
# UNIVERSITY OF GREENWICH

Final Year Project (COMP1682)

# INDIVIDUAL ASSIGNMENT

## Greenwich University Management System Project

| | |
|---|---|
| **Student Name** | VU THANH TRUONG |
| **Student ID** | 001479794 |
| **Student Group** | TCH2702 |
| **Programming** | Global Personal Competency |
| **Academic Year** | 2025 - 2026 |
| **Instructor** | NGUYEN HUU CHU |
| **Submission Date** | 29 November 2025 |

# TABLE OF CONTENTS

# TABLE OF FIHURES

# 1. INTRODUCTION

## 1.1. Motivation

The motivation for this project arises from the growing mismatch between the increasing complexity of academic operations at Greenwich Vietnam and the fragmented set of information systems currently used to support them. As an international joint programme, Greenwich Vietnam must simultaneously satisfy local regulatory requirements, internal management needs and quality assurance standards from the University of Greenwich in the UK, while also meeting the expectations of students and parents for transparent, up-to-date academic information. At present, these demands are handled by separate platforms such as the Academic Portal (AP), Course Management System (CMS) and Faculty Learning Management (FLM), each with its own interface, data model and workflow. This fragmentation forces users to log into multiple systems, repeat similar tasks in different places and manually cross-check information such as timetables, grades, tuition balances or support requests. In practice, this not only wastes time and increases operational and maintenance costs, but also creates confusion, inconsistent data and a poor user experience that can negatively affect student satisfaction and the institution's image. From a practical perspective, a unified web-based platform that consolidates the core functions of AP, CMS and FLM into a single, role-based environment has the potential to significantly simplify daily work for students, lecturers, academic staff and parents, reduce duplication of effort, improve the accuracy and consistency of data, and provide a more intuitive, mobile-friendly gateway to academic services. It also creates a more solid foundation for monitoring learning progress, providing timely support and making data-informed decisions at programme and institutional levels. At the same time, the project is academically and technically significant because it treats Greenwich Vietnam as a real-world case study for the design and implementation of an integrated academic management system using contemporary software engineering principles. The work requires systematic requirements analysis, modelling of academic processes and entities, and the proposal of a modular, scalable architecture that can support future extensions such as online examinations, blended learning, advanced learning analytics or integration with external services. Technically, the project provides an opportunity to apply and evaluate concepts such as layered architecture, domain-driven design, security and role-based access control, RESTful APIs and cloud deployment in a realistic higher-education context. By addressing both institutional needs and engineering challenges, the project not only aims to deliver a useful prototype for Greenwich Vietnam but also contributes to the broader discussion on how universities in emerging educational markets can evolve from isolated legacy systems to coherent, sustainable digital ecosystems for teaching, learning and academic administration.

## 1.2. Background & Problem Statement

The context of this project is the digital infrastructure currently supporting teaching, learning and academic administration at Greenwich Vietnam, an international joint programme operating in partnership with the University of Greenwich (UK). Over time, the institution has developed and adopted several separate web-based systems to serve different aspects of its operations, most notably the Academic Portal (AP) for student records, timetables and financial information, the Course Management System (CMS) for managing course content and learning activities, and the Faculty Learning Management (FLM) system for lecturers' teaching-related tasks. Each of these platforms has been built or configured with its own interface, data structures and workflows, and they are maintained as largely independent systems. As a result, academic and administrative processes such as enrolment, timetable management, attendance tracking, assessment, tuition payment, student support and reporting are distributed across multiple environments, requiring users to navigate between different URLs, logins and interaction patterns in their daily work.

Within this context, several specific problems have become increasingly apparent. First, the separation of AP, CMS and FLM leads to fragmented and sometimes inconsistent information: data about the same student, class or subject may be updated at different times in different systems, forcing staff to perform manual reconciliation and increasing the risk of errors. Second, students, lecturers and staff are required to sign in to multiple portals to complete tasks that are logically related (for example, checking a timetable in one system, accessing learning materials in another and confirming attendance or grades in a third), which is inconvenient, time-consuming and confusing, especially for new users. Third, this fragmentation makes it difficult for academic managers and support units to obtain a holistic view of learner progress, workload and support needs, because data is scattered across systems that do not provide a unified reporting layer. Finally, maintaining and evolving three separate platforms increases operational complexity and cost for the IT team, and slows down the institution's ability to introduce new features or respond to changing requirements. These issues affect a wide range of stakeholders: students who need clear and reliable access to academic and financial information; lecturers who must manage courses, assessments and learner engagement; programme coordinators and academic staff who oversee delivery and quality; support and finance staff who manage tuition, scholarships and student cases; IT staff responsible for system maintenance and data integrity; and parents or guardians who require accurate information about students' progress. Together, these problems define the core background and problem statement for the proposed development of a unified management system.

## 1.3. Project Domain

The proposed project lies within the **Education** domain, specifically at the intersection of higher education management and educational technology (EdTech). It focuses on the design and implementation of a web-based academic management platform that integrates multiple operational areas which are traditionally handled by separate systems. These areas include student enrolment and registration, programme and curriculum structures, timetable and class scheduling, attendance tracking, assessment and grading, academic records, tuition and scholarship information, lecturer evaluation, communication between stakeholders, and student support case management. From a domain perspective, the project addresses the growing need for universities to move from isolated, legacy portals toward integrated, user-centred digital environments that can support both day-to-day operations and longer-term quality assurance and decision-making. It aligns with broader trends in EdTech where institutions aim to provide seamless digital experiences, reduce administrative friction and create a consistent data backbone for teaching, learning and academic governance.

In terms of application scope, the system is primarily designed for **Greenwich Vietnam**, an international joint programme operating in collaboration with the University of Greenwich (UK). Within this context, the platform is intended to be used by a wide range of stakeholders: students who need to access timetables, course materials, grades and financial information; lecturers who manage classes, assessments, attendance and interaction with learners; academic and programme coordinators who oversee delivery, compliance and academic quality; support and administrative staff who handle tuition payments, scholarships, student records and support tickets; IT staff who maintain the digital infrastructure; and, to some extent, parents or guardians who require transparent information about students' progress and obligations. Although the initial implementation and evaluation are scoped to Greenwich Vietnam, the underlying problem and solution pattern are not unique to a single institution. Many universities, colleges and training organisations—especially those operating joint programmes or multi-campus models—face similar challenges with fragmented systems and disconnected user experiences. Therefore, the project can be viewed as a concrete case study with potential applicability to other higher education organisations that wish to unify their academic portals, improve operational efficiency and deliver a more coherent digital ecosystem for all participants in the teaching and learning process.

# 1.4. Proposed Technologies / Methods

The proposed academic management system will be developed as a full-stack Java web application using Spring Boot as the core framework, with Thymeleaf serving as the server-side template engine instead of a separate single-page application approach. The backend will leverage Spring Boot to implement controllers, services, business logic, and database interactions through Spring Data JPA and Hibernate. The project will be built and managed using Maven or Gradle, incorporating standard Spring modules including Spring Web MVC, Spring Data JPA, Spring Security, and Spring Validation, and Thymeleaf. The presentation layer will be rendered directly on the server using Thymeleaf templates combined with HTML5, modern CSS (via Bootstrap 5 or Tailwind CSS delivered through WebJars or CDN), and vanilla JavaScript or lightweight libraries (e.g., Alpine.js or HTMX) to achieve a responsive, role-aware user interface that works seamlessly across desktop, tablet, and mobile devices without requiring a separate frontend build pipeline. Form handling, data binding, and validation will follow the classic Spring MVC pattern using @ModelAttribute, @Valid, and BindingResult, allowing complex forms (e.g., student enrolment, assessment grading, timetable adjustment, and financial transactions) to be processed and re-rendered on the same Thymeleaf template with error messages displayed inline, providing excellent user experience while maintaining strong server-side control (Bauer and King, 2022; Cosmina et al., 2021). The application will adhere to a clean layered architecture (presentation → controller → service → repository → domain model) and incorporate selected Domain-Driven Design principles by organising bounded contexts around core academic domains such as enrolment, timetabling, assessment, finance, and student support (Evans, 2004; Vernon, 2013). Authentication and authorisation will be implemented using Spring Security with traditional form-based login and Role-Based Access Control (RBAC). Thymeleaf views will integrate Spring Security tags (sec:authorize, sec:authentication) to conditionally render UI elements according to user roles. Session management will rely on standard servlet sessions or Spring Session with Redis if clustering is required in future (Gutierrez, 2020). Analysis and design phases will employ standard software engineering artefacts including use case diagrams, class diagrams, sequence diagrams, activity diagrams, and entity-relationship (ER) models to ensure requirements are accurately captured and translated into implementation (Dennis et al., 2018). The relational database will remain MySQL, with a normalised schema representing key entities: campuses, programmes, curricula, course instances, students, lecturers, timetables, assessments, grades, and financial records. Core business algorithms (timetable conflict detection, prerequisite validation, attendance and grade calculation, financial clearance checks) will be encapsulated within service classes and invoked directly from controllers or Thymeleaf expressions when immediate display is needed. Development will follow an iterative and incremental Agile-inspired approach, delivering working prototypes of high-priority features (login, personal timetable view, class roster management, grade entry) early for stakeholder feedback (Beck et al., 2001). Supporting tools will include Git with GitHub/GitLab for version control and issue tracking; Figma for UI wireframing and prototyping; Postman for occasional API contract testing; JUnit 5, Mockito, and Spring Test for comprehensive unit and integration testing; and Docker/Docker Compose to containerise the Spring Boot application and MySQL database, ensuring identical behaviour across development, testing, and demonstration environments (Turnbull, 2018). IntelliJ IDEA or Visual Studio Code with Spring Tools will serve as the primary IDE, complemented by Logback logging and Spring Boot Actuator for basic monitoring. In summary, the adoption of Spring Boot

with Thymeleaf and @ModelAttribute results in a robust, maintainable, server-side rendered Java web application that is particularly well-suited for a large-scale capstone project, offering rapid development velocity, strong typing, excellent IDE support, and straightforward deployment while demonstrating mastery of enterprise Java patterns.

## 2. Key Phrases

[Unified academic management platform for Greenwich Vietnam]

[Integration of Academic Portal (AP), Course Management System (CMS) and Faculty Learning Management (FLM)]

[Role-based web portal for students, lecturers, staff and parents]

[Centralised academic, financial and student support information]

[Improved user experience and operational efficiency in higher education]

[Scalable and maintainable EdTech architecture for future extensions]

# 3. AIM & OBJECTIVES

## 3.1. Aim (Overall Goal)

This project aims to develop a unified, web-based academic management platform that integrates the existing Academic Portal (AP), Course Management System (CMS) and Faculty Learning Management (FLM) into a single role-based environment in order to reduce system fragmentation, improve the accuracy and consistency of academic and financial data, streamline and simplify day-to-day academic and administrative tasks, and enhance the overall digital experience for students, lecturers, academic and support staff, and parents at Greenwich Vietnam.

## 3.2. Objectives (Specific Goals)

**Objective 1: Develop a unified academic management platform**

Design and implement a web-based system that consolidates the core functions of the existing Academic Portal (AP), Course Management System (CMS) and Faculty Learning Management (FLM) into a single, role-based platform tailored to the operational context of Greenwich Vietnam.

**Objective 2: Improve usability and access to academic information**

Provide an intuitive, responsive user interface that allows students, lecturers, academic/support staff and parents to easily access timetables, course information, assessments, academic records, tuition details and support services through a single login, reducing confusion and duplicated effort.

**Objective 3: Enhance data consistency and support informed decision-making**

Establish a centralised data model and integration mechanisms that minimise discrepancies between academic, financial and support information, while enabling basic reporting and monitoring of learner progress, workload and key academic processes for management and quality assurance purposes.

**Objective 4: Build a scalable and maintainable technical foundation**

Apply appropriate software engineering principles and modern web technologies (e.g. layered architecture, RESTful APIs, secure authentication/authorisation) to create a system that is maintainable, extendable and capable of integrating additional modules or services in the future as Greenwich Vietnam's digital ecosystem evolves.

# 4. ACTIVITIES & DELIVERABLES

The project will be carried out through a series of structured and iterative activities that are fully aligned with the stated aim and objectives, with each activity producing tangible deliverables that demonstrate clear progress and serve as evidence of the work completed. The project will begin with comprehensive requirements gathering and analysis involving all key stakeholder groups at Greenwich Vietnam – including students, lecturers, academic coordinators, administrative and support staff, IT personnel, and parents – together with an in-depth study of the current workflows and data structures of the existing Academic Portal (AP), Course Management System (CMS), and Faculty Learning Management (FLM) systems, resulting in a formal requirements specification document and detailed process descriptions that highlight current pain points and desired improvements. Building on these foundations, the subsequent phase will focus on designing the overall system architecture and normalized database schema for the unified platform, defining the core modules, entities, relationships, integration points, and critical business rules that must be enforced across academic, financial, and student-support data, delivering high-quality architecture diagrams, module diagrams, a fully normalized database schema, and detailed documentation of essential business rules such as timetable conflict detection, data integrity constraints, and basic reporting requirements. The implementation phase will then develop the complete system using Java Spring Boot with the classic Spring MVC pattern, following a clean layered architecture (Controller – Service – Service – Repository – Domain/Entity) and employing server-side rendering with Thymeleaf templates, Spring Model Attributes, and form-backing objects to provide a seamless, role-based, single-sign-on web experience that unifies academic, financial, and support functionalities. Key development activities will include creating dedicated controllers and model attributes for each user role, building responsive and reusable Thymeleaf templates with layout dialects and fragments, implementing robust authentication and fine-grained authorization using Spring Security (form-based login, role-based and method-level security with @PreAuthorize), applying comprehensive validation at both entity and form levels (Hibernate Validator + BindingResult), and enforcing complex business rules such as schedule conflict checks, grade constraints, tuition status validation, and support-ticket workflows, ultimately producing fully functional modules for user & role management, timetable & class management, course & assessment viewing, financial overview, and student support case tracking, accompanied by clear documentation of controllers, model attributes, and Thymeleaf templates. Once a stable prototype is available, extensive functional testing and targeted usability testing will be conducted with representative users from all roles, generating detailed test plans, test reports, and user feedback records that drive iterative refinements to the interface, workflows, and performance. Finally, the project will deliver a polished, production-ready version of the unified academic management platform packaged as an executable JAR and optionally containerized with Docker for easy deployment and demonstration, together with complete technical documentation (covering system architecture, database schema, key controllers and model attributes, Spring Security configuration, and deployment instructions) and user-oriented documentation (quick-start guides and role-specific usage instructions with screenshots of the Thymeleaf interfaces), thereby providing conclusive evidence that the project has successfully progressed from thorough problem analysis to the delivery of a modern, cohesive, server-rendered web application built entirely on Spring Boot MVC, Thymeleaf, and Model Attributes that effectively eliminates the fragmentation of existing tools at Greenwich Vietnam.

UNIVERSITY OF GREENWICH

# 5. PLAN

## 5.1. Work Breakdown Structure (WBS)

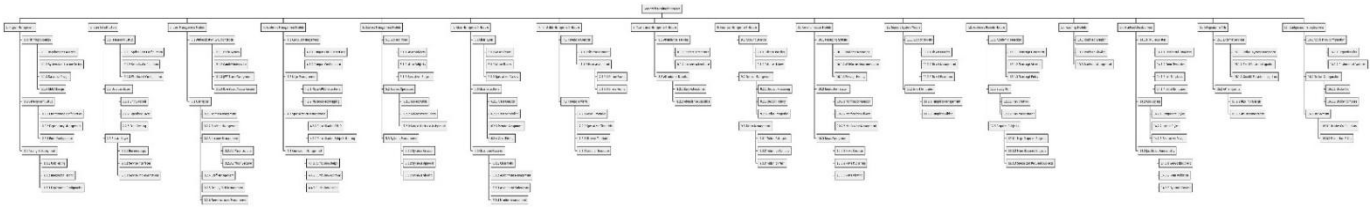

*Figure 1 Work Breakdown Structure (WBS)*

The Work Breakdown Structure for the Greenwich Graduation Project is a complete university management system developed on the Spring Boot framework, organized into 16 major work packages that systematically cover every phase from initial planning to production deployment. Package 1 – Project Management covers requirements gathering and analysis, overall system architecture design, database schema design, UI/UX design, Maven-based development environment setup, and establishment of testing and deployment pipelines. Package 2 – Core Infrastructure establishes the technical foundation, including Spring Boot application configuration, Spring Security implementation with role-based access control, WebSocket integration for real-time functionality, entity modeling, JPA repositories, initial data seeding, and a cleanly separated service layer. Package 3 – User Management Module delivers a JWT-based authentication system, OAuth2 integration, and comprehensive user role management for administrators, students, primary lecturers, guest lecturers, staff, deputy staff, and parents, each with distinct permissions. Package 4 – Academic Management Module handles campuses, academic programs, majors, specializations, and full curriculum frameworks with complete approval workflows. Package 5 – Subject Management Module supports classification of core subjects, elective subjects, and specialized courses, provides full CRUD operations, prerequisite management, lecturer assignment, and detailed syllabus handling. Package 6 – Class Management Module enables three class types, schedule creation, member assignment, lecture material posting, assignment distribution and submission, and class roster management. Package 7 – Timetable Management Module offers timetable generation, time-slot and classroom (physical or virtual) allocation, and schedule visualization from the perspective of programs, specializations, individual students, or lecturers. Package 8 – Attendance Management Module tracks attendance for both students and lecturers and produces detailed statistical reports. Package 9 – Financial Management Module manages student account balances, processes deposits through Stripe, maintains transaction histories, and calculates flexible tuition fees by campus and academic year. Package 10 – Communication Module includes a real-time chat system powered by WebSocket, push notification services, and internal news/announcement management. Package 11 – Support System Module provides ticket creation and tracking along and customizable email template management. Package 12 – Academic Records Module stores official transcripts, individual study plans, and monitors completion of required courses to meet graduation criteria. Package 13 – Roadmap Module facilitates long-term academic planning and progression tracking for students.

Package 14 – Frontend Development implements responsive HTML templates, CSS styling, client-side JavaScript with WebSocket connectivity, form validation, and dynamic UI effects. Package 15 – Integration & APIs incorporates Stripe payment processing, email services, OAuth2 providers, and delivers a fully documented RESTful API. Package 16 – Configuration & Deployment manages application properties, Docker containerization using Dockerfile and Docker Compose, and final production deployment to the Render platform. This structured approach ensures that every feature of a modern university management system is comprehensively addressed, with each module remaining independently maintainable while preserving strong integration, scalability, and long-term extensibility.

# 5.2. Gantt Chart (Timeline)

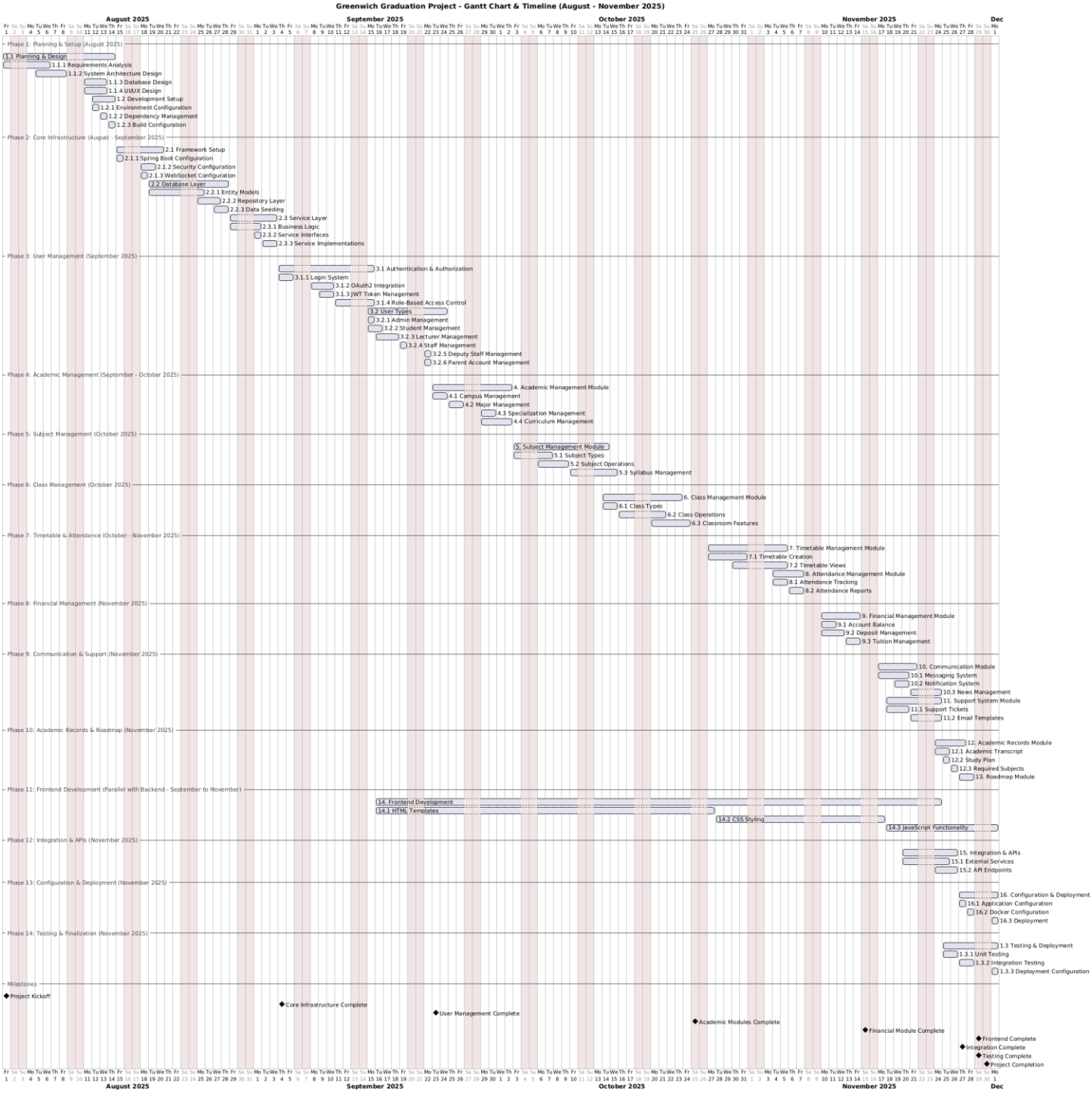


*Figure 2 Gantt Chart (Timeline)*

The Greenwich Graduation Project, a full-featured university management system built on Spring Boot, is scheduled for completion within a tight yet realistic 4-month timeline, running from August 1, 2025, to November 30, 2025. The project kicks off with the Planning & Setup phase during the first two weeks of August, delivering comprehensive requirements analysis, system architecture design, database schema, UI/UX prototypes, and full Maven-based development environment configuration by August 15. From mid-August to September 4, the team concentrates on establishing the Core Infrastructure, finalizing Spring Boot setup, Spring Security with RBAC, WebSocket for real-time features, entity models, repositories, data seeding, and a cleanly layered service architecture, creating a robust and secure backend foundation. September 2025 marks the intensive development sprint for critical modules: the complete User Management system featuring JWT authentication, OAuth2 integration, and differentiated role-based access for six user types (administrator, student, primary lecturer, staff, deputy staff, and parent) is delivered by mid-month, while parallel frontend development using responsive HTML templates, CSS frameworks, and real-time JavaScript begins in the third week. From late September through mid-October, the Academic Management (campuses, majors, specializations, and curriculum with approval workflows) and Subject Management (subject categorization, prerequisites, lecturer assignment, and syllabus management) modules are fully implemented. Starting mid-October, the team rolls out the operational core trio in rapid succession: Class Management (supporting theory, lab, and online classes with scheduling, materials, assignments, and rosters), Timetable Management (slot and room allocation with multi-view displays), and Attendance Management (tracking and reporting for students and lecturers), ensuring all essential academic functions are ready before November. The accelerated November phase focuses on completion and integration: Financial Management with Stripe payment gateway and flexible tuition calculation is finalized in the first week, Communication Module (real-time WebSocket chat, push notifications, and internal news) and Support System (ticketing and email templates) are completed by mid-month, followed immediately by Academic Records and long-term Roadmap modules. Frontend development runs concurrently from September and reaches full completion by late November, delivering a polished, responsive, and real-time-enabled user interface. The final week (November 24–30) is dedicated to end-to-end Integration & APIs (Stripe, SMTP services, OAuth2 providers, and fully documented RESTful endpoints), containerization with Docker and Docker Compose, production deployment to Render, and an intensive testing sprint covering unit, integration, and end-to-end tests. Key milestones are firmly set: project kickoff on August 1, core infrastructure ready by September 4, all academic modules completed by October 25, financial module delivered by November 15, and simultaneous frontend finalization, API integration, testing, and deployment targeted for November 29, with official project handover and go-live on November 30, 2025. This carefully orchestrated schedule, with strategic overlap between backend and frontend streams and prioritized delivery of high-impact modules, maximizes efficiency and guarantees on-time delivery of a high-quality, production-ready university management platform.

# 6. INITIAL REFERENCES

Bauer, C. and King, G. (2022) *Java Persistence with Hibernate*. 3rd edn. Manning Publications.

Beck, K. et al. (2001) 'Manifesto for Agile Software Development', Available at: https://agilemanifesto.org/ (Accessed: 29 November 2025).

Cosmina, I., Harrop, R., Schaefer, C. and Ho, C. (2021) *Pro Spring 6: An In-Depth Guide to the Spring Framework*. 6th edn. Apress.

Dennis, A., Wixom, B.H. and Roth, R.M. (2018) *Systems Analysis and Design*. 7th edn. Wiley.

Evans, E. (2004) *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley.

Gutierrez, F. (2020) *Pro Spring Boot 2*. 2nd edn. Apress.

Turnbull, J. (2018) *The Docker Book: Containerization is the new virtualization*. 2nd edn. Self-published.

Vernon, V. (2013) *Implementing Domain-Driven Design*. Addison-Wesley.