabi. For the AAS, there are separate modules for the export of the applications, the management of the applications that have been finalized, the archiving of those applications, and for the tracking of those completed applications.

The availability of separate modules for each of these departments indicates that the requirements of the system were thoroughly analysed and translated into the system itself. Furthermore, non-functional requirements of the system (such as security and access requirements) were also analyzed and reflected in the system. Thus, this first objective demonstrates that the credit transfer system can be used as a means of fulfilling the requirements that were identified from each of these departments.

6.4.2 Second Objective

The second objective of the project was to design a platform for the creation of a centralized digital repository for the records of the credit transfer process. Such an objective was achieved through the creation of a digital platform that was able to centralize the records of the process; it is a repository for applications, transcripts, syllabi, articulations, comparison records, and outcomes of evaluations and exports of those applications. By creating a platform that has modules for each of the various records of the credit transfer process, it becomes possible for each of those departments to have access to these records and references to them. Furthermore, by having those references to the various records within one system, it is possible to create a more efficient and effective credit transfer process within the university in general.

6.4.3 Third Objective

The third objective of the project was to develop a module for the comparison of syllabi from two different departments within the university. Such a module was developed through the implementation of a module that utilizes a staged prompting method to evaluate syllabi from two separate departments. This staging allows for the separate evaluation of aspects of each syllabi, which helps to allow the RP to have a more thorough and reviewable evaluation of each syllabi before making an approval decision. Through the availability of the comparison module for the syllabi, it is possible for each of the RPs to have a more thorough review of each syllabi prior to making decisions regarding the transfer of those credits. Furthermore, because the stage-based method of comparison is used, it is possible for this module to help support the RP in its role of approving syllabi for credit transfer into the students' overall degree programs.

6.5 Constraint and Limitation

In addition to the objectives that were achieved during the development of the system, there are also some limitations to the system that should be discussed.

6.5.1 Database Dependency and Maintenance

The first objective of this project was to analyze the requirement of an automated credit transfer system. As described in Section 1.1, that objective was achieved in fulfilling the identification and translation of the system requirements into the system itself. Those requirements were gathered through interviews with the students, HOP, RP, and AAS departments regarding the requirements of the credit transfer system and its individual departments.

6.5.2 AI Model Dependency and Output Consistency

The third objective was to develop a module to compare syllabi from two departments in the university. Such a system is based upon utilizing AI models to compare those syllabi from separate departments. Thus, one limitation to the system is that the information and knowledge of the AI models will have an impact upon the outputs of that module; the model will be dependent upon the type of model that is used and upon the knowledge of the syllabi that is uploaded by the students.

6.5.3 Dependency on Service APIs and External Endpoints

Another limitation of the system is in its dependency upon a variety of external services to function. Services such as Stirling PDF for document extraction, object storage for documents, Ollama for syllabus comparison models, and SMTP for email functionality will all be required to be connected and operational for the system to function accordingly.

6.5.4 Server Resource and Infrastructure Cost

Furthermore, more than just web hosting will be required to deploy the system live; database hosting, file storage, PDF processing, AI model hosting, and email functionality will all require some form of hosting services. Therefore, there will be an associated cost to the operation of the system for these resources.

6.5.5 Scalability and Performance Limitation

Another limitation to the system is in the scalability and performance of some of the operations that are to occur within the system. For instance, operations like document extraction, syllabi comparison, report generation, and file handling will be relatively resource-intensive operations in comparison to typical web

application functions. Thus, the system may experience a performance limitation with an increased number of applications that are submitted to the university.

6.6 Future Work and Recommendation

Based upon the constraints and limitations of the system, as well as through the feedback from the students and staff during acceptance testing of the system, the following recommendations can be made for future development of the system.

6.6.1 Built-In Document Sanitization and Redaction

One recommendation for the future is the implementation of a feature that allows for built-in document sanitization and redaction. This would allow for documents that are uploaded to the system to have certain information removed prior to the submission of the application. Such a feature would allow the system to automatically protect the personal information of the students and remove unnecessary information from being shared.

6.6.2 Supporting Documents for Syllabus Comparison

Another recommendation is that the system may be enhanced to allow for additional documents to be uploaded alongside the syllabus that is being submitted for comparison. Such additional documents could contain information about the course, the CLOs for the course, the textbooks that will be used in the course, and any additional related documents. These files would provide additional justification for the syllabi that are being uploaded and allow for the comparison software to create more detailed justification for approval of the syllabi for credit transfer.

6.6.3 AI Chatbot for User Support

Another potential area for improvement of the system is the implementation of an AI chatbot that may be able to support the users of the system. That chatbot could have questions about the system, requirements of the credit transfer system, the steps that must be taken to successfully complete an application, and other questions that may arise from the users of the system.

6.6.4 Fine-Tuning of the AI Model

Another recommendation for the future development of the system is the fine-tuning of the AI model upon which the system is based. Currently, the model that is used for comparison of the syllabi is general-purpose AI software with no experience in reviewing academic syllabi. Therefore, fine-tuning the AI model

upon syllabi documents and similar academic files will increase the accuracy and effectiveness of the comparisons between syllabi from two departments.

6.6.5 Production Analytics and Usage Monitoring

Another recommendation for the future is the implementation of analytics into the production of the system. Such analytics would allow the university departments to understand how the system is used; it will help to determine which steps in the process are the most difficult for the students, for which departments there are the most applications are submitted, and any other information about the system that could enable future improvements to the software.

6.6.6 Background Job Queue and Worker Processing

Another area for future development is the implementation of a background job queue and workers for the system, such as a Prisma-backed job queue. Such job workers could help with the performance of relatively-intensive processes for the system, such as extraction of documents from students' files, comparison of syllabi, generation of reports of students' programs, and processing of deadlines for each of those students. By implementing such workers, the system may become more reliable and efficient with its performance of those jobs, as well as it may allow the website to be more easily scalable.

6.7 Conclusion

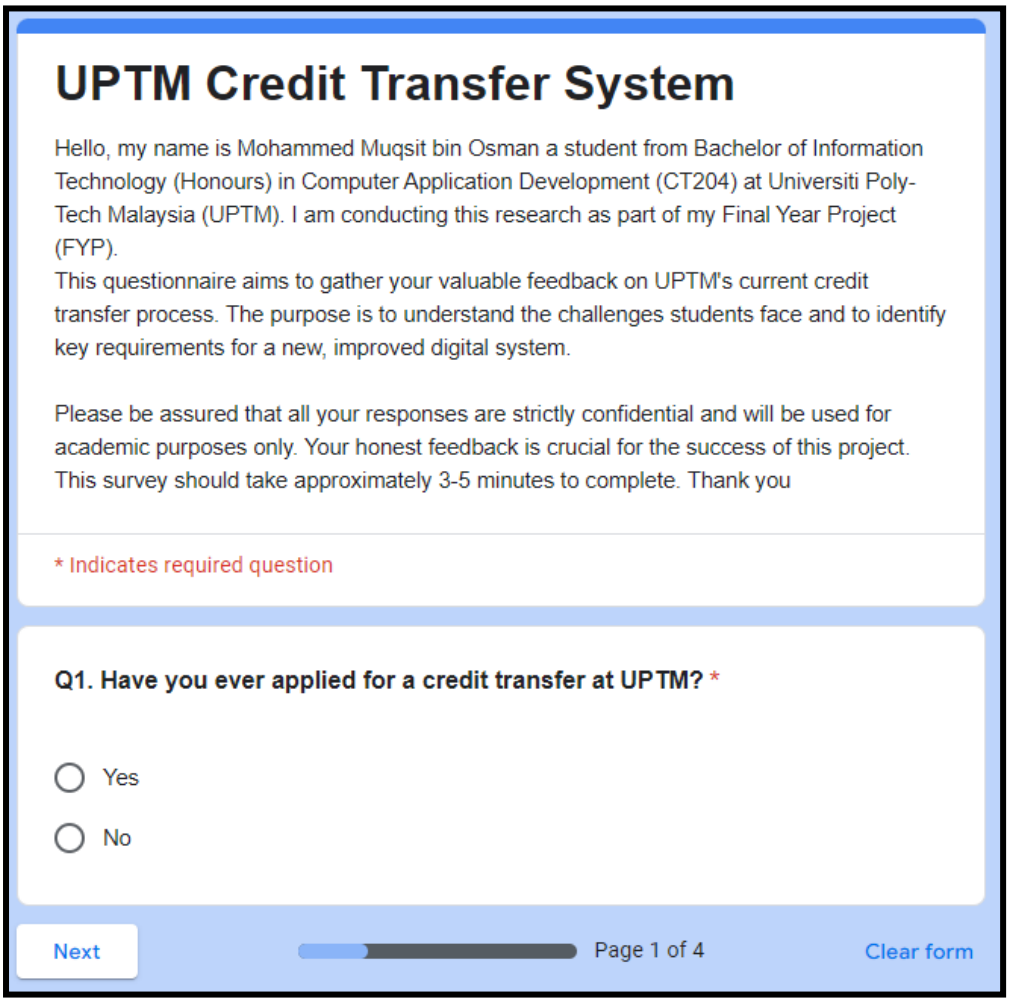
In this chapter, the project will be concluded through a review of the project schedule, risk management for the project, the achievements of the project, the constraints to the project, and the recommendations for future development of the project. According to the project schedule created for the project, the project will be divided into six main phases between August 2025 and April 2026. Furthermore, the Work Breakdown Structure and Gantt chart for the project will provide a more detailed review of the actual timeline for the project and its phases to be fulfilled. Risk management for the system included the identification of potential risks to the project, such as the following: database dependency, document extraction, AI system availability, access management, production environment, and communication through email. Furthermore, strategies were created for the elimination or mitigation of each of these risks. Thus, the project was developed with these risks in mind, but each of the mitigations for those risks ensured that the project could still be fulfilled within the scope of the identified risks.

The objectives of the project were accomplished in each of the three main objectives. The first objective required that the requirements of the system were analyzed. Such analysis included interviewing and surveying the students, the HOP, the RP, and the AAS departments regarding their requirements. Thus, the requirement analysis has been fulfilled. The second objective required that a platform is developed that can act as a repository for all of the records related to the credit transfer process; this platform includes the modules for students' applications, transcripts, syllabi, approvals, and decisions. Thus, that objective has been fulfilled. Finally, the third objective was to develop a module to allow the comparison of syllabi submitted by students from two different departments; that module is implemented and uses a staged method of prompting the AI model to allow for thorough and reviewable approval decisions from each of the RPs. Thus, this third objective has likewise been fulfilled.

In recognizing the constraints and limitations to the system (its databases, its AI model, the services upon which it is reliant, the cost of hosting the system, and its potential scalability), recommendations for the future development of the system can be made to expand upon the current system and fulfill any needs that may emerge for the departments of UPTM. Recommendations for the future development of the system will be made regarding built-in document redaction, supporting documents for syllabi comparison, development of an AI chatbot for the system, fine-tuning of the AI model, production analytics, and background job queue workers. Overall, then, the UPTM Credit Transfer System can be concluded as a successful implementation of the project that was proposed. Based upon the design, development, testing, and evaluation of the system, the project has successfully created a system that fulfills its purpose for the current departments of the university. Thus, within the scope of the project, the system has achieved its purpose for the institution and allows for the development of future improvements to the system overall.

Appendix A – Questionnaire & Interview Questions

i. Questionnaire Question (Pre-Development)



UPTM Credit Transfer System

Hello, my name is Mohammed Muqsit bin Osman a student from Bachelor of Information Technology (Honours) in Computer Application Development (CT204) at Universiti Poly-Tech Malaysia (UPTM). I am conducting this research as part of my Final Year Project (FYP).
This questionnaire aims to gather your valuable feedback on UPTM's current credit transfer process. The purpose is to understand the challenges students face and to identify key requirements for a new, improved digital system.

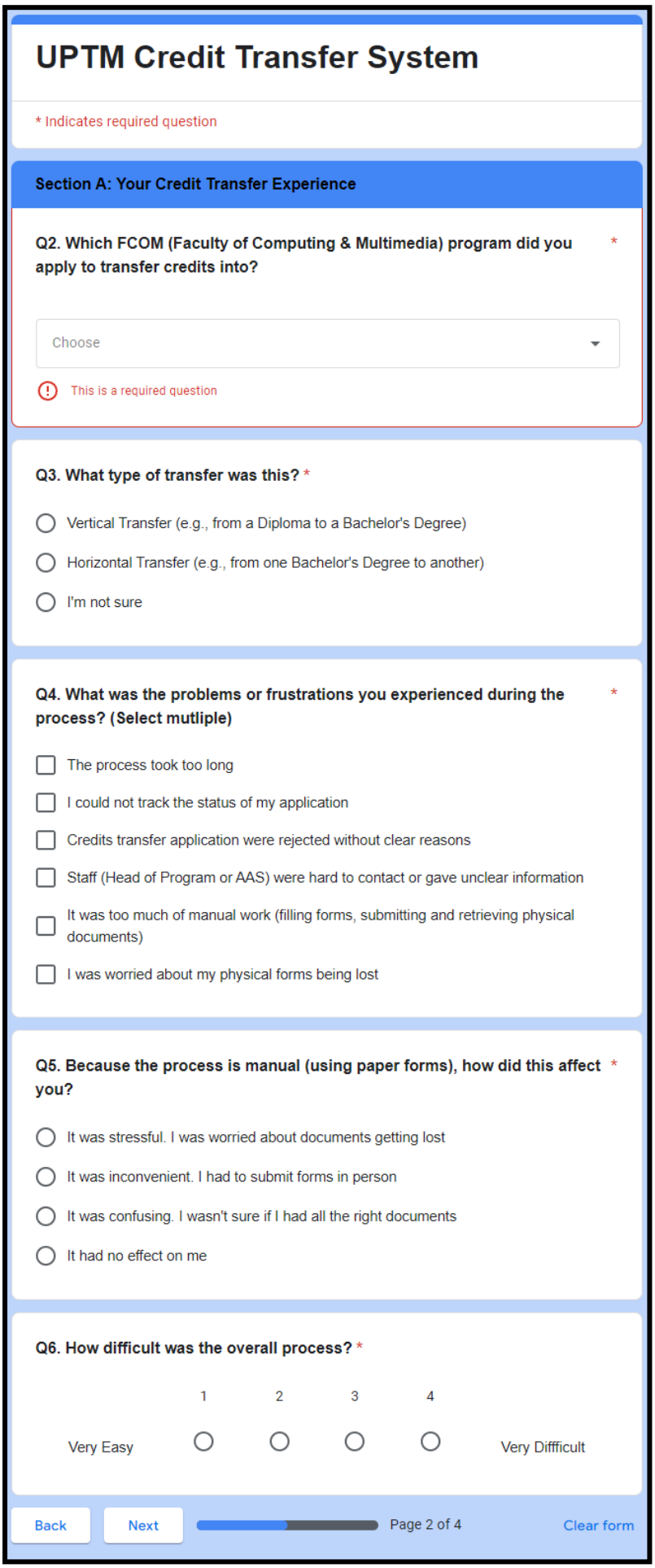
Please be assured that all your responses are strictly confidential and will be used for academic purposes only. Your honest feedback is crucial for the success of this project. This survey should take approximately 3-5 minutes to complete. Thank you

* Indicates required question

Q1. Have you ever applied for a credit transfer at UPTM? *

- ◯ Yes
- ◯ No

Next Page 1 of 4 Clear form



UPTM Credit Transfer System

* Indicates required question

Section A: Your Credit Transfer Experience

Q2. Which FCOM (Faculty of Computing & Multimedia) program did you apply to transfer credits into? *

Choose

This is a required question

Q3. What type of transfer was this? *

- ○ Vertical Transfer (e.g., from a Diploma to a Bachelor's Degree)
- ○ Horizontal Transfer (e.g., from one Bachelor's Degree to another)
- ○ I'm not sure

Q4. What was the problems or frustrations you experienced during the process? (Select mutliple) *

- ☐ The process took too long
- ☐ I could not track the status of my application
- ☐ Credits transfer application were rejected without clear reasons
- ☐ Staff (Head of Program or AAS) were hard to contact or gave unclear information
- ☐ It was too much of manual work (filling forms, submitting and retrieving physical documents)
- ☐ I was worried about my physical forms being lost

Q5. Because the process is manual (using paper forms), how did this affect you? *

- ○ It was stressful. I was worried about documents getting lost
- ○ It was inconvenient. I had to submit forms in person
- ○ It was confusing. I wasn't sure if I had all the right documents
- ○ It had no effect on me

Q6. How difficult was the overall process? *

	1	2	3	4	
Very Easy	○	○	○	○	Very Diffficult

Back Next Page 2 of 4 Clear form

UPTM Credit Transfer System

* Indicates required question

Section B: Expectations

Q7. Do you agree that the credit transfer process at UPTM should be fully digital (online submission, tracking, and approval)? *

	1	2	3	4	
Strongly disagree	○	○	○	○	Strongly agree

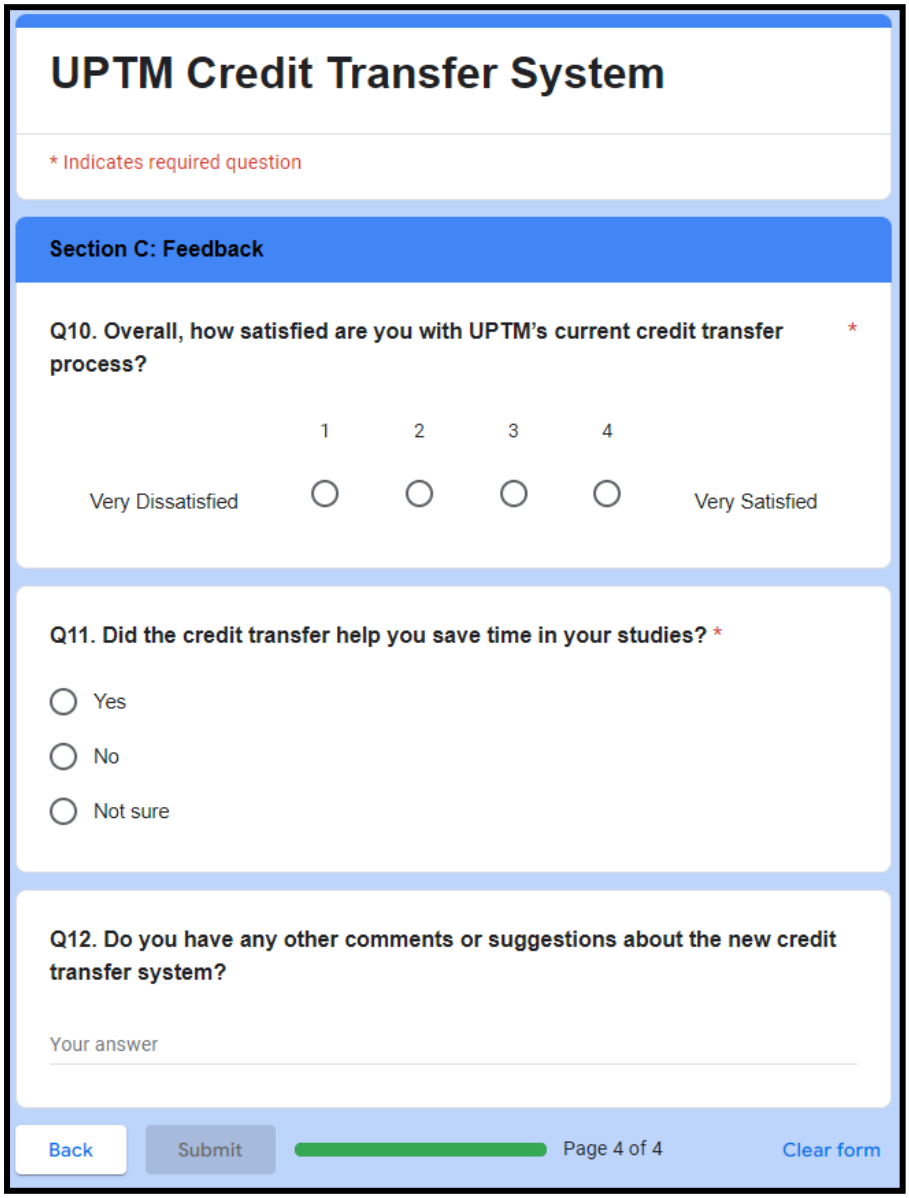
Q8. What features do you expect from the new online credit transfer system, how important are the following features to you? *

	Not Important	Important	Very Important
Easy-to-use online portal with a clean and simple interface (good UI/UX)	○	○	○
Step-by-step user guide (e.g., video or PDF) explaining how to apply	○	○	○
Public articulation list showing courses from other institutions that have already been approved for credit transfer	○	○	○
Real-time status tracking (e.g., "Application received," "Under review," "Approved")	○	○	○
Mobile-friendly access (works well on phones and tablets)	○	○	○

Q9. What is the single biggest improvement you hope to see from a new online system? *

- ○ Transparency: Being able to track my application status from start to finish.
- ○ Clarity: Knowing exactly what to do and what documents are needed.
- ○ Convenience: Submitting everything online with no physical paper forms.
- ○ Consistency: Knowing the rules are applied fairly to everyone.

Back Next Page 3 of 4 Clear form



UPTM Credit Transfer System

* Indicates required question

Section C: Feedback

Q10. Overall, how satisfied are you with UPTM's current credit transfer process? *

	1	2	3	4	
Very Dissatisfied	○	○	○	○	Very Satisfied

Q11. Did the credit transfer help you save time in your studies? *

- ○ Yes
- ○ No
- ○ Not sure

Q12. Do you have any other comments or suggestions about the new credit transfer system?

Your answer

Back Submit Page 4 of 4 Clear form

ii. Interview question (Pre-Development)

a. HOP Interview Question

No.	Question
1	How long have you been involved as a head of program, and how familiar are you with the credit transfer process?
2	Can you describe the main steps you currently follow when a student applies for credit transfer?
3	What kind of information or documents do you usually collect from students during the process?
4	How do you communicate and coordinate with Resource Persons (RPs) and AAS staff throughout the process?
5	What are the most common challenges or delays you experience with the current manual system?
6	How do you currently keep track of application progress or approval status?
7	How do you verify that all required steps (such as RP review and AAS update) are completed before approval?
8	What problems do you face when retrieving or reviewing previous credit transfer records?
9	What key features or functions would you like to see in an automated system to make your work easier?
10	In your opinion, how could a digital system improve the efficiency and transparency of the overall process?

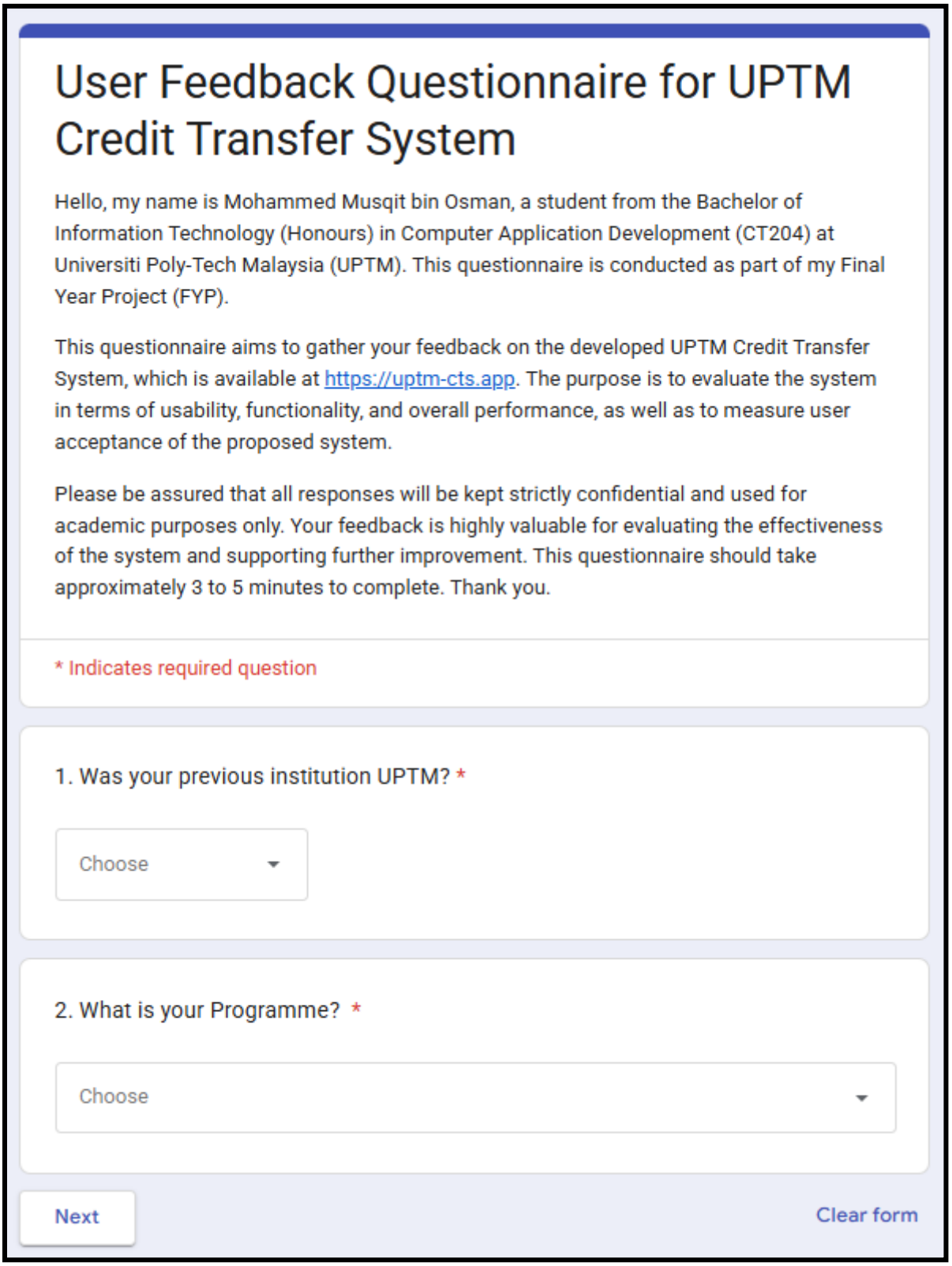
b. HOP Interview Question

No.	Question
1	How long have you been serving as a Resource Person, and how often are you involved in credit transfer evaluations?
2	Can you describe your role in the credit transfer process and what tasks you perform during an evaluation?
3	What criteria do you normally use to determine whether two courses are equivalent?
4	How do you usually obtain the syllabi or course outlines needed for your comparison?
5	What are the main challenges you face when comparing syllabi manually?
6	How do you currently record or document your evaluation results?
7	Would you prefer if the system store comparison data and documents in the cloud or keep them as physical copies?
8	In your opinion, do you agree if the syllabus comparison process is automated using AI technology? Why or why not?

c. AAS Interview Question

No.	Question
1	Can you walk me through what you usually do after receiving a completed credit transfer form from a coordinator or student?
2	After a credit transfer application is approved and entered into the Campus Management System (CMS), how do you store or archive the physical form?
3	Have you ever experienced cases where physical credit transfer forms were damaged, misplaced, or lost? What happened?
4	Do you find it difficult to process applications when the handwriting on the form is hard to read? Has this caused any issues?
5	What usually happens if a student submits a form with incorrect or incomplete information?
6	Besides handwriting or lost forms, what other issues or challenges have you faced in processing credit transfers?
7	What's your opinion on digitalizing the credit transfer application process—for example, using an online form instead of paper?

iii. Questionnaire question (Post-Development)



User Feedback Questionnaire for UPTM Credit Transfer System

Hello, my name is Mohammed Musqit bin Osman, a student from the Bachelor of Information Technology (Honours) in Computer Application Development (CT204) at Universiti Poly-Tech Malaysia (UPTM). This questionnaire is conducted as part of my Final Year Project (FYP).

This questionnaire aims to gather your feedback on the developed UPTM Credit Transfer System, which is available at https://uptm-cts.app. The purpose is to evaluate the system in terms of usability, functionality, and overall performance, as well as to measure user acceptance of the proposed system.

Please be assured that all responses will be kept strictly confidential and used for academic purposes only. Your feedback is highly valuable for evaluating the effectiveness of the system and supporting further improvement. This questionnaire should take approximately 3 to 5 minutes to complete. Thank you.

* Indicates required question

1. Was your previous institution UPTM? *

Choose

2. What is your Programme? *

Choose

Next Clear form

User Feedback Questionnaire for UPTM Credit Transfer System

* Indicates required question

Section B: Objective

3. The system meets my needs in applying for credit transfer. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

4 The workflow of the system matches the steps I need to complete as a student *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

5. The system makes it easier to store and manage my credit transfer application records. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

6. The system makes it easier to access and review my application information in one place. *

- ○ Strongly Agree
- ○ Agree
- ○ Disagree
- ○ Strongly Disagree

7. The system helps support a clearer course matching process. *

- ○ Strongly Agree
- ○ Agree
- ○ Disagree
- ○ Strongly Disagree

8. The system makes syllabus-related evaluation feel more consistent and transparent. *

- ○ Strongly Agree
- ○ Agree
- ○ Disagree
- ○ Strongly Disagree

Back Next Clear form

User Feedback Questionnaire for UPTM Credit Transfer System

* Indicates required question

Section C: Functional Requirements

9. The system allows me to view course mapping information clearly. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

10. The system allows me to download my CTA result easily. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

11. The system allows me to track my application status clearly. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

12. The system allows me to update or complete missing application documents easily. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

Back Next Clear form

User Feedback Questionnaire for UPTM Credit Transfer System

* Indicates required question

Section D: Non-Functional Requirements

13. The system is easy to understand and use. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

14. The system responds quickly when I log in, submit my application, or view my application details. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

15. I feel confident that my personal and academic information is protected when using the system. *

- ◯ Strongly Agree
- ◯ Agree
- ◯ Disagree
- ◯ Strongly Disagree

Back Submit Clear form

iv. Interview question (Post-Development)

a. HOP Interview Question

No.	Question
1	Does the system achieve the objective of automating the credit transfer application process?
2	Does the system achieve the objective of supporting academic evaluation with AI-assisted comparison?
3	Does the system achieve the objective of providing centralized tracking and record management?
4	Are the functional requirements for HOP (Table 3.3) adequately supported by the system?
5	Does the articulation management feature improve how you maintain course mappings?
6	Does the system provide acceptable performance during normal operations?
7	Is the system interface clear and easy to use for your tasks?
8	Does the system enforce proper access control and data security?
9	Would you recommend using this system for the actual credit transfer process at UPTM?

b. RP Interview Question

No.	Question
1	Does the system achieve the objective of automating the credit transfer application process?
2	Does the system achieve the objective of supporting academic evaluation with AI-assisted comparison?
3	Does the system achieve the objective of providing centralized tracking and record management?
4	Are the functional requirements for RP (Table 3.4) adequately supported by the system?
5	Does the AI comparison meet the expectation you expressed in the earlier interview about wanting automated similarity percentages?
6	Does the system provide acceptable performance during the comparison process?
7	Is the system interface clear and easy to use for your evaluation tasks?
8	Does the system maintain proper records of your evaluation decisions?
9	Would you recommend using this system for the actual credit transfer evaluation process?

c. AAS Interview Question

No.	Question
1	Does the system achieve the objective of automating the credit transfer application process?
2	Does the system achieve the objective of providing centralized tracking and record management?
3	Does the system reduce the issues you experienced with physical form handling?
4	Are the functional requirements for AAS (Table 3.5) adequately supported by the system?
5	Does the export feature meet your operational needs for updating CMS?
6	Does the system provide acceptable performance when processing exports?
7	Is the system interface clear and easy to use for your administrative tasks?
8	Does the system preserve proper records of export activities?
9	Would you recommend using this system for the actual credit transfer administrative process?

Appendix B – User Manual

This appendix presents the operational user manual for the implemented UPTM Credit Transfer System (CTS). A role-specific version of the same manual is also available within the system through the built-in documentation module for Student, HOP, RP, and AAS users. The following sections summarize the main workflows and interfaces for each role.

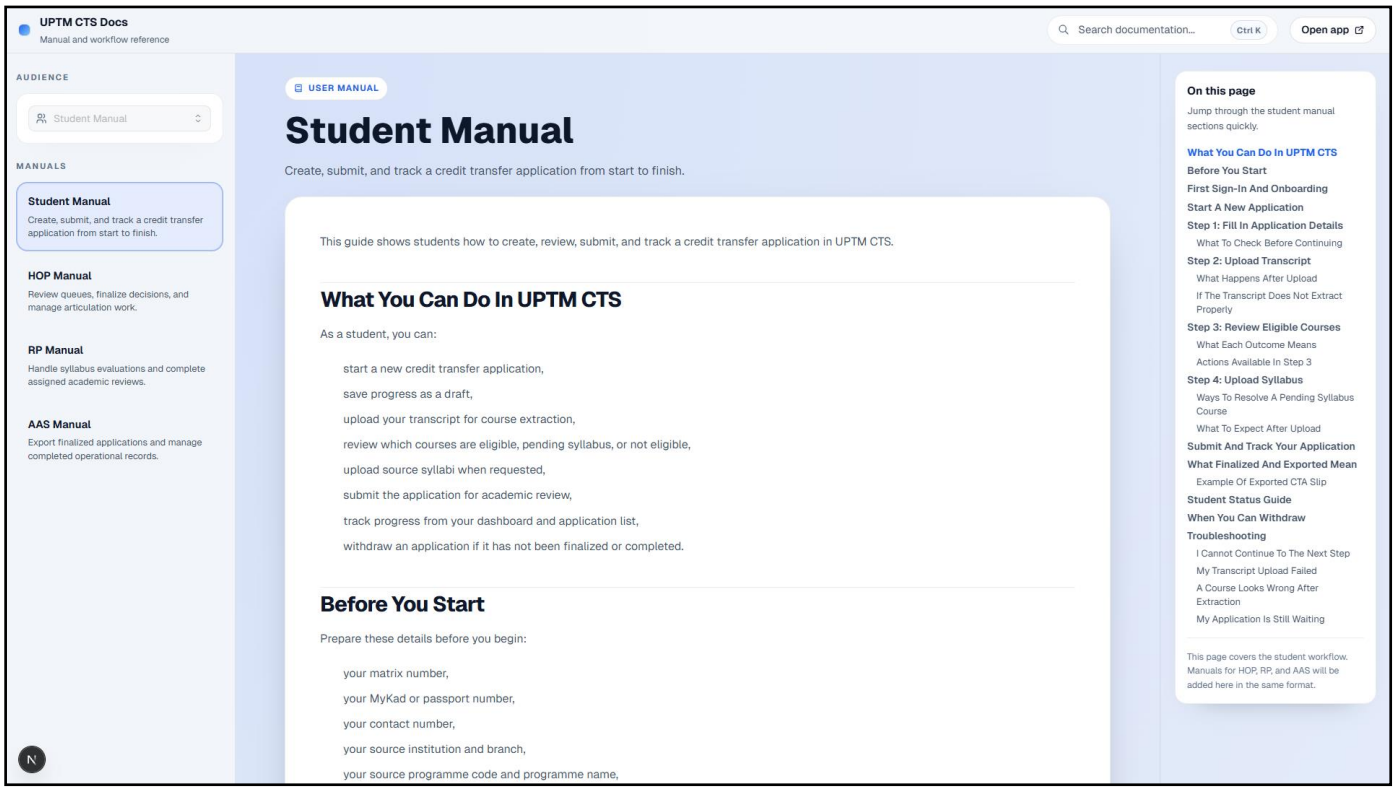


Figure 1: Student Manual

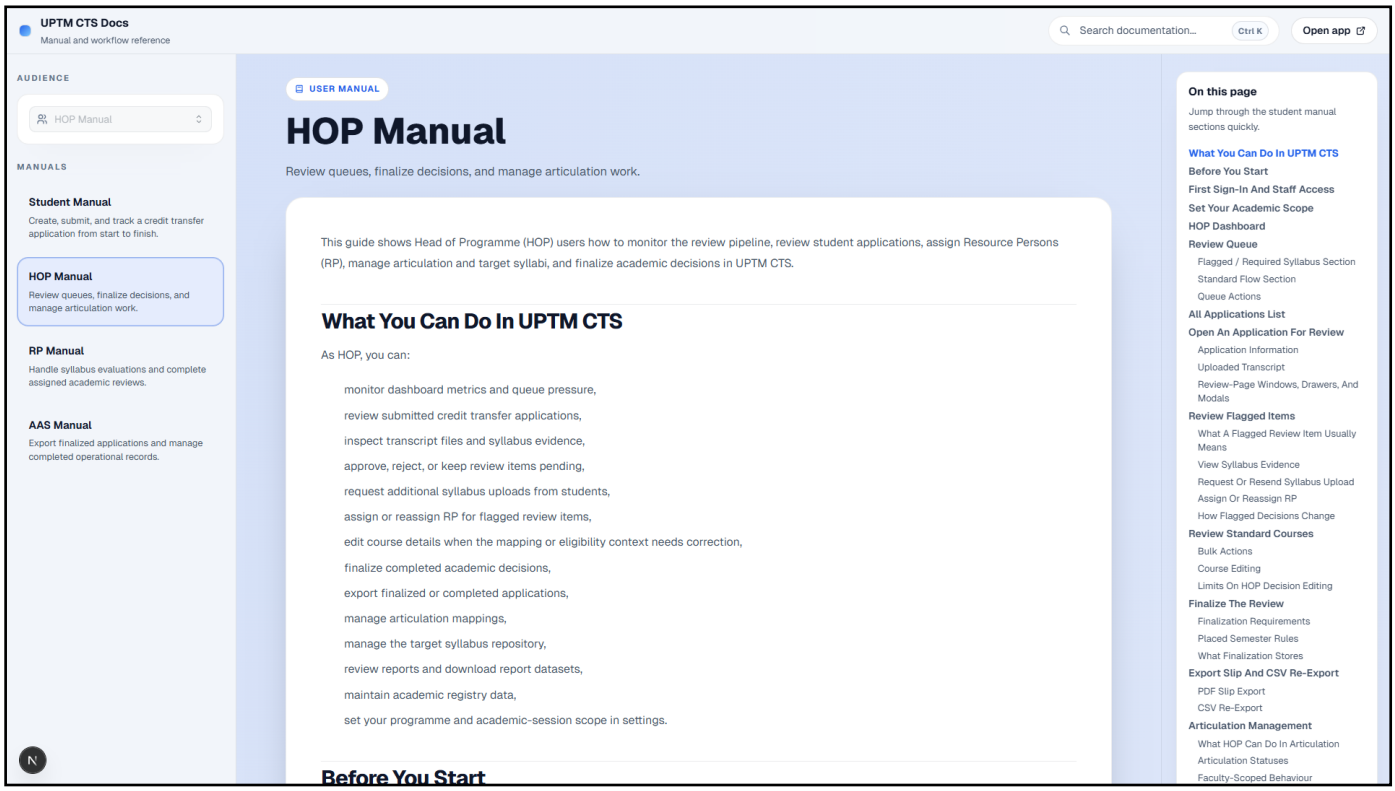


Figure 2: HOP Manual

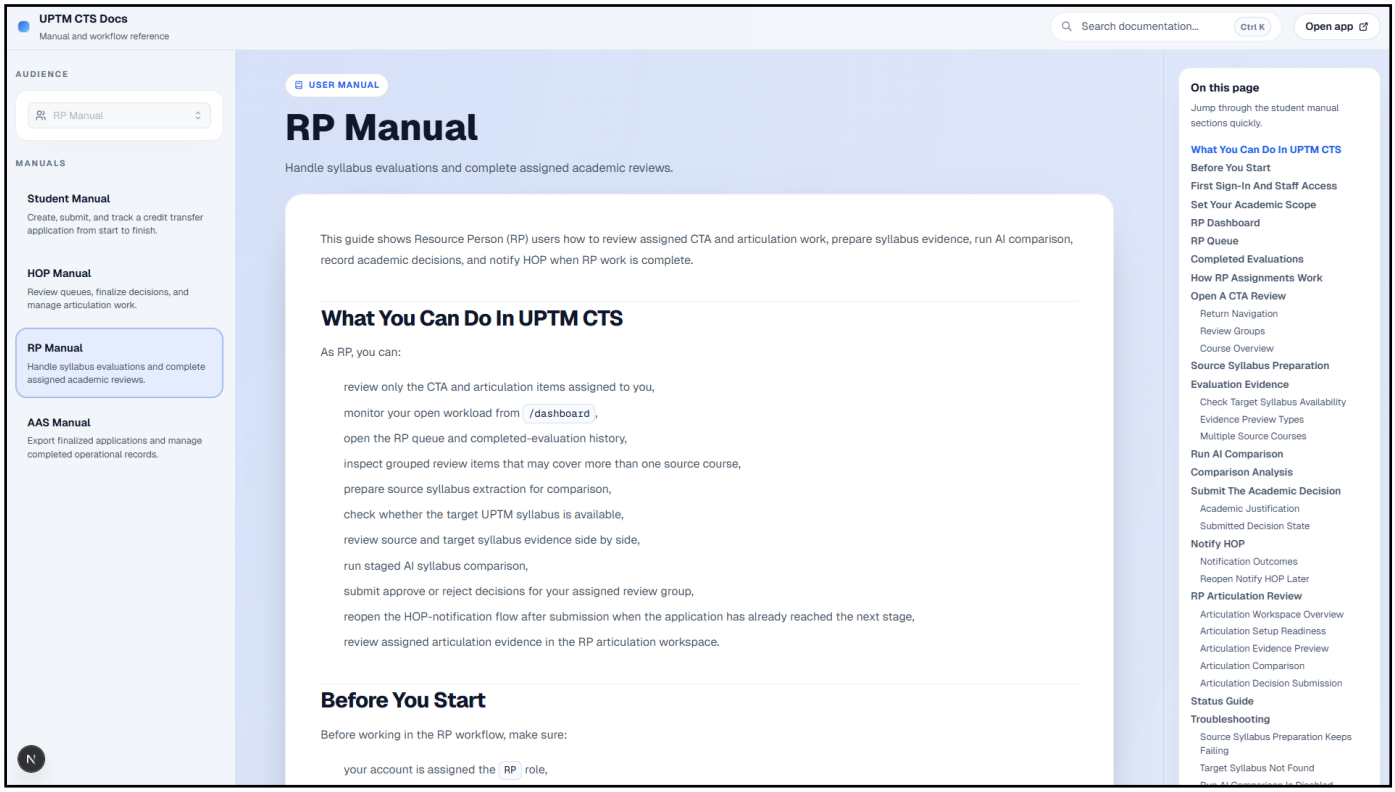


Figure 3: RP Manual

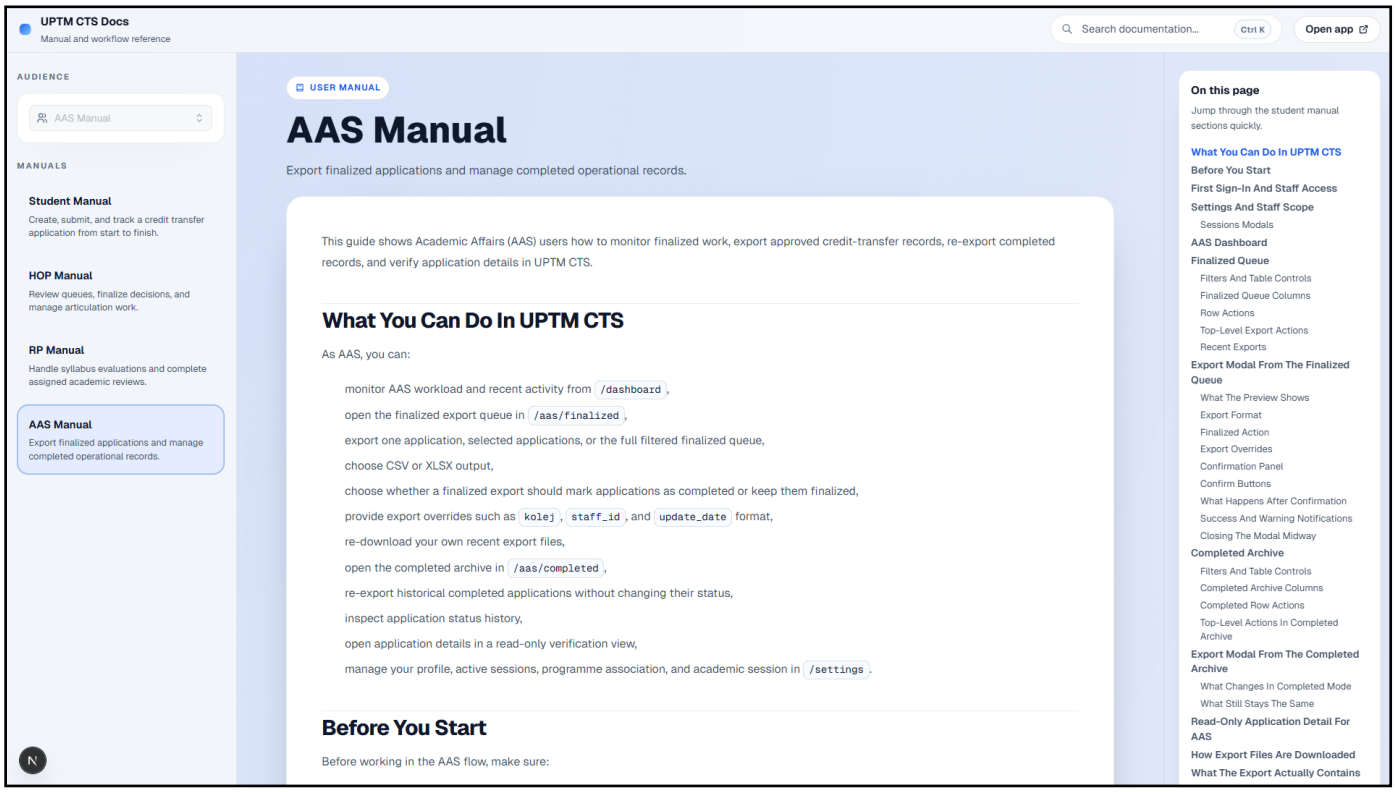


Figure 4: AAS Manual

Student Guide

This guide shows students how to create, review, submit, and track a credit transfer application in UPTM CTS.

What You Can Do In UPTM CTS

As a student, you can:

- start a new credit transfer application,
- save progress as a draft,
- upload your transcript for course extraction,
- review which courses are eligible, pending syllabus, or not eligible,
- upload source syllabi when requested,
- submit the application for academic review,
- track progress from your dashboard and application list,
- withdraw an application if it has not been finalized or completed.

Before You Start

Prepare these details before you begin:

- your matrix number,
- your MyKad or passport number,
- your contact number,

- your source institution and branch,
- your source programme code and programme name,
- your target UPTM programme,
- your academic session,
- your latest CGPA and graduation year if available,
- a transcript in PDF format,
- syllabus files for flagged courses if you already have them.

First Sign-In And Onboarding

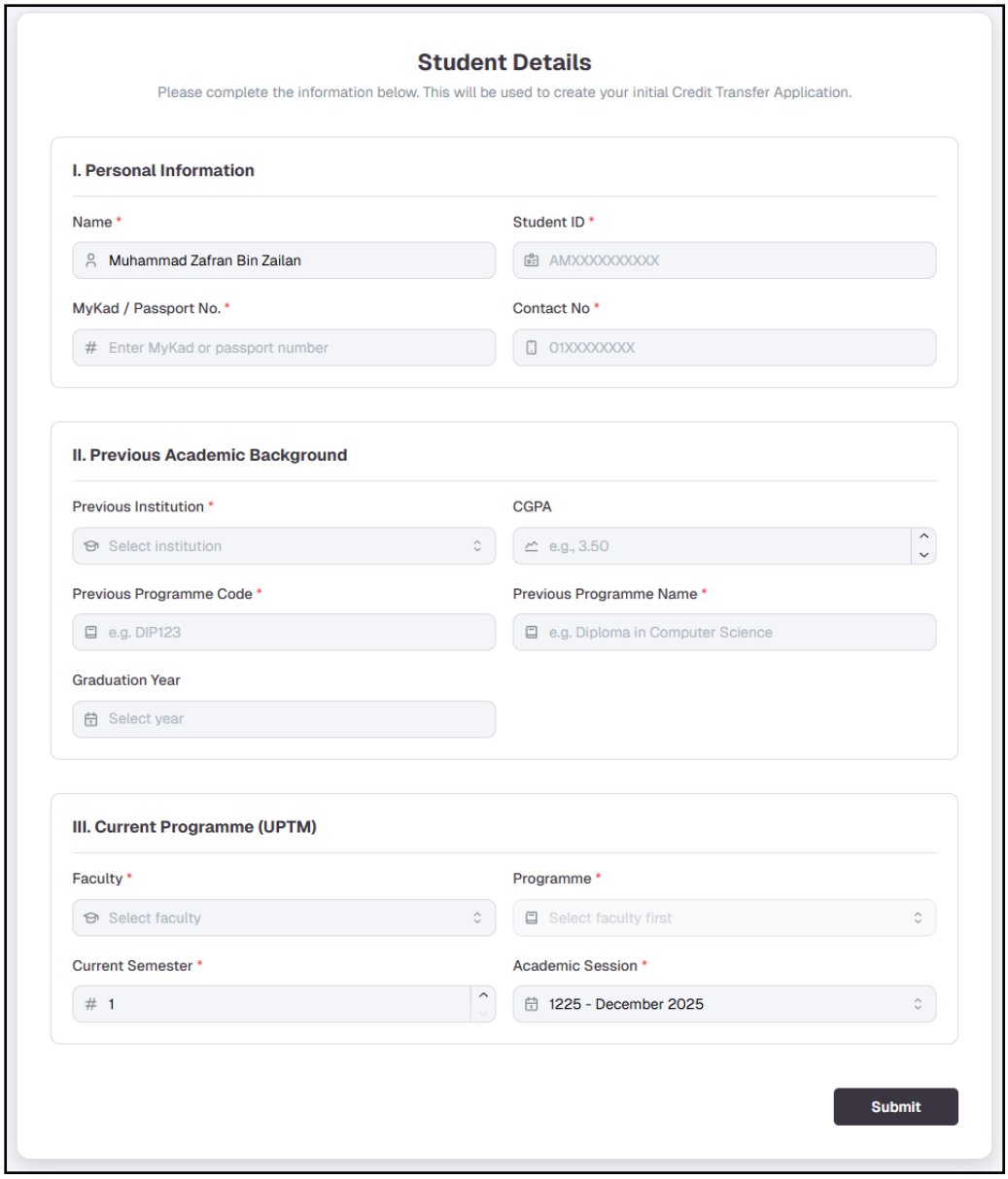


Figure 5: Student onboarding form.

When you sign in for the first time, the system may send you to the onboarding page before you reach the dashboard.

Use onboarding to complete:

- your personal details,
- your previous institution and branch,
- your previous programme code and programme name,
- your current UPTM programme,
- your current semester,
- your academic session.

If your institution or branch is not listed, choose **Other (Not Listed)** and enter the correct name manually.

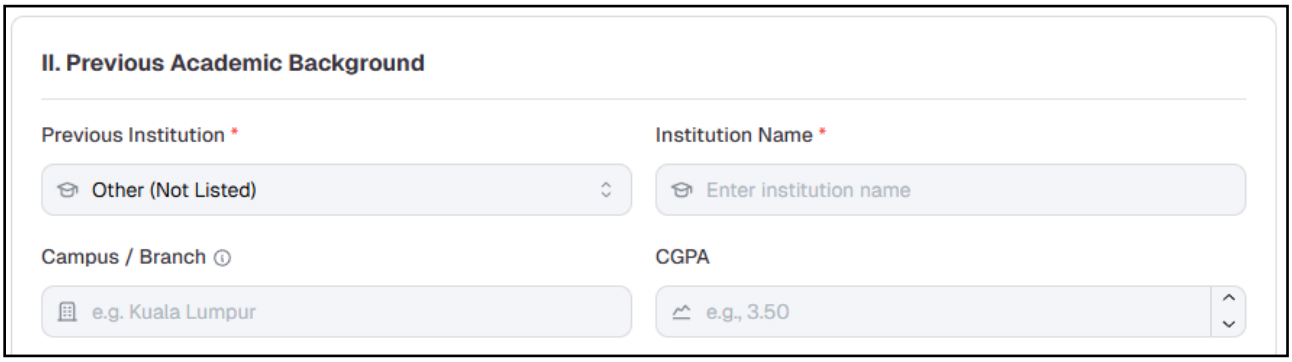


Figure 6: Onboarding institution other option.

After you submit onboarding:

- your profile is marked as complete,
- the system creates your first draft credit transfer application,
- you are redirected to the dashboard.

Routes involved:

- `/sign-in`
- `/onboarding`
- `/dashboard`

Start A New Application

Open the dashboard and choose **New Application**.

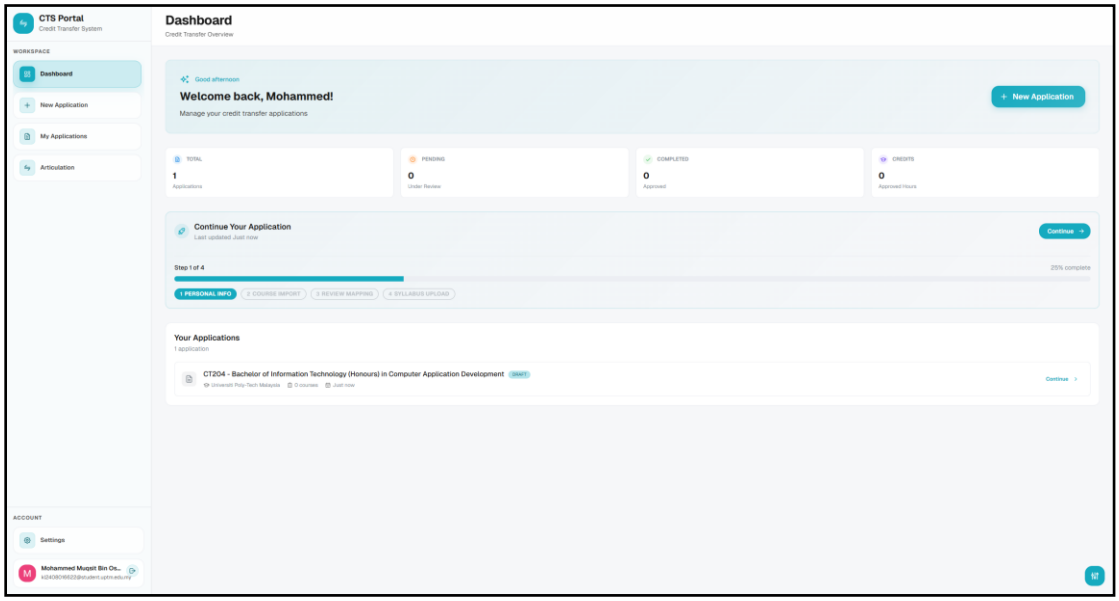


Figure 7: New application button on dashboard.

The system creates or reuses your current draft application and sends you to the first step.

Routes involved:

- `/dashboard`
- `/cta/new`
- `/cta/new/step-1`

Step 1: Fill In Application Details

Use Step 1 to confirm your academic profile and source-study details before uploading the transcript.

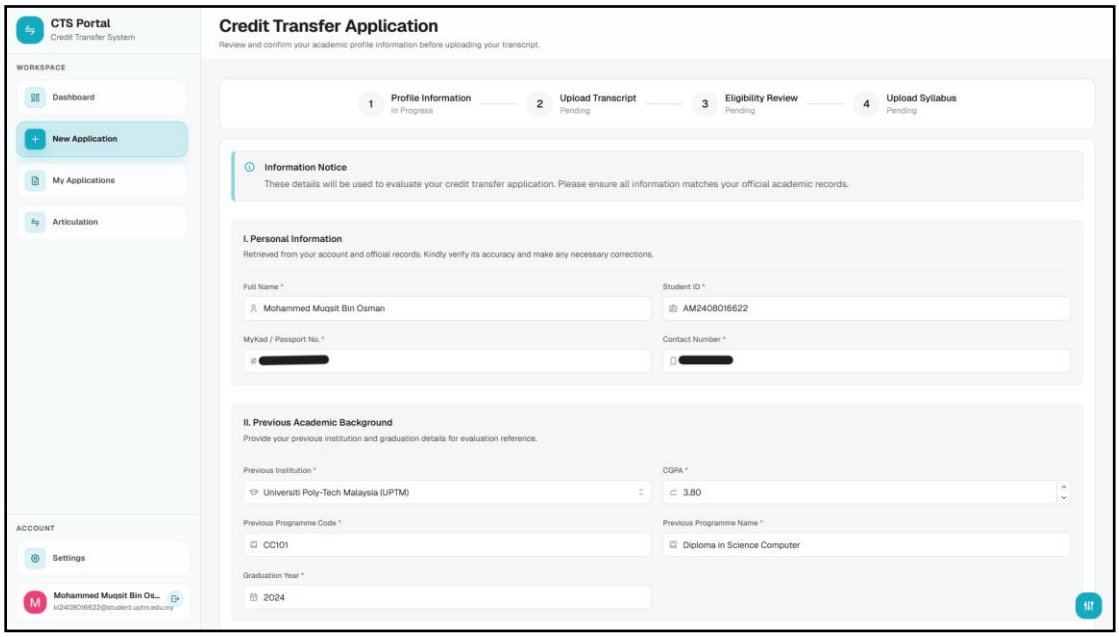


Figure 8: Step 1 application details form.

Enter or review:

- student matrix number,
- MyKad or passport number,
- current semester,
- contact number,
- source institution and branch,
- source programme code and programme name,
- source graduation year,
- source CGPA,
- credit transfer type,
- target UPTM programme,
- academic session.

What To Check Before Continuing

Before moving to Step 2, make sure:

- the source institution is correct,

- the target UPTM programme is correct,
- the academic session is correct,
- the programme and branch information matches your transcript,
- required fields are filled in.

You can save the page as a draft if you are not ready to continue.

Step 2: Upload Transcript

Use Step 2 to upload your transcript and let the system extract your courses.

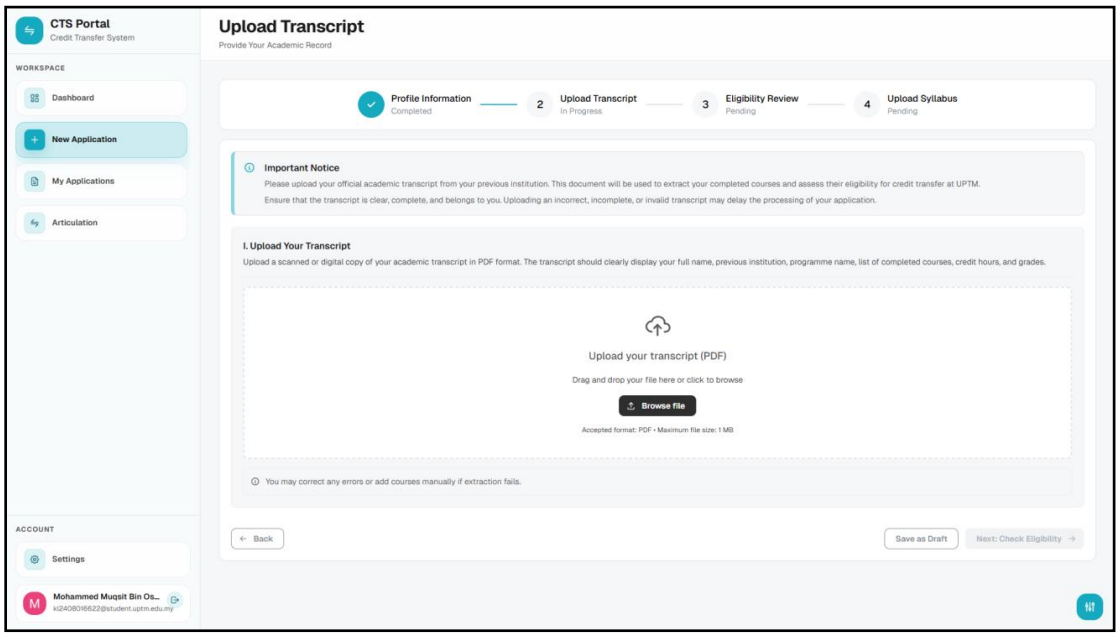


Figure 9: Step 2 transcript upload page.

Accepted file:

- PDF only

The upload flow is:

1. Select your transcript PDF.
2. Wait for the system to convert and analyze the document.
3. Review the extracted course list.
4. Add, edit, or delete courses if the extraction needs correction.

What Happens After Upload

After the transcript is processed, the system tries to:

- extract course information from the transcript,
- match your source courses to existing articulation data,
- check basic eligibility such as grade and credit-hour rules,
- prepare a course list for your review in Step 3.

If The Transcript Does Not Extract Properly

If the result looks incomplete or inaccurate:

- verify that the uploaded file is a readable PDF,
- try another copy of the transcript if the scan quality is poor,
- edit the extracted courses manually before continuing,
- remove incorrect courses and add missing ones where needed.

Step 3: Review Eligible Courses

Use Step 3 to review the extracted course list before submission.

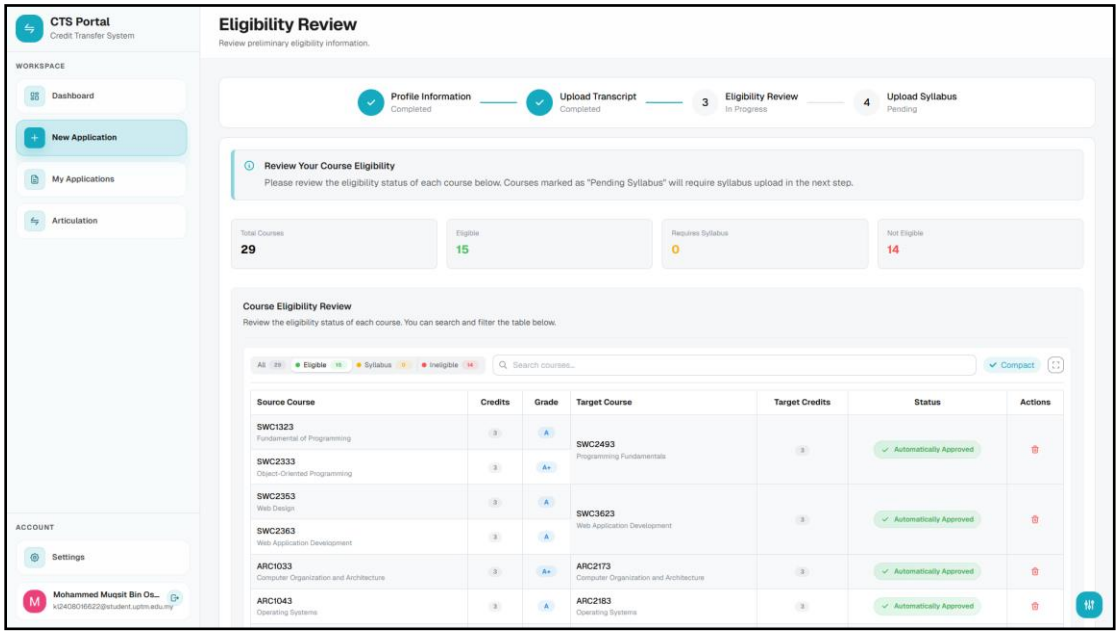


Figure 10:: Step 3 course review page.

The page groups courses into outcomes such as:

- automatically approved,
- pending syllabus,
- grade below minimum,
- credits below target,
- not eligible for transfer.

What Each Outcome Means

- **Automatically Approved** means the system found a valid articulation match and the course passed the basic eligibility checks.
- **Pending Syllabus** means the course may still be transferable, but additional syllabus evidence is needed.
- **Grade Below Minimum** means the course does not meet the minimum grade rule.
- **Credits Below Target** means the source course does not meet the target credit requirement.
- **Not Eligible For Transfer** means the course cannot move forward in the current application.

Actions Available In Step 3

You can:

- save as draft,
- search and review the course table,
- delete courses you do not want included,
- continue to Step 4 if any course still needs a syllabus,
- submit directly if no syllabus is required.

Step 4: Upload Syllabus

Step 4 only appears when at least one course still needs source-course syllabus evidence.

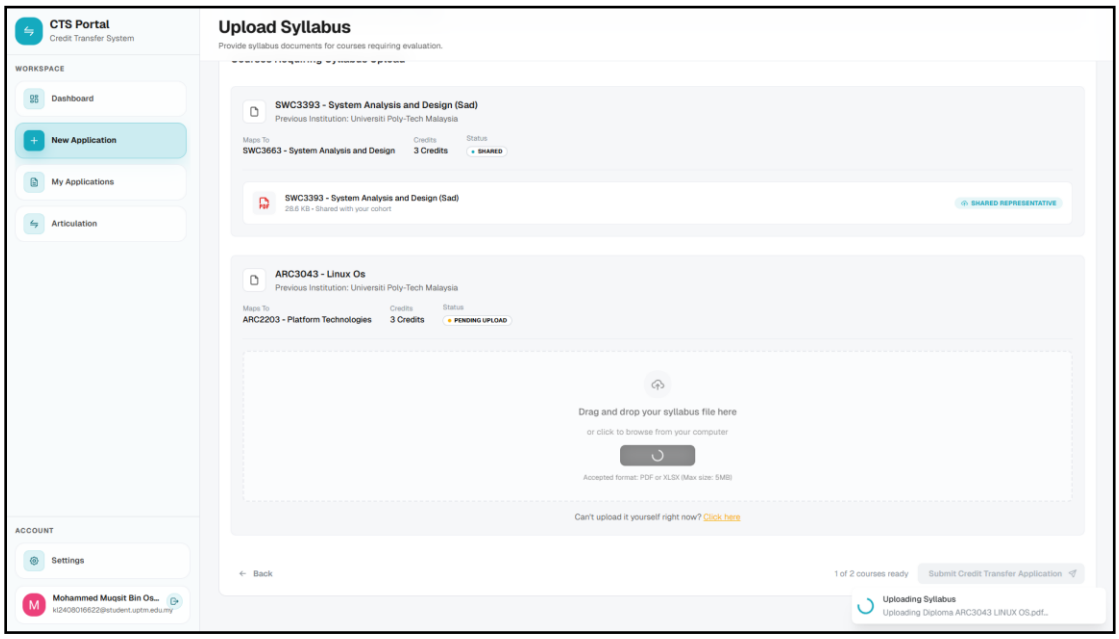


Figure 11: Step 4 syllabus upload page.

Accepted files:

- PDF
- XLSX

If you upload an Excel workbook with multiple sheets, the system asks you to choose the correct worksheet.

Ways To Resolve A Pending Syllabus Course

For each flagged course, you can:

- upload the syllabus now,
- mark that you will upload it later,
- mark that you cannot provide it,
- wait for a shared representative syllabus if the workflow supports it.

What To Expect After Upload

After a syllabus is uploaded:

- the file is attached to the selected course,
- the application keeps moving through the evidence workflow,
- the course can become ready for academic review.

If you choose **Upload Later** or **Cannot Provide**, the application may remain in an awaiting-syllabus state until the evidence is resolved or the deadline passes.

Submit And Track Your Application

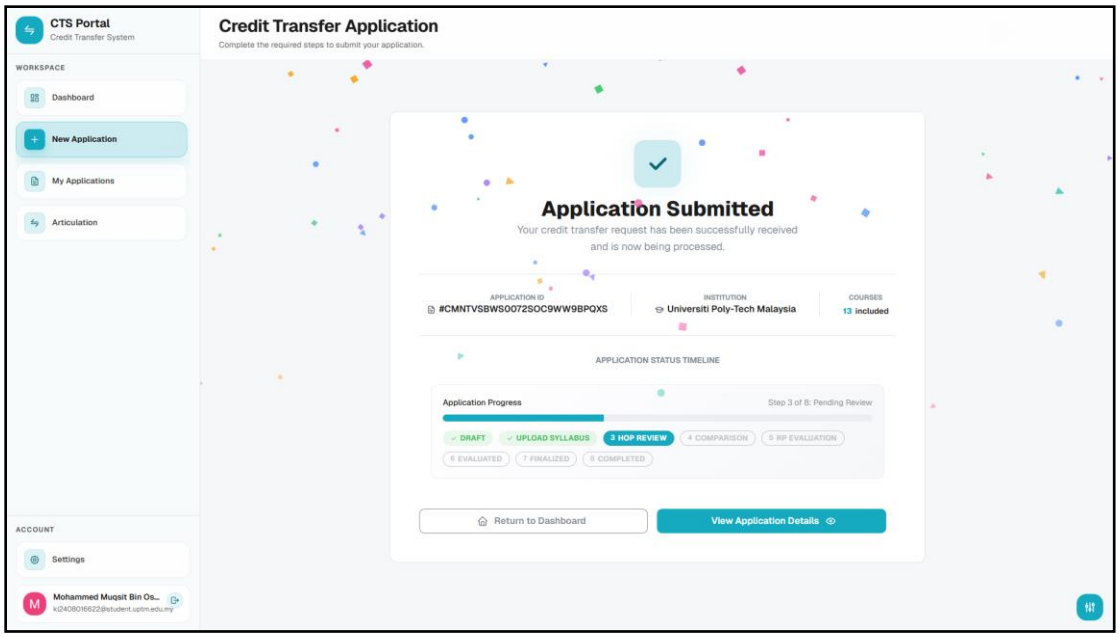


Figure 12: Application Submission Success Page.

After student-side requirements are complete, the application moves into staff review.

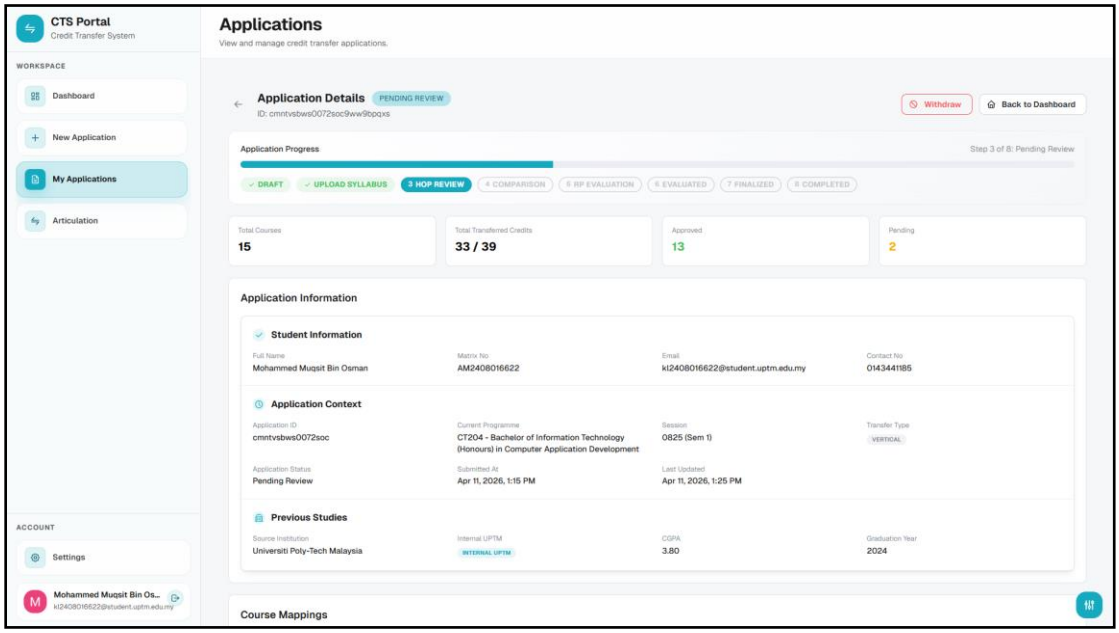


Figure 13: Application Detail Overview.

Use these pages to monitor progress:

- `/dashboard`
- `/applications`
- `/applications/[id]`

From the application list and detail view, you can review:

- the current status,
- the target programme,
- source institution details,
- the number of courses included,
- whether syllabus uploads are still pending,
- course-level decisions and supporting syllabus records.

What Finalized And Exported Mean

Near the end of the process, you may see the application move through these final states:

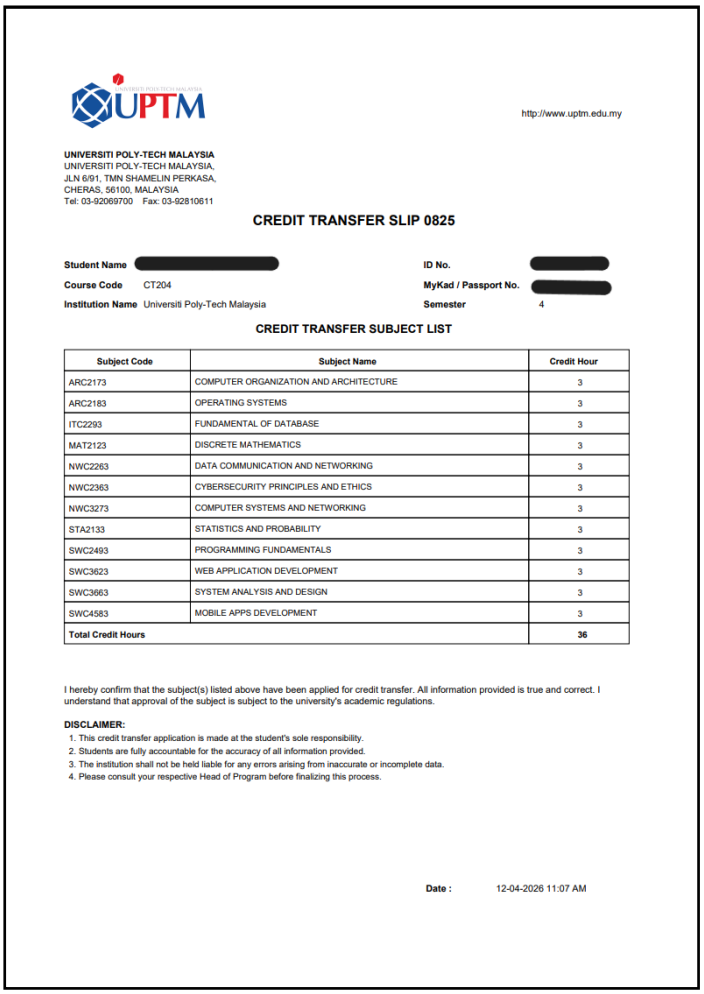
- **FINALIZED** means the academic decision has been locked by HOP.
- **COMPLETED** means AAS has exported the finalized result into the downstream record flow and closed the case.

As a student:

- you do not perform the export yourself,
- you can still open the application to review the outcome,
- you can no longer edit or withdraw the application after it reaches these final states.

Example Of Exported CTA Slip

The image below shows an example of the exported CTA slip PDF generated after the application is completed.



UPTM

http://www.uptm.edu.my

UNIVERSITI POLY-TECH MALAYSIA
UNIVERSITI POLY-TECH MALAYSIA,
JLN 6/91, TMN SHAMELIN PERKASA,
CHERAS, 56100, MALAYSIA
Tel: 03-92069700 Fax: 03-92810611

CREDIT TRANSFER SLIP 0825

Student Name [redacted] **ID No.** [redacted]
Course Code CT204 **MyKad / Passport No.** [redacted]
Institution Name Universiti Poly-Tech Malaysia **Semester** 4

CREDIT TRANSFER SUBJECT LIST

Subject Code	Subject Name	Credit Hour
ARC2173	COMPUTER ORGANIZATION AND ARCHITECTURE	3
ARC2183	OPERATING SYSTEMS	3
ITC2293	FUNDAMENTAL OF DATABASE	3
MAT2123	DISCRETE MATHEMATICS	3
NWC2263	DATA COMMUNICATION AND NETWORKING	3
NWC2363	CYBERSECURITY PRINCIPLES AND ETHICS	3
NWC3273	COMPUTER SYSTEMS AND NETWORKING	3
STA2133	STATISTICS AND PROBABILITY	3
SWC2493	PROGRAMMING FUNDAMENTALS	3
SWC3623	WEB APPLICATION DEVELOPMENT	3
SWC3663	SYSTEM ANALYSIS AND DESIGN	3
SWC4583	MOBILE APPS DEVELOPMENT	3
Total Credit Hours		**36**

I hereby confirm that the subject(s) listed above have been applied for credit transfer. All information provided is true and correct. I understand that approval of the subject is subject to the university's academic regulations.

DISCLAIMER:
1. This credit transfer application is made at the student's sole responsibility.
2. Students are fully accountable for the accuracy of all information provided.
3. The institution shall not be held liable for any errors arising from inaccurate or incomplete data.
4. Please consult your respective Head of Program before finalizing this process.

Date : 12-04-2026 11:07 AM

Figure 14: Example exported CTA slip PDF.

Student Status Guide

Status	Meaning
DRAFT	You are still preparing the application.
AWAITING_SYLLABUS_UPLOAD	One or more courses still need syllabus evidence.
PENDING_HOP_REVIEW	Your submission is complete and waiting for Head of Programme review.
SYLLABUS_COMPARISON_IN_PROGRESS	The system is running syllabus comparison work.
RP_EVALUATION_PENDING	The application is waiting for Resource Person evaluation.
EVALUATION_COMPLETED	Academic evaluation is done and awaiting finalization.
FINALIZED	The decision has been finalized by HOP.
COMPLETED	The case has already been exported and closed by AAS.
WITHDRAWN	You withdrew the application.

When You Can Withdraw

You can withdraw an application while it is still in a student or review-stage workflow.

You cannot withdraw an application once it is:

- `FINALIZED`,
- `COMPLETED`.

Troubleshooting

I Cannot Continue To The Next Step

Check for:

- missing required fields,
- invalid transcript file type,
- invalid or incomplete institution and programme details,
- unresolved syllabus requirements.

My Transcript Upload Failed

Check that:

- the file is in PDF format,
- the file is readable and not corrupted,
- the scan is clear enough for extraction,
- the file size is within the current upload limit shown by the system.

A Course Looks Wrong After Extraction

In Step 2, review the extracted rows and:

- edit incorrect course details,
- add missing courses,
- delete rows that should not be included.

My Application Is Still Waiting

Open the application detail page and check:

- whether any course is still pending syllabus,
- whether the application is already with HOP or RP,
- whether the status has moved into evaluation or finalization.

HOP Guide

This guide shows Head of Programme (HOP) users how to monitor the review pipeline, review student applications, assign Resource Persons (RP), manage articulation and target syllabi, and finalize academic decisions in UPTM CTS.

What You Can Do In UPTM CTS

As HOP, you can:

- monitor dashboard metrics and queue pressure,
- review submitted credit transfer applications,
- inspect transcript files and syllabus evidence,
- approve, reject, or keep review items pending,
- request additional syllabus uploads from students,
- assign or reassign RP for flagged review items,
- edit course details when the mapping or eligibility context needs correction,
- finalize completed academic decisions,
- export finalized or completed applications,
- manage articulation mappings,
- manage the target syllabus repository,
- review reports and download report datasets,
- maintain academic registry data,
- set your programme and academic-session scope in settings.

Before You Start

Before working in the HOP workflow, make sure:

- your account is assigned the `HOP` role,
- your programme association and academic session are set in `/settings`,
- the target programme admission semester is configured correctly when you expect to finalize applications,
- target syllabi needed for RP comparison are available in `/syllabus`,
- you understand that `FINALIZED` locks the academic decision and `COMPLETED` means AAS has already exported the case.

First Sign-In And Staff Access

HOP users do not go through the student onboarding form.

When your staff role has been assigned correctly:

- sign in through `/sign-in`,
- the system auto-marks the account as onboarded,
- you are redirected to `/dashboard`.

If you are sent to `/staff-awaiting`, your staff email exists in the system but the correct role has not been assigned yet. In that situation, you cannot continue into the HOP workflow until the role is updated.

Routes involved:

- `/sign-in`
- `/dashboard`
- `/staff-awaiting`

Set Your Academic Scope

Figure 15: HOP Settings.

Open `/settings` and review your:

- programme association,
- academic session.

This scope matters because it influences:

- the default academic session used on HOP dashboard and report filters,
- the programme and faculty context shown in staff settings,
- the target programme options available in faculty-scoped HOP workflows such as articulation creation and import.

If the current academic session is missing, you can create it from the settings page before saving your staff scope.

Route involved:

- `/settings`

HOP Dashboard

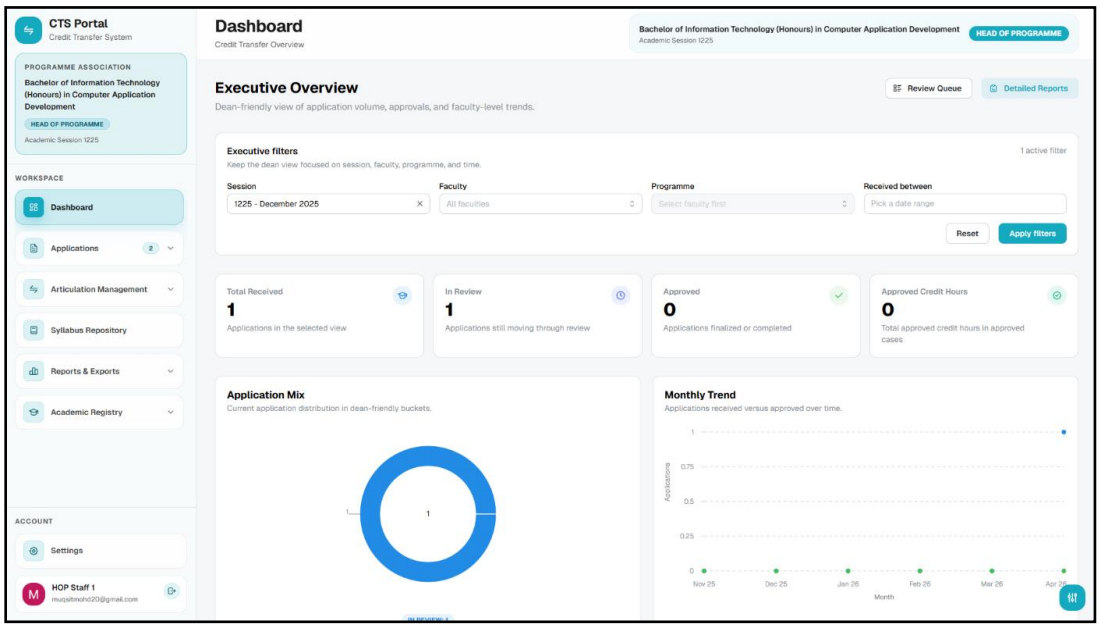


Figure 16: HOP Dashboard.

Use `/dashboard` as your high-level monitoring page.

The dashboard is not the main place to make case decisions. It is the place to understand:

- how many applications are currently in the workflow,
- how many are still in review,
- how many are already finalized or completed,
- which institutions are generating the most volume,
- which queue stages are building operational pressure,
- which applications are ageing the longest.

The HOP dashboard also includes filters, so you can narrow the view by academic session and other reporting dimensions before opening the detailed queue or reports pages.

Review Queue

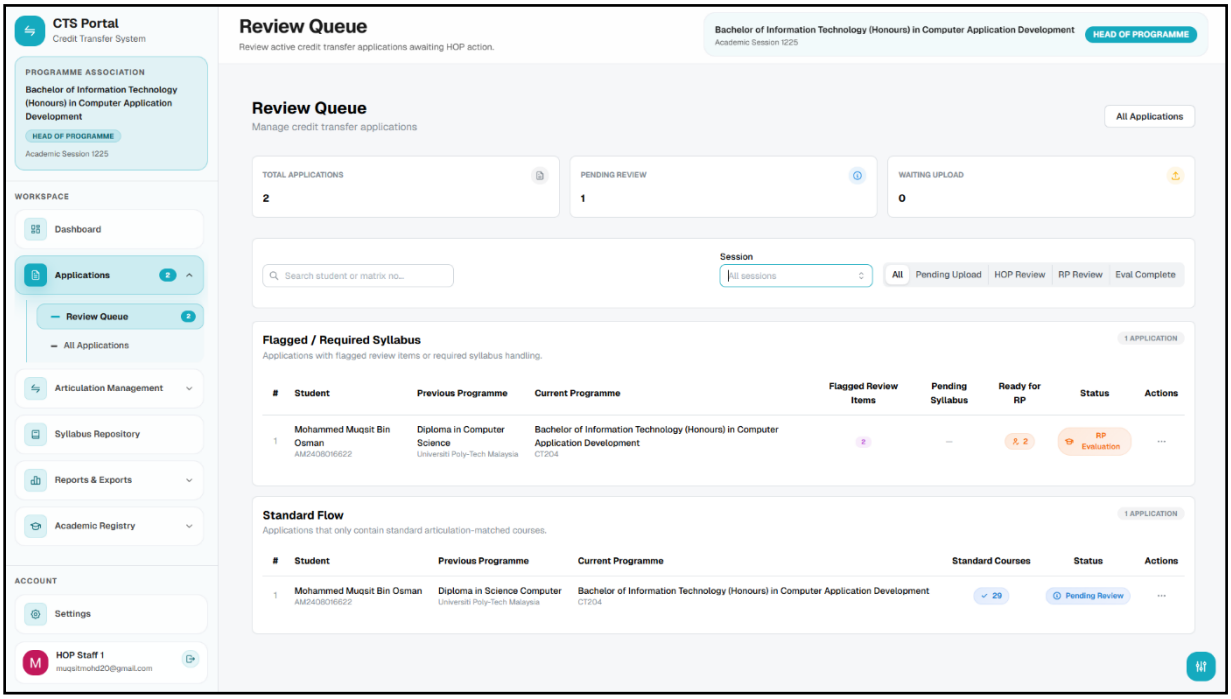


Figure 17: HOP Review Queue

Open `/reviews/hop` to work through applications that currently need HOP attention.

The page includes:

- summary cards for total applications, pending review, and waiting upload,
- search by student name or matrix number,
- session filter,
- quick status filter for `AWAITING_SYLLABUS_UPLOAD`, `PENDING_HOP_REVIEW`, `RP_EVALUATION_PENDING`, and `EVALUATION_COMPLETED`,
- two separate sections so you can distinguish flagged work from standard-flow work quickly.

Flagged / Required Syllabus Section

The **Flagged / Required Syllabus** section shows applications that still contain flagged review work or unresolved syllabus handling.

For each application, you can see:

- student name and matrix number,
- previous institution and programme,
- target UPTM programme,
- number of flagged review items,
- how many flagged units are still missing syllabus evidence,
- how many flagged units are ready for RP assignment,
- the current application status.

Use this section when the case still needs evidence handling, RP assignment, or final academic judgment on flagged items.

Standard Flow Section

The **Standard Flow** section shows applications whose current visible work is mainly standard articulation-matched course review.

For each application, you can see:

- student information,
- source and target programme context,
- number of standard courses,
- current application status.

Use this section when the case does not currently depend on unresolved flagged syllabus work.

Queue Actions

Each queue row provides:

- **View** to open the case,
- **History** to open the application status-history modal.

The status-history modal shows:

- the full status timeline from oldest to newest,
- the actor who performed or triggered the change,
- notes attached to each status-history entry,
- the application summary card on the right for quick context.

The same history modal is reused from `/reviews/hop/all`, so the layout and information are consistent whether you open it from the queue or the full list.

All Applications List

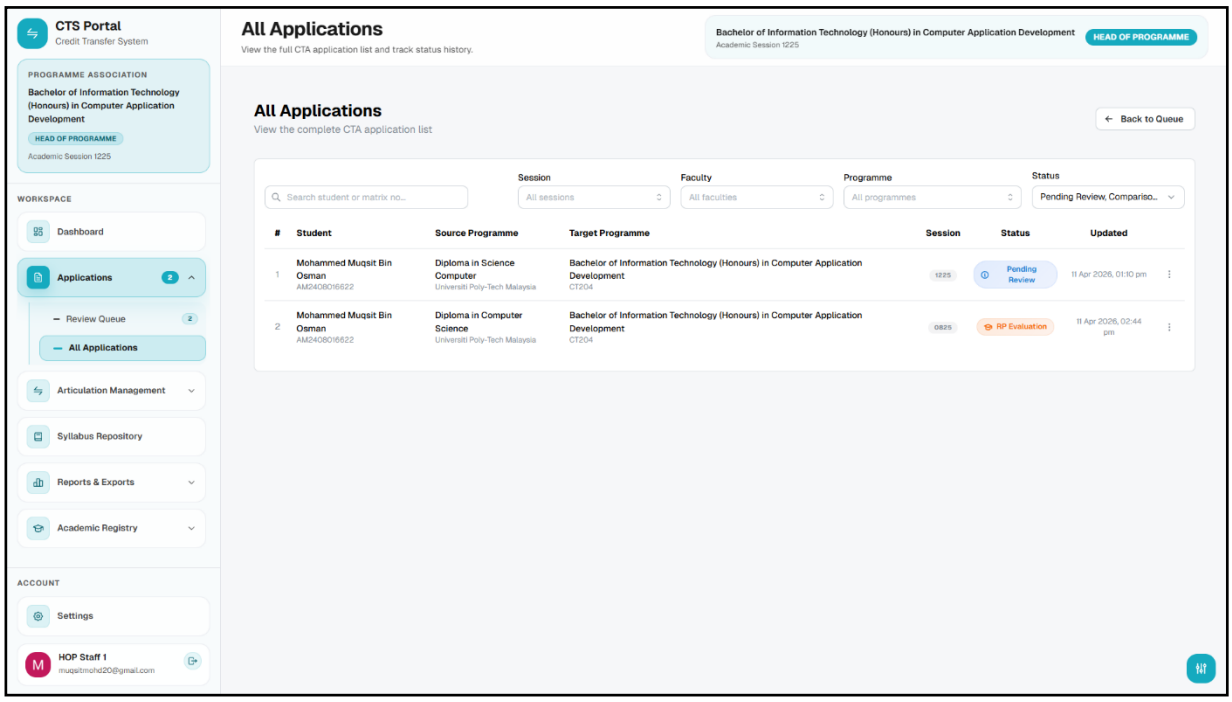


Figure 18: All Applications Page

Open `/reviews/hop/all` when you need a complete list, not only active queue work.

This page supports:

- search by student name or matrix number,
- filter by academic session,
- filter by faculty,
- filter by programme,
- multi-select status filter,
- access to view, export, and history actions.

Use this page when you need to:

- locate historical cases,
- find finalized or completed applications,
- export from outside the main queue,
- check withdrawn or completed items,
- review a broader filtered population by faculty or programme.

Export actions only appear when the application is already:

- `FINALIZED`, or
- `COMPLETED`.

Important modal behavior on this page:

- **History** opens the same status-history modal used on the queue page,
- **Export** opens the HOP export modal,

- the **Status** filter uses a dropdown menu with checkbox selections, so you can keep multiple statuses active at once instead of switching to only one status.

Open An Application For Review

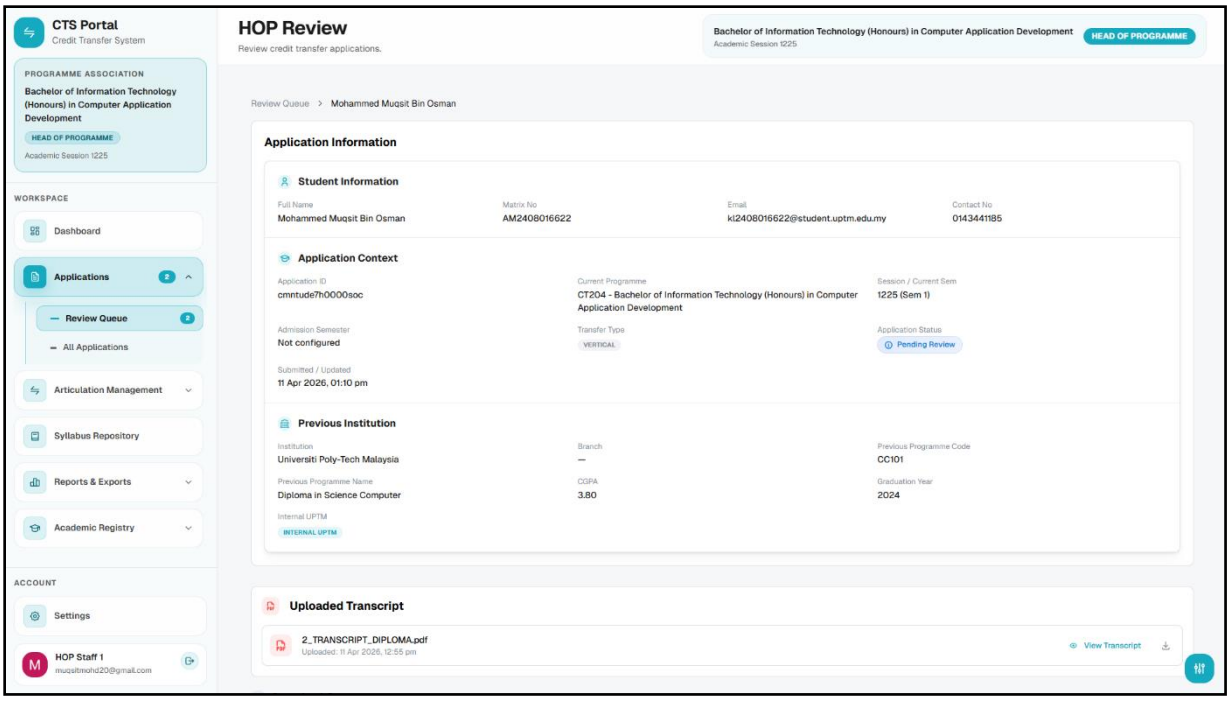


Figure 19: HOP Application Review

The main actionable review page is:

- `/reviews/hop/[applicationId]`

From this page, HOP can work through the case in detail.

The page includes:

- breadcrumbs back to the queue,
- **Application Information**,
- **Uploaded Transcript**,
- **Flagged Review Items**,
- **Standard Courses**,
- finalization controls.

Application Information

The **Application Information** panel summarizes:

- student information,
- source institution and branch,
- source programme,
- target UPTM programme,
- session and application identifiers,

- status-related context.

Use this section first to verify you are reviewing the correct case before changing course-level decisions.

Uploaded Transcript

The **Uploaded Transcript** section lists the transcript file records attached to the application.

For each transcript, HOP can:

- open the transcript in an in-page window,
- open it in a new tab,
- download the file.

Use the transcript before changing decisions if the extracted course rows or source context look inconsistent.

The **View Transcript** button is a menu. The in-page window option is marked experimental and opens a draggable viewer inside the current screen.

Review-Page Windows, Drawers, And Modals

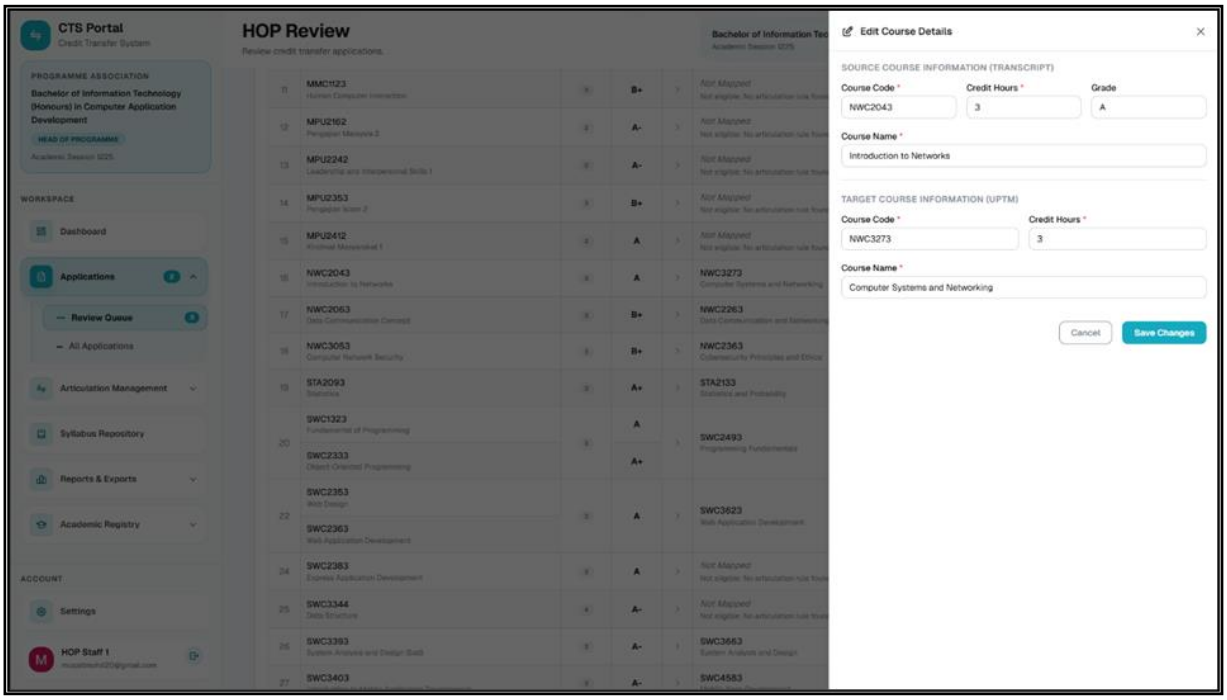


Figure 20: HOP Application Review

The HOP review page uses several overlays:

- **Edit Course Details** opens as a right-side drawer,
- **Assign Resource Person** opens as a large centered modal,
- **Resource Person Information** opens as a small read-only modal,
- **Finalize Review** opens as a confirmation modal with placed-semester input,
- shared confirmation prompts open for bulk actions, RP reassignment, and syllabus-email resend,
- syllabus and transcript previews can open in draggable in-page windows.

These overlays matter because the action is not always executed immediately. Some actions need extra confirmation text, some need required fields, and some are blocked while saving is in progress.

Review Flagged Items

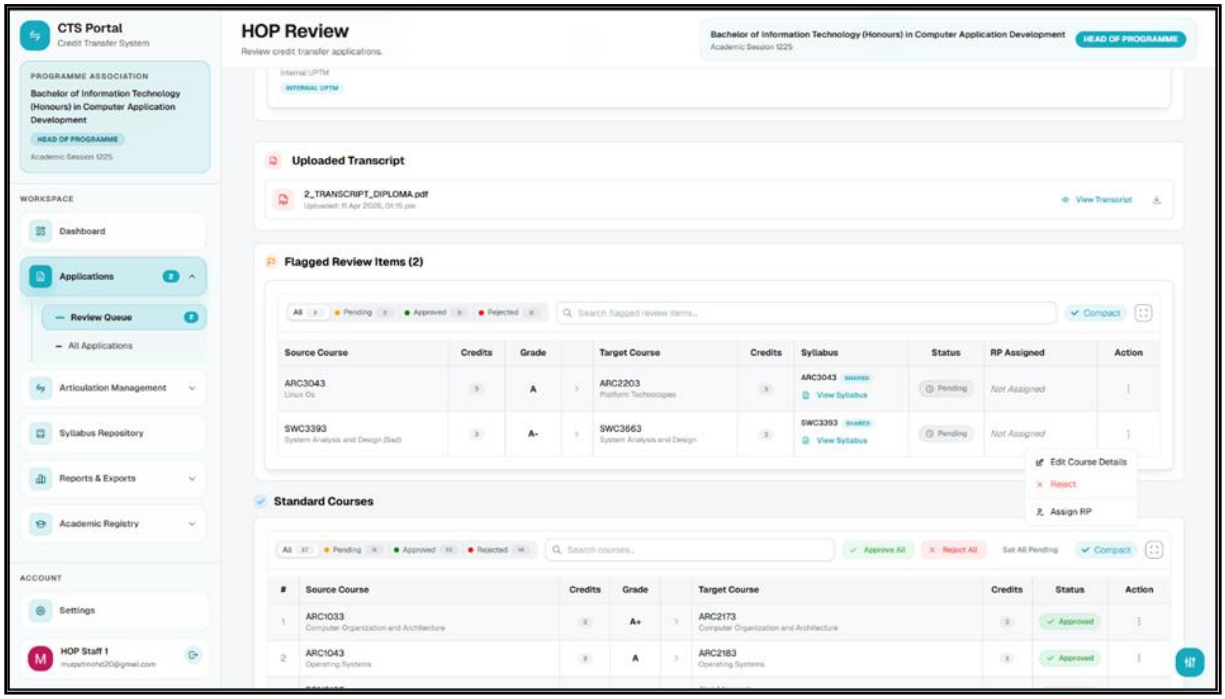


Figure 21: HOP Application Review

Flagged review items are the most important HOP workload because they depend on additional evidence, RP input, or manual academic judgment.

The **Flagged Review Items** table supports:

- search,
- status filter,
- row-level actions,
- syllabus-related follow-up,
- RP assignment.

Typical flagged-unit decisions are:

- `PENDING`,
- `APPROVED`,
- `REJECTED`.

What A Flagged Review Item Usually Means

A flagged review item normally means one of these situations:

- the course matched only a proposed or manual articulation rule,
- the course still needs a syllabus,
- the course belongs to a shared review scope,
- the course needs RP comparison before HOP can finalize the outcome.

One flagged review unit may represent:

- one course,
- multiple composite source courses,
- a shared review group that affects other linked applications or articulation rows.

View Syllabus Evidence

HOP review is based on the effective syllabus, not only a direct upload from the current student.

In practice, the available evidence can come from:

- a direct source syllabus upload on the course,
- a shared representative syllabus linked through the shared syllabus case.

If no real representative syllabus exists, the unit is not ready for RP assignment.

What you will see in the table:

- **Shared** badge when the effective syllabus comes from the shared syllabus case,
- **Direct** badge when the effective syllabus comes from the current student-side upload,
- **Required** or another syllabus-intent badge when usable evidence is still missing.

When a syllabus is available, the **View Syllabus** action opens a menu with:

- **Open in Window** for the in-page draggable viewer,
- **Open in New Tab** when a real stored file is available.

Request Or Resend Syllabus Upload

Use **Request Syllabus Upload** when flagged courses still do not have real syllabus evidence.

What happens when you use it:

- if this is the first request, the application is moved to `AWAITING_SYLLABUS_UPLOAD`,
- the system attempts to send a syllabus-upload email to the student,
- a status-history note is recorded,
- the request can be resent later from the same page if the application is still awaiting upload.

Important behavior:

- if email delivery fails, the request is still recorded in history,
- if no flagged course is actually missing a syllabus, the request action is blocked,
- this action is meant to send the student back to the evidence-upload flow, not restart the whole application.

Confirmation behavior:

- the first request normally runs immediately and shows a loading notification while the system updates the application,
- if the application is already `AWAITING_SYLLABUS_UPLOAD`, pressing the button opens a confirmation modal first,
- the resend modal does not require typed confirmation, but you must still click the resend action,
- the result notification tells you whether the email was sent, saved with warning, or failed completely.

Assign Or Reassign RP

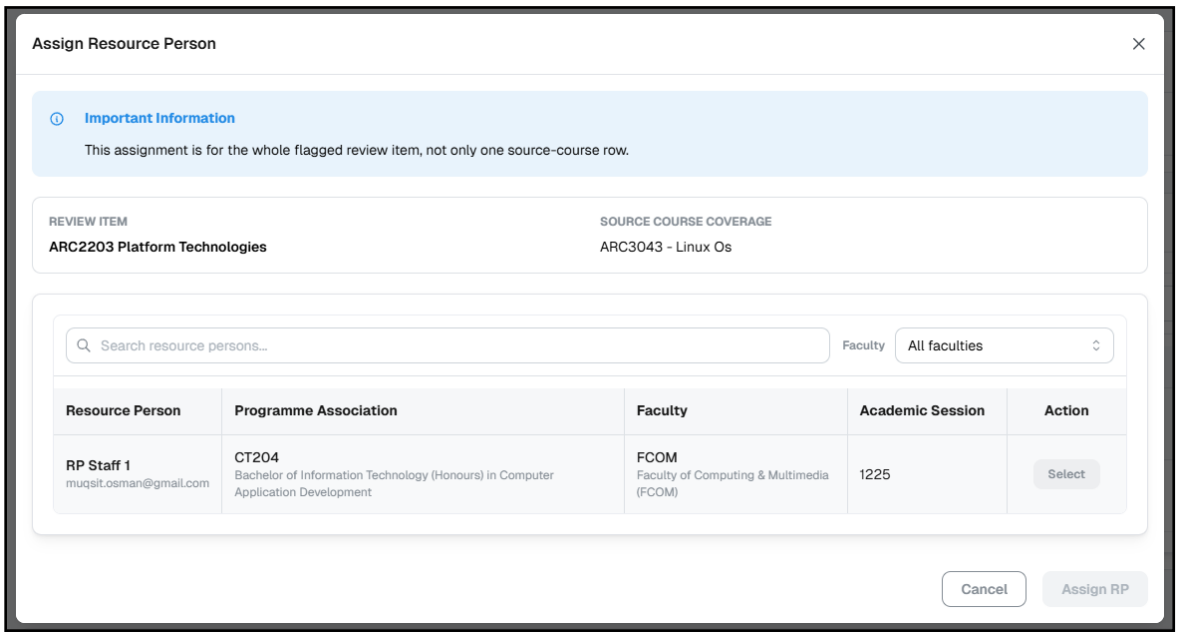


Figure 22: Assign or Reassign RP modal

Use **Assign Resource Person** when a flagged review item has usable representative syllabus evidence and should move into RP evaluation.

The RP assignment modal lets you:

- search RP users,
- filter RP users by faculty,
- inspect RP programme association,
- inspect RP academic-session association,
- select the RP to assign.

Important behavior:

- RP assignment is stored at shared-review-group level,
- one assignment can affect multiple linked CTA courses and articulation rows,
- reassigning RP resets the shared RP decision back to pending,
- if no representative syllabus exists, RP assignment is blocked.

Modal and confirmation behavior:

- the assignment modal starts with an alert explaining that the assignment covers the whole flagged review item, not only one source-course row,
- composite flagged items show the full source-course coverage inside the modal,
- if some member syllabi are still missing, the modal warns you before assignment,
- while assignment is running, the modal cannot be dismissed accidentally,
- **View RP Info** opens a small read-only modal with the assigned RP identity,

- **Reassign RP** opens a confirmation modal first and requires you to type the displayed target course code, or the source course code if no target code is present, before you can continue into the assignment modal.

How Flagged Decisions Change

Flagged review items do not behave exactly like standard rows.

In the current HOP screen:

- HOP can directly reject a flagged review item from the action menu,
- RP-related actions are handled through assignment, reassignment, and downstream RP review,
- approved flagged outcomes can return through the shared-review workflow and then appear in the flagged status column,
- workflow synchronization is recalculated after the relevant review updates.

Review Standard Courses

The **Standard Courses** table is used for non-flagged rows and straightforward review handling.

The table supports:

- search,
- decision filter,
- row grouping for composite or shared source-code patterns,
- row highlighting for user-modified items,
- bulk actions.

Bulk Actions

HOP can use these bulk actions on standard courses:

- **Approve All**
- **Reject All**
- **Set All Pending**

Important behavior:

- courses rejected by CTA eligibility rules are skipped,
- courses missing target-course information are skipped where necessary,
- the UI warns when some rows cannot be included in the bulk action.

Confirmation behavior:

- each bulk action opens a confirmation modal before anything is saved,
- you must type `CONFIRM` exactly before the action button becomes available,
- the modal message tells you how many rows will be updated and how many rows will be skipped,
- the same confirmation modal also shows a copy control for the expected confirmation text.

Course Editing

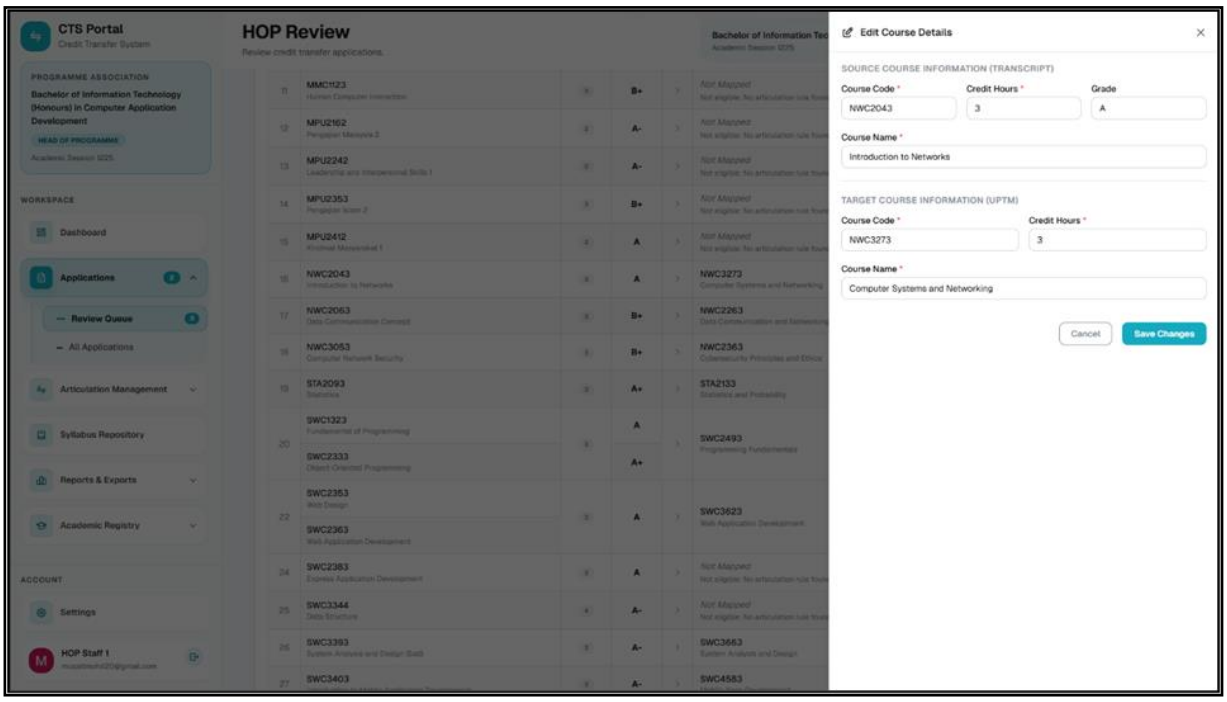


Figure 23: Edit Course Details Drawer

Use **Edit Course Details** when the extracted or mapped row needs correction.

HOP can update:

- source course code,
- source course name,
- source credit hours,
- source grade,
- target course code,
- target course name,
- target credit hours.

Important behavior:

- editing marks the course as user-modified,
- eligibility-relevant changes trigger a new eligibility evaluation pass,
- application workflow status is re-synced after reevaluation,
- if the course belongs to a composite block, target-course details are synchronized across the other rows in the same composite block.

Drawer behavior:

- the editor opens from the right side of the screen,
- composite rows show one source-course form section per member row and one shared target-course section,
- the save button stays disabled until all required fields are filled and all credit-hour values are greater than `0`,
- while saving is in progress, escape, outside click, and the close button are blocked,

- after save, the UI shows a running course-check notification and refreshes the case details.

Limits On HOP Decision Editing

HOP cannot freely override every rejected course from the review screen.

If a course was already rejected by CTA eligibility rules, the review action is blocked with the message that these courses cannot be changed from HOP review.

In that situation, the correct next step is usually to:

- inspect the transcript and mapping,
- fix course metadata if it is wrong,
- check the articulation context,
- then let the system reevaluate the row.

Finalize The Review

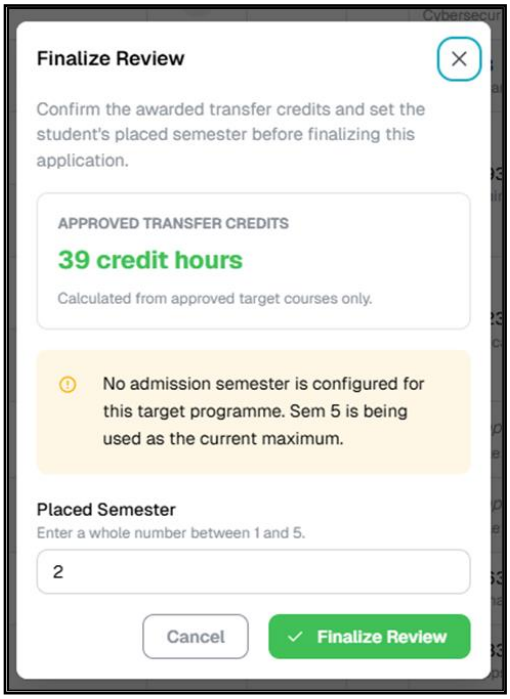


Figure 24: Finalize Review Modal

Use **Finalize Review** only after every reviewable item is resolved.

The button stays disabled when:

- any standard course still has a pending decision,
- any flagged review item still has a pending decision.

Finalization Requirements

Before finalizing, confirm:

- every standard course is approved or rejected,
- every flagged review item is approved or rejected,

- the placed semester is correct,
- the approved credit total looks correct for the target programme.

Modal behavior:

- pressing **Finalize Review** opens a modal instead of finalizing immediately,
- the modal shows the current approved transfer-credit total,
- the placed-semester field is prefilled with the suggested value based on the student's current semester and the allowed maximum,
- the modal shows an informational alert about the maximum semester rule,
- validation errors stay inline inside the modal until corrected.

Placed Semester Rules

During finalization, HOP must enter **Placed Semester**.

Validation rules:

- it must be a whole number greater than `0`,
- it cannot exceed the target programme admission semester,
- if the programme admission semester is not configured, the current fallback maximum is Sem `5`.

What Finalization Stores

When finalization succeeds:

- application status becomes `FINALIZED`,
- the HOP reviewer ID is stored,
- total approved credit hours are stored,
- placed semester is stored,
- status history is updated,
- generic workflow synchronization no longer moves the application automatically.

`FINALIZED` is the academic lock point for the case.

Export Slip And CSV Re-Export

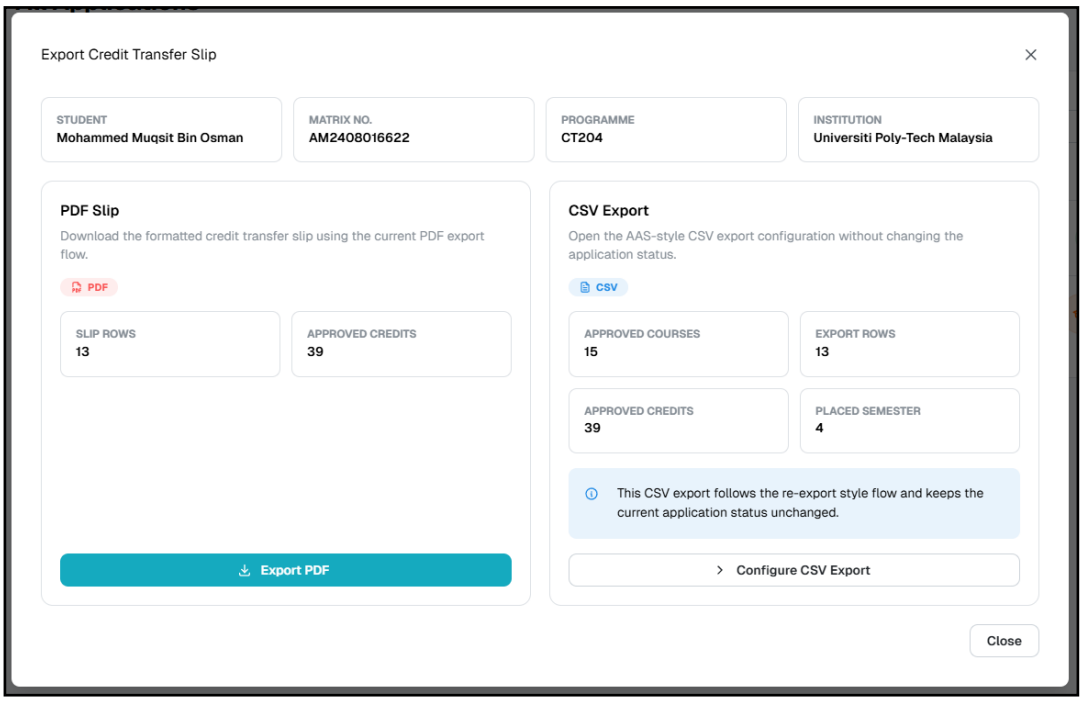


Figure 25: Export CTA Slip

HOP can export once an application is:

- `FINALIZED`, or
- `COMPLETED`.

You will encounter export actions from:

- `/reviews/hop/all`,
- HOP read-only application detail views for finalized or completed applications.

PDF Slip Export

The PDF option downloads the formatted credit transfer slip.

The export preview shows:

- student name,
- matrix number,
- programme,
- source institution,
- number of approved courses,
- approved credit hours.

Use PDF when you need the formal slip document in the current export format.

In the HOP export modal, PDF appears as one export card in the opening format-selection view. If the application is not exportable, the card shows the rejection reason instead of enabling the export button.

CSV Re-Export

HOP also has a CSV re-export flow for finalized or completed cases.

This flow:

- opens an export-configuration modal,
- keeps the application status unchanged,
- generates an AAS-compatible CSV structure for the selected application.

The CSV configuration requires:

- `staff_id`,
- `kolej` mode using either past institution name or short code,
- `update_date` format,
- confirmation text that exactly matches your HOP identity.

The export is blocked when:

- the application has no exportable approved rows,
- the confirmation text does not match your identity,
- the staff ID is missing.

What the export modal looks like:

- the first view shows export-format cards for **PDF Slip** and **CSV Export**,
- choosing **Configure CSV Export** switches the same modal into a second configuration view,
- the CSV view includes export overrides, the included-application summary, and the confirmation panel,
- the confirmation panel requires the exact HOP identity string shown on screen,
- a **Back to export options** button returns to the first modal view without closing the modal.

Articulation Management

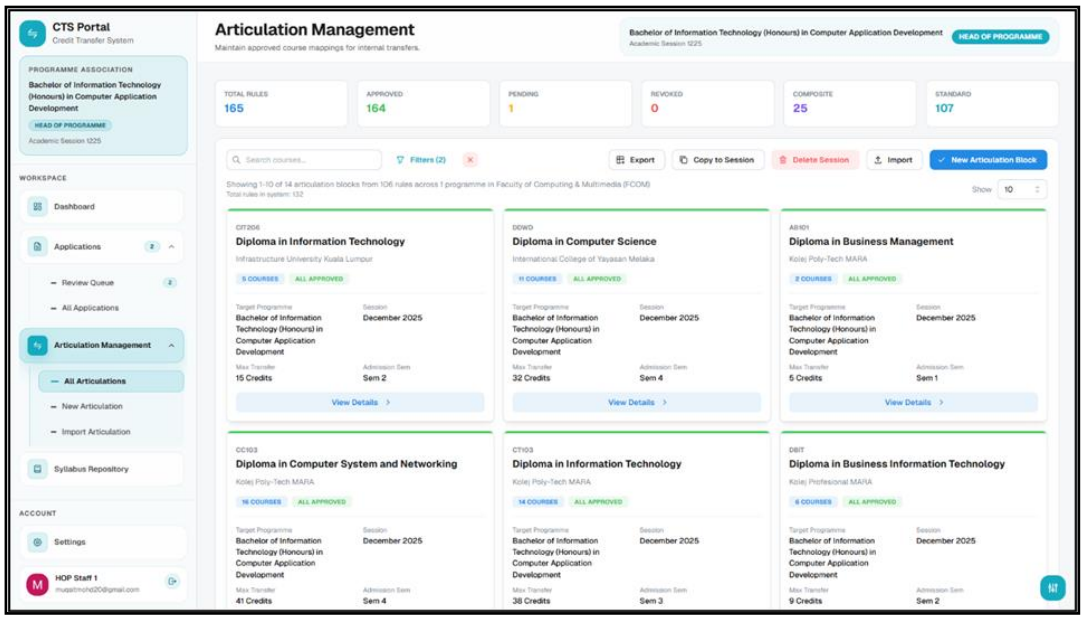


Figure 25: Articulation Management Page

Use the articulation module when HOP needs to maintain the mapping library that powers student matching.

Main routes:

- `/articulation`
- `/articulation/new`
- `/articulation/details/[id]/new`
- `/articulation/details/[id]`
- `/articulation/edit/[id]`
- `/articulation/import`
- `/articulation/report/[id]`

What HOP Can Do In Articulation

HOP can:

- create a new articulation block,
- add the first articulation row in the detail-new flow,
- add more mappings to an existing block,
- edit block-level source context,
- edit or delete articulation rows,
- approve or revoke articulation rules,
- assign RP for articulation review,
- upload supporting syllabi,
- copy articulation blocks into another academic session,
- import reviewed mappings from Excel,
- export articulation data for offline sharing or audit.

Articulation Statuses

The main articulation statuses are:

- `PROPOSED`
- `APPROVED`
- `REVOKED`

These statuses matter because:

- approved rules can drive automatic CTA matching,
- proposed rules can still appear as manual flagged matches,
- revoked rules should no longer be treated as active approved mappings.

Faculty-Scoped Behaviour

When HOP has a faculty-linked programme scope, target programme choices in manual articulation create and import flows are restricted to that faculty context.

Target Syllabus Repository

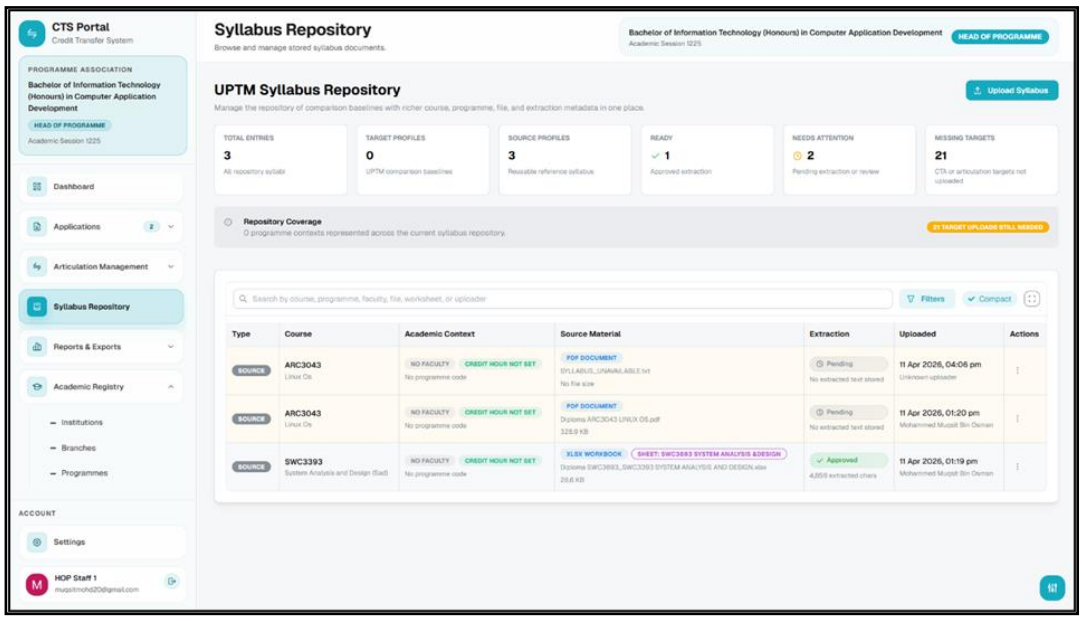


Figure 26: Sylalbus Repository Page

Use `/syllabus` to maintain target UPTM syllabi used in comparison workflows.

This page lets HOP:

- upload target syllabus files,
- inspect extraction state,
- retry extraction,
- edit syllabus metadata,
- delete repository entries,
- create manual syllabus entries when needed.

Upload Rules

Accepted files:

- PDF
- XLSX

Important behavior:

- PDF files go through text extraction,
- XLSX files support worksheet selection,
- the upload flow can prompt for worksheet choice when the workbook has multiple sheets,
- course, programme, faculty, and credit-hour metadata can be stored alongside the file.

Repository Use

The syllabus repository is not just storage.

It is used as the comparison baseline for:

- RP syllabus evaluation,
- shared review workflows,
- articulation-related evidence review.

If a target syllabus is missing, RP comparison work may stall even when the student-side source syllabus is already available.

Reports And Analytics

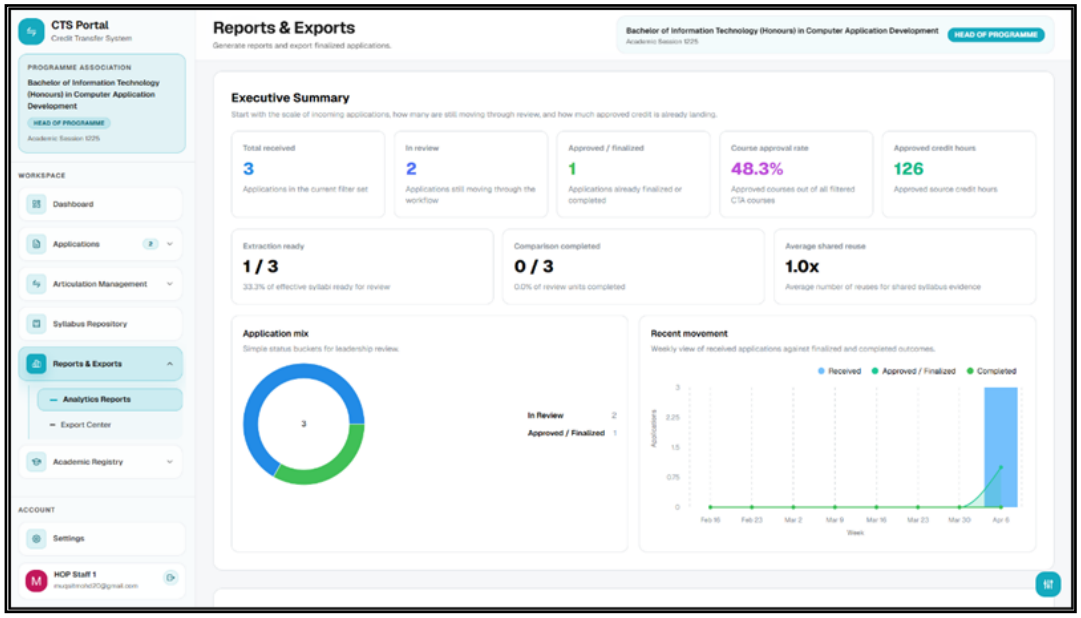


Figure 27: Reports and Analytics Page

Use `/reports` for leadership-style monitoring.

This page includes:

- executive summary metrics,
- application mix,
- recent movement trends,
- institution insights,
- queue and SLA visuals,
- backlog ageing,
- oldest active applications.

Use reports when you need:

- faculty or programme-level oversight,
- institutional trends,
- review-pipeline pressure analysis,
- a higher-level leadership summary instead of row-by-row processing.

Export Center

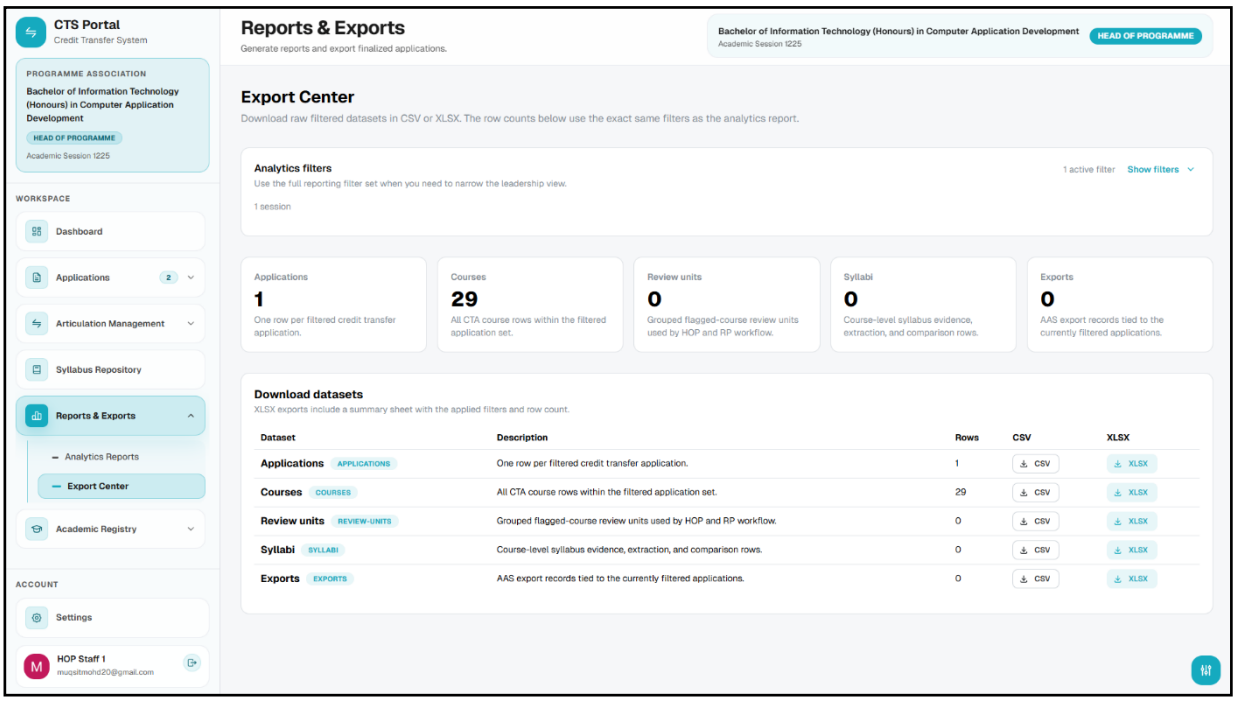


Figure 28: Report Export Center

Use `/reports/exports` when you need raw filtered datasets instead of dashboard charts.

The Export Center supports CSV and XLSX downloads for datasets such as:

- applications,
- courses,
- review units,
- syllabi,
- exports.

These downloads use the same filter logic as the analytics report, so the dataset output matches the reporting scope you selected.

Academic Registry

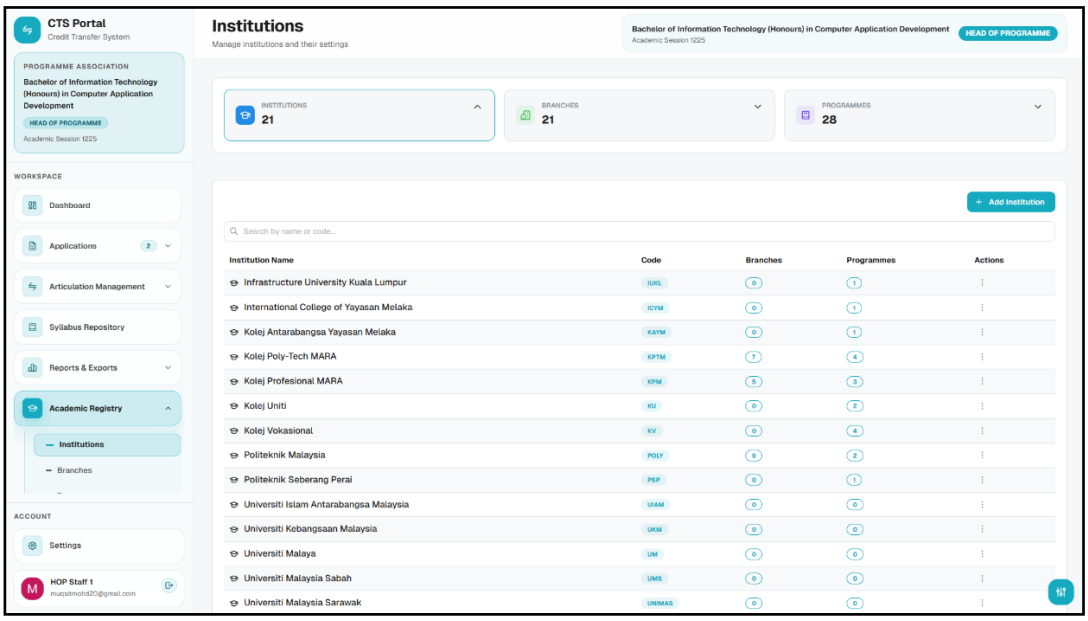


Figure 29: Academic Registry Page

Use `/academic-registry` to maintain master data that drives onboarding, articulation, application source context, and reporting.

The registry covers:

- institutions,
- branches,
- programmes.

HOP can:

- add new records,
- edit existing records,
- deactivate or delete where allowed.

Important behavior:

- deletion is guarded when dependent records exist,
- branch and programme metadata feeds into articulation and application context,
- programme metadata also matters for finalization rules such as admission-semester validation.

HOP Status Guide

These are the application statuses HOP will usually work with:

Status	Meaning for HOP
`PENDING_HOP_REVIEW`	The case is ready for HOP review and decision work.
`AWAITING_SYLLABUS_UPLOAD`	Student-side syllabus evidence is still missing or was requested by HOP.

SYLLABUS_COMPARISON_IN_PROGRESS	A comparison run is currently executing for shared review work.
RP_EVALUATION_PENDING	Evidence is ready and the case is waiting for RP decision work.
EVALUATION_COMPLETED	RP work is done and the case is back with HOP for finalization.
FINALIZED	HOP has locked the academic decision.
COMPLETED	AAS has exported the finalized case and closed it operationally.
WITHDRAWN	The student withdrew the application.

Troubleshooting

I Cannot Assign RP

Check whether:

- the review item already has a representative syllabus,
- the application is still in a reviewable status,
- the selected user really has the RP role.

If the syllabus is still missing, request student upload first or confirm whether a shared representative syllabus already exists.

Finalize Review Is Disabled

Check whether:

- any standard course is still PENDING,
- any flagged review item is still PENDING,
- the target programme admission semester is configured correctly,
- the placed semester you want to enter is within the allowed maximum.

Request Syllabus Upload Did Not Send An Email

The request can still be recorded even if email delivery fails.

Open the application history and check the notes:

- the application may already be in AWAITING_SYLLABUS_UPLOAD,
- the note may show that email delivery failed,
- you can resend the request later from the review page.

Export Is Not Available

Check whether:

- the application is already FINALIZED or COMPLETED,

- there are approved courses with target-code and credit-hour data,
- the required CSV confirmation text matches your HOP identity exactly,
- the `staff_id` field has been filled in for CSV export.

I Cannot Find A Case In The Queue

Check:

- the search text,
- the selected session,
- the queue status filter,
- whether the case has already moved to `FINALIZED`, `COMPLETED`, or `WITHDRAWN`.

If necessary, open `/reviews/hop/all` and widen the status filters to locate the case.

RP Guide

This guide shows Resource Person (RP) users how to review assigned CTA and articulation work, prepare syllabus evidence, run AI comparison, record academic decisions, and notify HOP when RP work is complete.

What You Can Do In UPTM CTS

As RP, you can:

- review only the CTA and articulation items assigned to you,
- monitor your open workload from `/dashboard`,
- open the RP queue and completed-evaluation history,
- inspect grouped review items that may cover more than one source course,
- prepare source syllabus extraction for comparison,
- check whether the target UPTM syllabus is available,
- review source and target syllabus evidence side by side,
- run staged AI syllabus comparison,
- submit approve or reject decisions for your assigned review group,
- reopen the HOP-notification flow after submission when the application has already reached the next stage,
- review assigned articulation evidence in the RP articulation workspace.

Before You Start

Before working in the RP workflow, make sure:

- your account is assigned the `RP` role,
- your staff scope is saved in `/settings`,
- HOP has already assigned the relevant CTA or articulation work to you,
- a usable source syllabus and target syllabus exist for the review item,
- you understand that RP decisions can apply to a shared review group, not only one student row.

First Sign-In And Staff Access

RP users do not use the student onboarding form.

When your staff role has been assigned correctly:

- sign in through `/sign-in`,
- the system auto-marks the account as onboarded,
- you are redirected to `/dashboard`.

If you are sent to `/staff-awaiting`, your staff email exists in the system but your stored role is still not ready for RP access. You cannot continue into the RP workflow until the role is corrected.

Routes involved:

- `/sign-in`
- `/dashboard`

- `/staff-awaiting`

Set Your Academic Scope

Open `/settings` and review your:

- programme association,
- academic session.

This scope is used by staff review and reporting flows, including RP dashboard filtering and current working context.

Route involved:

- `/settings`

RP Dashboard

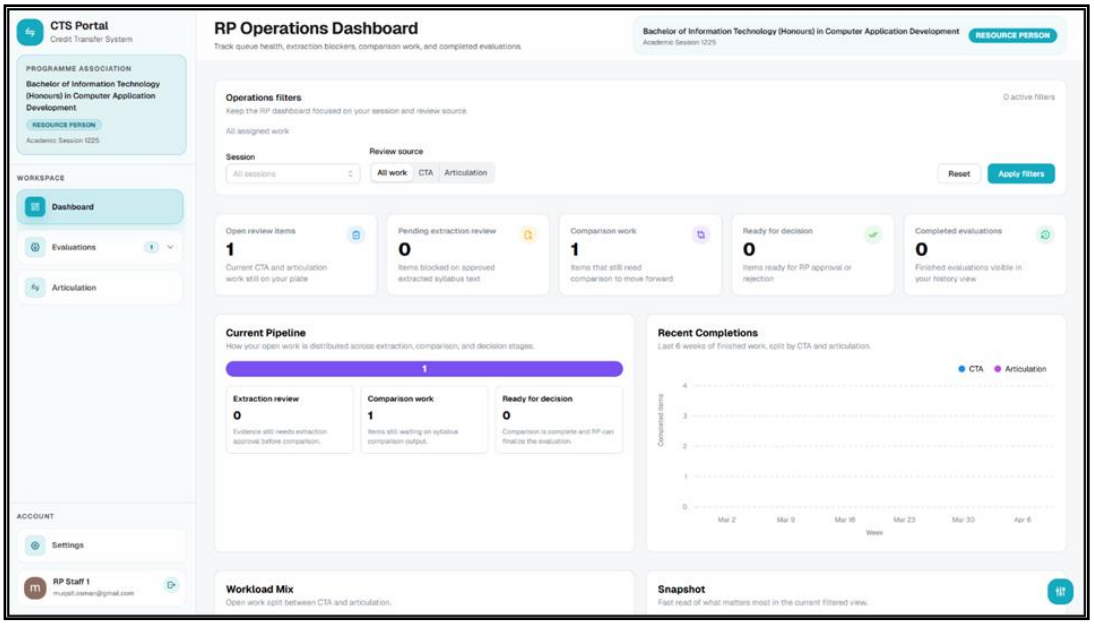


Figure 30: RP Dashboard

Use `/dashboard` as your operations overview.

The RP dashboard includes:

- an **Operations filters** toolbar with session and review-source filters,
- KPI cards for open items, pending extraction, comparison work, ready-for-decision items, and completed evaluations,
- a **Current Pipeline** section,
- **Recent Completions** trend charts,
- **Workload Mix** for CTA and articulation work,
- a **Snapshot** section,
- **Queue Preview**,
- **Recent Completed Evaluations**.

Review-source filters can show:

- all assigned work,
- CTA only,
- articulation only.

Use the dashboard to decide where to start. Use the RP queue and RP review pages to actually perform the evaluation.

RP Queue

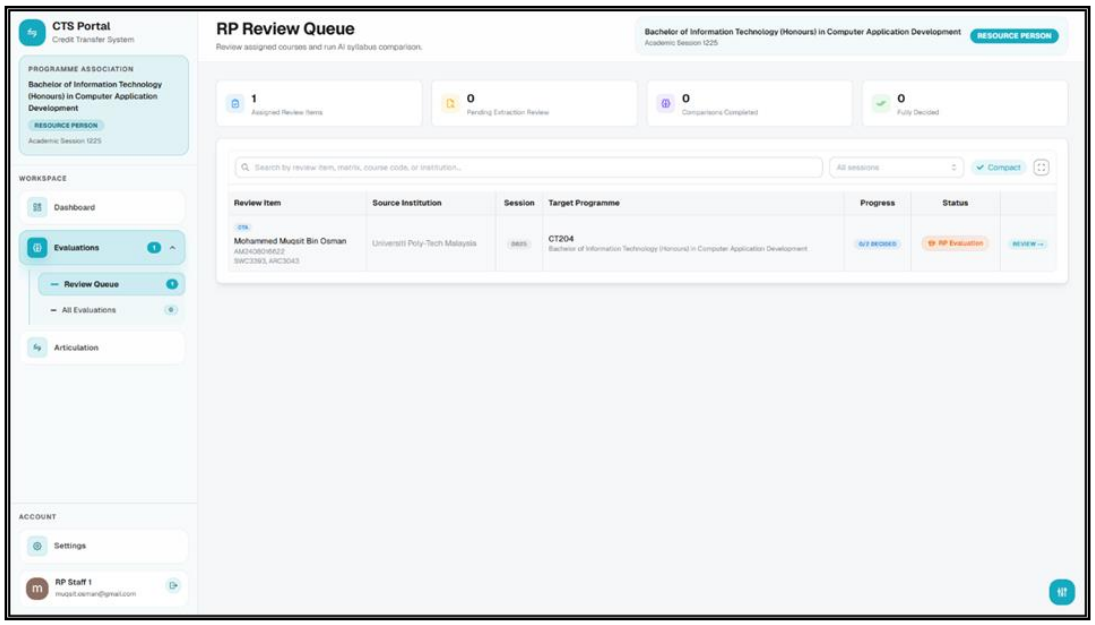


Figure 31: RP Evaluation Queue

Open `/reviews/rp` to work on open items that still need RP action.

The queue includes:

- search,
- session filter,
- metric cards for assigned review items, pending extraction review, comparisons completed, and fully decided items,
- both CTA and articulation work in one list.

Each queue row shows:

- review source badge: `CTA` or `ARTICULATION`,
- review item title and subtitle,
- source institution,
- session,
- target programme,
- progress in `decided / total review units`,
- current status,

- **Review** link.

Important behavior:

- the queue only shows items that still have unresolved RP work,
- completed items move out of this queue and into `/reviews/rp/all`,
- CTA and articulation items share the same queue surface, so always check the review-source badge before opening a case.

Completed Evaluations

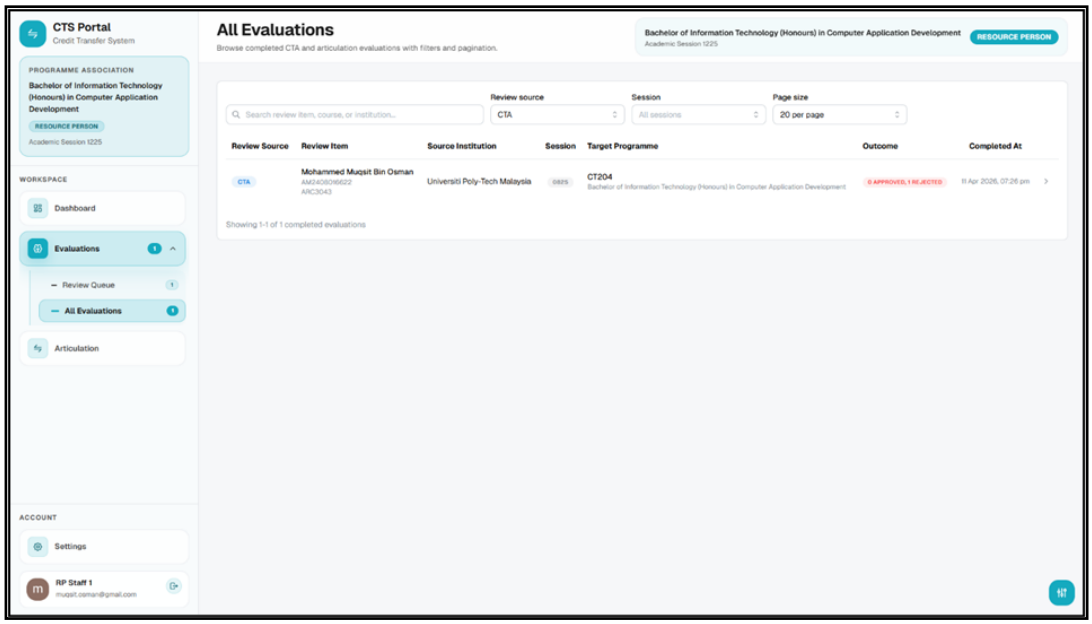


Figure 32: All Evaluations Page

Open `/reviews/rp/all` to review completed RP history.

This page supports:

- search by review item, course, or institution,
- review-source filter using `CTA`, `ARTICULATION`, or `ALL`,
- session filter,
- page-size selection,
- pagination.

Each row shows:

- review source,
- review item,
- source institution,
- session,
- target programme,
- outcome summary,
- completed timestamp.

Important behavior:

- CTA rows open `/reviews/rp/[applicationId]?from=all`,
- articulation rows open `/articulation/view/[id]?from=all`,
- rows are clickable, so the whole row acts like a quick-open action,
- this page is history-oriented, not an open-work queue.

How RP Assignments Work

RP work is grouped by shared academic review scope, not strictly by one visible row.

In practice, one RP assignment can cover:

- one flagged CTA course,
- a composite CTA group with multiple source courses,
- shared CTA rows reused across applications,
- articulation rows linked into the same shared review group.

This matters because:

- one comparison run can feed more than one record,
- one RP decision can propagate to linked CTA courses and articulation rows,
- the visible student application may show only one slice of a wider shared review group.

Open A CTA Review

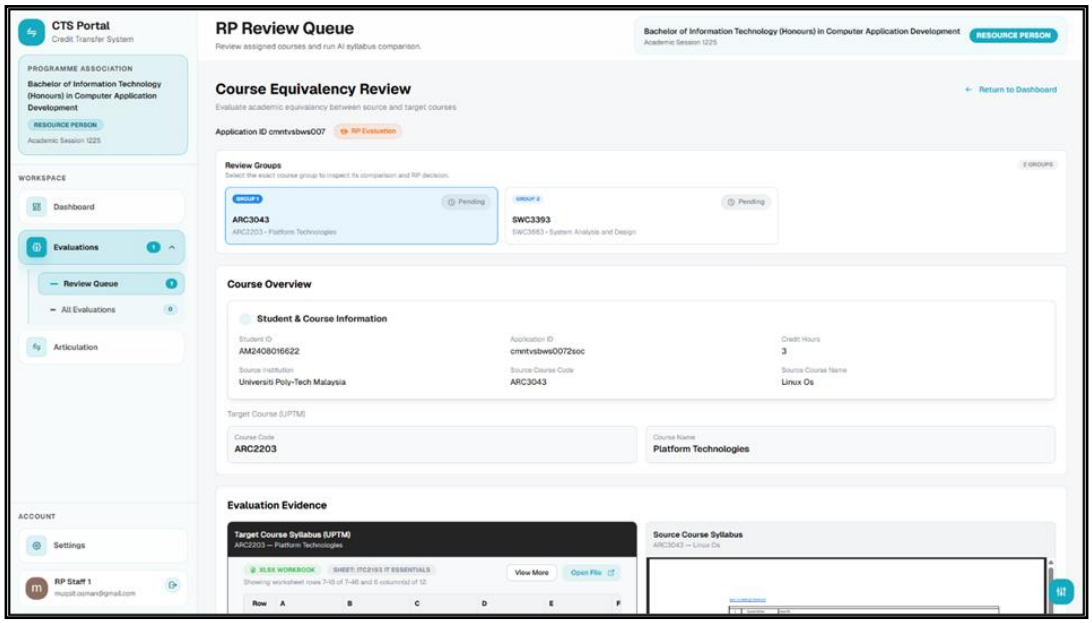


Figure 33: RP Review

The main CTA review route is:

- `/reviews/rp/[applicationId]`

From the queue, this page opens the active pending group. From completed history, it opens in read-only history mode and can also restore a previously completed group.

Return Navigation

The page supports two return paths:

- from the queue, it routes back to `/reviews/rp`,
- from completed history, it routes back to `/reviews/rp/all`.

Important current behavior:

- on CTA review pages opened from the queue, the return button label may read `Return to Dashboard`,
- the button still routes back to `/reviews/rp`, not the dashboard.

Review Groups

If the application contains more than one RP-assigned group, the page shows **Review Groups** at the top.

Each group card shows:

- group number,
- member count when the group is composite,
- source course codes,
- target course,
- current RP decision status.

Use these group cards to switch between assigned review groups without leaving the page.

Course Overview

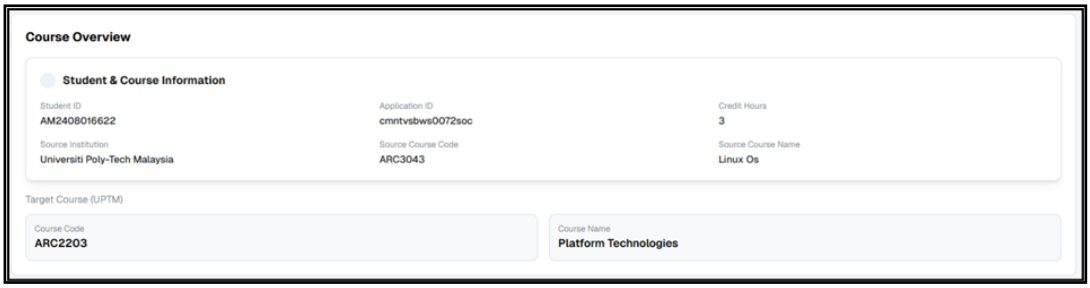


Figure 34: Course Overview

The **Course Overview** panel summarizes:

- student ID,
- application ID,
- credit hours,
- source institution,
- source course code,
- source course name,
- target course code,
- target course name.

Use this panel to confirm you are reviewing the correct source-target equivalency before you start comparing evidence.

Source Syllabus Preparation

Before comparison can run, source syllabus extraction must be ready.

When the page detects source syllabi that still need preparation, you will see an alert:

- **Preparing Source Syllabus** while the system is working,
- **Source Syllabus Preparation Required** when preparation is still needed,
- **Source Syllabus Preparation Failed** when extraction could not be prepared cleanly.

Important behavior:

- the page automatically attempts source-syllabus preparation on load when needed,
- the **Retry Preparation** button lets you rerun the process manually,
- preparation runs across the whole visible review group, not only one row,
- if some syllabi still fail, the page keeps the error summary visible.

Evaluation Evidence

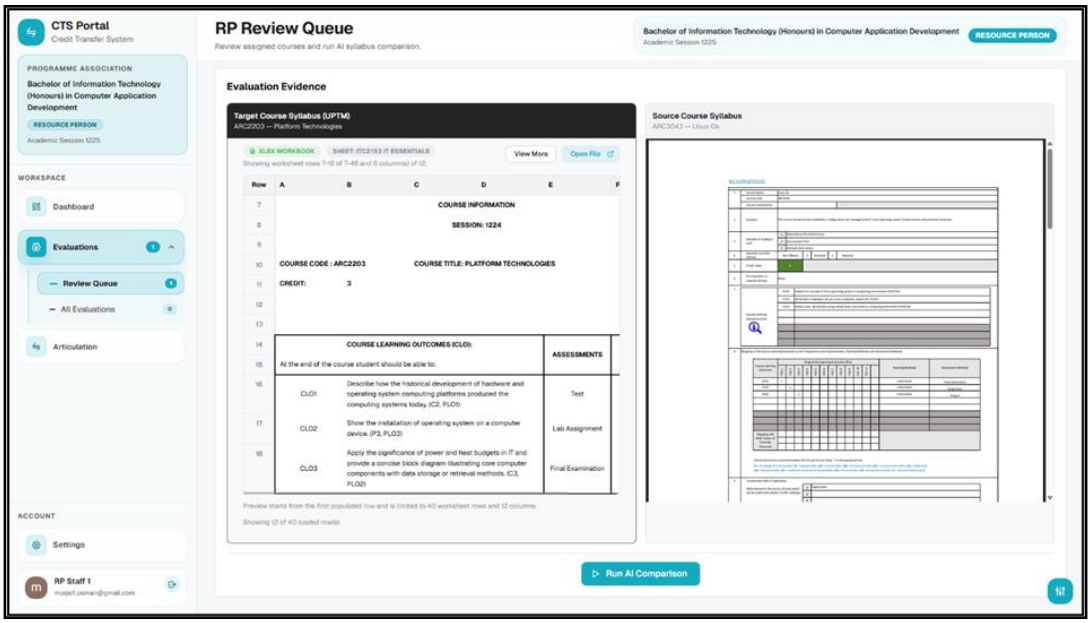


Figure 35: Evaluation Evidence

Once at least one approved source syllabus is ready, the **Evaluation Evidence** section becomes available. This section is the main side-by-side evidence workspace.

Check Target Syllabus Availability

If target evidence has not been resolved yet, you will see **Check Target Syllabus Availability**.

What happens next:

- if the target syllabus exists, the page loads the target evidence,
- if it does not exist, the page shows a warning that the target syllabus has not been uploaded,

- the warning includes a button to open `/syllabus` in a new tab so HOP can upload it.

Important behavior:

- when source extraction is already approved, the page usually auto-checks the target syllabus for you,
- target comparison remains blocked until the target syllabus exists and its extraction is approved.

Evidence Preview Types

The source and target panes support two file behaviors:

- PDF files render inline inside the page,
- XLSX files render as workbook previews inside the page.

For XLSX previews, the page shows:

- worksheet name,
- visible row range,
- visible column count,
- **View More** or **Show Less** when the preview is long,
- **Open File** to open the original workbook in a new tab.

Multiple Source Courses

When the review group contains more than one source course:

- the source side becomes a tabbed area,
- each tab represents one source course,
- you can switch tabs without losing the rest of the review page state.

Run AI Comparison

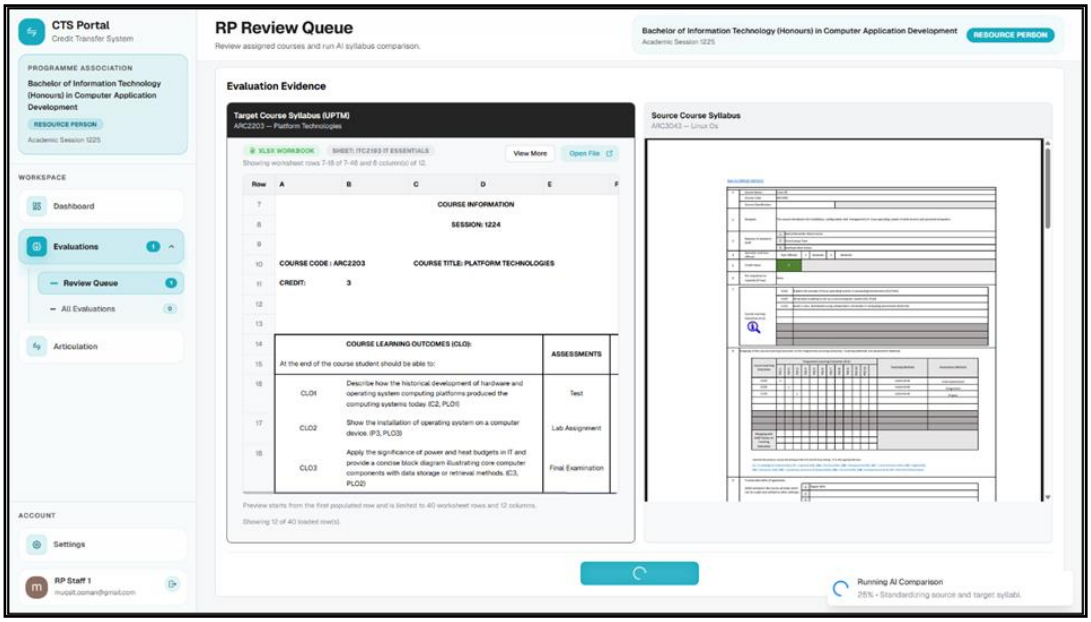


Figure 36: Run AI Comparison

Use **Run AI Comparison** when both sides of the evidence are ready.

The button appears only when:

- source syllabus evidence is available,
- target syllabus exists,
- a comparison result has not already been loaded in the current view.

The button becomes unavailable when:

- the comparison is already processing,
- source extraction is still incomplete,
- the target syllabus is missing or not ready.

What happens when you run it:

- the UI shows a staged progress notification,
- linked applications move to `SYLLABUS_COMPARISON_IN_PROGRESS`,
- the shared comparison pipeline runs,
- successful completion moves linked applications back to `RP_EVALUATION_PENDING`,
- the finished comparison result is shown in the comparison-analysis section.

The staged comparison pipeline includes logical stages such as:

- `PREPROCESS_SOURCE`
- `PREPROCESS_TARGET`
- `STANDARDIZE_SOURCE`
- `STANDARDIZE_TARGET`
- `COMBINE_SOURCES`
- `MAPPING`
- `VERIFICATION`
- `SCORING`
- `SUMMARY`

Comparison Analysis

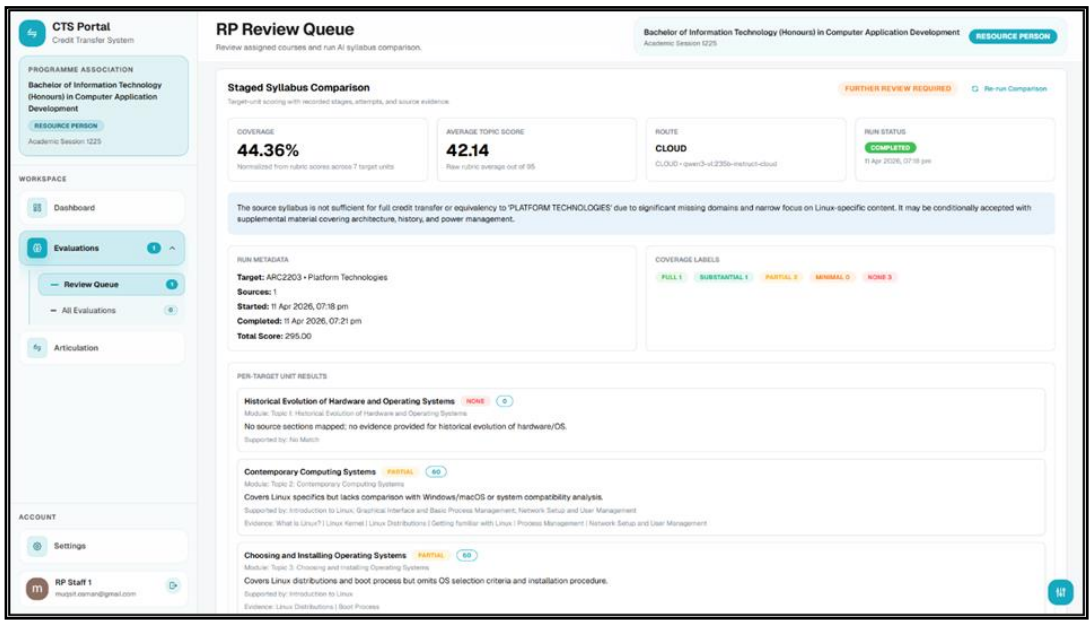


Figure 37: Comparison Analysis

When a comparison result exists, the page shows the staged comparison analysis view.

This view is not just a score panel. It is the main comparison readout used before the RP decision is submitted.

Important behavior:

- the rerun action uses the same comparison flow again,
- the rerun button stays tied to the same assigned review group,
- comparison telemetry is preserved in the shared review records.

Submit The Academic Decision

The **Academic Decision** section appears only after comparison results exist.

RP can choose:

- **Approve Equivalency**
- **Reject Equivalency**

The page also provides:

- an academic-justification textarea,
- a submit button,
- read-only display after a decision has already been submitted.

Important behavior:

- there is no extra typed-confirmation modal before submission,
- RP submits directly from the page,
- the decision is applied at shared-review-group level,
- one submission can update multiple linked CTA rows and articulation rows,

• after submission, the decision panel becomes read-only.

Academic Justification

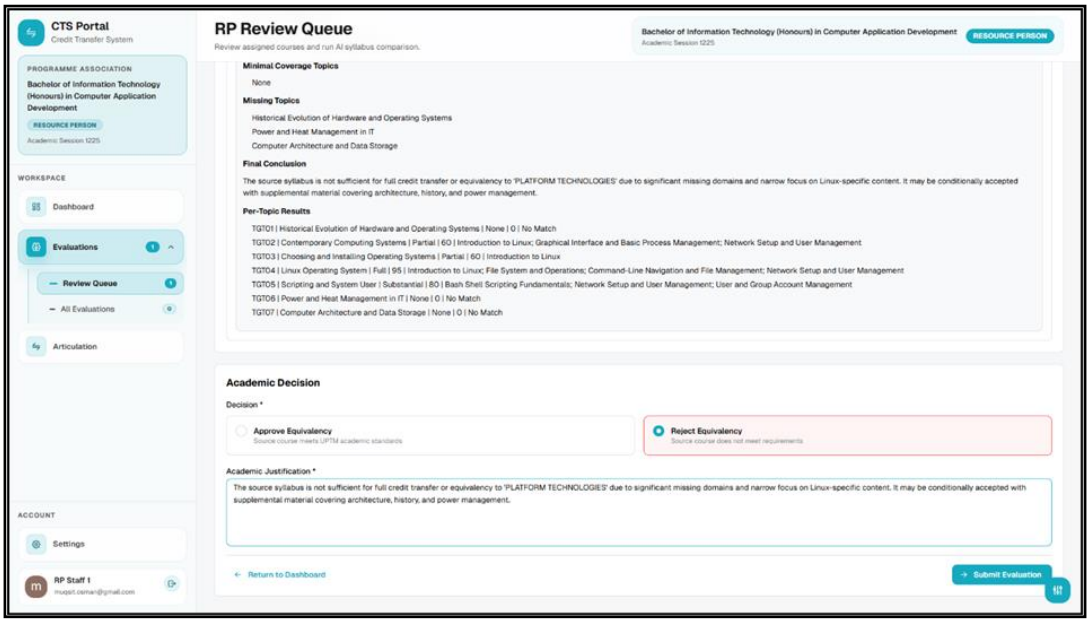


Figure 38: Acadmic Decision

The UI labels academic justification as required and it should be completed as part of the review record. Best practice:

- explain the equivalency reasoning clearly,
- refer to learning outcomes, scope coverage, and missing content where relevant,
- make the justification strong enough that HOP can understand the decision without redoing the entire RP analysis.

Submitted Decision State

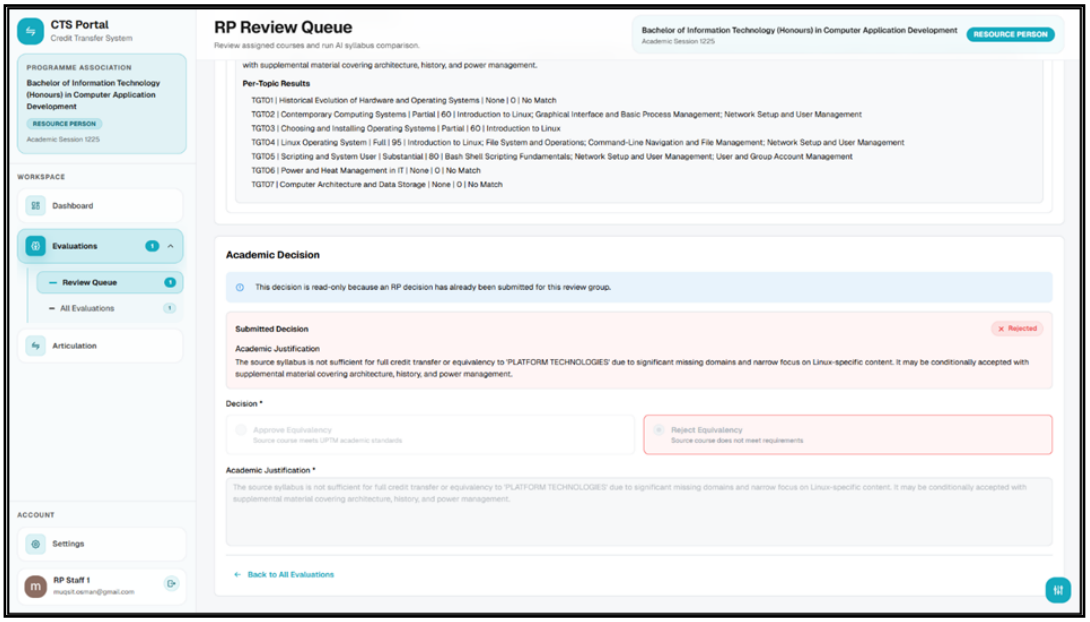


Figure 39: Submitted Decision

After submission, the review page shows:

- the submitted decision badge,
- the recorded academic justification,
- a read-only alert explaining that the RP decision is locked for this review group.

Notify HOP

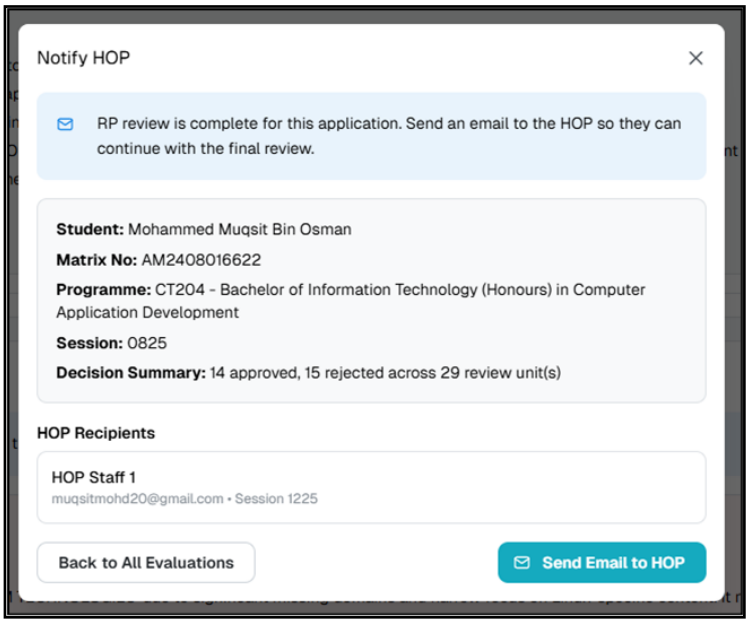


Figure 40: Notify HOP Modal

When RP submission causes the application to reach `EVALUATION_COMPLETED`, the page opens the **Notify HOP** modal automatically.

This modal shows:

- student name,
- matrix number,
- programme,
- session,
- approved and rejected counts,
- HOP recipients for that programme.

Important modal behavior:

- the modal is centered,
- outside click does not close it,
- escape does not close it,
- the email button is disabled if no HOP recipient is configured.

Notification Outcomes

When you send the notification:

- success shows a completion notification and closes the modal,
- failure keeps the modal available so you can retry,
- both success and failure are recorded into application status history notes.

Reopen Notify HOP Later

If the application is already in one of these statuses:

- `EVALUATION_COMPLETED`
- `FINALIZED`
- `COMPLETED`

and the visible review is already read-only, RP can still reopen the notify-HOP flow using **Notify HOP** from the decision section.

RP Articulation Review

RP also reviews assigned articulation work, not only CTA applications.

The articulation RP workspace is opened from:

- queue items with the `ARTICULATION` badge,
- completed history rows that open `/articulation/view/[id]?from=all`.

Articulation Workspace Overview

The articulation RP page shows:

- breadcrumbs and return navigation,
- report-readiness summary,
- assigned RP,
- source-syllabus count,
- target-syllabus status,
- extraction readiness,
- side-by-side evidence,
- comparison analysis,
- academic decision section.

Important behavior:

- if you are not the assigned RP, the page is effectively read-only,
- HOP can also view this page in a read-only support role,
- the articulation page becomes the full RP workspace once evidence and assignment are ready.

Articulation Setup Readiness

Before comparison can run, the page checks whether:

- an RP has been assigned,

- a target syllabus has been selected,
- all required source syllabi exist,
- extraction is approved on both source and target evidence.

If setup is incomplete, the page shows the exact blocking issues instead of allowing comparison.

Articulation Evidence Preview

Articulation evidence behaves slightly differently from CTA evidence:

- PDF files render inline,
- non-PDF files do not render inline,
- workbook-style files are opened in a new tab instead of using the CTA workbook-preview layout.

Articulation Comparison

The articulation **Run AI Comparison** button is enabled only when:

- you are the assigned RP,
- setup is complete,
- extraction is ready,
- the comparison is not already processing.

Important behavior:

- the articulation comparison also uses the shared staged comparison pipeline,
- the result can feed back into linked CTA work through the shared review workflow,
- completing the articulation comparison can refresh CTA and HOP review surfaces.

Articulation Decision Submission

The articulation decision section closely mirrors the CTA RP decision area:

- approve or reject cards,
- academic-justification textarea,
- read-only state after submission,
- submit button disabled until a decision is selected,
- comparison must already be complete before non-pending submission is accepted.

Unlike the CTA path, there is no HOP notification modal on the articulation review page.

Status Guide

These CTA application statuses are the main ones RP will encounter:

Status	Meaning for RP
`RP_EVALUATION_PENDING`	The case is ready for RP comparison and academic decision.
`SYLLABUS_COMPARISON_IN_PROGRESS`	A comparison run is currently processing for the shared review group.

EVALUATION_COMPLETED	RP work for that application is complete and HOP is the next owner.
FINALIZED	HOP has already locked the application decision. RP review stays read-only.
COMPLETED	AAS has already exported and closed the finalized application. RP review stays read-only.

These articulation outcomes are the main ones RP will see in articulation-related queue/history items:

Status	Meaning for RP
PROPOSED	The articulation rule is still pending academic confirmation.
APPROVED	The articulation review outcome is accepted for approved use.
REVOKED	The articulation outcome is no longer valid for active use.

Troubleshooting

Source Syllabus Preparation Keeps Failing

Check whether:

- the file really exists,
- the source syllabus is not a CANNOT_PROVIDE placeholder,
- extraction has enough time to finish after retry,
- the review item is actually assigned to you.

If the page still shows a preparation failure, retry again and then ask HOP to inspect the underlying syllabus record.

Target Syllabus Not Found

Check whether:

- the target course code is correct,
- HOP has uploaded the correct UPTM target syllabus,
- the uploaded target syllabus has finished extraction.

Use the button that opens /syllabus in a new tab if HOP needs to fix the repository entry.

Run AI Comparison Is Disabled

Check whether:

- source preparation is still running,
- any selected source syllabus still lacks approved extraction,

- the target syllabus is missing,
- the target syllabus extraction is not approved yet,
- the comparison is already processing.

I Submitted A Decision But Cannot Edit It

That is expected for the current RP workflow.

Once an RP decision has been submitted for the review group:

- the decision section becomes read-only,
- the stored decision and justification are shown back to you,
- follow-up changes require workflow intervention rather than another RP edit from the same page.

Notify HOP Cannot Be Sent

Check whether:

- the application has reached `EVALUATION_COMPLETED`, `FINALIZED`, or `COMPLETED`,
- the application really includes review work assigned to you,
- at least one HOP email recipient exists for the target programme,
- the email service is available.

If delivery fails, the failure is still written into status history and the modal can be retried.

I Cannot Find My Review Item In The Queue

Check:

- the search text,
- the selected session,
- whether the item is already completed and now lives in `/reviews/rp/all`,
- whether the item is CTA or articulation work,
- whether HOP assigned the review to another RP instead of you.

I Opened A CTA Review And The Back Button Label Looks Wrong

This is a current UI label issue, not a permission problem.

On CTA RP review pages opened from the queue:

- the button label may say `Return to Dashboard`,
- the actual route still goes back to `/reviews/rp`.

AAS Guide

This guide shows Academic Affairs (AAS) users how to monitor finalized work, export approved credit-transfer records, re-export completed records, and verify application details in UPTM CTS.

What You Can Do In UPTM CTS

As AAS, you can:

- monitor AAS workload and recent activity from /dashboard,
- open the finalized export queue in /aas/finalized,
- export one application, selected applications, or the full filtered finalized queue,
- choose CSV or XLSX output,
- choose whether a finalized export should mark applications as completed or keep them finalized,
- provide export overrides such as kolej, staff_id, and update_date format,
- re-download your own recent export files,
- open the completed archive in /aas/completed,
- re-export historical completed applications without changing their status,
- inspect application status history,
- open application details in a read-only verification view,
- manage your profile, active sessions, programme association, and academic session in /settings.

Before You Start

Before working in the AAS flow, make sure:

- your account has been assigned the AAS role,
- you understand that FINALIZED means academically locked by HOP,
- you understand that COMPLETED means AAS export processing has already been performed,
- you have the external UPTM staff_id value required for export,
- you know that only approved rows with a target course code and target credit hours become export rows,
- you know that **Recent exports** only shows files uploaded by your own AAS account.

First Sign-In And Staff Access

AAS users do not use the student onboarding form.

When your staff role has been assigned correctly:

- sign in through /sign-in,
- the system auto-marks the account as onboarded,
- you are redirected to /dashboard.

If your account is signed in but still does not have the AAS role, you will not be able to open the AAS workspace.

Routes involved:

- /sign-in

- /dashboard

Settings And Staff Scope

Open /settings when you need to manage your account, sessions, or staff workspace context.

The page includes three parts:

- **Account** for your name, email, and role display,
- **Sessions** for viewing and revoking active sign-ins,
- **Programme Association** for choosing a programme and academic session.

Important AAS behavior:

- your export permission comes from the AAS role itself,
- the saved programme and academic session affect the shell header and sidebar context,
- they are useful for staff workspace consistency, but they are not the thing that grants export access.

Sessions Modals

The settings page uses two small modals:

- **Sign out other sessions** keeps the current session active and revokes the others,
- **Create academic session** lets you create or reuse a 4-digit academic session code in MMYY format.

Use the sessions section when you need to check where your account is signed in or force old sessions to close.

AAS Dashboard

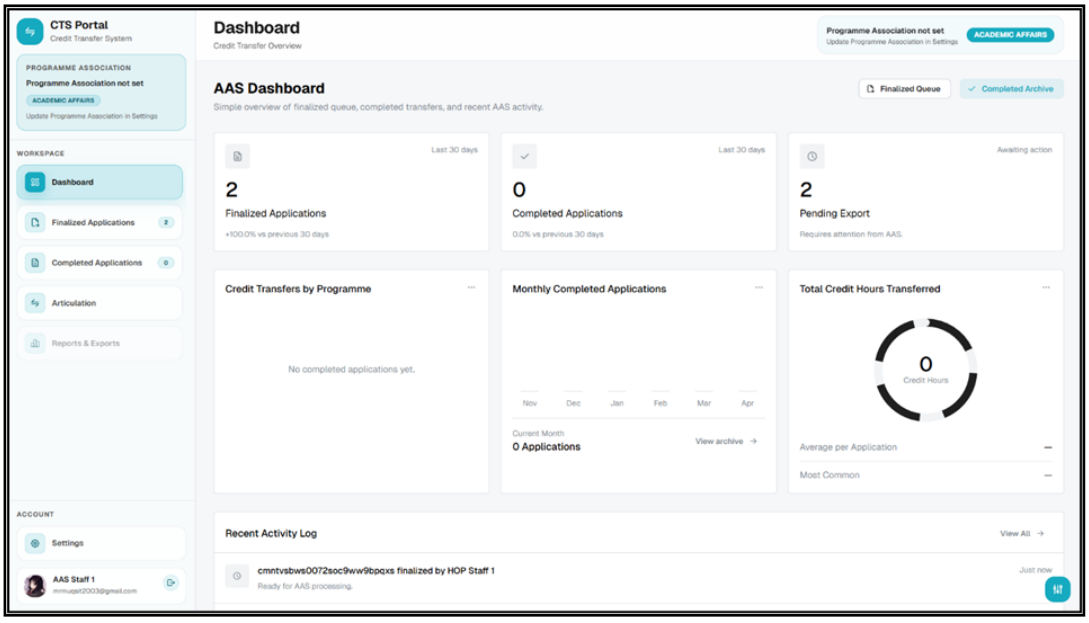


Figure 41: AAS Dashboard

Use /dashboard as your overview page.

The AAS dashboard shows:

- KPI cards for finalized applications, completed applications, and pending export count,

- top completed programmes,
- monthly completed-application trend,
- total transferred credit hours,
- recent activity across finalized, completed, and export events.

Use the dashboard to understand workload and recent operational movement.

The dashboard also gives you direct buttons to:

- /aas/finalized
- /aas/completed

Finalized Queue

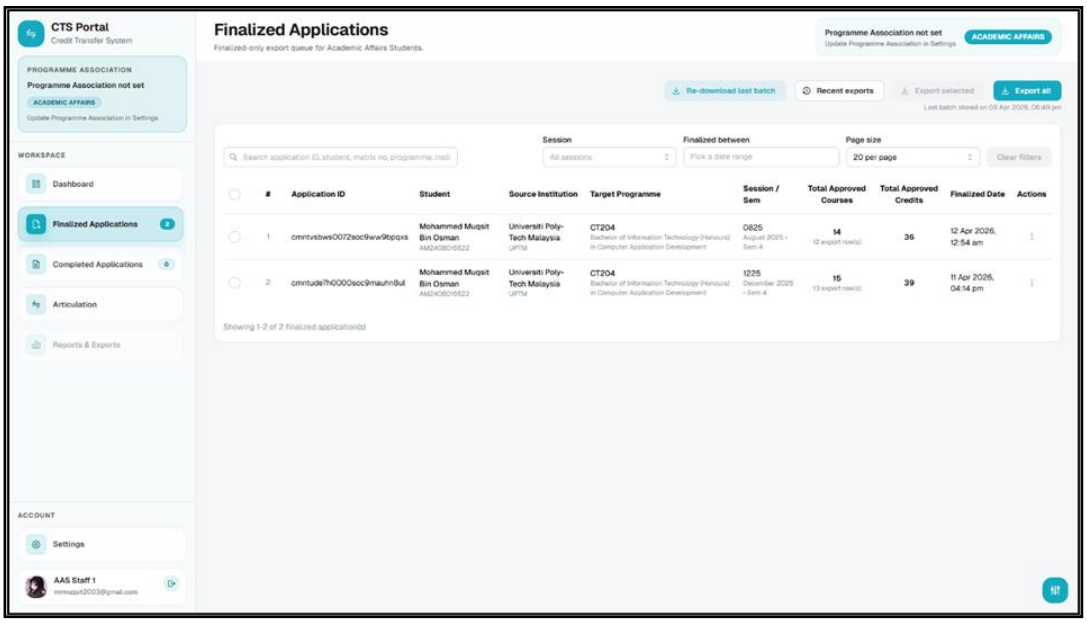


Figure 42: AAS Finalize Queue

Open /aas/finalized to work on applications that HOP has already finalized and AAS can now process operationally.

This page shows only applications whose current status is FINALIZED.

Filters And Table Controls

The finalized queue supports:

- search by application ID, student, matrix number, programme, or institution,
- session filter,
- finalized date-range filter,
- page-size selector,
- pagination,
- clear filters.

Important selection behavior:

- the header checkbox selects only the rows on the current page,
- selection is cleared when filters or page change,
- **Export all** uses the current filters across the whole result set, not only the visible page.

Finalized Queue Columns

Each row shows:

- running number,
- application ID,
- student name and matrix number,
- source institution and short code,
- target programme code and name,
- session and placed semester,
- total approved courses,
- total approved credits,
- finalized date,
- action menu.

Row Actions

Each finalized row provides:

- **View**
- **Export**
- **History**

View opens /applications/[id]?from=aas-finalized.

History opens the shared status-history modal. The modal shows:

- the application summary,
- the full status timeline in oldest-to-newest order,
- who triggered each entry,
- notes attached to each status change.

Top-Level Export Actions

At the top of the page, AAS can use:

- **Re-download last batch**
- **Recent exports**
- **Export selected**
- **Export all**

Important behavior:

- **Re-download last batch** appears only when you have already generated at least one export file,
- **Recent exports** lists stored export files created by your own AAS account,
- **Export selected** stays disabled until at least one row is selected,

- **Export all** stays disabled until at least one finalized application exists in the current filtered result.

Recent Exports

The recent-exports menu shows:

- the export timestamp,
- the stored filename.

This list is user-scoped:

- it shows only files uploaded by the currently signed-in AAS user,
- another AAS user cannot see or download your export files through this menu.

Export Modal From The Finalized Queue

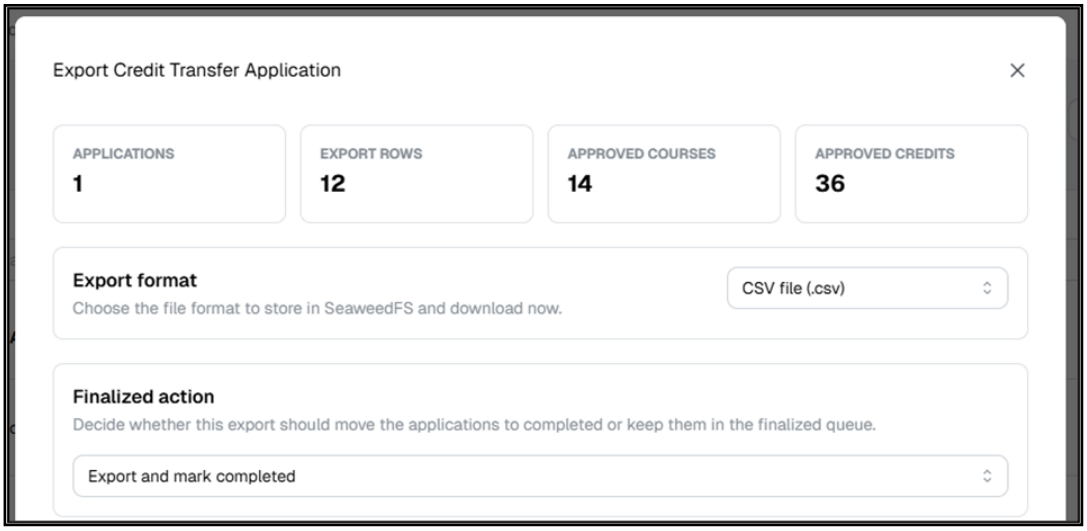


Figure 43: AAS Export CTA

The finalized queue uses one shared export modal for:

- single-application export,
- selected-application export,
- filtered export of the entire queue result.

The modal is centered and opens only after the preview is loaded successfully.

While the export request is being submitted:

- outside click is blocked,
- Escape close is blocked,
- the modal cannot be dismissed until the request finishes.

What The Preview Shows

Before you confirm, the modal shows summary cards for:

- applications,
- export rows,
- approved courses,

- approved credits.

It also shows a scrollable list of included applications with:

- student name,
- matrix number,
- application ID,
- target programme,
- source institution,
- session,
- semester,
- export row count,
- finalized timestamp.

Export Format

In finalized mode, the modal defaults to CSV.

You can still switch to either:

- CSV file (.csv)
- Excel (.xlsx)

The chosen file is downloaded immediately after a successful export and also stored in **Recent exports**.

Finalized Action

Only the finalized queue shows the **Finalized action** selector.

You must choose one of these:

- **Export and mark completed**
- **Re-export only and keep finalized**

Use **Export and mark completed** when AAS is finishing the operational workflow.

Use **Re-export only and keep finalized** when you need a fresh file but do not want the applications to leave the finalized queue yet.

Export Overrides

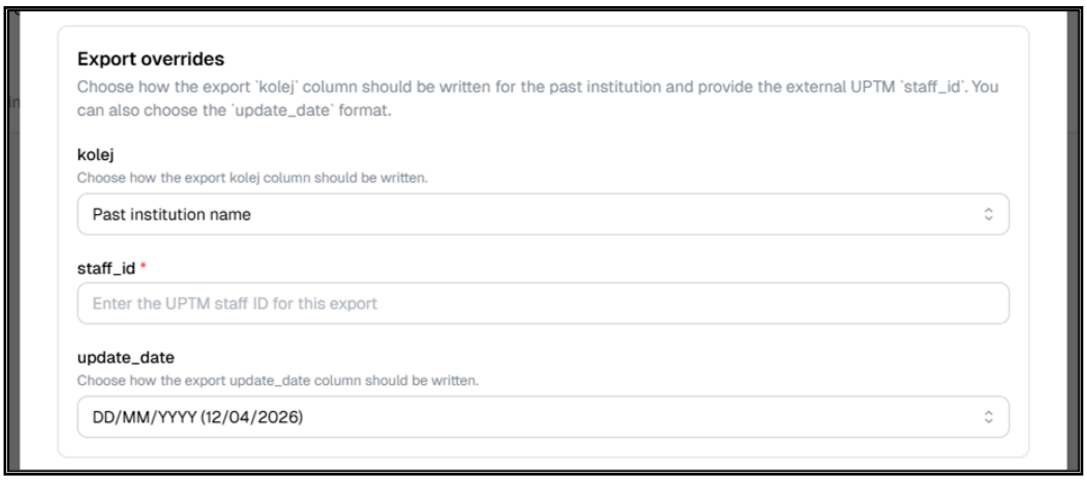


Figure 44: AAS Export Overrides Config

The modal requires export settings before confirmation.

`kolej`

You must choose how the kolej column should be written:

- **Past institution name**
- **Past institution code (shortname)**

If institution code is missing and you choose code mode, the system falls back to the institution name.

`staff_id`

You must enter the external UPTM staff_id.

The export cannot continue while this field is empty.

`update_date`

You can choose one of these formats:

- YYYY-MM-DD
- DD/MM/YYYY
- DD-MM-YYYY
- DD-MMM-YYYY

The modal shows a live example beside each option.

Confirmation Panel

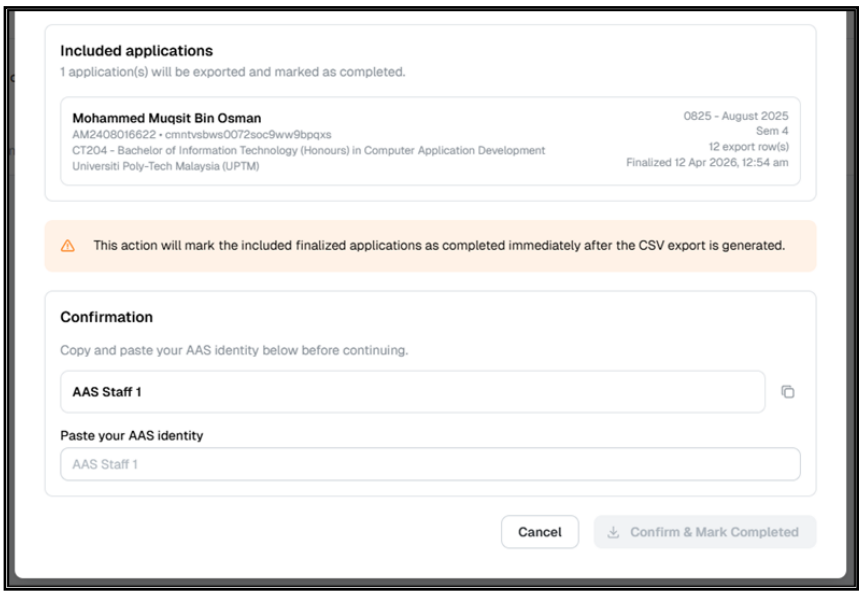


Figure 45: AAS Export Confirmation Panel

The confirmation section shows your AAS identity string, which is:

- your saved display name, or
- your email if no display name is set.

To continue:

- copy the displayed identity if needed,
- paste it into the confirmation field exactly,
- make sure the field matches with no extra differences.

Important behavior:

- the field shows an inline Value does not match error when the text is wrong,
- the confirm button stays disabled until the text matches exactly,
- the confirm button also stays disabled while preview is loading or staff_id is empty.

Confirm Buttons

The main action button changes based on what you selected:

- **Confirm & Mark Completed**
- **Confirm & Re-export**

What Happens After Confirmation

If you choose **Export and mark completed**:

1. the export file is generated as CSV or XLSX,
2. the file is stored permanently in the export store,
3. approved export rows are written to CreditTransferExport,
4. the included applications move from FINALIZED to COMPLETED,

5. application status history is updated,
6. student completion emails are attempted,
7. the download starts automatically,
8. the queue refreshes.

If you choose **Re-export only and keep finalized**:

1. a fresh export file is generated,
2. the file is stored permanently in **Recent exports**,
3. the applications stay in FINALIZED,
4. the download starts automatically.

Success And Warning Notifications

After a successful finalized export:

- a green notification appears when export and email delivery are fully clean,
- a yellow notification appears when export succeeds but some completion emails fail or are skipped.

The notification can include email summary counts such as:

- sent,
- failed,
- skipped.

Closing The Modal Midway

If you change export settings and try to close the modal before confirming, the browser shows a discard confirmation.

The warning message is different depending on mode:

- in normal finalized export, it reminds you that no applications will be marked completed unless you confirm,
- in re-export mode, it reminds you that no file will be generated unless you confirm.

Completed Archive

Open /aas/completed to review applications that have already been exported and operationally closed.

This page shows only applications whose current status is COMPLETED.

Filters And Table Controls

The completed archive supports:

- search,
- session filter,
- completed date-range filter,
- page-size selector,
- pagination,
- clear filters.

Selection behavior is the same as the finalized queue:

- page checkbox selects only the current page,
- selection clears when you change filters or page,
- **Export all** acts on the whole filtered completed result, not only visible rows.

Completed Archive Columns

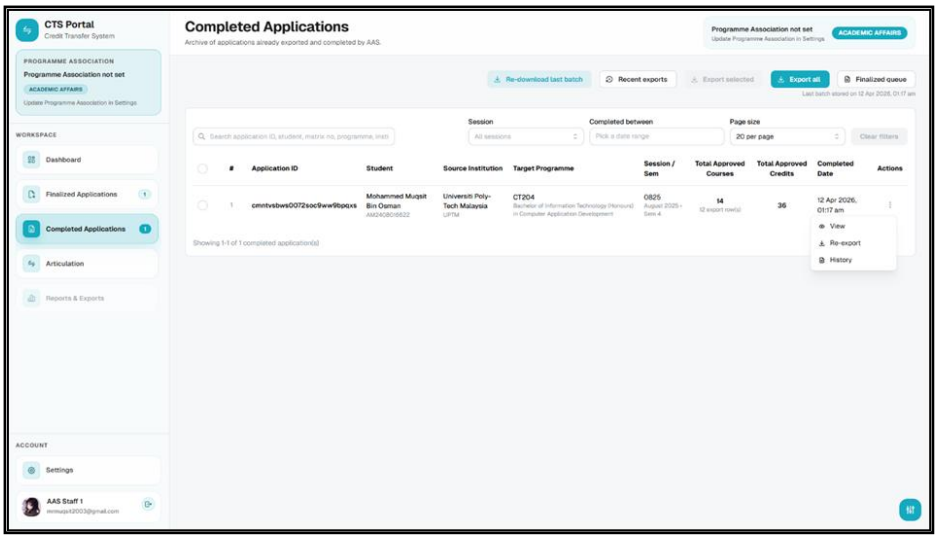


Figure 46: Completed Archive

The completed archive shows almost the same columns as the finalized queue, but the last date column is:

- **Completed Date**

Completed Row Actions

Each completed row provides:

- **View**
- **Re-export**
- **History**

View opens /applications/[id]?from=aas-completed.

History opens the same status-history modal used elsewhere.

Top-Level Actions In Completed Archive

At the top of the page, AAS can use:

- **Re-download last batch**
- **Recent exports**
- **Export selected**
- **Export all**
- **Finalized queue**

Use **Finalized queue** when you want to jump back to applications that are still awaiting operational completion.

As with the finalized page:

- **Export selected** stays disabled until at least one completed row is selected,
- **Export all** stays disabled until the current filtered completed result contains at least one application.

Export Modal From The Completed Archive

The completed archive reuses the same export modal, but its behavior is different.

What Changes In Completed Mode

In completed mode:

- the modal defaults to XLSX,
- the **Finalized action** selector is removed,
- the warning alert always says the export will not change status,
- the main button is always **Confirm & Re-export**.

Completed-mode export is always a re-export only.

The selected completed applications stay in COMPLETED.

What Still Stays The Same

Completed-mode export still requires:

- export preview,
- kolej choice,
- staff_id,
- update_date format,
- exact identity confirmation.

The generated file is still:

- downloaded immediately,
- stored in **Recent exports**,
- available later through the AAS download route.

Read-Only Application Detail For AAS

When you press **View** from either AAS page, the app opens the shared application detail screen in AAS read-only mode.

Important AAS behavior on that screen:

- the page is for verification only,
- AAS cannot edit student or course data there,
- AAS does not perform export completion from that detail page,
- the page shows a read-only notice explaining that export and completion actions are not available there and are handled from the queue,
- the back button changes based on where you came from.

Back navigation:

- from /aas/finalized, the button goes back to the finalized queue,
- from /aas/completed, the button goes back to the completed archive.

How Export Files Are Downloaded

Stored export files are downloaded through:

- /api/aas/exports/[id]

Important access rules:

- you must be signed in as AAS,
- the file must be an EXPORT file,
- the file must have been uploaded by your own account.

The route then redirects to a time-limited presigned download URL.

Practical meaning:

- old links are not meant to be shared permanently,
- if another AAS user generated the export, you cannot download it through your own session,
- the safest way to access files is through **Recent exports** or **Re-download last batch**.

What The Export Actually Contains

Each export row is built from approved transfer outcomes only.

The exported columns are:

- stud_id
- subject_code
- cdt_hrs
- kolej
- notes
- session
- semester
- staff_id
- update_date
- update_time

Important behavior:

- rows are created only for approved items with target course code and target credit hours,
- if an application has approved courses but some are missing target-code or credit-hour data, those incomplete rows are not exported,
- one application can produce multiple export rows,
- the queue shows both approved course count and export row count, which are not always identical.

Status Guide

These are the statuses AAS needs to understand most clearly:

Status	Meaning for AAS
FINALIZED	HOP has locked the academic decision and the application is ready for AAS export processing.
COMPLETED	AAS export processing has already been performed and the application is now operationally closed.

You may also see earlier statuses in application history, but AAS actions are centered on FINALIZED and COMPLETED.

Troubleshooting

The Confirm Button Is Disabled

Check whether:

- preview is still loading,
- staff_id is still empty,
- the confirmation text does not exactly match your displayed identity,
- there are no exportable rows in the resolved scope.

Export Selected Does Nothing Useful

Check whether:

- you selected at least one row,
- you are still on the same page where you made the selection,
- you changed filters or pages and cleared the selection without noticing.

If you want the whole filtered result across all pages, use **Export all** instead.

Export All Is Disabled

Check whether the current filtered result actually contains any applications.
If not, clear or widen the search, session, and date filters first.

Some Approved Courses Did Not Appear In The Export

Only rows with both:

- a target course code,
- a target credit-hour value

become export rows.
Compare:

- **Total Approved Courses**
- **Export Rows**

If those numbers differ, the case has approved outcomes that are not fully exportable yet.

I Exported But The Application Stayed Finalized

Check the **Finalized action** setting you used.

If you selected **Re-export only and keep finalized**, that behavior is expected.

Use **Export and mark completed** when you want the application to leave the finalized queue.

I See An Email Warning After Export

That means the export itself succeeded, but one or more student completion emails were:

- failed, or
- skipped.

The file is still stored and the application can still be completed successfully even when email delivery shows warnings.

I Cannot Download An Older Export

Check whether:

- the file appears under your own **Recent exports**,
- you are signed in with the same AAS account that generated it.

The download route only allows access to export files uploaded by the current user.

The Modal Says The Application Is No Longer Available For Export

This means the export scope changed before confirmation.

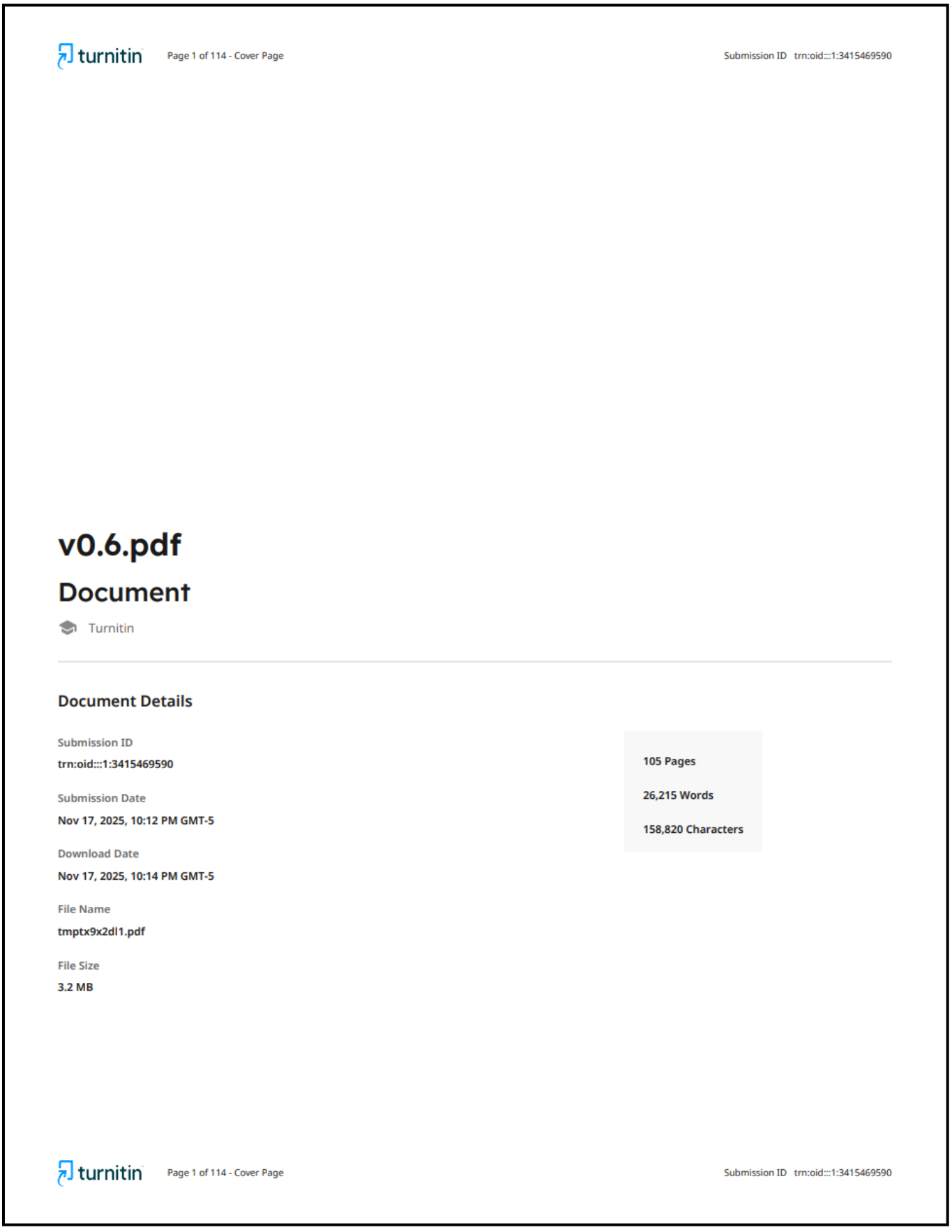
Typical reasons:

- the application status changed,
- one of the selected rows left the queue,
- the filtered result is no longer the same as when the modal was opened.

Refresh the page and reopen the export modal before trying again.

Appendix C – Turnitin Result

i. Turnitin Result FYP1 (Chapter 1-3)



turnitin Page 1 of 114 - Cover Page Submission ID trn:oid:::1:3415469590

v0.6.pdf

Document

Turnitin

Document Details

Submission ID
trn:oid:::1:3415469590

Submission Date
Nov 17, 2025, 10:12 PM GMT-5

Download Date
Nov 17, 2025, 10:14 PM GMT-5

File Name
tmptx9x2dl1.pdf

File Size
3.2 MB

105 Pages

26,215 Words

158,820 Characters

turnitin Page 1 of 114 - Cover Page Submission ID trn:oid:::1:3415469590

9% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- **126** Not Cited or Quoted 9%
  Matches with neither in-text citation nor quotation marks
- **4** Missing Quotations 0%
  Matches that are still very similar to source material
- **2** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

6% Internet sources
4% Publications
6% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

ii. Turnitin Result FYP2 (Chapter 4-6)

turnitin Page 1 of 160 - Cover Page Submission ID trn:oid:::1:280124984

FYP_CHAPTER_4_5_6.docx

My Files
My Files
University of Khartoum

Document Details

Submission ID
trn:oid:::1:280124984

Submission Date
Apr 12, 2026, 5:45 PM GMT+5

Download Date
Apr 12, 2026, 5:47 PM GMT+5

File Name
FYP_CHAPTER_4_5_6.docx

File Size
14.0 MB

158 Pages

29,975 Words

194,837 Characters

turnitin Page 1 of 160 - Cover Page Submission ID trn:oid:::1:280124984

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

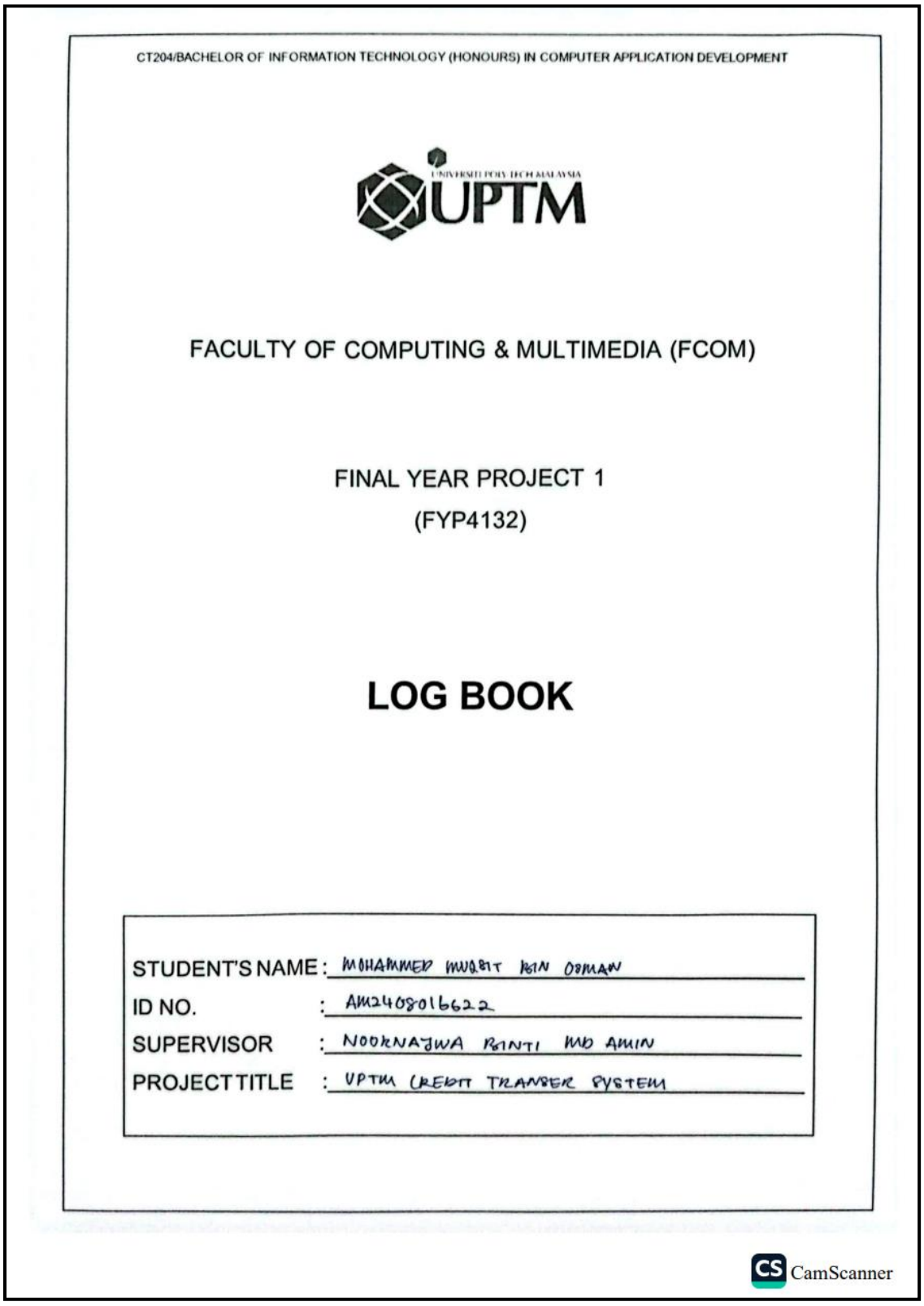
What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

Appendix D – Logbook

i. Log Book FYP1 (FYP4132)



CT204/BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT

UNIVERSITI PERTAHANAN MALAYSIA
UPTM

FACULTY OF COMPUTING & MULTIMEDIA (FCOM)

FINAL YEAR PROJECT 1
(FYP4132)

LOG BOOK

STUDENT'S NAME : MOHAMMED MUQSIT BIN OSMAN
ID NO. : AM2408016622
SUPERVISOR : NOORNAJWA BINTI MD AMIN
PROJECT TITLE : UPTM CREDIT TRANSFER SYSTEM

CamScanner

CT204/BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT

Date/ Week		Agenda	Next Agenda	Signature (Supervisor)
6/8/25	1	Title consultation for FYP	FYP title confirmation	
15/8/25	2	Title confirmation in FYPMS	Proposal writing	
20/8/25	3	Identify problem statement & objective	proposal writing continuation	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 7/11/25
27/8/25	4	Verify & submission proposal	Chapter 1 Report writing	
3/9/25	5	Chapter 1 review	Do correction for Chapter 1 & start writing Chapter 2	
10/9/25	6	Discuss & Investigation	Review Chapter 2 Investigation	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 7/11/25
24/9/25	7	Verify Ch.2 ~~Invest~~ Investigation & start related works	Review related works (Ch 2) & start Ch.3	
1/10/25	8	review methodology	Start Ch.3 requirement	
8/10/25	9	review functional & non-functional	Start preparing question for questionnaire & interviews	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia
15/10/25	10	Finalized question for questionnaire & interview	Start Ch.3 Analysis	7/11/25

CamScanner

CT204/BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT

22/10/25	11	Review Ch.3 Analysis	Start Ch.3 Use case and Flow Chart diagram	
29/10/25	12	Review use case and flow chart diagram	~~Start~~ Finishing Report & start slide presentation	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 7/11/25
7/11/25	13	Review finalizes Report & slide Presentation	Presentation	
	14	Presentation		

CamScanner

ii. Log Book FYP1 (FYP4144)

CT204 / BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT

FACULTY OF COMPUTING & MULTIMEDIA (FCOM)

FINAL YEAR PROJECT 2

FYP4144

LOG BOOK

STUDENT'S NAME : MOHAMMED MUQSIT BIN OSMAN

ID NO. : AM2408016622

SUPERVISOR : NOORNAJWA BINTI MD AMIN

PROJECT TITLE : UPTM Credit Transfer System

CT204 / BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) IN COMPUTER APPLICATION DEVELOPMENT

Week		Agenda	Next Agenda	Signature (Supervisor / Coordinator)
23/12/25	1	Brainstorming, Introduction to the course and Title selection	wireframe design for system	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
31/12/25	2	Renew wireframe design	Start system developing and Chapter 4	
7/1/26	3	Review system feature and its requirement	Obtain student data and organize workflow	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
21/1/26	4	Review Chapter 4 and organize database	Fix chapter 4 and continue developing	
28/1/26	5	Develop LTA workflow and integrate with database	Continue system implementation and improve interface	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
MID-TERM BREAK				NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
4/2/26	6	Continue system implementations and	start unit testing for developed module	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
11/2/26	7	Develop student module	Debug systems error	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
25/2/26	8	Develop HOP module	prepping & conformation on RP module	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
4/3/26	9	Develop RP module	Confirmation on AAS module	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
11/3/26	10	Develop AAS module	Fixing All user module	
18/3/26	11	Review complete system before Client testing	Client testing for feedback	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
25/3/26	12	Review poster and slide presentation	Fix poster and slide presentation	
30/3/26	13	Fully demo system	Presentation	NOORNAJWA BINTI MD AMIN Ketua Program CT204 Fakulti Pengkomputeran & Multimedia Universiti Poly-Tech Malaysia 9/4/26
2/4/26	14	Presentation	submission report poster n slide	

References

Abdul Aziz, F. F. ., Isa, R., Amin, N. M. ., Azizan, N. ., Kamarzaman, N. S. ., & Jalal, S. F. (2025). Bridging Academic Boundaries: A Design and UAT Study of the Automated Credit Transfer System (ACTS). *Journal of Ecohumanism*, *4*(4), 953 –. https://doi.org/10.62754/joe.v4i4.6820

Al-Fedaghi, S. (2020). Modeling the Realization and Execution of Functions and Functional Requirements. *IJCSIS) International Journal of Computer Science and Information Security*, *18*(3). https://arxiv.org/pdf/2004.00078.pdf

AL-Msie'deen, R., Blasi, A. H., Salman, H. E., Alja'afreh, S. S., Abadleh, A., Alsuwaiket, M. A., Hammouri, A., Al_Nawaiseh, A. J., Tarawneh, W., & Al-Showarah, S. A. (2022). Detecting commonality and variability in use-case diagram variants. *ArXiv:2203.00312 [Cs]*, *100*(04). https://arxiv.org/abs/2203.00312

Alotaibi, M. (2021). The Role of Information Systems in Enhancing the Implementation of Administrative Decisions. *International Journal of Business and Management*, *17*(1), 1. https://doi.org/10.5539/ijbm.v17n1p1

Ayoub, R. (2017). Understanding SMART Objectives / A Comprehensive Guide. *Zenodo*. https://doi.org/10.5281/zenodo.14640654

Azizan, N., Isa, R., Abdul Aziz, F. F., & Amiruddin, M. (2021). The Development of a Web-based Credit Transfer Application (CTA) for Higher Academic Institution: From Feasibility Study to Testing Phase. *IOP Conference Series: Materials Science and Engineering*, *1062*(1), 012041. https://doi.org/10.1088/1757-899x/1062/1/012041

Bhattacharya, S. (2023). The Software Principles of Design for Data Modeling: Gathering Requirements – A Detailed Guide. In D. Samanta (Ed.), *The Software Principles of Design for Data Modeling* (pp. 226-247). IGI Global Scientific Publishing. https://doi.org/10.4018/978-1-6684-9809-5.ch017

Boyle, A. (2022). *Data collection*. https://doi.org/10.31219/osf.io/h9an8

Cao, H. (2024). *Recent advances in text embedding: A Comprehensive Review of Top-Performing Methods on the MTEB Benchmark*. https://doi.org/10.48550/arxiv.2406.01607

Chandrasekaran, D., & Mago, V. (2022). Automating Transfer Credit Assessment-A Natural Language Processing-Based Approach. *Computers, Materials & Continua*, *73*(2), 2257–2274. https://doi.org/10.32604/cmc.2022.027236

Chand, S. P. (2025). Methods of Data Collection in Qualitative Research: Interviews, Focus Groups, Observations, and Document Analysis. *Advances in Educational Research and Evaluation*, *6*(1), 303–317. https://doi.org/10.25082/aere.2025.01.001

DeepLearning.AI. (2023). *Natural Language Processing (NLP) - A Complete Guide*. Www.deeplearning.ai. https://www.deeplearning.ai/resources/natural-language-processing/

Dongmo, C. (2024). A Review of Non-Functional Requirements Analysis Throughout the SDLC. *Computers*, *13*(12), 308. https://doi.org/10.3390/computers13120308

Dorimé-Williams, M., Couturier, L., & Kadlec, A. (2025, July). *Filling the Transfer Data Gap: Implications for Policy and Practice*. MDRC. https://www.mdrc.org/work/publications/filling-transfer-data-gap

Enevoldsen, K., Chung, I., Kerboua, I., Kardos, M., Mathur, A., Stap, D., Gala, J., Siblini, W., Krzemiński, D., Indra, W. G., Sturua, S., Utpala, S., Ciancone, M., Schaeffer, M., Sequeira, G., Misra, D., Dhakal, S., Rystrøm, J., Solomatin, R., & Çağatan, Ö. (2025). *MMTEB: Massive Multilingual Text Embedding Benchmark*. ArXiv.org. https://arxiv.org/abs/2502.13595

Gupta, S. (2023). *Decoupled Architecture & Microservices - Saurabh Gupta - Medium*. Medium; Medium. https://medium.com/@saurabh.engg.it/decoupled-architecture-microservices-29f7b201bd87

Kulkarni, S. (2025). *Role of Faculty in Credit Transfer and Mapping in Higher Education*. 145–176. https://doi.org/10.4018/979-8-3693-7635-5.ch006

Metto, E., Miriam, M., & Benson N., K. (2022). A Study of the Management of Student Records in Academic Registrars' Offices in Kenyan Universities. *African Journal of Empirical Research*, *3*(1), 68–77. https://doi.org/10.51867/ajernet3.1.8

MQA. (2016). *The Official Portal Of MQA*. Mqa.gov.my. https://www.mqa.gov.my/new/faq_credittransfers.cfm#gsc.tab=0

NACAC. (2022, October). *The Transfer Process Defined*. National Association for College Admission Counseling (NACAC). https://www.nacacnet.org/the-transfer-process-defined/

Ninja, N. (2023). *TF-IDF: Weighing Importance in Text*. Let's Data Science. https://letsdatascience.com/tf-idf/

Nogueira, R., & Cho, K. (2019). *Passage Re-ranking with BERT*. ArXiv.org. https://arxiv.org/abs/1901.04085?spm=a2ty_o01.29997173.0.0.298bc921Ha427c&file=1901.04085

Nurdiana, D., Adhi Susilo, Dwi Astuti Aprijani, & Andri Suryadi. (2022). Sistem Informasi Alih Kredit Mata Kuliah (SIAKSI) untuk Mahasiswa berbasis Web. *EDUMATIC Jurnal Pendidikan Informatika*, *6*(2), 384–393. https://doi.org/10.29408/edumatic.v6i2.6753

Nurdiana, D., Susilo, A., Aprijani, D. A., & Suryadi, A. (2021). THE DEVELOPMENT OF WEB-BASED CREDIT TRANSFER APPLICATIONS IN THE FACULTY OF SCIENCE AND TECHNOLOGY UNIVERSITAS TERBUKA (A CASE STUDY IN THE INFORMATION SYSTEM PROGRAM). *International Journal of Global Operations Research*, *2*(4), 150–161. https://doi.org/10.47194/ijgor.v2i4.84

Okeke, N. (2021). *Agile Methodology: Meaning, advantages, disadvantages & more*. TargetTrend. https://targettrend.com/agile-methodology-meaning-advantages-disadvantages-more/

Okon, G., Umoren, E., & Philips, K. (2023). Academic Records Management Systems (ARMS) and Students' Academic Records Maintenance in Nigerian Universities. *Library Philosophy and Practice (E-Journal)*. https://digitalcommons.unl.edu/libphilprac/7936/

Osman, M. M. B. (2023). CT204 credit transfer system [Unpublished diploma project]. Universiti Poly-Tech Malaysia.

Qwen. (2024, November 29). *Qwen (Qwen)*. Huggingface.co. https://huggingface.co/Qwen

QwenLM. (2025). *GitHub - QwenLM/Qwen3-Embedding*. GitHub. https://github.com/QwenLM/Qwen3-Embedding?tab=readme-ov-file

Reimers, N., & Gurevych, I. (2019). *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. ArXiv.org. https://arxiv.org/abs/1908.10084

Scott, A., Campbell, K. K., Carey, J. P., Velez, L., Ambardekar, A. P., & Scott, D. J. (2024). Understanding ACGME Standards for Simulation: A Document Analysis of Institutional and Program Requirements. *Journal of Graduate Medical Education*, *16*(6), 691–700. https://doi.org/10.4300/jgme-d-24-00127.1

Srinivas, T., A. David Donald, G. Thippanna, C. Madiletty, & B. Thulasi Thanmai. (2023). Exploring the Uncharted: A Research Problem Statement. *Zenodo (CERN European Organization for Nuclear Research)*, *5*(3). https://doi.org/10.5281/zenodo.8214100

Stang, J., Kroeger, S., & Zaeh, M. F. (2023). Semi-structured expert interview-based requirements elicitation for a digitalized production ramp-up. *Procedia CIRP*, *120*, 279–284. https://doi.org/10.1016/j.procir.2023.08.050

StraighterLine. (2022, February). *What is Transfer Credit?* Straighterline.com. https://www.straighterline.com/blog/what-is-transfer-credit

Subari, A., & Fauzi, A. (2019). Credit Transfer System (CTS) Design in Academic Information System of Diponegoro University. *Journal of Vocational Studies on Applied Research*, *1*(1), 1–4. https://doi.org/10.14710/jvsar.v1i1.4293

UKPLab. (2019). *SentenceTransformers Documentation — Sentence Transformers documentation*. Sbert.net. https://sbert.net/examples/sentence_transformer/applications/retrieve_rerank/README.html

UM. (2025). *UNIVERSITI MALAYA USER MANUAL Pathway to Credit Transfer and Exemption: Student Application (STUDENT) Workstream: -Enrolment*. https://engine.um.edu.my/ug/wow2026/Lampiran%2011%20-%20UserManual_PaCE%20_v1_%20Student.pdf

Zhang, Y., Li, M., Long, D., Zhang, X., Lin, H., Yang, B., Xie, P., Yang, A., Liu, D., Lin, J., Huang, F., & Zhou, J. (2025). *Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models*. ArXiv.org. https://arxiv.org/abs/2506.05176

Zhi, Q., Zhou, Z., Morisaki, S., & Yamamoto, S. (2019). An Approach for Requirements Elicitation using Goal, Question, and Answer. *2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI)*, 847–852. doi:10.1109/IIAI-AAI.2019.00172