UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING

INDUSTRIAL TRAINING REPORT

DEVELOPMENT OF WEB-BASED SYSTEMS AT THE SOFTWARE ENGINEERING RESEARCH GROUP (SERG), UNIVERSITI TEKNOLOGI MALAYSIA

AHMED KHAIRY

(A19EC4044)

2025/2026

BACHELOR OF COMPUTER SCIENCE

(SOFTWARE ENGINEERING)

TRAINING PLACE:	Software Engineering Research Group (SERG), Faculty of Computing, Universiti Teknologi Malaysia
TRAINING PERIOD:	20 WEEKS (9 MARCH 2025 – 24 JULY 2025)
SUPERVISORS:	DR. SHAHLIZA BINTI ABD HALIM
	TS. DR. NOR AZIZAH BINTI SA'ADON
REPORT DATE:	JUNE 2026

ABSTRACT

This report documents a 20-week industrial training placement at the Software Engineering Research Group (SERG), Faculty of Computing, Universiti Teknologi Malaysia. The training centred on the design, development, testing, and documentation of two web-based systems: the UTM Course Hub and the Active Library User (ALU) application. The UTM Course Hub is a multi-user, full-stack administrative web application built for the Faculty of Computing. It comprises a vanilla JavaScript single-page frontend, a backend of twenty-two n8n Cloud workflows exposing REST webhook endpoints, and a Supabase Postgres database of around one hundred and forty course records across six relational tables secured by row-level security. Every mutating endpoint verifies a Supabase Auth JSON Web Token and enforces role-based access for administrators and students. The system supports catalogue management, study plan curation, an administrator-only credit-transfer module that extracts transcript text with Google Cloud Vision and predicts UTM equivalents through a Groq large-language-model matcher, and a chat assistant powered by a LangChain agent with twelve database-backed tools. The ALU application is a frontend-only gamified platform for Perpustakaan Awam Johor targeting primary school children, running entirely in the browser with localStorage as its sole persistence layer. Both systems followed an iterative software development lifecycle spanning requirement gathering, design, implementation, testing, and formal use case documentation. The placement strengthened competencies in full-stack development, database design, workflow automation, frontend engineering, and technical documentation, alongside professional skills in time management, problem-solving, and stakeholder communication.

ABSTRAK

Laporan ini mendokumentasikan latihan industri selama 20 minggu di Kumpulan Penyelidikan Kejuruteraan Perisian (SERG), Fakulti Komputeran, Universiti Teknologi Malaysia. Latihan memberi tumpuan kepada reka bentuk, pembangunan, pengujian dan pendokumentasian dua sistem berasaskan web: UTM Course Hub dan aplikasi Active Library User (ALU). UTM Course Hub ialah aplikasi web pentadbiran bersifat full-stack dan berbilang pengguna yang dibangunkan untuk Fakulti Komputeran. Ia merangkumi antara muka hadapan satu halaman menggunakan JavaScript tulen, backend yang terdiri daripada dua puluh dua aliran kerja n8n Cloud yang mendedahkan titik akhir webhook REST, serta pangkalan data Supabase Postgres yang menyimpan kira-kira seratus empat puluh rekod kursus dalam enam jadual hubungan yang dikawal oleh dasar keselamatan peringkat baris. Setiap titik akhir yang mengubah data mengesahkan JSON Web Token daripada Supabase Auth dan menguatkuasakan capaian berdasarkan peranan bagi pentadbir dan pelajar. Sistem ini menyokong pengurusan katalog kursus, penyusunan pelan pengajian, modul pemindahan kredit khusus pentadbir yang mengekstrak teks transkrip menggunakan Google Cloud Vision dan meramalkan kesetaraan UTM melalui pemadanan berasaskan model bahasa besar Groq, serta pembantu sembang berasaskan ejen LangChain dengan dua belas alat berdasarkan pangkalan data. Aplikasi ALU pula ialah platform bergamifikasi yang hanya melibatkan antara muka hadapan, dibangunkan untuk Perpustakaan Awam Johor bagi kanak-kanak sekolah rendah, dan beroperasi sepenuhnya dalam pelayar menggunakan localStorage sebagai satu-satunya lapisan kepersisan. Pembangunan kedua-dua sistem mengikut kitaran hayat pembangunan perisian secara berulang. Latihan ini mengukuhkan kecekapan dalam pembangunan web full-stack, reka bentuk pangkalan data, automasi aliran kerja, kejuruteraan antara muka hadapan, dan pendokumentasian teknikal, di samping kemahiran profesional dalam pengurusan masa, penyelesaian masalah, dan komunikasi.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

UTM	-	Universiti Teknologi Malaysia
SERG	-	Software Engineering Research Group
PPAJ	-	Perpustakaan Awam Johor
ALU	-	Active Library User
SDLC	-	Software Development Life Cycle
API	-	Application Programming Interface
REST	-	Representational State Transfer
CRUD	-	Create, Read, Update, Delete
OCR	-	Optical Character Recognition
UC	-	Use Case
UI	-	User Interface
UX	-	User Experience
SQL	-	Structured Query Language
RLS	-	Row Level Security
CDN	-	Content Delivery Network

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Industrial training is a supervised placement that lets undergraduate students spend an extended period working inside a real organisation before they graduate. At Universiti Teknologi Malaysia (UTM) it is treated as a graduation requirement rather than an optional extra, because it forces students to translate what they have learned in lectures into deliverables that a stakeholder actually depends on. The exposure to how professional teams communicate, plan, review and ship also gives students a much clearer idea of the pace and expectations of a working software environment.

Within the Faculty of Computing, the placement is scheduled in the final year of study once the student has cleared the technical prerequisites, so that the intern is able to contribute meaningfully from the first week rather than spending the whole placement catching up. The faculty also insists that the role sit within a computing discipline, for example software engineering, database engineering or data analytics, so that the training reinforces the student's academic specialisation instead of pulling them into unrelated work.

This report documents a 20-week industrial training placement undertaken at the Software Engineering Research Group (SERG) within the Faculty of Computing, UTM. The placement focused on the design and development of two web-based software systems, the UTM Course Hub and the Active Library User (ALU) application, together with the formal documentation required to deliver them to a professional standard. This experience provided meaningful exposure to the complete software development lifecycle and to the engineering judgement required when academic theory meets real project constraints.

1.2 Organization Background

Figure 1.1 Logo of Universiti Teknologi Malaysia

Universiti Teknologi Malaysia (UTM) is one of Malaysia's premier public research universities, with its main campus located in Johor Bahru, Johor. As a leading institution in engineering and technology education, UTM is committed to producing competent graduates and advancing knowledge through teaching, research, and innovation. The university hosts a wide range of faculties and research groups that collaborate with industry and the wider community on applied projects.

The Faculty of Computing is one of UTM's foremost faculties, offering undergraduate and postgraduate programmes in software engineering, computer science, data engineering, and related computing disciplines. Beyond teaching, the faculty operates internal administrative systems and undertakes applied software development that serves both its own operations and the surrounding community. The industrial training documented in this report was conducted within this environment.

Within the Faculty of Computing, the Software Engineering Research Group (SERG) carries out research and development across areas such as software design, software quality, requirements engineering, and applied software solutions. The training was hosted by SERG under the supervision of Dr. Shahliza Binti Abd Halim, providing the opportunity to contribute to practical, real-world systems while being mentored by experienced academic staff.

1.3 Organization Structure

The structure of the Faculty of Computing reflects a well-defined hierarchy that promotes clarity of roles across its academic departments and research groups.

The industrial training primarily involved collaboration with SERG, which operates as a research and development unit within the faculty.

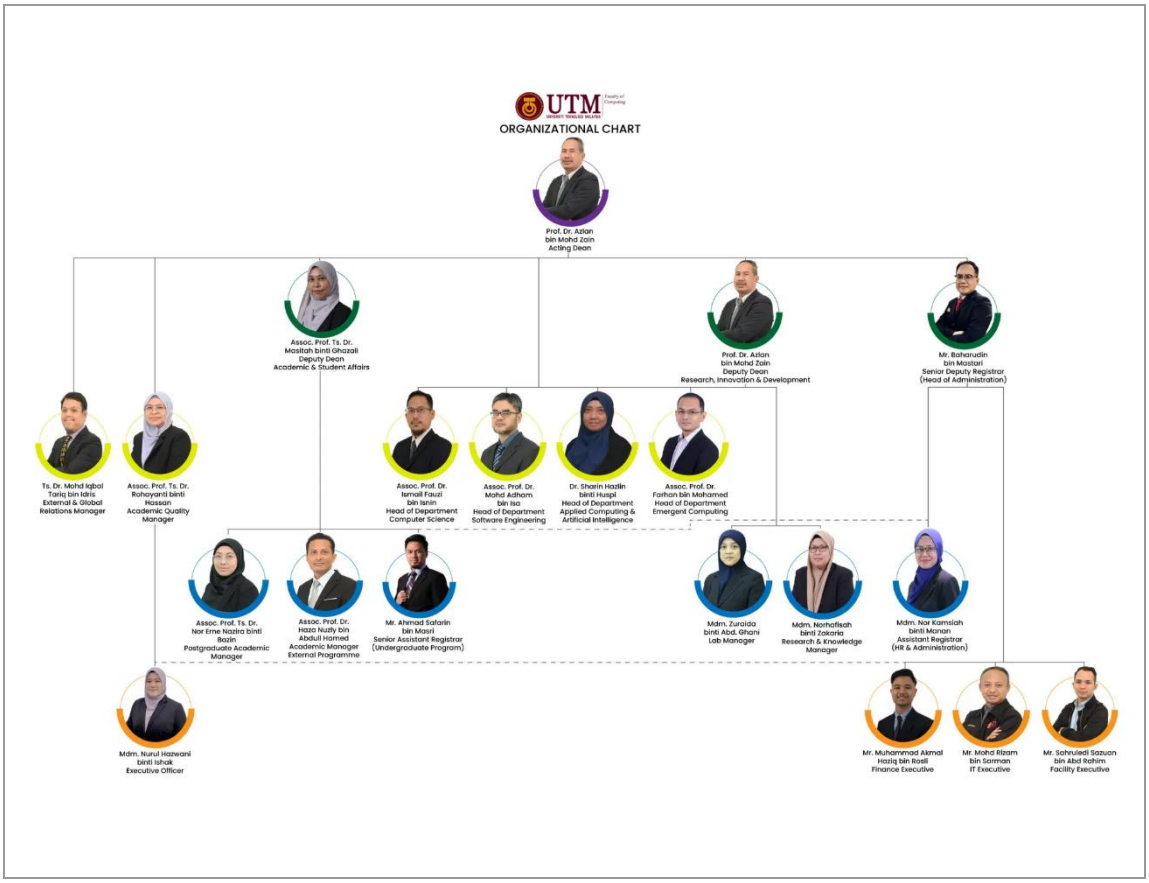


Figure 1.2 Faculty of Computing Organisation Structure

1.3.1 Faculty of Computing

The Faculty of Computing is led by the Dean, supported by Deputy Deans for academic, research, and student affairs. The faculty is organised into academic departments and research groups that oversee teaching delivery, postgraduate supervision, and research activities. Administrative units within the faculty manage student records, the course catalogue, programme planning, and credit-transfer evaluation, which are precisely the administrative workflows that the UTM Course Hub was designed to support.

This administrative context was important to the training, because the first project addressed a genuine operational need within the faculty rather than a hypothetical scenario. Understanding how the catalogue and credit-transfer processes are handled in practice directly shaped the requirements for the system.

1.3.2 Software Engineering Research Group (SERG)

The Software Engineering Research Group (SERG) is a research unit within the Faculty of Computing focused on advancing software engineering practice and producing applied software solutions. The group's interests span software modelling, software quality and testing, requirements engineering, and the development of tools that support both academic operations and community partners.

During the placement, SERG provided supervision, project direction, and review for both systems developed. The group's dual interest in internal faculty tooling and community-facing applications was reflected in the two assigned projects: an internal administrative platform for the faculty, and a public-facing educational application for a community library.

1.4 Internship Job Scope

During the industrial training, responsibilities encompassed the full development lifecycle of two web-based systems, from requirement gathering through to deployment and documentation. The key areas of work are summarised below.

(a) **Full-Stack Web Development:** Designing and building the UTM Course Hub as a complete web application, including a vanilla JavaScript frontend, a backend of n8n Cloud workflows exposing REST webhook endpoints, and a Supabase Postgres database.

(b) **Frontend Application Development:** Developing the Active Library User (ALU) application as a self-contained, browser-only system using HTML, CSS, and JavaScript, with localStorage as its sole persistence mechanism.

(c) **Database Design and Workflow Automation:** Designing the relational schema for the Course Hub, migrating the data store from MongoDB to Supabase Postgres, and orchestrating backend operations as independent n8n workflows.

(d) **Credit-Transfer Automation:** Implementing a client-side document processing pipeline using pdf.js and Tesseract.js to extract course codes from uploaded forms and match them against the live catalogue.

(e) **Requirements Analysis and Documentation:** Producing formal use case documentation for both systems, comprising nineteen use cases for the Course Hub and sixteen for the ALU application.

Working on these two systems back to back was where the placement genuinely paid off in skills terms. Full-stack orchestration, browser-only architecture, relational schema design, and formal use case documentation all became a regular part of the daily workflow rather than isolated coursework, and together they leave a solid technical base to build on in a first professional software role.

1.5 Summary of Chapter

In summary, this chapter introduced the industrial training programme and provided an overview of the host organisation, the Faculty of Computing at Universiti Teknologi Malaysia, and the Software Engineering Research Group within it. A clear understanding of the organisation's structure and operational needs proved essential in aligning the training's goals with real requirements.

The placement at SERG offered a valuable platform to develop technical skills while contributing to two impactful projects: the UTM Course Hub for the faculty and the ALU application for Perpustakaan Awam Johor. Together these projects provided exposure to two distinct architectural contexts, full-stack and frontend-only, establishing a strong basis for the detailed account presented in the following chapter.

CHAPTER 2

SPECIFIC DETAILS ON PROJECTS

2.1 Introduction

This chapter documents the twenty-week placement carried out at the Software Engineering Research Group (SERG), Faculty of Computing, UTM, focusing on the specific tasks that were assigned and the two web-based systems that were built and delivered during the period.

The first and more technically extensive project was the UTM Course Hub, a full-stack administrative web application developed for the Faculty of Computing to manage the course catalogue, curate study plans, and evaluate credit-transfer applications. The second was the Active Library User (ALU) application, a frontend-only gamified platform developed for Perpustakaan Awam Johor to encourage library engagement among primary school children. This chapter details the objectives, the systems, the execution process, the tools used, the timeline, the knowledge applied, and the challenges encountered throughout the training.

2.2 Objectives of Projects

The primary objective of the industrial training was to contribute practical software solutions that serve the operational and community needs associated with the Faculty of Computing, while developing professional software engineering competencies. The specific objectives are listed below.

(a) **Streamlining Administrative Operations:** Developing the UTM Course Hub to digitise and centralise course catalogue management, study plan curation, and credit-transfer evaluation, replacing manual and fragmented processes with a single web application.

(b) **Automating Credit-Transfer Evaluation:** Building a module that extracts course codes from uploaded credit-transfer forms using client-side Optical Character Recognition and matches them against the catalogue, reducing manual checking effort.

(c) **Encouraging Community Library Engagement:** Developing the ALU application to motivate primary school children to use the library through a points-and-badges reward system, interactive games, and a bilingual interface.

(d) **Technical Development:** Gaining hands-on experience across the full stack, including frontend development, relational database design, workflow automation, and REST API integration.

(e) **Professional Exposure:** Adhering to professional standards through requirement gathering, supervisor collaboration, iterative improvement, and formal documentation of all delivered work.

2.3 Industrial Training Projects

The placement focused on developing web-based solutions that address real needs at the Faculty of Computing and within the local community. The two key projects, presented in the order they were prioritised, were the UTM Course Hub and the Active Library User (ALU) application.

2.3.1 UTM Course Hub

The UTM Course Hub is a multi-user, full-stack administrative web application developed for the Faculty of Computing to address inefficiencies in maintaining the course catalogue, building programme study plans, and evaluating credit-transfer applications. The system was developed iteratively across twelve major versions, culminating in version 7.12. It is delivered as a single-page interface served from an n8n workflow, backed by twenty-two n8n Cloud workflows and a Supabase Postgres database containing around one hundred and forty course records. Access is governed by Supabase Auth JSON Web Tokens verified on every mutating endpoint, and users are assigned one of two roles, administrator or student, each with a tailored feature scope. The key features of the system include:

(a) **Catalogue Management:** Administrators can list, filter by programme, search, create, inline-edit, and delete course records, with deletions cascading to any study plan that references the course.

(b) **Study Plan Curation:** Administrators can create named study plans, add and remove courses, save the plan to the database atomically, rename plans, and print a UTM-branded version directly from the browser.

(c) **Credit-Transfer Evaluation:** Administrators upload a student transcript in Portable Document Format or image form; the system extracts the text using Google Cloud Vision (with pdf.js used for text-layer Portable Document Format documents and Tesseract.js retained as a client-side fallback), then invokes a Groq large-language-model matcher that reads the extracted text against a faculty guideline table and returns predicted UTM equivalents together with the student header. The administrator approves or rejects each predicted row with check boxes, may manually add or override any UTM code, and finally saves the record. Every saved record is stored under a dedicated Transfer Records tab, where an administrator can generate an official Approved-only printout or a Full printout that also lists the disapproved rows.

The database for the Course Hub was originally prototyped on MongoDB but was migrated to Supabase Postgres mid-project to resolve connection-stability issues and to take advantage of relational integrity. The six-table schema, comprising courses, plans, plan_courses, credit_transfers, profiles, transfer_mappings, and transfer_guidelines, applies foreign key relationships, cascade deletes, JavaScript Object Notation binary columns where appropriate, and row-level security policies that restrict data access according to the caller's role and identity. In addition to the administrative catalogue and study-plan features, the system exposes a general-purpose chat assistant built on a LangChain agent with twelve database-backed tools, allowing users to query and manipulate the catalogue and their own study plans through natural language. This project directly supports the faculty's objective of operating efficient, technology-supported administrative workflows.

2.3.2 Active Library User (ALU) Application

The Active Library User (ALU) application is a frontend-only gamified web application developed for Perpustakaan Awam Johor (PPAJ), Cawangan Taman Universiti, to encourage library engagement among primary school children. Unlike the Course Hub, the ALU application has no backend or database; it runs entirely in

the browser and uses localStorage as its sole persistence layer, a deliberate choice made because the library had no server infrastructure to host a backend. The system is organised around the following functional areas:

(a) **Registration and Rewards:** Children and parents register, choose avatars, and earn points and badges; the system tracks total points, levels, and a badge collection of eight earnable badges.

(b) **Book Donation and Reuse:** Members submit donation forms online and browse a pool of donated books, earning points for each interaction.

(c) **Catalogue and Games:** A 15-book catalogue with Dewey Decimal classification, search, and a sorting activity, alongside five progressively unlocked games covering library knowledge, word hunting, Dewey matching, title unscrambling, and story sequencing.

(d) **Bilingual Interface:** A single-button toggle switches the entire interface between English and Bahasa Malaysia, with the preference persisted to localStorage and applied on every page load.

The ALU application was accompanied by a formal Use Case Specification covering sixteen use cases. Because all data is held on a single device, the leaderboard rankings are populated using a representative sample dataset, a constraint that was documented explicitly in the specification to set accurate expectations.

2.4 Project Execution

This section walks through how the two systems, the UTM Course Hub and the ALU application, were actually built over the twenty-week placement. The execution followed short iterations with weekly check-ins, which kept the scope realistic, made supervisor feedback easy to incorporate, and made progress visible from week to week.

2.4.1 Development Workflow

Work was split into distinct phases so that the placement did not become a single unbroken sprint. Both systems used an Agile flavour of the Software Development Life Cycle: features were sliced small, reviewed with the supervisor, then merged before the next slice began. This showed up most clearly on the Course

Hub, which was maintained as a versioned artefact and progressed through numbered releases from an early prototype all the way to version 7.12.

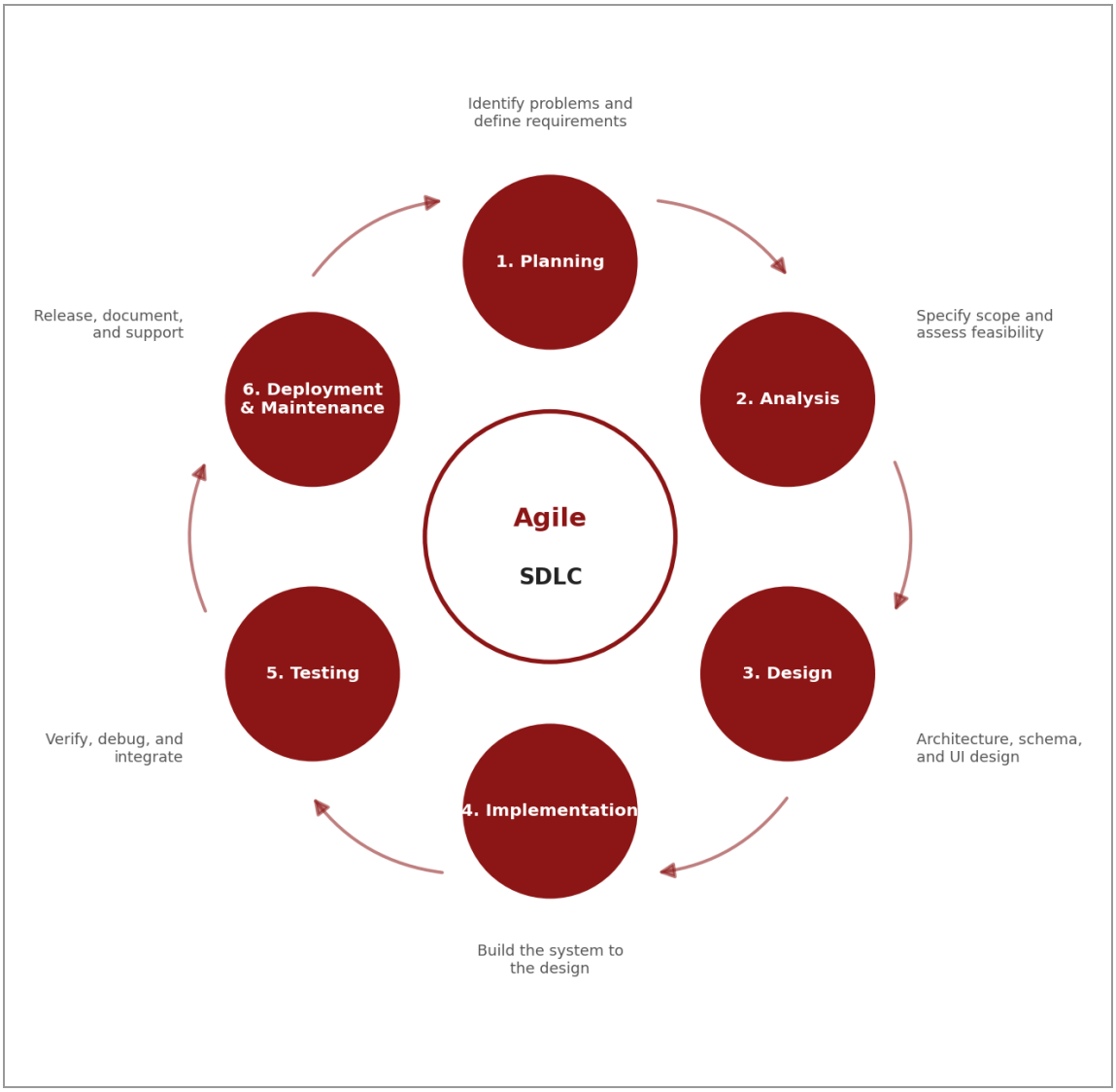


Figure 2.1 Agile Methodology in the SDLC

2.4.2 Requirement Gathering and Planning

Execution began with requirement gathering and planning. Discussions with the supervisor at SERG identified the operational pain points the Course Hub needed to address, namely fragmented catalogue maintenance and manual credit-transfer checking and clarified the engagement goals that the ALU application needed to meet for its young audience. Based on these findings, the scope of each system and its modules was defined and prioritised, with the Course Hub taken first as the larger and more technically involved deliverable.

2.4.3 Initial Setup

Once planning was complete, the development environment was prepared. Visual Studio Code was configured as the primary editor, an n8n Cloud workspace was set up to host the backend workflows, and a Supabase project was provisioned to host the Postgres database. The initial relational schema was designed in Supabase, and the base single-page frontend was scaffolded. For the ALU application, the localStorage data model and page structure were established up front to keep state handling consistent as the feature set grew.

2.4.4 Design and Development

The design and development phases focused on building the user interfaces and implementing the core functionality of each system. For the Course Hub, this involved developing the catalogue, plan, and credit-transfer modules against the n8n backend, and integrating pdf.js and Tesseract.js for client-side document processing. For the ALU application, this involved building the registration, donation, catalogue, games, and leaderboard modules, together with the bilingual interface toggle.

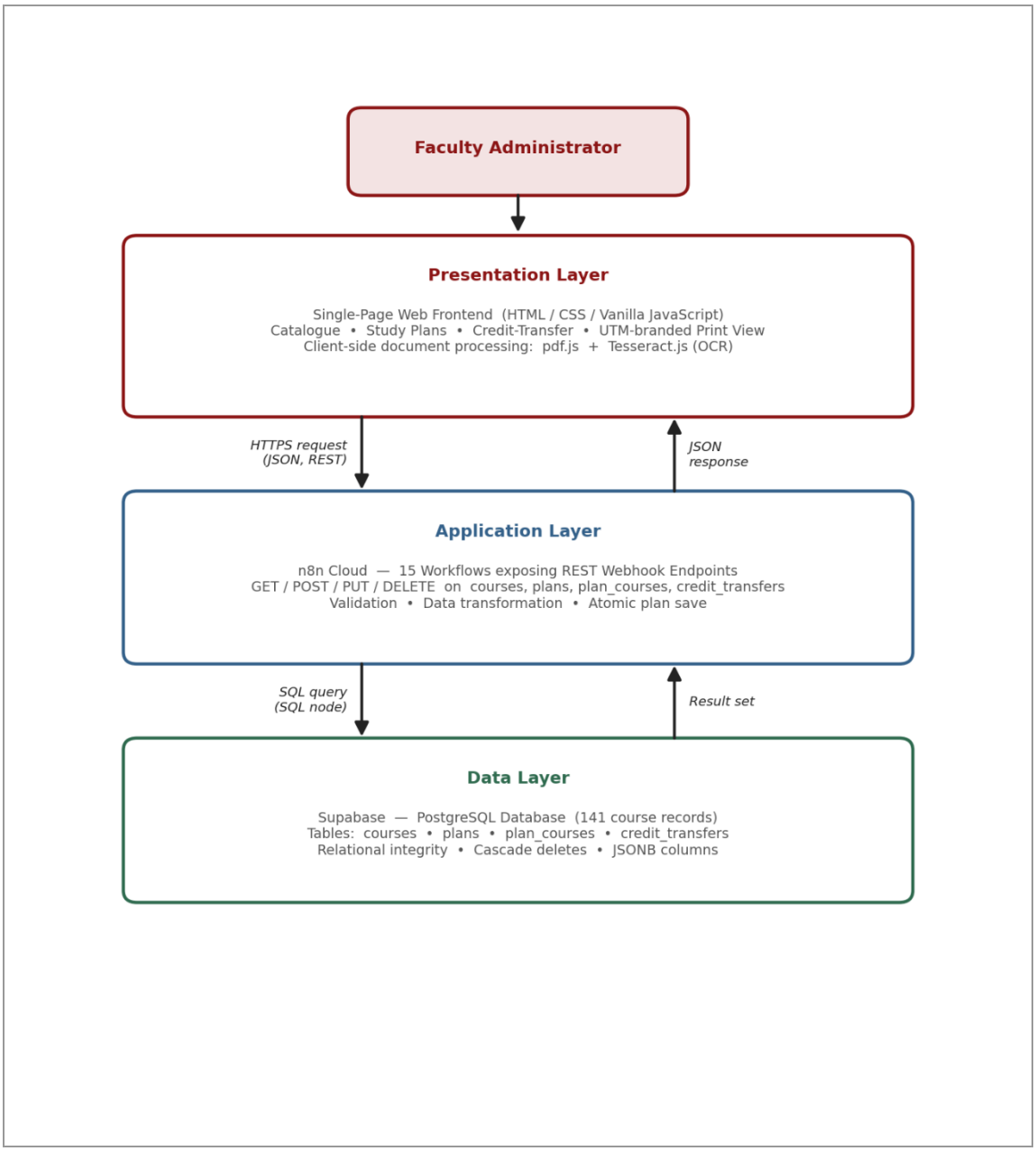


Figure 2.2 UTM Course Hub System Architecture

Figure 2.3 UTM Course Hub Database Schema (Supabase)

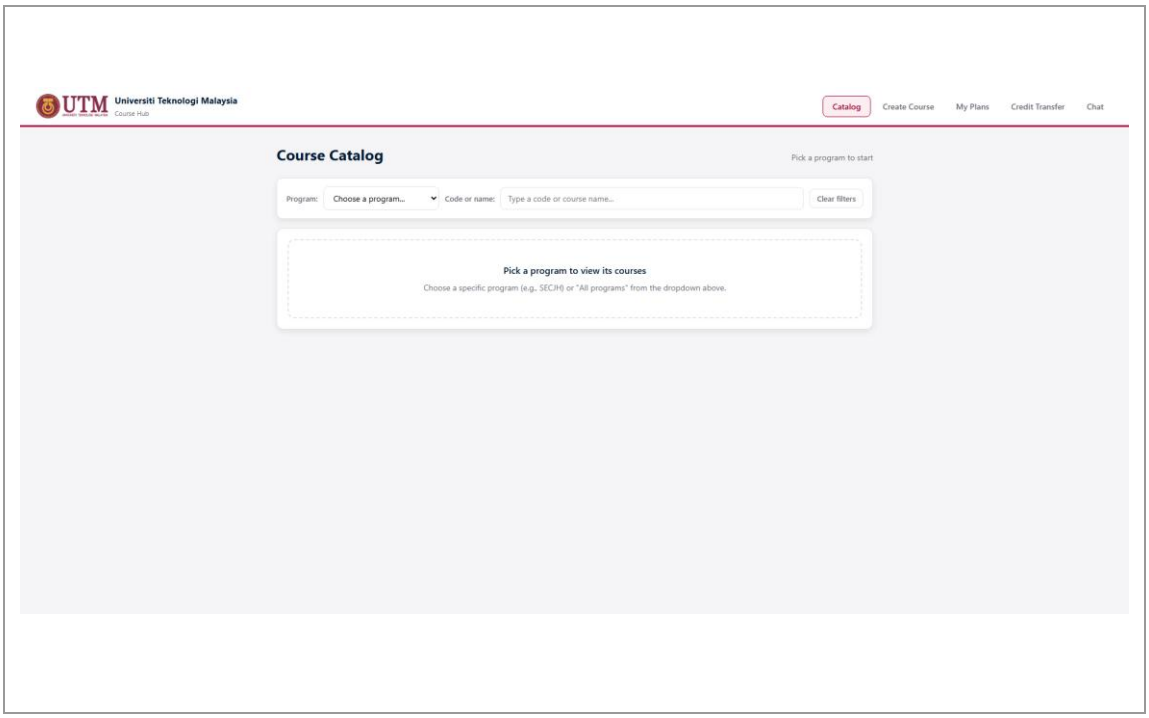


Figure 2.4 UTM Course Hub User Interface

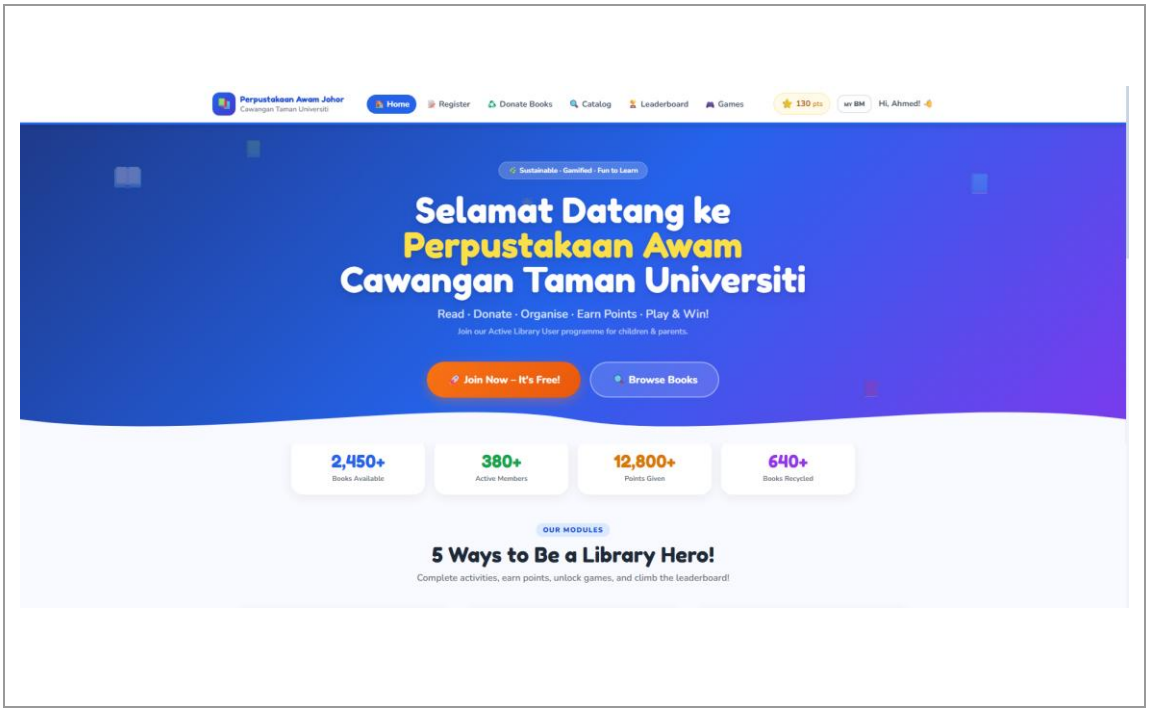


Figure 2.5 ALU Application User Interface

2.4.5 Testing and Debugging

Regular testing and debugging were carried out throughout development to ensure reliability. For the Course Hub, this included resolving cross-origin resource sharing (CORS) misconfigurations, JSON parsing errors in the listing endpoints, and

a blank print-page issue in the plan print view. The OCR pipeline was refined iteratively to handle imperfect scans, most notably by normalising the common misreading of the digit 2 as the letter Z within course code suffixes. For the ALU application, testing focused on validating the points and badge logic and ensuring consistent behaviour across pages.

2.4.6 Communication and Feedback

Collaboration with the supervisor played a significant role in the projects' success. Regular meetings were held to present progress and incorporate feedback. Suggestions arising from these reviews, such as refinements to the credit-transfer matching behaviour and improvements to the catalogue editing experience, were implemented to strengthen the systems.

2.4.7 Final Deployment and Handover

The closing weeks of the placement were dedicated to handover: deploying the two systems and writing formal use case specifications for each. The Course Hub was pushed to n8n Cloud, with Supabase hosting the Postgres database and Auth service, so any future maintainer can inspect, run or edit the workflows in the same place they were built. The ALU application, having no server component, was packaged as a static bundle that runs on any file server or from the local file system, which keeps its maintenance profile as low as possible.

2.4.8 Execution Conclusions

Delivering these two systems produced more than a working codebase: the Course Hub tackled a real administrative bottleneck inside the faculty, while the ALU application answered a genuine engagement need for a public library. Building them end to end sharpened technical judgment in full-stack and pure-frontend contexts respectively, and made the mechanics of communicating with a non-developer stakeholder much more concrete.

2.5 Software and Hardware Used

The following two tables list the hardware and software that were used across the two projects. Each item is included because it appeared repeatedly in the daily

workflow rather than as a one-off tool, and the split between server-backed and browser-only systems drove several of the choices.

2.5.1 Hardware

The hardware listed below supported the development, testing, and deployment of both systems throughout the placement.

Table 2.1 Hardware Used for Project Development

Hardware	Description
Development Laptop	Personal laptop used for all project-related tasks, including development, debugging, and testing. Specifications: Intel Core i5-series processor, 16GB RAM, Windows 11.
External Monitor	Secondary display connected to the laptop to enhance productivity through multi-window work, such as simultaneous editing, browser testing, and reviewing the n8n workflow canvas.

2.5.2 Software Requirements

The software tools below were used for coding, backend orchestration, database management, document processing, testing, and documentation.

Table 2.2 Software Used for Project Development

Software	Description
Visual Studio Code	Primary code editor used to develop the frontend of both systems in HTML, CSS, and JavaScript.
n8n Cloud	Workflow orchestration platform hosting the fifteen REST webhook endpoints that form the Course Hub backend.
Supabase (Postgres)	Managed PostgreSQL database and authentication service used to host and administer the Course Hub's

Software	Description
	six relational tables together with role-based access control.
pdf.js (v3.11.174)	Client-side library used to extract text from credit-transfer PDFs that contain an electronic text layer.
Tesseract.js (v5.0.5)	In-browser Optical Character Recognition library used to read scanned or photographed credit-transfer forms.
Google Chrome DevTools	Used for debugging frontend behaviour, inspecting network requests, and validating localStorage state.
localStorage (Web Storage API)	Client-side persistence mechanism for the ALU application, storing profiles, points, badges, and language preference.
Microsoft Word	Used to prepare the use case specifications and the industrial training report.

2.6 Project Timeline

The Gantt chart below illustrates the 20-week timeline of the industrial training. The placement began with reporting duty and environment setup, followed by the development of the UTM Course Hub as the primary deliverable, the subsequent development of the ALU application, and finally the preparation of documentation and this report. The structured timeline ensured efficient task allocation and steady progress against the training objectives.

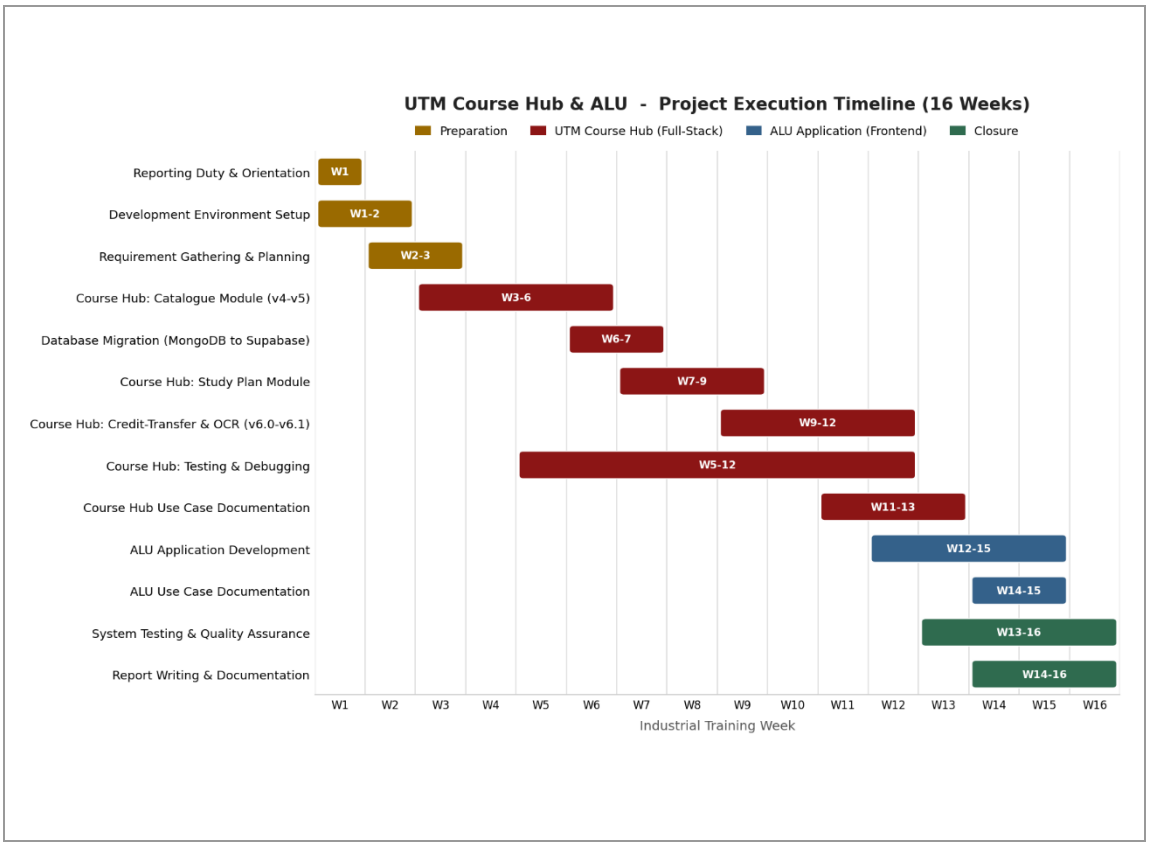


Figure 2.6 Gantt Chart of Project Execution Timeline

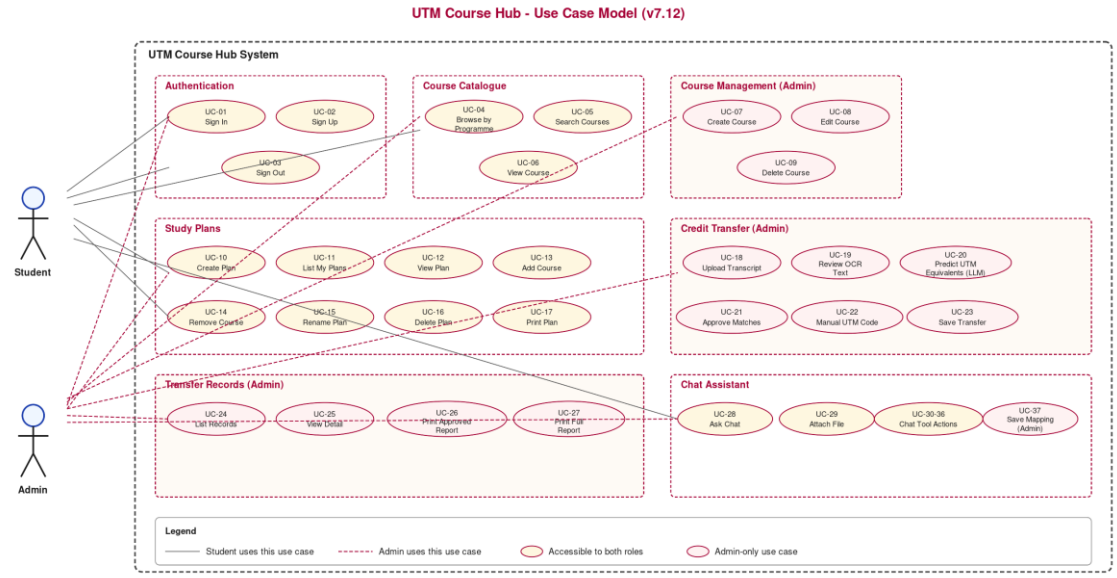


Figure 2.7 UTM Course Hub Use Case Model

2.7 Theoretical and Practical Knowledge Applied

The industrial training required the application of both theoretical and practical knowledge gained during the course of study. The key areas applied are discussed in this section.

2.7.1 Web Development and Frameworks

Both systems were built from raw HTML, CSS, and JavaScript without a framework such as React or Vue. In the Course Hub, this vanilla frontend talked to a workflow-driven backend over REST, and in the ALU application it held the entire product on its own. Working at this level made day-to-day competence with the Document Object Model, the Fetch API, event delegation and manual state management non-negotiable, and demonstrated that a considered vanilla stack can still ship a responsive interface without dragging in a framework.

2.7.2 Database Management

Theoretical knowledge of relational database systems and SQL was applied in designing and managing the Supabase Postgres database for the Course Hub. Understanding of entity-relationship modelling supported the creation of the four-table schema with foreign keys and cascade deletes, while the use of JSONB columns for credit-transfer evaluation results illustrated the practical trade-off between strict normalisation and flexible storage. The migration from MongoDB to Postgres reinforced the differences between document and relational data models.

2.7.3 Workflow Automation and Backend Engineering

The Course Hub backend was implemented as a set of independent n8n Cloud workflows, each exposing a single REST endpoint. This required learning to model resource-oriented operations as workflows, configure webhook triggers, write data-transformation expressions, and connect to Supabase through SQL nodes, providing practical insight into workflow automation as a backend architecture.

2.7.4 Software Development Life Cycle (SDLC)

The SDLC phases of planning, design, development, testing, and deployment were applied to both systems, ensuring systematic progression. Agile practices such as iterative delivery and the incorporation of supervisor feedback were used throughout,

most visibly in the versioned evolution of the Course Hub from early prototypes to version 7.12.

2.7.5 API Development and Integration

Hands-on experience was gained in designing and testing REST endpoints for the Course Hub, including GET, POST, PUT, and DELETE operations on the courses, plans, and credit_transfers resources. The frontend consumed these endpoints asynchronously, with defensive handling of empty or malformed responses to ensure the interface never failed silently.

2.7.6 Optical Character Recognition and Client-Side Processing

The credit-transfer module applied document-processing techniques entirely on the client side, using pdf.js for text-layer extraction and Tesseract.js for OCR. Concepts such as text normalisation and string similarity were applied through a post-processing step that corrected common OCR errors and a Dice bigram similarity routine that suggested catalogue matches for unrecognised codes.

2.7.7 Collaboration and Stakeholder Communication

Regular communication with the supervisor was essential for clarifying requirements, presenting progress, and incorporating feedback. Translating administrative and community needs into precise technical requirements, and reporting progress clearly, strengthened professional communication skills that complemented the technical work.

2.7.8 Problem-Solving and Debugging

Practical experience was gained in debugging and resolving issues across both systems, including CORS misconfiguration, JSON parsing failures, a blank print page, database connection instability, and imperfect OCR output. The ability to isolate root causes and apply targeted fixes was significantly enhanced through the placement.

2.8 Problems Faced, General Skills and Implementation of Task

Several difficulties surfaced during the placement, ranging from platform-level constraints to timing pressure between two parallel projects. Working through them was arguably where the most learning happened, because each one demanded a

specific technical fix and, at the same time, forced clearer prioritisation and communication with the supervisor.

2.8.1 Problems Faced

The problems encountered throughout the completion of the projects are discussed in this section.

2.8.1.1 Database Connection Instability and Migration

The Course Hub was initially prototyped on MongoDB, but recurring connection-stability issues with the hosted cluster disrupted development. Resolving this required migrating the entire data store to Supabase Postgres mid-project, redesigning the data as four relational tables, and rewriting the backend workflows to use SQL. The migration delayed feature work but ultimately produced a more reliable and maintainable system.

2.8.1.2 OCR Accuracy and Post-Processing

The credit-transfer flow originally relied on Tesseract.js, which produced imperfect output for scanned forms, most commonly misreading the digit 2 as the letter Z within course code suffixes. A normalisation step and a Dice bigram similarity fallback were introduced to mitigate this. As the credit-transfer module matured, the primary Optical Character Recognition path was migrated to Google Cloud Vision, whose Document Text Detection endpoint produced substantially cleaner text; Tesseract.js was retained as a fallback. An additional review step was added so that the administrator can inspect and edit the extracted text before it is sent to the Groq large-language-model matcher, guarding against residual misreads such as Z-for-2 in the transcript codes.

2.8.1.3 CORS and JSON Parsing Errors

During integration of the frontend with the n8n backend, cross-origin resource sharing misconfigurations initially blocked requests, and several listing endpoints failed when returning empty bodies. These were resolved by correcting the webhook response headers and adding defensive JSON parsing that falls back gracefully when a response body is empty.

2.8.1.4 localStorage Constraints in the ALU Application

Relying on localStorage as the sole persistence layer meant that user data was confined to a single device and that a genuinely shared leaderboard was not possible. These constraints were managed by establishing a disciplined key-naming convention up front and by populating the leaderboard with a representative sample dataset, with the limitation documented clearly in the use case specification.

2.8.1.5 Time Management Across Two Projects

Delivering two systems of differing complexity within a single placement required careful prioritisation. The Course Hub was treated as the primary deliverable and advanced first, with the ALU application developed in parallel where capacity allowed, ensuring that both systems reached a complete and documented state by the end of the training.

2.8.1.6 Frontend Persistence inside the n8n Cloud Sandbox

Once the Course Hub was deployed to n8n Cloud as a single-page application served from a Respond to Webhook node, an unexpected constraint became apparent. Every response served by the n8n Cloud webhook is delivered with a hard-coded Content Security Policy sandbox directive that omits the allow-same-origin token. This blocks all browser storage application programming interfaces, including localStorage, sessionStorage, and cookies, and it also blocks the Web Locks application programming interface that the Supabase JavaScript client relies on to serialise authentication requests. In practice this meant that a page refresh silently signed the user out and wiped the chat panel, and that the Supabase JavaScript client would hang during sign-in.

The mitigations applied were, in order: overriding the Content Security Policy header on the Respond to Webhook node (rejected by the n8n Cloud proxy), disabling session persistence on the Supabase client, stubbing the navigator.locks application programming interface with a no-operation shim, and finally bypassing the Supabase JavaScript client for authentication entirely by calling the Supabase Auth REST endpoints directly. A permanent fix would have required migrating the single-page application to a static host such as Cloudflare Pages, which would relax the sandbox and restore browser storage. In consultation with the industrial supervisor, this migration was deferred and the current build accepts that a refresh will sign the user out, a trade-off documented explicitly in the accompanying system overview.

2.8.2 General Skills Gained

The general skills gained throughout the execution of the projects are discussed in this section.

2.8.2.1 Communication and Collaboration

Effective communication with the supervisor enabled clear requirement gathering, regular progress updates, and the incorporation of feedback, ensuring that both systems remained aligned with the intended objectives.

2.8.2.2 Adaptability

The need to switch between two architecturally different systems, and to migrate the Course Hub's data store mid-project, strengthened the ability to adapt quickly to changing technical requirements and to learn unfamiliar tools such as n8n and Supabase.

2.8.2.3 Problem Solving

Debugging issues such as database instability, CORS errors, and imperfect OCR output sharpened analytical thinking and the ability to resolve complex problems systematically.

2.8.2.4 Project Management

Managing two parallel deliverables using iterative, milestone-based planning developed practical project management skills, particularly in prioritising the larger Course Hub while keeping the ALU application on track.

2.8.3 Implementation Management

This section describes how the implementation of the projects was managed to handle the workload effectively.

2.8.3.1 Task Prioritization

Work was broken into smaller milestones, with priorities set by deadline and dependency. The Course Hub was developed in phases, beginning with the database schema and catalogue module, followed by plan curation and finally the credit-transfer module.

2.8.3.2 Iterative Development

Iterative delivery and incremental testing ensured consistent progress. The Course Hub's versioned releases, tracked in its change log from version 4 through version 7.12, provided a clear record of how the system evolved without discarding completed work.

2.8.3.3 Documentation

Detailed documentation of each system, including architecture, database schema, API endpoints, and full use case specifications, ensured traceability and ease of future maintenance.

2.9 Summary of Chapter

Chapter 2 documented the two systems that carried the placement: the UTM Course Hub, an admin-facing full-stack application serving the Faculty of Computing, and the ALU application, a browser-only gamified platform serving Perpustakaan Awam Johor. Building the two side by side stretched skills across full-stack orchestration, database schema design, workflow automation and framework-free frontend engineering, and gave two very different pictures of what shipping software to a live user actually looks like.

Despite challenges such as database migration, integration errors, and imperfect OCR output, these obstacles facilitated personal growth and strengthened problem-solving ability. The use of modern tools such as n8n, Supabase, pdf.js, and Tesseract.js enhanced technical proficiency, while close collaboration with the supervisor fostered a deeper understanding of professional project execution. This chapter has detailed the projects, methods, tools, timeline, and challenges of the training, providing a strong foundation for the reflections presented in the next chapter.

CHAPTER 3

OVERALL INFORMATION OF INDUSTRIAL TRAINING

3.1 Introduction

Chapter 3 steps away from the mechanics of the two projects and reflects on what the placement produced in a broader sense: the specific know-how absorbed from the supervisor and from working through the projects, an honest read on where the work went well, and a note on the areas that could still be sharpened after the placement ends.

3.2 Knowledge and Skills Gained

Most of what was learned during the placement came from a combination of direct supervision and the pressure of shipping working code. The supervisor, Dr. Shahliza Binti Abd Halim, helped keep the work anchored to what the eventual users actually needed. In practice this meant repeated cycles of clarifying requirements before writing code, thinking about the design before opening the editor, and revisiting decisions once a feature had been in front of the stakeholder for a while.

The most significant technical growth came from building the UTM Course Hub as a complete full-stack system. Designing the Supabase schema, orchestrating the n8n backend, and connecting them to a vanilla JavaScript frontend required holding the entire architecture in mind at once, a form of end-to-end thinking that academic exercises rarely demand. The credit-transfer module further developed skills in client-side document processing and string-matching algorithms.

The ALU application contributed a complementary set of skills, demonstrating how a useful system can be delivered under tight architectural constraints using only the browser's native capabilities. Reference materials such as the official documentation for n8n, Supabase, pdf.js, and Tesseract.js were instrumental in advancing these skills, while writing the use case specifications for both systems strengthened the discipline of formal requirements analysis.

3.3 Constructive Comments on Overall Task Performance

Looking back on the twenty weeks, the training clearly stretched both technical and non-technical capability. The two projects showed the ability to move a concept all the way to a delivered product for two different stakeholders, the Faculty of Computing and Perpustakaan Awam Johor. Features such as the credit-transfer prediction pipeline and the ALU game module were not part of the initial specification, and were introduced after conversations with the supervisor about what would actually be useful, which is a habit worth keeping.

One notable strength was adaptability in learning new tools such as n8n, Supabase, pdf.js, and Tesseract.js, and in migrating the Course Hub's data store mid-project without losing momentum. Strong problem-solving ability was evident in resolving database instability, CORS and JSON parsing errors, and imperfect OCR output.

There remain areas for improvement. Time management was occasionally challenged by the need to advance two systems in parallel and by unexpected technical setbacks such as the database migration. More frequent intermediate reviews and earlier feedback cycles with end users could have surfaced certain design limitations sooner. Continued attention to documentation clarity and presentation would further strengthen communication with stakeholders.

3.4 Chapter Summary

This chapter reflected on the industrial training experience, summarising the knowledge and skills gained while evaluating task performance and areas for improvement. It highlighted technical expertise acquired in full-stack web development, database design, workflow automation, frontend engineering, and OCR-based document processing, alongside professional skills such as problem-solving, adaptability, and communication. Guidance from the supervisor and the use of official documentation played a crucial role in navigating challenges and delivering both the UTM Course Hub and the ALU application. The chapter concluded by recognising significant technical growth while identifying time management and stakeholder communication as areas for further development.

CHAPTER 4

CONCLUSION

4.1 Introduction

Chapter 4 closes the report by weighing the placement as a whole: what was actually delivered, where the difficult stretches sat and how they were dealt with, and what should carry forward into future work. The two delivered systems, the UTM Course Hub and the Active Library User (ALU) application, are used as the concrete evidence, since between them they cover full-stack backend orchestration, relational schema design, workflow automation and browser-native document processing.

4.2 Overall Achievement of the Industrial Training

The placement produced two distinct working systems in the hands of their intended users, each with its own set of measurable achievements. The two subsections that follow detail the specific outcomes of each project separately.

4.2.1 UTM Course Hub

The UTM Course Hub was delivered as a live multi-user full-stack administrative web application for the Faculty of Computing. It comprises a vanilla JavaScript single-page frontend, a backend of twenty-two n8n Cloud workflow endpoints exposing REST webhooks, and a Supabase Postgres database of around one hundred and forty course records across six relational tables secured by row-level security. Every mutating endpoint verifies a Supabase Auth JSON Web Token and enforces role-based access for administrator and student roles.

On top of this foundation, the system delivers catalogue management, study plan curation with printable UTM-branded reports, an administrator-only credit-transfer evaluation module powered by Google Cloud Vision OCR and a Groq large-language-model matcher against a faculty guideline table, a Transfer Records history with approved-only and full print modes, and an integrated chat assistant built on a LangChain agent with twelve database-backed tools. The Course Hub was

accompanied by a formal use case specification covering thirty-seven use cases across the entire feature set.

4.2.2 Active Library User (ALU) Application

The Active Library User (ALU) application was delivered as a self-contained frontend-only gamified web platform for Perpustakaan Awam Johor, targeting primary school children. The application runs entirely in the browser using localStorage as its sole persistence layer, deliberately chosen because the library had no server infrastructure to host a backend.

The delivered features cover member registration with avatars and a reward system built around points, levels and eight collectible badges; book donation and reuse workflows; a fifteen-book Dewey Decimal catalogue with search and a sorting activity; five progressively unlocked learning games; a leaderboard populated from a representative dataset; and a full English/Bahasa Malaysia toggle persisted across sessions. The ALU application was accompanied by its own formal use case specification covering sixteen use cases.

4.3 Issues and Challenges During Industrial Training

A handful of difficulties genuinely shaped the placement rather than passing without incident. The largest was the reliability of the original MongoDB store behind the Course Hub, which was replaced mid-placement by a Supabase Postgres cluster and forced a redesign of every backend endpoint around a relational schema. Alongside this, the first attempts to wire the frontend to the n8n backend surfaced cross-origin misconfiguration and empty-body JSON parsing bugs that had to be worked through before the two halves of the system could speak to each other reliably.

The credit-transfer module presented an ongoing technical challenge in achieving reliable code extraction from imperfect scans, which was addressed by migrating the primary Optical Character Recognition path to Google Cloud Vision and by introducing a Groq large-language-model matcher that reads the transcript against a faculty guideline table. A further deployment-time challenge emerged after the Course Hub was hosted on n8n Cloud, whose Content Security Policy sandbox blocks browser storage and forced a rebuild of the authentication layer against the Supabase Auth REST endpoints. Finally, advancing two systems of differing complexity within

a single placement made time management demanding, requiring careful prioritisation of the larger Course Hub while keeping the ALU application on schedule.

4.4 Conclusion

The two delivered systems together illustrate the breadth of the placement. The UTM Course Hub is a multi-user, full-stack, cloud-hosted application: a vanilla JavaScript frontend, twenty-two n8n Cloud workflow endpoints, and a Supabase Postgres database with row-level security and JSON Web Token authentication, extended with a Groq large-language-model matcher for credit-transfer prediction and a LangChain chat assistant. The Active Library User (ALU) application is its architectural opposite: a browser-only gamified platform with localStorage as its sole persistence layer, no backend, and a bilingual English/Bahasa Malaysia interface designed for primary school children in Perpustakaan Awam Johor. Between them, the two systems cover full-stack orchestration, relational schema design, workflow automation, authentication, large-language-model integration, and framework-free frontend engineering.

Taken as a whole, the placement translated the academic groundwork of the degree into something concrete: two live systems with real users. The challenges along the way, from the mid-placement database migration to the n8n sandbox limits, were more useful than any smooth path would have been, because each one demanded a specific technical decision and a way to justify it to the supervisor. The Course Hub and the ALU application also happened to sit at opposite architectural extremes, one full-stack and multi-user, one browser-only and single-device, which made the placement double as a practical comparison of how the same engineering discipline plays out in very different constraints.

More than any single feature, the placement reinforced how much of professional software work is adaptability and communication rather than raw coding. The database change and the imperfect OCR pipeline demanded specific technical fixes; the surrounding conversations, weekly reviews, and course corrections mattered just as much. The training as a whole leaves a much clearer picture of what a professional software role looks like in practice, and a solid base of skills to bring into the first full-time role after graduation.

REFERENCES

n8n GmbH (2024) 'n8n Documentation: Workflow Automation and Webhook Nodes'. Available at: https://docs.n8n.io (Accessed: May 2026).

Supabase Inc. (2024) 'Supabase Documentation: PostgreSQL Database and Auth'. Available at: https://supabase.com/docs (Accessed: May 2026).

Mozilla Developer Network (2024) 'Web Storage API: Window.localStorage'. Available at: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage (Accessed: October 2025).

Mozilla (2024) 'pdf.js: A General-Purpose PDF Rendering Library'. Available at: https://mozilla.github.io/pdf.js (Accessed: April 2026).

Project Naptha (2024) 'Tesseract.js: Pure JavaScript OCR for the Browser'. Available at: https://tesseract.projectnaptha.com (Accessed: April 2026).

Dice, L. R. (1945) 'Measures of the Amount of Ecologic Association Between Species', Ecology, 26(3), pp. 297–302.

PostgreSQL Global Development Group (2024) 'PostgreSQL 16 Documentation: JSON Types and Functions'. Available at: https://www.postgresql.org/docs (Accessed: May 2026).

Deterding, S., Dixon, D., Khaled, R. and Nacke, L. (2011) 'From Game Design Elements to Gamefulness: Defining Gamification', Proceedings of the 15th International Academic MindTrek Conference, pp. 9–15.

OCLC (2024) 'Introduction to the Dewey Decimal Classification'. Available at: https://www.oclc.org/en/dewey.html (Accessed: May 2026).

APPENDICES

APPENDIX A: Gantt Chart of Internship Timeline

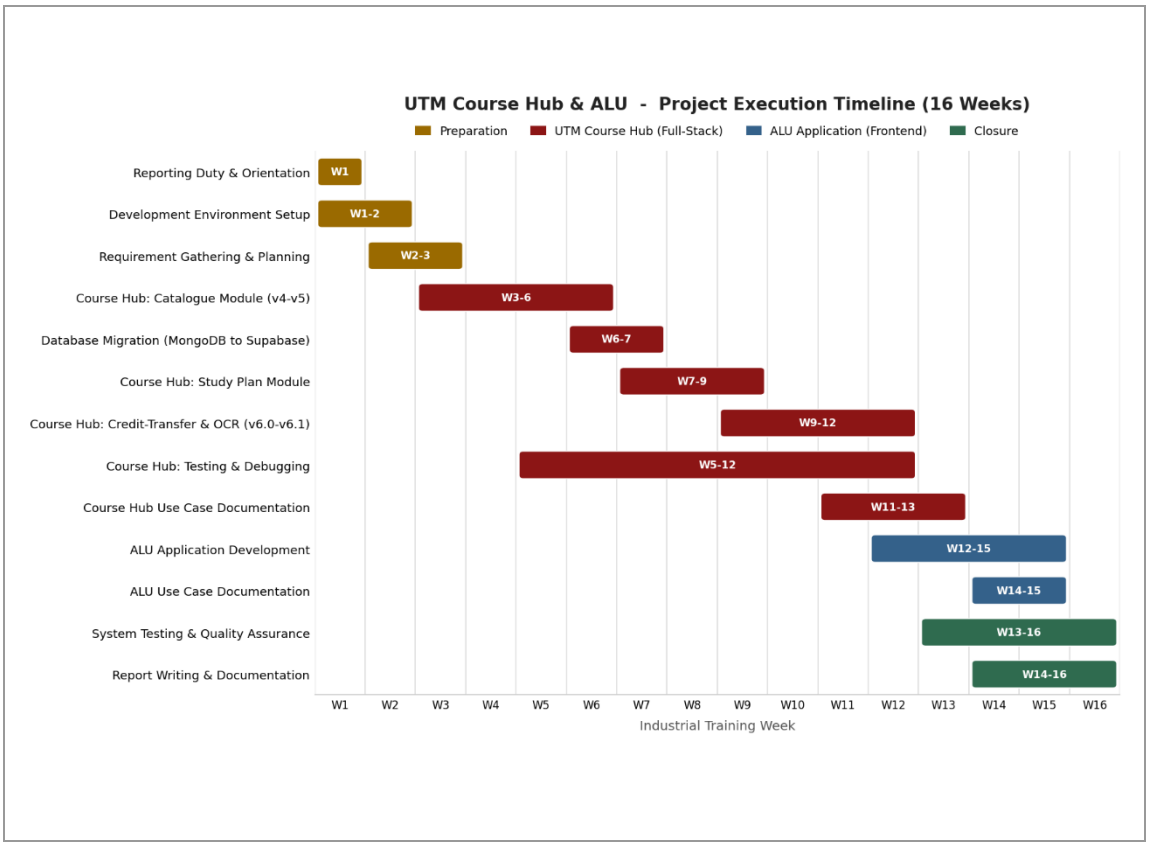


Figure A.1 Gantt Chart of Internship Timeline

APPENDIX B: Gantt Chart of UTM Course Hub Development

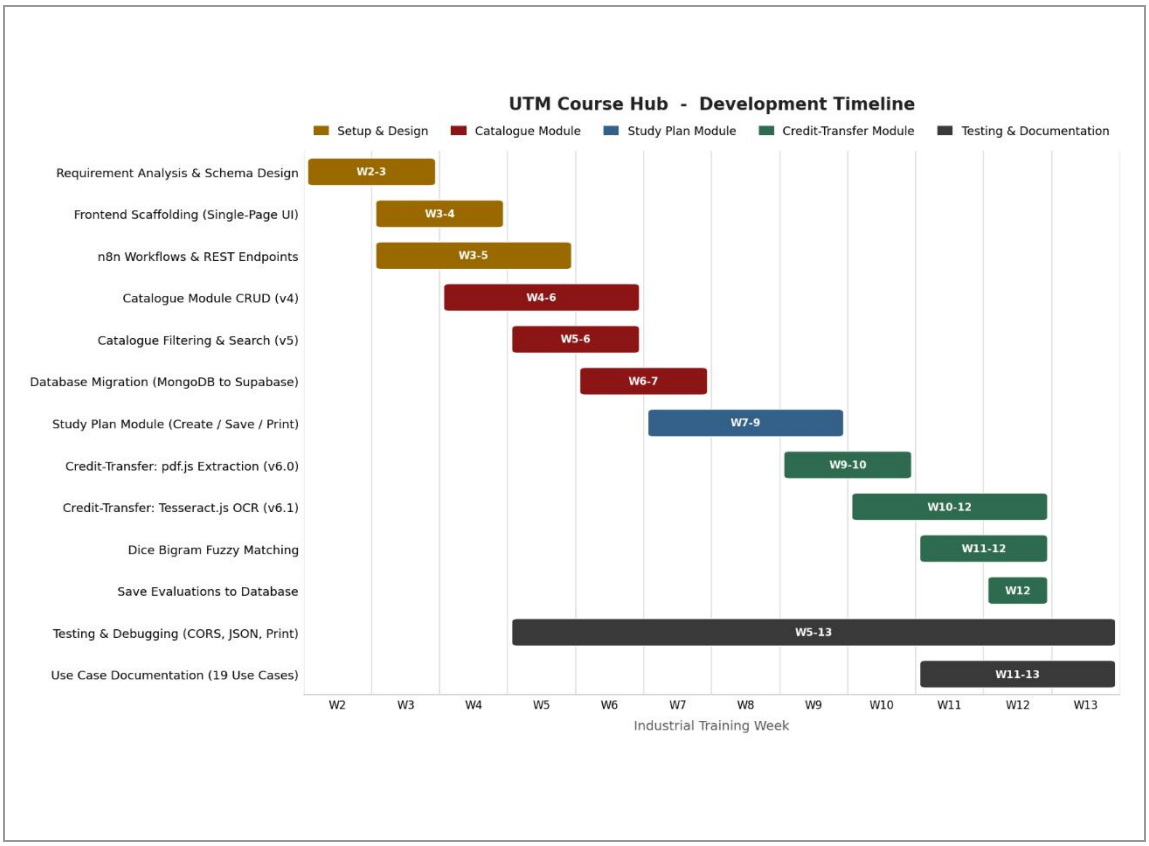


Figure B.1 Gantt Chart of UTM Course Hub Development

APPENDIX C: Gantt Chart of ALU Development

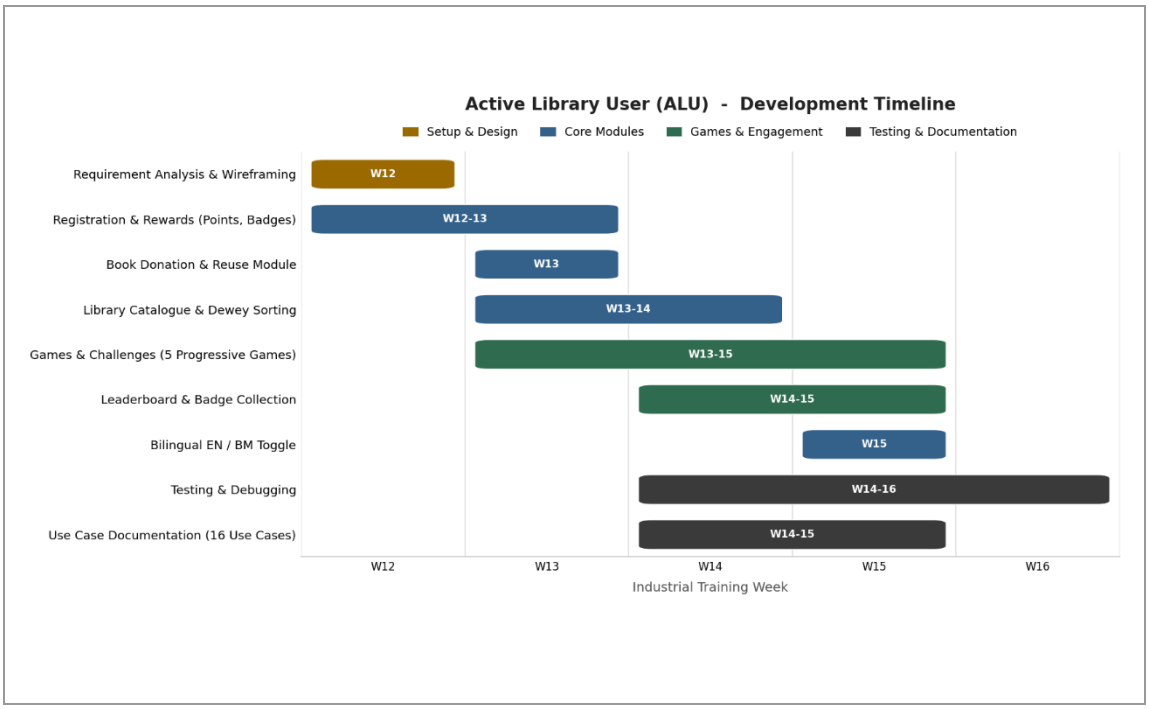


Figure C.1 Gantt Chart of ALU Development

APPENDIX D: Industrial Training Achievements & Checklists

Sekretariat Latihan Industri Sekolah Komputeran, Universiti Teknologi Malaysia, 81310 SKUDAI, JOHOR
Fax: 07-5565044 Tel: 07-5532008

INDUSTRIAL TRAINING ACHIEVEMENTS

(This form must be filled by student and must be attached in the Industrial Training report)

Student's Name : AHMED KHAIRY ABDELLATIF ABDELSALAM

Organisation : SOFTWARE ENGINEERING RESEARCH GROUP (SERG), UTM

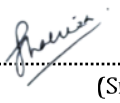
No.	Task (List all tasks have been completed)	Month of Task Achieved				
		Month 1	Month 2	Month 3	Month 4	Month 5
1	Orientation, development environment setup, requirement gathering and system design for the UTM Course Hub	✓				
2	Develop the UTM Course Hub core modules (course catalogue and study plan curation) and migrate the data store from MongoDB to Supabase Postgres	✓	✓	✓		
3	Develop the credit-transfer evaluation module, evolving from client-side OCR to Google Cloud Vision with a Groq large-language-model matcher and the Transfer Records history			✓	✓	✓
4	Extend the Course Hub into a multi-user application with Supabase Auth (JWT), role-based access, and an integrated chat assistant built on a LangChain agent			✓	✓	✓
5	Design and develop the Active Library User (ALU) application for Perpustakaan Awam Johor: frontend-only, localStorage, gamified, bilingual				✓	✓
6	System testing, quality assurance, and formal use case documentation for both systems				✓	✓
7	Prepare the industrial training report and final handover documentation					✓

Deliverable / Training reflection *(Outcomes that have been achieved)*
• Delivered the UTM Course Hub v7.12, a multi-user full-stack administrative web application for the Faculty of Computing (vanilla JavaScript frontend, twenty-two n8n Cloud workflows, Supabase Postgres with six relational tables secured by row-level security, JSON Web Token authentication for administrator and student roles).

- Delivered the credit-transfer evaluation module powered by Google Cloud Vision OCR and a Groq large-language-model matcher against a faculty guideline table, together with a Transfer Records history offering approved-only and full print reports.
- Delivered the Active Library User (ALU) application for Perpustakaan Awam Johor, a frontend-only gamified web platform with member registration, book donation and reuse, catalogue browsing, five interactive games, a leaderboard, and a bilingual English/Bahasa Malaysia interface.
- Authored formal use case specifications covering thirty-seven use cases across both systems, together with a system overview and hosting brief.
- Deployed both systems: the Course Hub on n8n Cloud with a Supabase-hosted database, and the ALU application as a static frontend runnable from any web host.

Student Signature:  Date: 15/7/2026

Approval

Organizations' Supervisor:	Faculty Supervisor:
	
(Signature)	(Signature)
Name: Dr. Shahliza Binti Abd Halim	Name: Ts. Dr. Nor Azizah Binti Sa'adon
Date: 16/07/2026	Date: 16/07/2026

APPENDIX E: Industrial Training Checklists (Placement)

No.	Activities/Tasks	Tick (✓)	Endorse by and date
1.	Report Duty To The Organization Approved by faculty	✓	13/3/2026
2.	Perform Report Duty Verification on ITS, verified by organization supervisor.	✓	13/3/2026
3.	Contact faculty supervisor to inform the job scope and organization information	✓	20/3/2026
4.	Update of Industrial Training site (address). Inform faculty supervisor and JKL, if any changes.	-	
5.	Updating Industrial Training Logbook online – daily basis	✓	
6.	Ensure that organization supervisor able to login to ITS successfully (Organization supervisor get ITS userid and password).	✓	
7.	Faculty Supervisor Visit. Date (physical): 15/7/2026	✓	15/7/2026
8.	Industrial Training Presentation.	✓	15/7/2026
9.	Performance evaluation by organization supervisor in ITS	✓	17/7/2026
10.	Submission of Industrial Training Logbook.	✓	13/7/2026
11.	Submission of Industrial Training Report with checklist and achievement form as Appendix.	✓	16/7/2026
12.	Fill in Industrial Training Performance Evaluation by student in ITS.	-	Inaccessible in ITS
13.	End Industrial Training	✓	24/7/2026

Note:

1. Italic activities are optional depending on student situation.

IMPORTANT: This checklist must be put as attachment in the industrial training report.

APPENDIX F: UTM Course Hub Use Case Model

The following diagram presents the complete use case model for the UTM Course Hub version 7.12, grouped into subsystems and annotated by role. Solid connectors indicate access available to a student; dashed connectors indicate access restricted to an administrator.

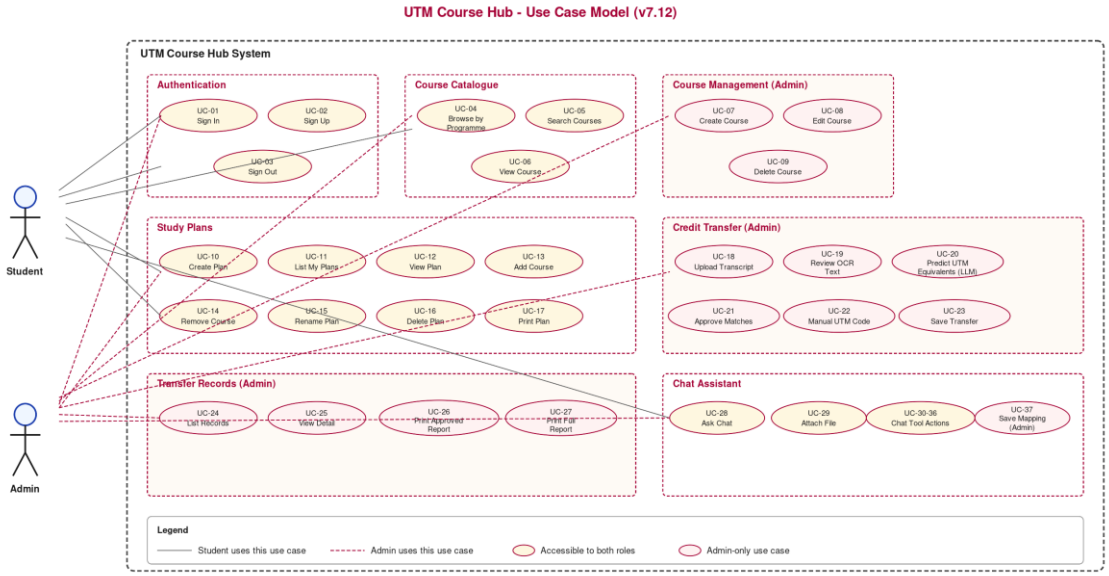


Figure F.1 UTM Course Hub Use Case Model (Appendix)