

Interim Report

Level 02

Learning Management System Software (LMS)

Team Laser

Team 'Laser' Group Members

Index No	Name
224112A	Kulasekera B.H.A.R
224074G	T. A. Hettige
224254J	R.Sharan
224256R	J.Dinuja
224236G	M.B.F. Bushra

Supervisors

Dr. S C Premaratne

Ms. A Pirapaharan

.....

.....

Faculty of Information Technology

University of Moratuwa

2024

Contents

Contents.....	2
Chapter 1 - Introduction.....	4
1.1 Introduction.....	4
1.2 Background and Motivation.....	5
1.3 Aim and Objectives.....	6
1.4 Problem in Brief.....	7
1.5 Proposed Solution.....	7
Chapter 2 - Existing Solution.....	9
2.1 Introduction.....	9
2.2 Similar Products.....	9
2.3 Summary.....	10
Chapter 3 - Technologies Adapted.....	11
3.1 Introduction.....	11
3.2 Front-end Technology.....	11
3.2.1 Next JS.....	11
3.3 Back-end Technology.....	12
3.3.1 SpringBoot.....	13
3.4 Database Technology.....	13
3.4.1 PostgreSQL Server.....	14
Chapter 4 - Proposed Solution.....	15
4.1 Introduction.....	15
4.2 Software Development Model.....	15
4.3 Users, Activities, Inputs, and Outputs.....	16
4.4 Process.....	23
4.5 Our Approach.....	25
4.5.1 Notification system.....	26
4.6 Project Management.....	26
4.7 Summary.....	27
Chapter 5 – Analysis and Design.....	28
5.1 Use Case Diagram.....	28
5.2 Activity Diagram.....	29
5.3 Use Case Diagram.....	70

5.4 Class Diagram.....	70
5.5 ER Diagram.....	70
5.5 Sequence Diagram of Student Interface.....	73
Chapter 6 – Implementation.....	80
6.1 Login Page.....	80
6.2 Administrator’s Dashboard.....	81
6.3 Lecturer’s Dashboard.....	82
6.4 Student’s Dashboard.....	83
6.5 Student’s Profile Page View.....	84
6.6 Student’s Password Changing Page View.....	84
6.7 Student’s Interface.....	85
6.8 Lecturer’s Interface.....	85
Appendix A – Individual Contribution.....	86
Appendix B – Action Plan.....	94
Chapter 7 – Reference.....	96

Chapter 1 - Introduction

1.1 Introduction

The **Mahapola Port & Maritime Academy (MPMA)** aims to modernize its educational infrastructure by transitioning from its current paper-based system to a sophisticated and efficient Learning Management System (LMS). The existing LMS, which is rarely used and fails to meet the academy's operational needs, will be replaced by a custom-designed platform leveraging contemporary technologies.

This project, undertaken by **IT - Group Laser**, is a collaborative effort to develop a user-friendly and feature-rich LMS tailored to MPMA's unique requirements. The system will integrate seamlessly with the proposed Enterprise Resource Planning (ERP) system, facilitating a unified platform for managing academic and administrative activities.

The proposed LMS will include functionalities such as course management, On-the-Job Training (OJT) tracking, exam and certification handling, and role-based access for different user categories, including administrators, lecturers, and students. By introducing a scalable and maintainable solution, this project aims to enhance the efficiency, accessibility, and overall educational experience at MPMA.

Furthermore, the LMS will provide MPMA with the flexibility to adopt cloud-based deployment and other emerging technologies, ensuring future adaptability and long-term usability, while the current deployment will be on a Virtual Machine in the MPMA On-Premise Servers. The project underscores the academy's commitment to leveraging digital transformation to meet the evolving demands of maritime education and training.

1.2 Background and Motivation

The Mahapola Port & Maritime Academy (MPMA) plays a key role in training and educating future maritime professionals. However, its current reliance on paper-based systems and an underutilized Learning Management System (LMS) limits its ability to deliver modern, efficient, and accessible learning experiences. This outdated approach leads to inefficiencies in managing courses, student progress, and administrative tasks.

With the increasing demand for digital transformation in education, MPMA recognizes the need for a modern LMS that integrates advanced technologies to improve productivity and learning outcomes. The shift from traditional methods to a streamlined, automated system will not only enhance user experiences but also align with global trends in maritime education and training.

The motivation for this project stems from the academy's vision to modernize its operations and provide a seamless learning experience for students and staff. By addressing current limitations and integrating the LMS with a proposed Enterprise Resource Planning (ERP) system, this project aims to set a foundation for a future-proof system that supports innovation and scalability.

Furthermore, as the maritime industry grows and evolves, MPMA requires a system that can adapt to new challenges, such as handling On-the-Job Training (OJT) management and improving exam and certification processes. This project is driven by a commitment to delivering a high-quality, user-friendly LMS that will transform how education and training are managed at MPMA.

1.3 Aim and Objectives

Aim:

- The primary goal is to create a user-friendly website that enables customers to design and purchase fully customizable 3D miniature frames. By focusing on personalization, the website will provide an intuitive platform for users to create unique and meaningful products, while ensuring a smooth shopping experience through reliable backend support.

Objectives:

- ✓ Create an easy-to-use LMS with role-based access for administrators, lecturers, and students.
- ✓ Create an easy-to-use LMS with role-based access for administrators, lecturers, and students.
- ✓ Add features for managing courses, assignments, exams, and certifications.
- ✓ Build a module to track and manage On-the-Job Training (OJT).
- ✓ Ensure the system is scalable and easy to maintain.
- ✓ To track customer satisfaction regarding the work done.
- ✓ Deploy the system on a Virtual Machine (VM) or cloud as needed
- ✓ Provide clear user guides and training for MPMA staff.

1.4 Problem in Brief

The Mahapola Port & Maritime Academy is facing challenges with its outdated paper-based system and underperforming Learning Management System (LMS). The current system is not capable of efficiently managing essential tasks like course management, student tracking, exams, certifications, and On-the-Job Training (OJT). This causes delays, increases administrative burden, and leads to a fragmented system where different processes are not connected. The academy needs a modern LMS that integrates all these functions into one platform, improves efficiency, and can grow with future needs.

1.5 Proposed Solution

The proposed solution is a modern Learning Management System (LMS) designed to streamline the management of courses, student progress, exams, certifications, and On-the-Job Training (OJT) at the Mahapola Port & Maritime Academy (MPMA). This LMS will replace the outdated paper-based system and the underperforming current LMS, providing a more efficient and user-friendly platform for both administrators and students.

The LMS will offer several key features, including course management, student progress tracking, and exam handling, making it easier for lecturers and administrators to manage academic activities. The system will also include an OJT management module, allowing the academy to track and evaluate students' practical training.

Key functionalities of the LMS include:

- **Course Management:** Enables lecturers to create, manage, and update courses, upload learning materials, assign tasks, and grade students based on Quizzes and Submissions.
- **Student Tracking:** Tracks student progress throughout courses and provides detailed reports on performance.
- **Exam and Certification Management:** Facilitates the creation, management, and grading of exams, and issues certificates to students upon course completion.
- **OJT Management:** Manages On-the-Job Training activities by scheduling, tracking,

and evaluating students' training experiences.

- **User Role Management:** Provides role-based access for administrators, lecturers, and students, ensuring secure and appropriate access to features.
- **ERP Integration:** Integrates with the MPMA's existing ERP system to streamline administrative tasks and ensure seamless data sharing.
- **Dashboard and Reporting:** Offers a dashboard for easy access to reports, performance data, and notifications about upcoming deadlines, events, and updates.

This LMS will improve the efficiency of managing academic activities, reduce administrative workload, and enhance the learning experience for students and lecturers. The system will be scalable, allowing for future updates and the possibility of cloud deployment, ensuring that MPMA's needs are met both now and in the future.

Chapter 2 - Existing Solution

2.1 Introduction

The development of a Learning Management System (LMS) for the Mahapola Port & Maritime Academy requires a thorough understanding of what is currently available in the market. By analyzing similar products, it becomes possible to identify their strengths, weaknesses, and areas they do not serve sufficiently. This chapter focuses on evaluating mature LMS platforms in order to identify their strengths and limitations. Such an analysis provides critical insights into how the proposed LMS can differentiate itself and better cater to the academy's unique operational and educational needs.

2.2 Similar Products

Today, many LMS platforms cater to multiple industries, including education, corporate training, and specialized areas like maritime studies. Common choices in this regard include Moodle, Blackboard, and Canvas; these are popular because they are in wide use and boast strong features. Moodle is particularly appreciated because it is open-source, customizable, and low in cost, making it a first choice for schools that have the skills to manage it. On the other hand, Blackboard is much stronger with its user-friendly interface and great integrations, though it's usually very costly to license. Canvas has a modern design and cloud-first approach that gives users accessibility and convenience but less flexibility in customization.

While these systems do have general features such as course management, user tracking, and assignment handling, they are not designed specifically with maritime education in mind. For example, none of these platforms handle On-the-Job Training (OJT) well or work easily with enterprise resource planning (ERP) systems that are very important for MPMA. Besides, the steep learning curves and costs of using these platforms may act as barriers for schools that want solutions tailored to their needs and budgets.

2.3 Summary

A review of the current LMS platforms presents their strengths and weaknesses in meeting specific needs of MPMA. While most of the functionality required by the maritime academies is provided through tools like Moodle and Blackboard, there is still not complete support by these for special maritime training needs, especially in OJT tracking and smooth integration with ERP. These gaps highlight the need for creating a special LMS that has advanced features and fits well with the academy's goals. By understanding the good and bad points of these solutions, the new system aims to provide a custom, effective, and expandable platform that helps MPMA achieve its digital transformation vision.

Chapter 3 - Technologies Adapted

3.1 Introduction

To build the proposed solution, various technologies are used. They can be divided into the following categories.

- Front-end technology
 - Next.js
- Back-end technology
 - SpringBoot
- Database technology
 - PostgreSQL

3.2 Front-end Technology

The front-end of the Learning Management System (LMS) is what users will see and interact with. We've chosen **Next.js** for the front-end development to make the system fast and easy to use.

Next.js is a tool that works with React to help build websites quickly. It makes the LMS load faster and work smoothly on all devices. With Next.js, we can easily add features and ensure that students, lecturers, and administrators have a great experience using the system.

3.2.1 Next JS

Next.js is a powerful framework built on top of React, designed for building fast and scalable web applications, especially for front-end development. It provides additional features like server-side rendering (SSR) and static site generation (SSG), which can significantly improve the performance and SEO of the Learning Management System (LMS).

One of the key advantages of Next.js is its ability to pre-render pages at build time or

on-demand. This helps ensure that the LMS pages load quickly, even for users with slower internet connections. By improving load times, Next.js enhances the overall user experience, which is crucial for an LMS where users will be frequently accessing course materials, assignments, and grades.

Next.js also makes it easier to structure and manage a large application like the LMS. The framework supports automatic code splitting, so only the necessary code is loaded for each page, improving performance. It also provides a seamless way to handle routing, which is essential for managing different parts of the LMS, such as courses, student profiles, and exam management.

Another benefit of using Next.js is its strong community and ecosystem. It has a wealth of resources, plugins, and tools that can help the development team build, test, and maintain the LMS efficiently. Since the development team may not have extensive experience with front-end development, this can be a significant advantage, allowing them to leverage the community's support and resources.

With the speed, flexibility, and robust developer support offered by Next.js, it is an ideal choice for building the front-end of the LMS, ensuring that the system is efficient, user-friendly, and easily maintainable in the long run.

3.3 Back-end Technology

The back-end of the Learning Management System (LMS) manages all the main functions of the system. To build the back-end, we have chosen **Spring Boot**.

Spring Boot is a simple and powerful tool for creating back-end applications. It helps us quickly set up the LMS's back-end, handling tasks like user logins, course management, and saving data. Spring Boot also makes it easy to connect to databases and other services, ensuring the system runs smoothly and can handle lots of information.

3.3.1 SpringBoot

Spring Boot is a popular choice for back-end development because it provides a powerful and efficient platform for building web-based applications. It is flexible and scalable, allowing developers to create applications that can run across different environments and handle growing amounts of data and users.

One of the key benefits of Spring Boot is that it comes with a wide range of built-in features and libraries, which make it easy for developers to handle common back-end tasks like database access, security, and handling HTTP requests. It also supports RESTful APIs, making it ideal for building modern web applications like the LMS and working with a mobile application in the future.

Spring Boot simplifies development by offering tools for building, testing, and deploying applications quickly. It has a strong ecosystem with a large developer community, ensuring that developers can find plenty of resources and support to build and maintain the LMS effectively.

3.4 Database Technology

The database is an important part of the system as it stores all the necessary data. For the LMS, we have chosen **PostgreSQL** as the database technology.

PostgreSQL is a reliable and efficient open-source database. It is good for handling large amounts of data, like student records, courses, and grades. It can perform complex tasks quickly, ensuring the LMS runs smoothly. PostgreSQL is easy to connect with web applications and provides strong security to keep the data safe, while allowing concurrent write operations.

3.4.1 PostgreSQL Server

PostgreSQL is a widely used open-source database management system (DBMS) known for its reliability, flexibility, and robust features. It is favored by many developers for building scalable and high-performance applications, including for web-based systems like the LMS.

Some key features of PostgreSQL are:

Reliability: PostgreSQL is known for its stability and reliability, making it an excellent choice for applications that require consistent uptime and data integrity.

Performance: PostgreSQL is designed to handle large amounts of data and complex queries efficiently. It is optimized for high performance, ensuring that data retrieval and manipulation are fast.

Security: PostgreSQL offers strong security features, such as data encryption, access controls, and authentication mechanisms, ensuring that sensitive data is well protected from unauthorized access.

Scalability: PostgreSQL is highly scalable, allowing it to handle increasing amounts of data and traffic as the application grows, making it suitable for long-term use.

Management: PostgreSQL provides robust tools for managing data, including backup and recovery options, performance monitoring, and tuning, ensuring that the database runs smoothly and can be easily maintained.

Chapter 4 - Proposed Solution

4.1 Introduction

Our system, the Learning Management System (LMS), will be used to manage courses, student engagement, and lecturer collaboration. Through the software, we aim to provide an interface that connects instructors, students, and administrators efficiently. In this system, instructors can create and manage courses, assign materials, and track student progress, while students can easily access course content, participate in assignments, and monitor their learning journey.

Admins will have the ability to manage the entire LMS, including assigning instructors to courses and ensuring the smooth running of the system. The system will also allow students to interact with course materials, track their learning progress, and receive updates on assignments and deadlines. Additionally, the system will provide features like notifications for upcoming assignments, course updates, and important announcements. It will also support real-time updates, making it easier for both students and instructors to stay informed. The LMS will aim to enhance communication, streamline course management, and improve the overall learning experience for both students and instructors.

4.2 Software Development Model

We are using the Agile Scrum methodology to develop the Learning Management System (LMS) project. This approach allows us to work in iterative cycles (sprints), enabling us to adapt to any changes in user requirements. After each sprint, we will have a deliverable that can be reviewed and tested, ensuring the system meets the needs of the end users. By adopting Agile Scrum, we can incorporate continuous feedback from users, leading to a more user-focused and flexible development process. Additionally, working with Agile Scrum will give us valuable experience in using a widely adopted project management method, which is essential in modern software development.

4.3 Users, Activities, Inputs, and Outputs

Users		
Lecturer	Activities	<ul style="list-style-type: none">• Login using username and password• View assigned courses• Edit profile• Manage lectures• Manage assignments and quizzes• Manage Materials• Interact with Students• Receive notifications and Student Queries• Edit account details• View and Manage Course details• Participate in forums , discussions , or messaging• Edit user details• View Enrolled Students• Download Student Activities• Send Announcements• Report Generation• Add Students with Admin approval• Record Student Attendance• Redirect to DMS (or Continue with Moodle) based on user settings• Display Document Management Features• Select Document• Add Metadata

		<ul style="list-style-type: none"> • Store Document in DMS Storage • Notify relevant users • Fetch Document from DMS Storage • Allow user to download • Enter Search Query • Fetch Matching Documents • Display Results
	Inputs	<ul style="list-style-type: none"> • Username and Password • Setting Course Parameters • Sending Announcements • Lecture details • Questions and Answers for Quizzes • Report Requests • Moodle User Roles • User Access Settings • Selected Document • Document Metadata • Search Query • Download Request
	Outputs	<ul style="list-style-type: none"> • Course materials • Dashboard updates • Student Queries

		<ul style="list-style-type: none"> • Student Quizzes answers and submissions • Notifications • Authentication Result • User Roles and Permissions • Moodle Dashboard Access • DMS Dashboard Access • Document Management Options • Document Uploaded • Download Document • Document Search Results
Administrator	Activities	<ul style="list-style-type: none"> • Login using username and password • Manage courses • Send notifications • Change password • Assign lectures and student to relevant course • Authentication User • Sync Roles and Permissions • Access Moodle Dashboard • Redirect to DMS (or Continue with Moodle) based on user settings • Display Document Management Features • Select Document • Add Metadata • Store Document in DMS Storage • Notify relevant users • Fetch Document from DMS Storage • Allow user to download

		<ul style="list-style-type: none"> • Enter Search Query • Fetch Matching Documents • Display Results
	Input	<ul style="list-style-type: none"> • User name and password • Content Notifications • Course details • Course assignments details • User management details • System configurations • User Login Credentials • Moodle User Roles • User Access Settings • Selected Document • Document Metadata • Search Query • Download Request
	Output	<ul style="list-style-type: none"> • Feedback from lectures and students • User management outputs • Course management outputs • Content management outputs • Notification outputs • Error handling and status outputs • Reports • Messages and from lectures • Notifications • System status update • Authentication Result • User Roles and Permissions

		<ul style="list-style-type: none"> • Moodle Dashboard Access • DMS Dashboard Access • Document Management Options • Document Uploaded • Notifications • Download Document • Document Search Results
Student	Activities	<ul style="list-style-type: none"> • Login using username and password • View enrolled courses • Edit profile • Participate in assignments and quizzes • Interact with lectures • Receive notifications • Edit account details • View Enrolment details • Participate in forums , discussions , or messaging • Edit user details
	Inputs	<ul style="list-style-type: none"> • Username and Password • Assignments submission • Quiz responses • attach any private files • Update personal information • Request for certificate

	Outputs	<ul style="list-style-type: none"> • Access Course materials • Download Course Materials • Dashboard updates • Assignment feedback and grades • Notifications • confirmation messages • real time quiz timer • Quiz results or feedback
--	---------	---

Trainers	Activities	<ul style="list-style-type: none"> • Login using username and password • View enrolled courses • Track Task Progress • Assign Tasks • Review Log submissions • Provide Feedback • Edit account details • View Enrolment details • Participate in forums , discussions , or messaging • Edit user details
	Inputs	<ul style="list-style-type: none"> • Username and Password • Assignments submission • Quiz responses
	Outputs	<ul style="list-style-type: none"> • Course materials • Dashboard updates • Assignment feedback and grades • Notifications • confirmation messages

4.4 Process

The Learning Management System (LMS) is designed to streamline and enhance the process of teaching, learning, and course management. The system starts with the administrator, who has the responsibility of setting up the platform by creating accounts for students, lecturers, and themselves. Administrators also create and manage courses by defining course details, uploading relevant materials, and assigning lecturers to handle specific subjects or classes. Once these initial setups are complete, lecturers gain access to the system to upload course-related content such as lecture notes, recorded videos, assignments, and quizzes. They also define grading criteria and deadlines for assignments and assessments.

Students access the LMS using their credentials and have a personalized dashboard that displays their enrolled courses, notifications, and upcoming deadlines. Students can enroll in courses, download learning materials, and submit assignments directly through the platform. The system tracks submission dates and automatically flags any late submissions for lecturer review. For quizzes and exams, the system provides automated grading features for objective questions, which allows students to get instant feedback. Subjective questions are reviewed manually by lecturers, who then provide feedback and grades.

The LMS also includes an attendance management feature that allows lecturers to track and record student attendance for every class. This data is accessible to both students and administrators, ensuring transparency. Additionally, progress dashboards are provided for both students and lecturers. Students can monitor their grades, assignment statuses, and overall performance in each course, while lecturers can track student engagement and identify those who may need additional support.

Communication between students and lecturers is facilitated through an integrated chat or messaging system. This feature allows students to ask questions, seek clarification on assignments, or discuss course-related topics with their peers and lecturers. The system also sends notifications and reminders to students about assignment deadlines, upcoming exams, and important announcements.

Administrators have additional responsibilities, such as monitoring the overall performance of the system, ensuring that courses and accounts are kept up to date, and archiving old courses to maintain system efficiency. They also have the authority to deactivate accounts for users who are no longer part of the institution and remove outdated course materials.

The LMS is designed to be flexible and scalable, allowing institutions to adapt it to their specific needs. It provides real-time access to learning materials, enhances collaboration between students and lecturers, and ensures that teaching and learning processes are efficient and effective.

4.5 Our Approach

In our LMS project, the developer's approach involves using modern technologies to ensure a smooth and efficient experience for both users and administrators. For the front-end, we are using Next.js with Server-Side Rendering (SSR). SSR enables the pages to be rendered on the server before being sent to the client, which improves the performance and SEO of the application. By dynamically generating content on the server, Next.js ensures that elements such as course details, lecturer profiles, and student progress load quickly and efficiently, providing a fast and responsive user experience. The SSR approach helps in faster load times and better search engine visibility, making the LMS system more scalable and user-friendly.

For the back-end, we are utilizing Spring Boot, which is a robust framework that enables the development of secure, maintainable, and scalable server-side applications. Spring Boot provides a seamless connection between the front-end and back-end, ensuring the smooth flow of data, such as course management, student details, and lecturer assignments.

To manage the data and ensure the system can handle complex queries and relationships, we are using PostgreSQL as our database solution. PostgreSQL offers high performance, reliability, and advanced features, allowing us to manage and store all the data related to the courses, lecturers, students, and user interactions efficiently.

With this combination of Next.js, Spring Boot, and PostgreSQL, we are able to create a fast, reliable, and scalable LMS system that meets the needs of both students and administrators.

4.5.1 Notification system

The notification system in the LMS ensures that students, lecturers, and administrators stay informed about important updates. It includes notifications for assignment deadlines, course updates, exam schedules, feedback, and system maintenance. These notifications are delivered via push notifications. Users can customize their notification settings, mute alerts, or view a history of past notifications. Notifications are triggered automatically based on actions like submitting assignments or uploading course materials, helping users stay organized and engaged. This system improves communication, productivity, and ensures that all users are always informed about relevant events and deadlines.

4.6 Project Management

The project management for our LMS system is carried out using a structured and collaborative approach. We follow the Agile Scrum methodology, which allows us to work in iterative cycles and adapt to changing requirements effectively. For scheduling and task tracking, we utilize Gantt charts and ClickUp, ensuring clear timelines and responsibilities.

Our team is divided into specific roles to handle the different aspects of the project efficiently. For instance, UI designs for the dashboard and login page are created using Figma, ensuring a user-friendly and visually appealing interface. The system development involves Next.js for the front-end, Spring Boot for the back-end, and PostgreSQL for the database, with a notification system to keep users updated.

Containerization using Docker ensures seamless development and deployment, while the system operates on virtual machines hosted on-premises, providing control and reliability. This comprehensive project management approach ensures that the LMS system is developed on time and meets all user expectations effectively.

4.7 Summary

The LMS (Learning Management System) project is designed to provide a comprehensive platform for managing courses, lectures, and student interactions in an educational environment. Administrators can create, update, and manage course content, while students can enroll, attend classes, view materials, and track their performance. The system is developed using modern technologies: Next.js for the front-end, Spring Boot for the back-end, and PostgreSQL for the database. A notification system keeps users informed about course updates, deadlines, and other important events.

For containerization, Docker is used to ensure a consistent and scalable development and deployment process. The LMS is compatible with modern web browsers, including Chromium-based browsers, Safari, and Firefox, providing a seamless experience across desktop and mobile platforms. For deployment, the system will be hosted on-premises, with all associated hardware and infrastructure costs covered by MPMA.

The development follows the Agile Scrum methodology, ensuring flexibility and iterative development to meet user requirements. This approach guarantees a reliable, user-friendly, and scalable solution for educational institutions.

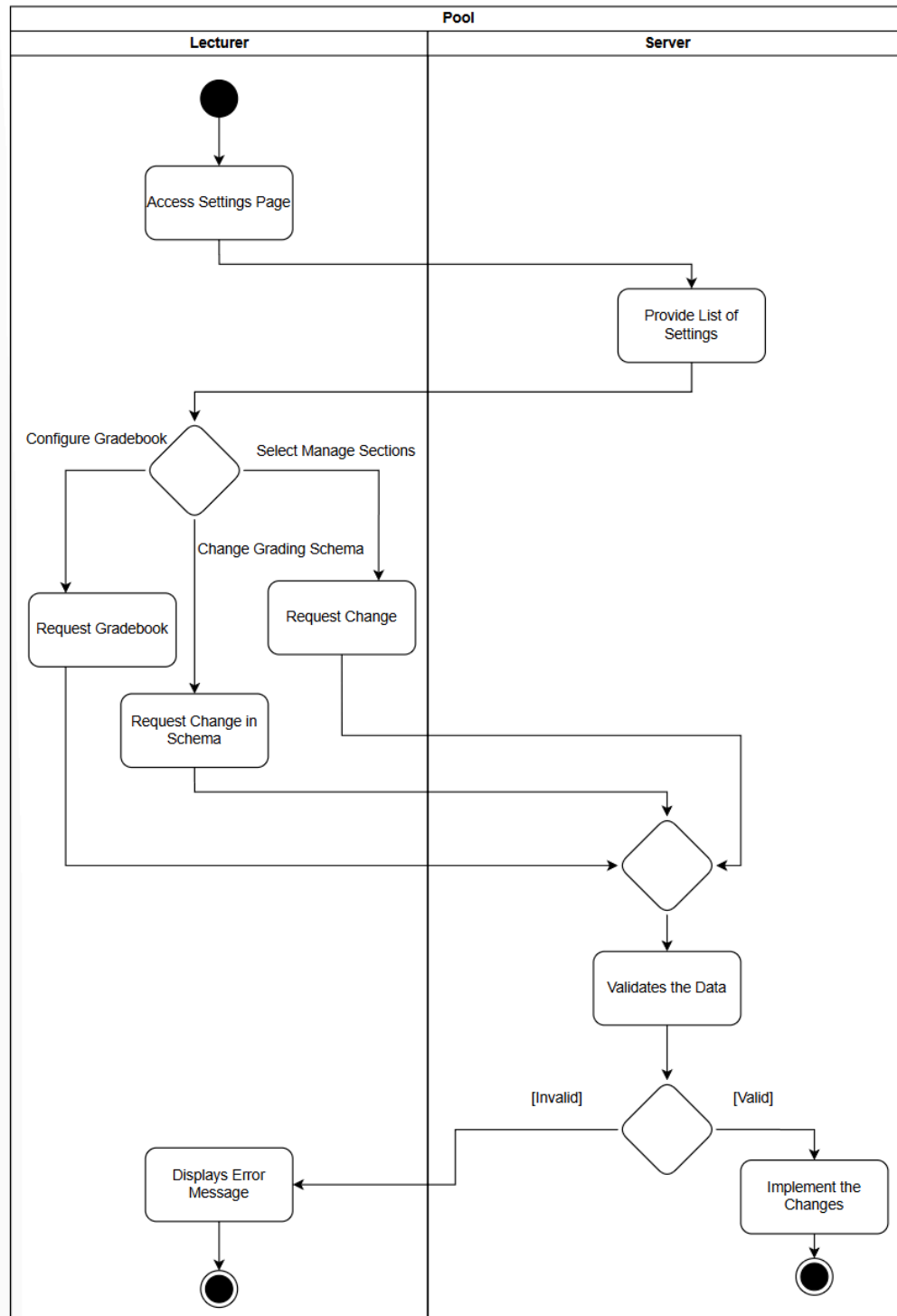
Chapter 5 – Analysis and Design

5.1 Use Case Diagram

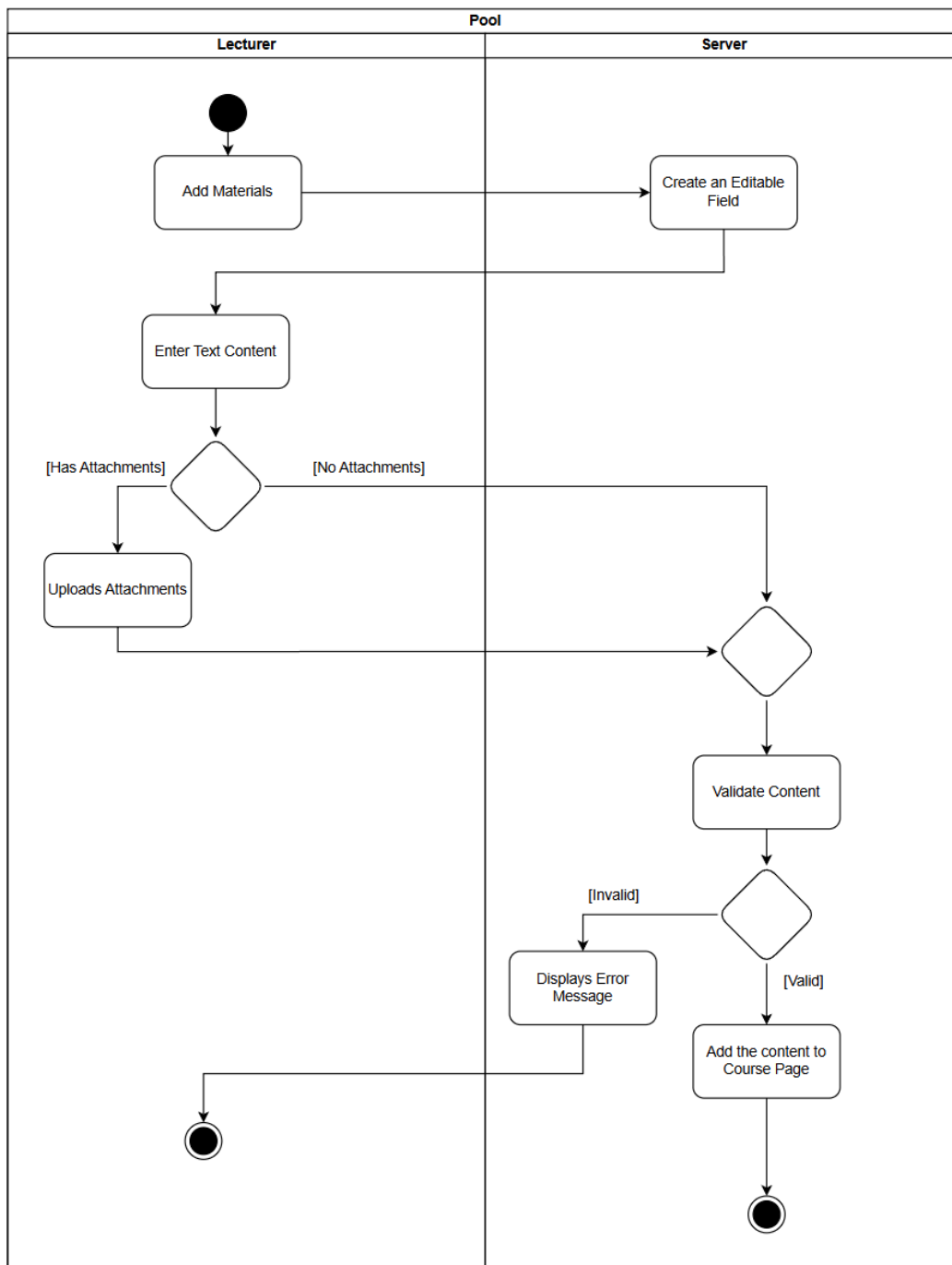
5.2 Activity Diagram

Lecturer Interface

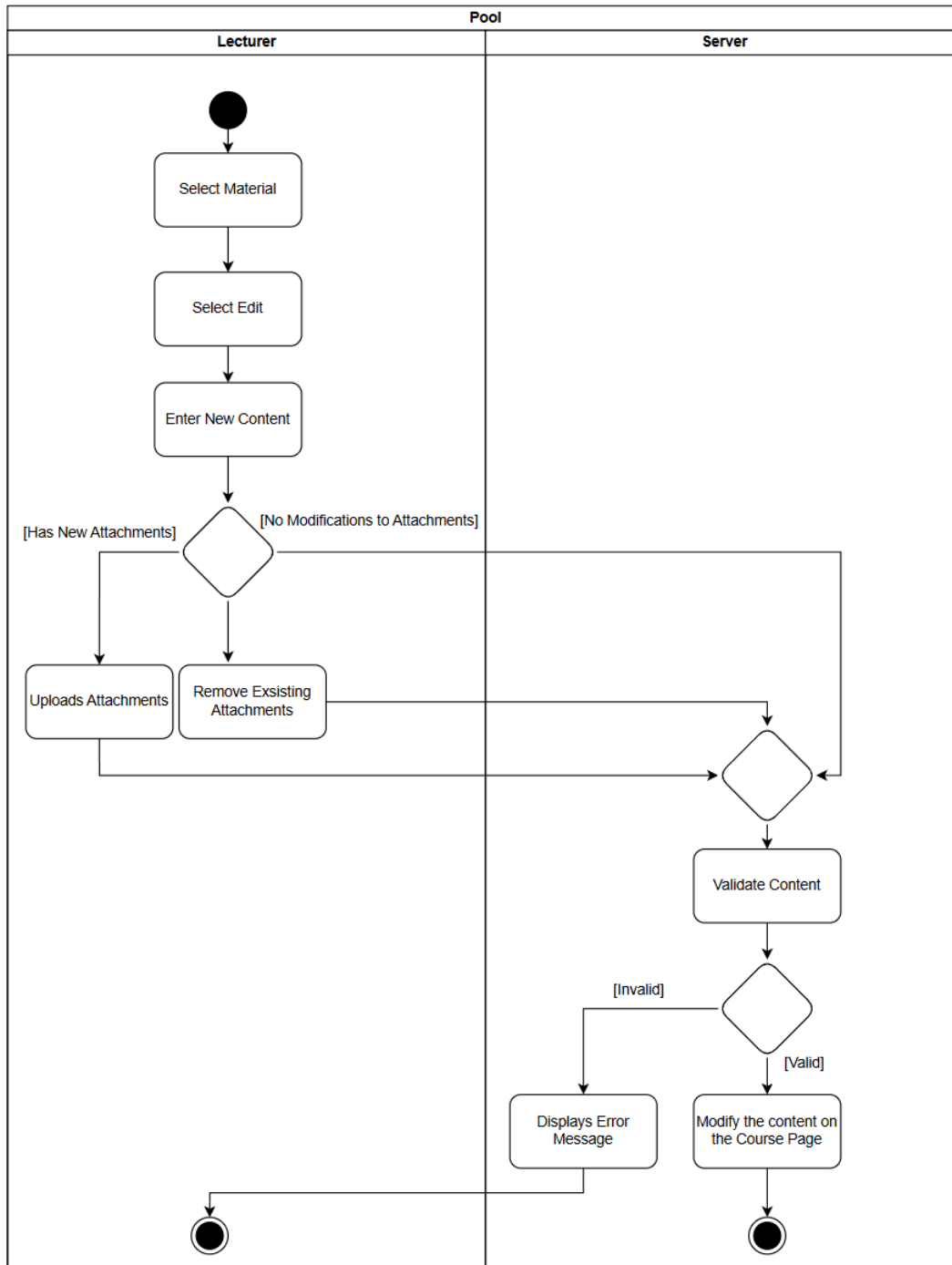
Adjust Course Settings



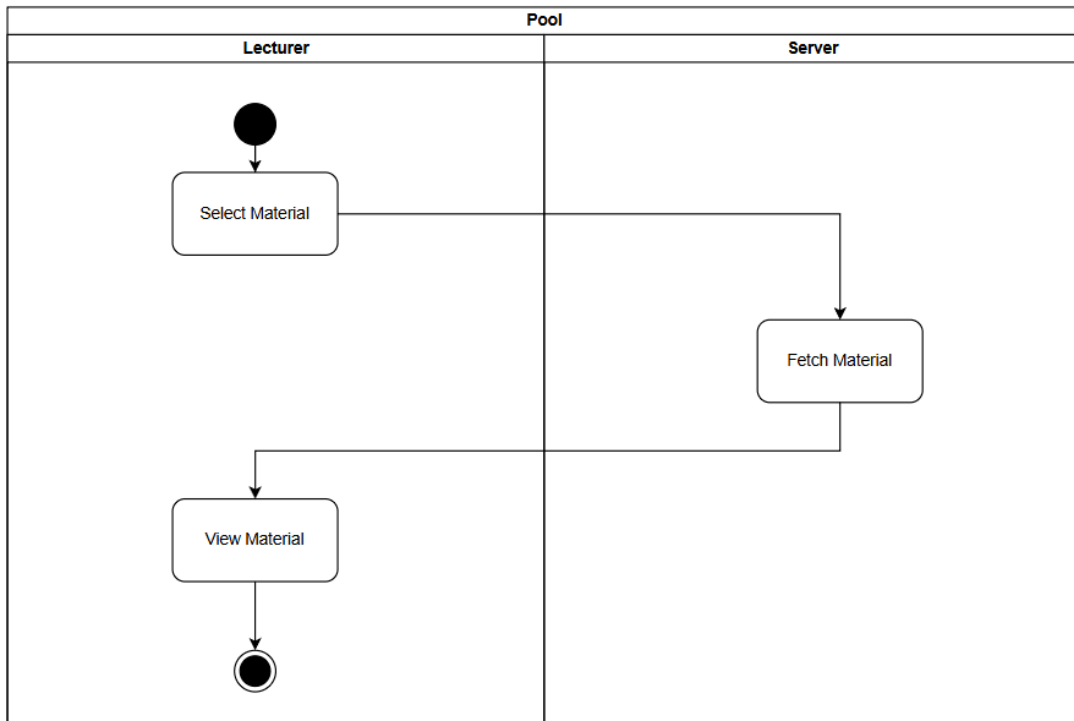
Add Text/Materials



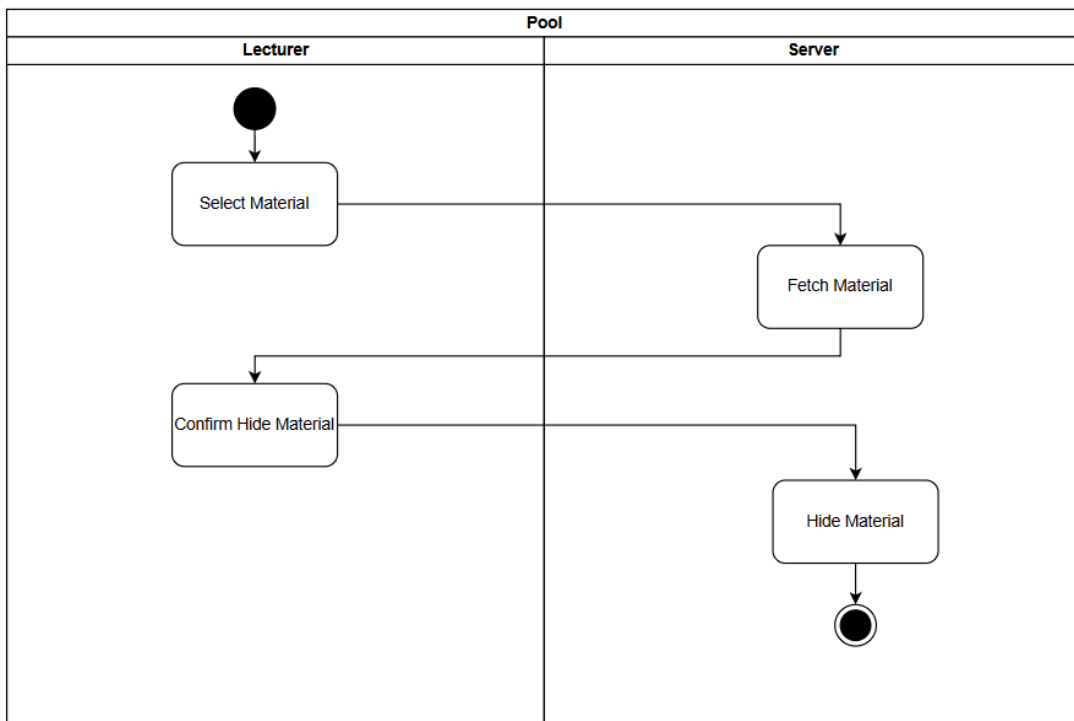
Edit Materials



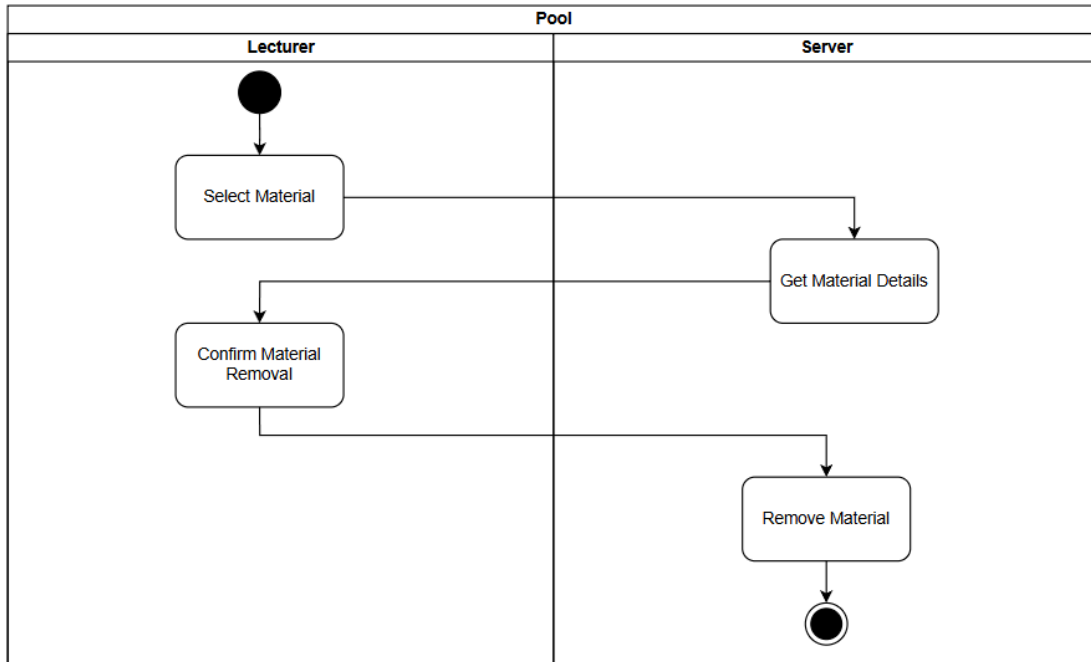
Preview Materials

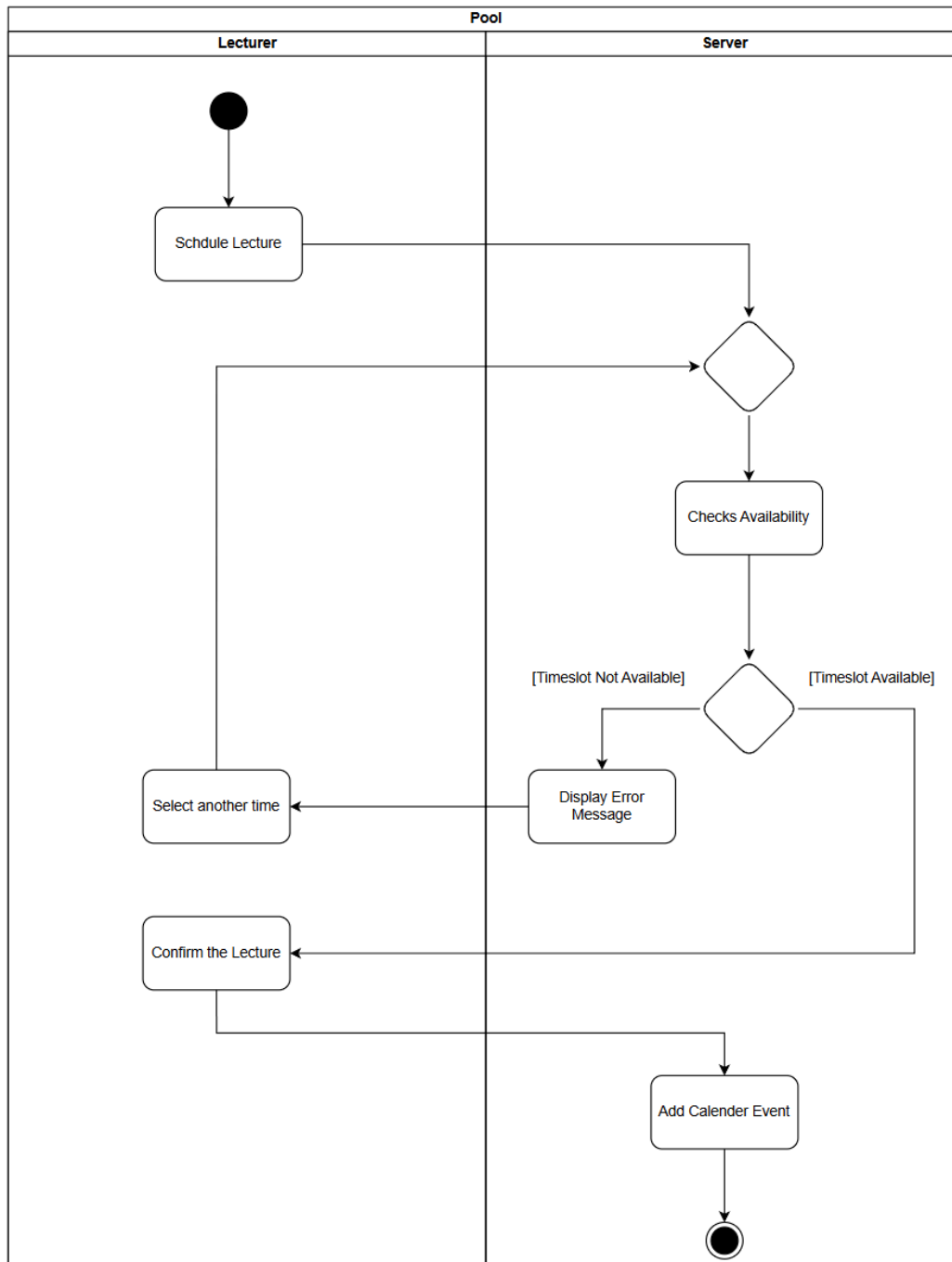


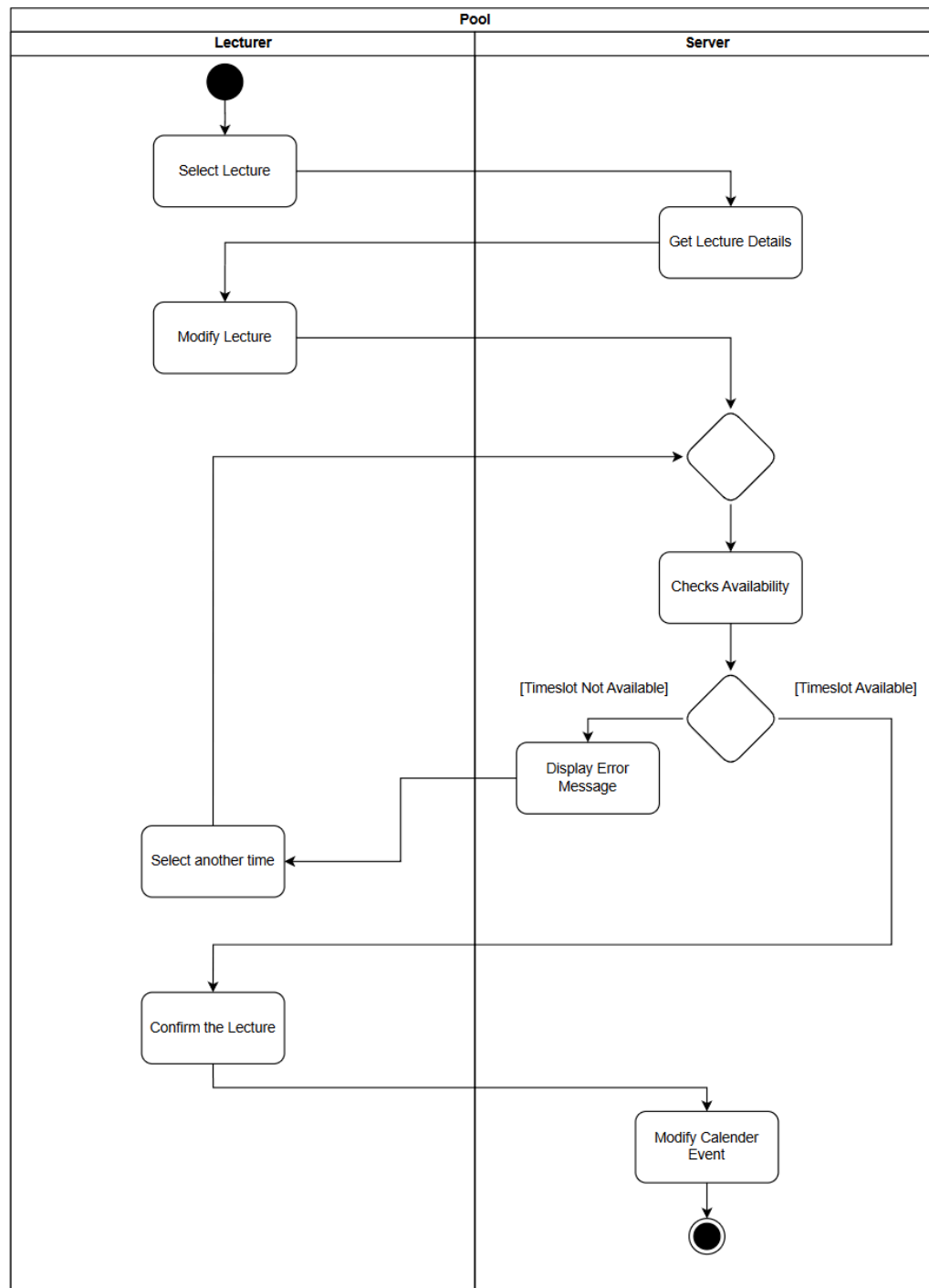
Hide Materials



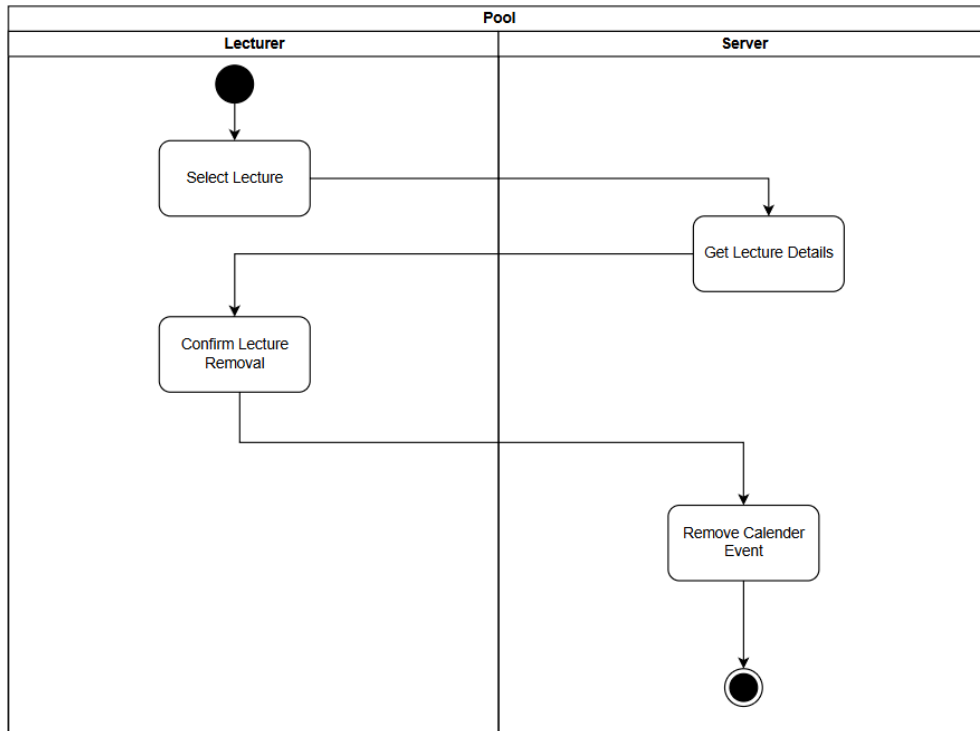
Material - Remove



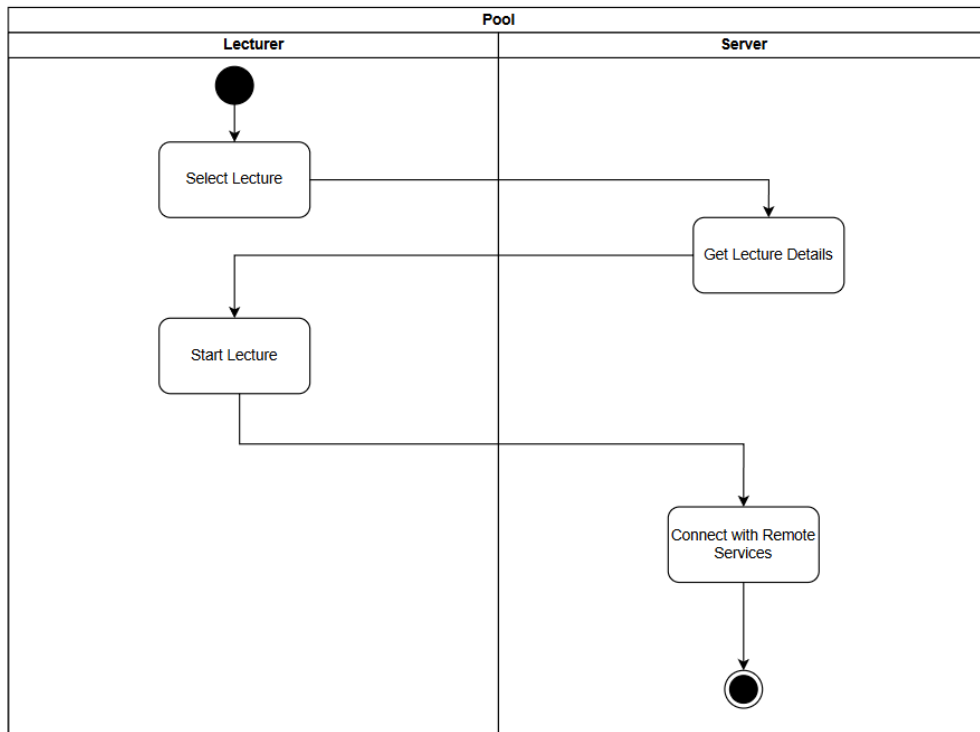


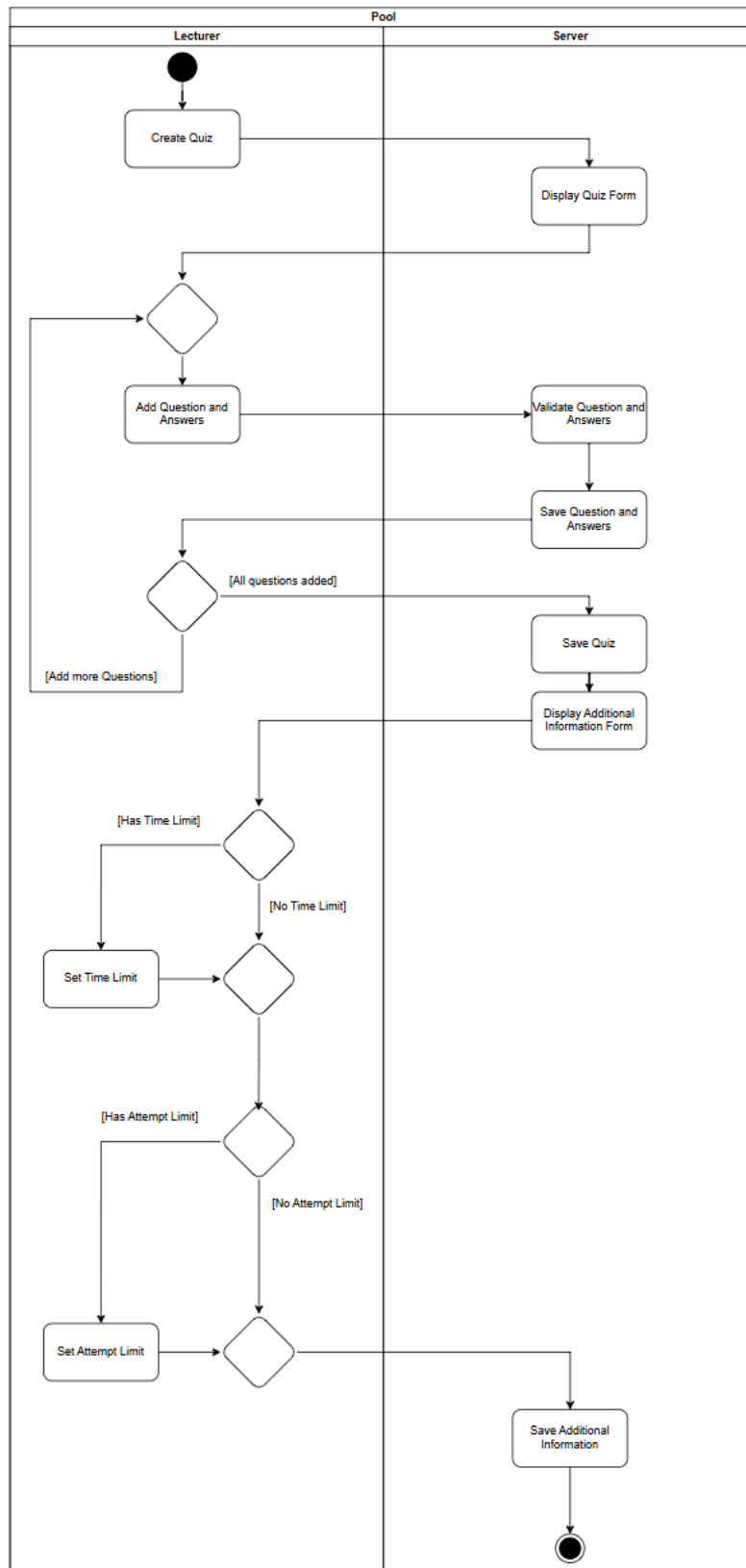


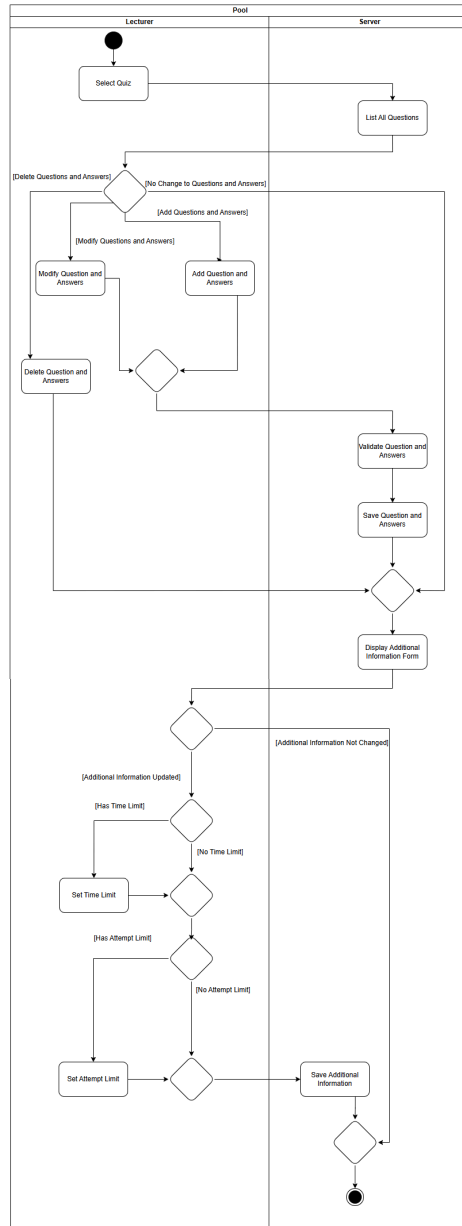
Lecture Management - Modify (Remove)



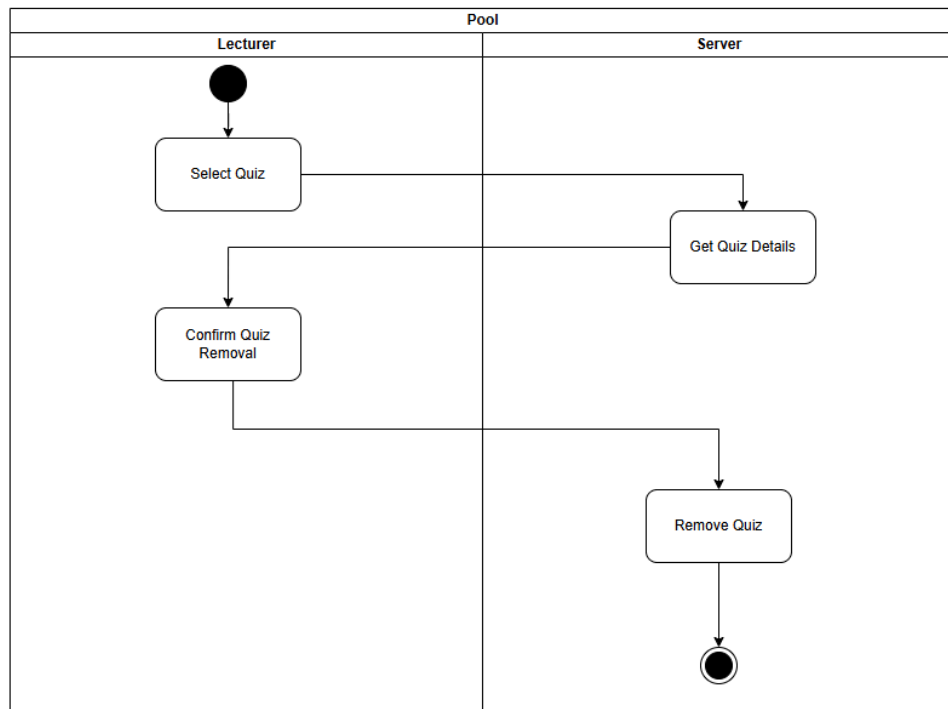
Lecture Management - Conduct



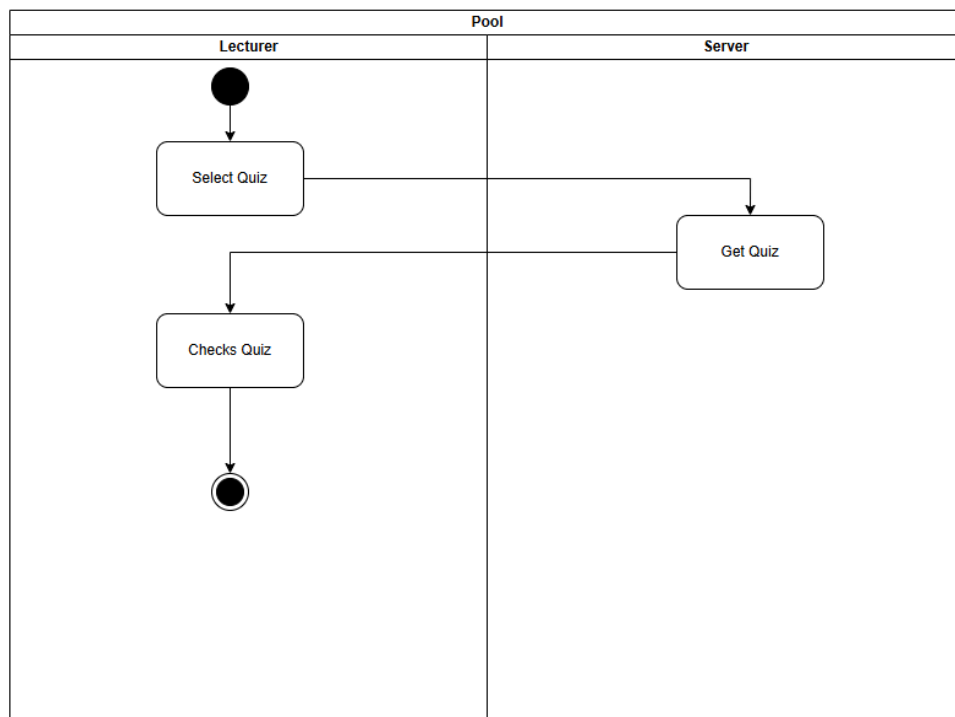




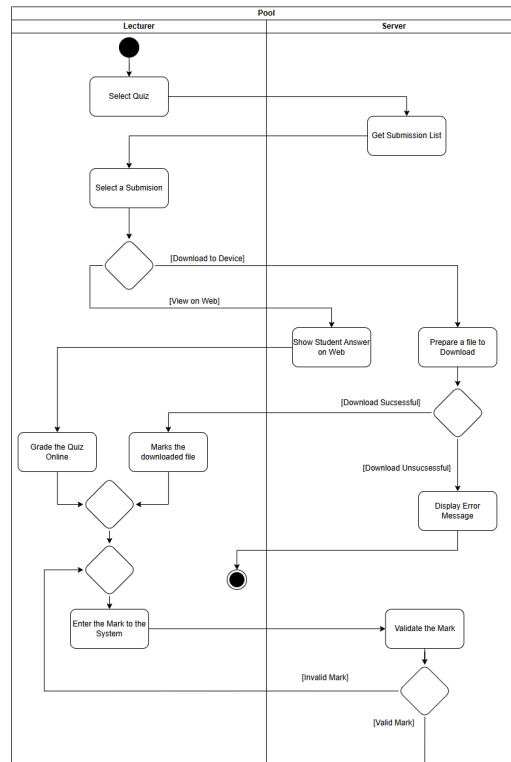
Quiz Management - Remove



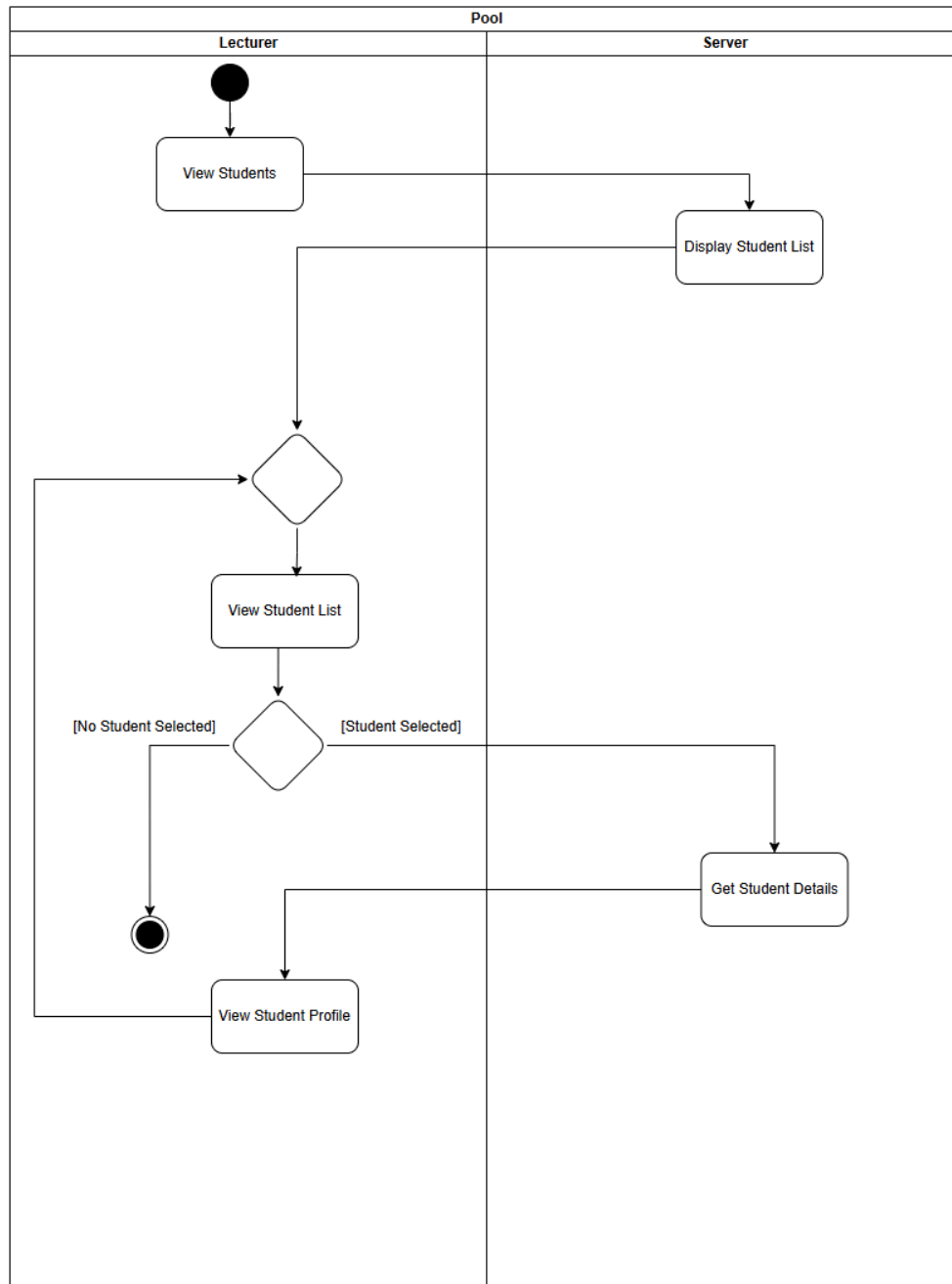
Quiz Management - Preview



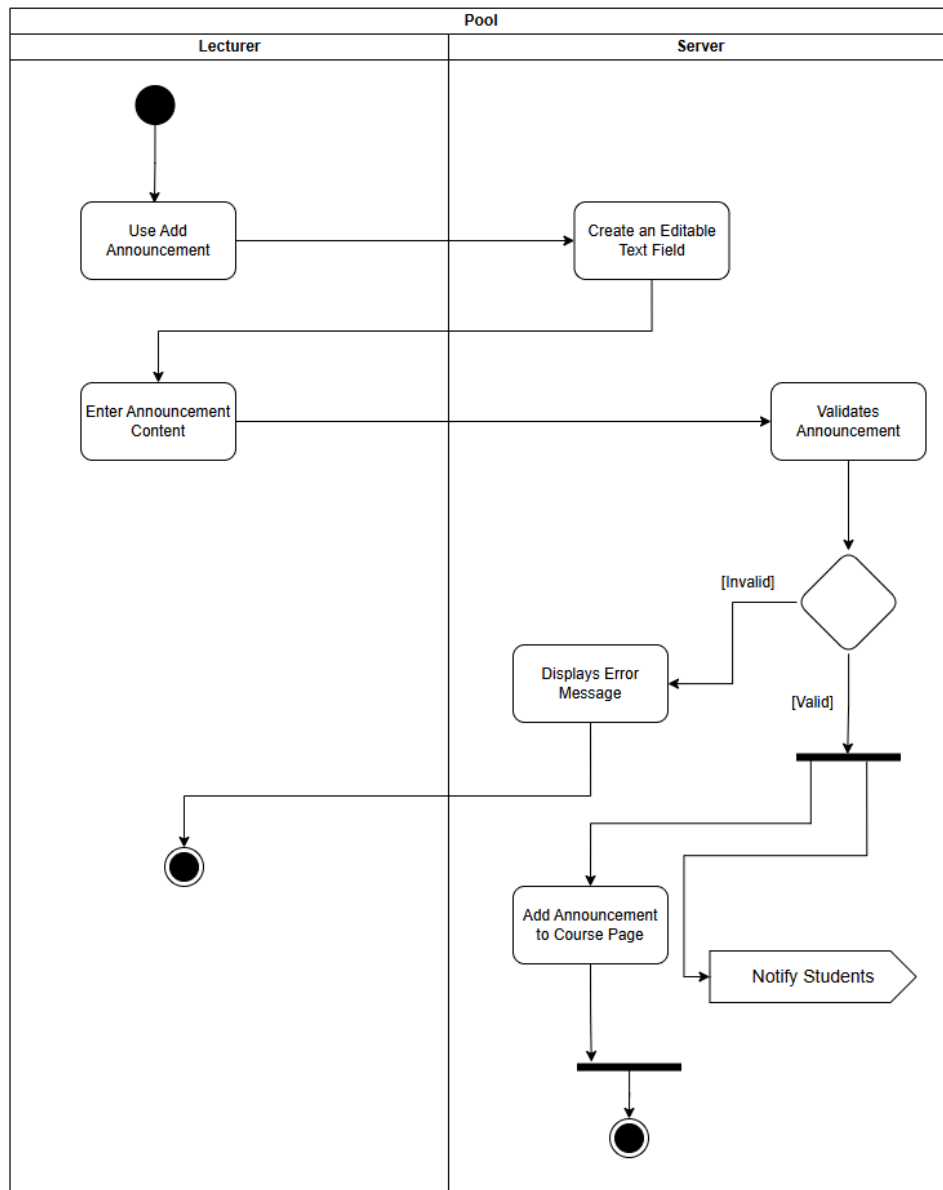
Grade Quizzes



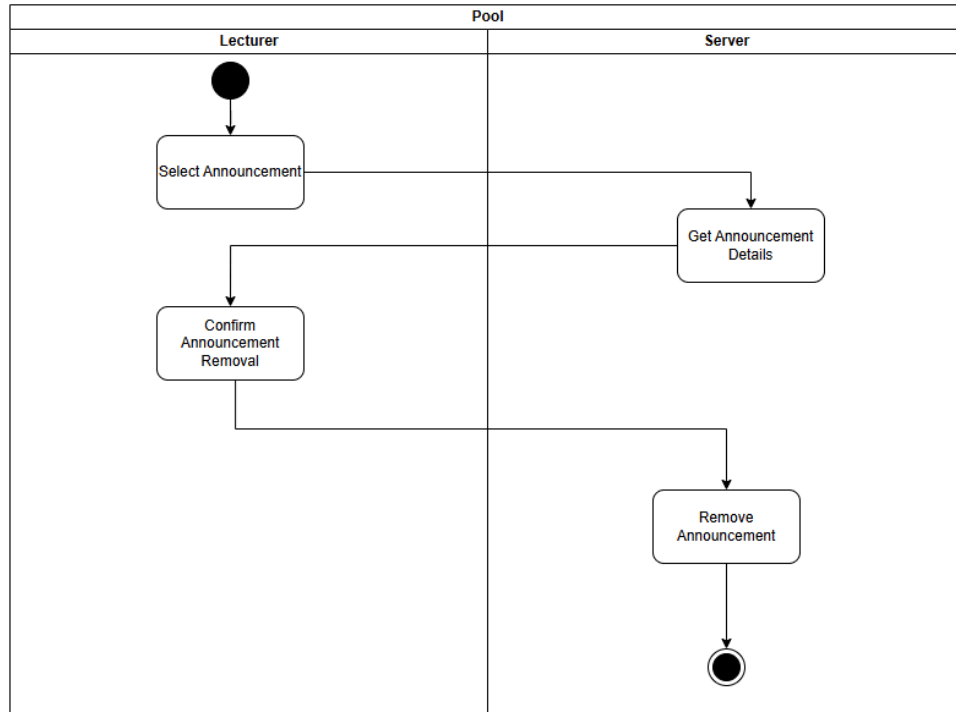
View Enrolled Students



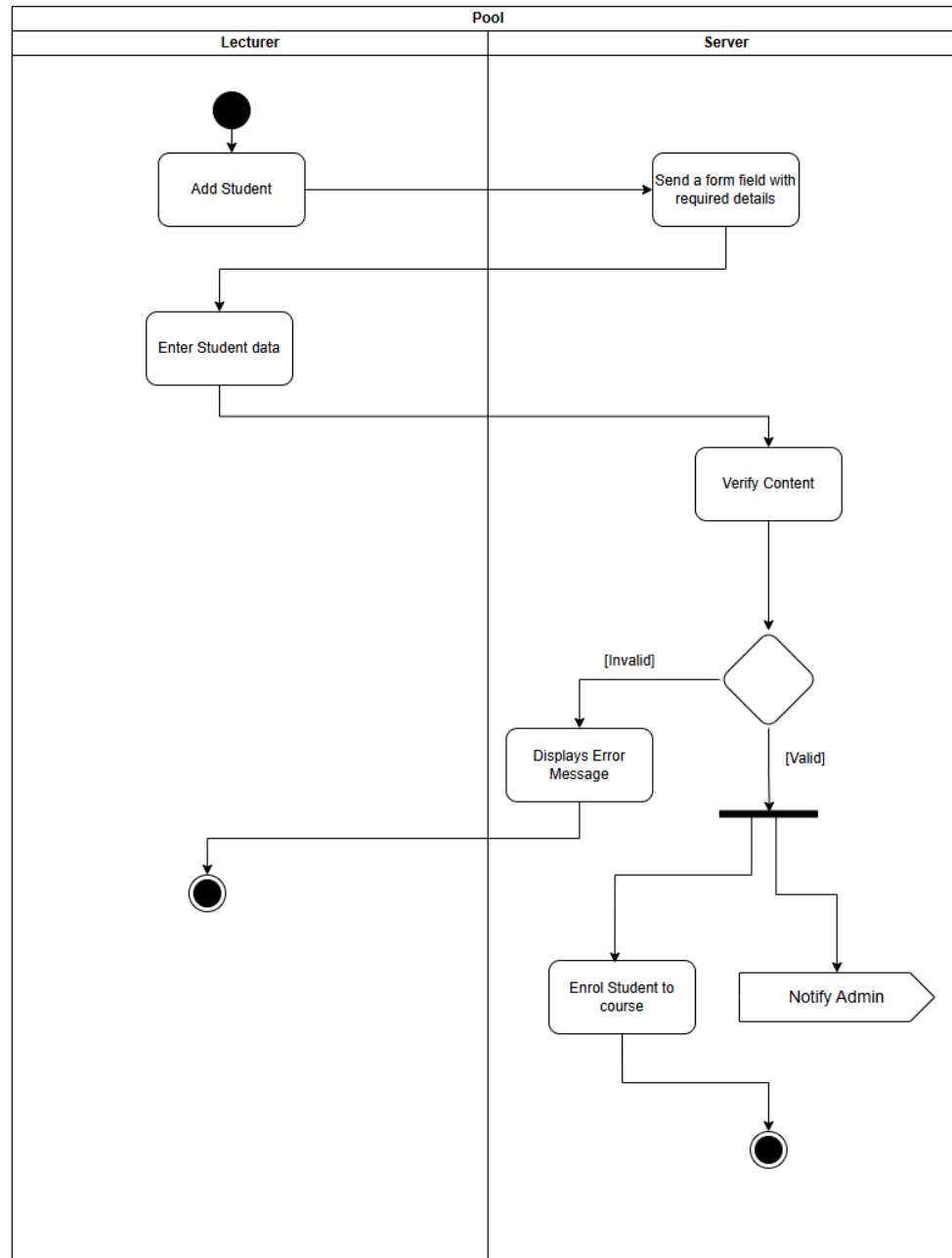
Send Announcement



Announcement - Remove

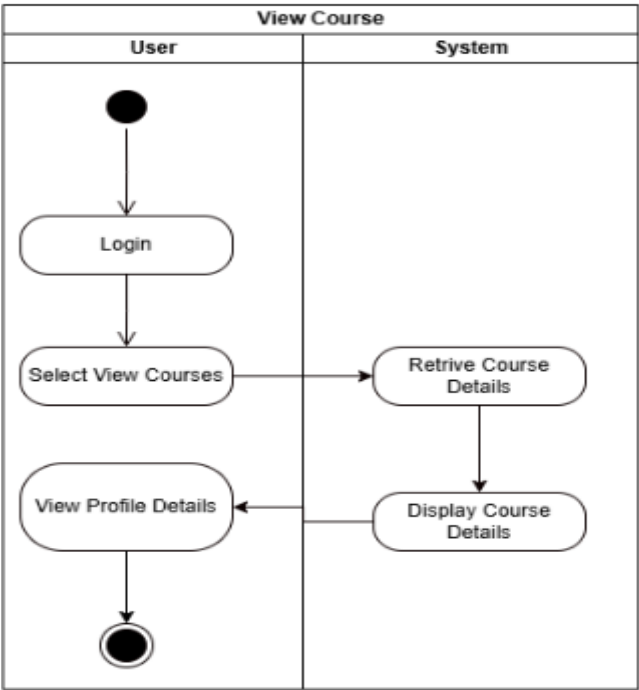
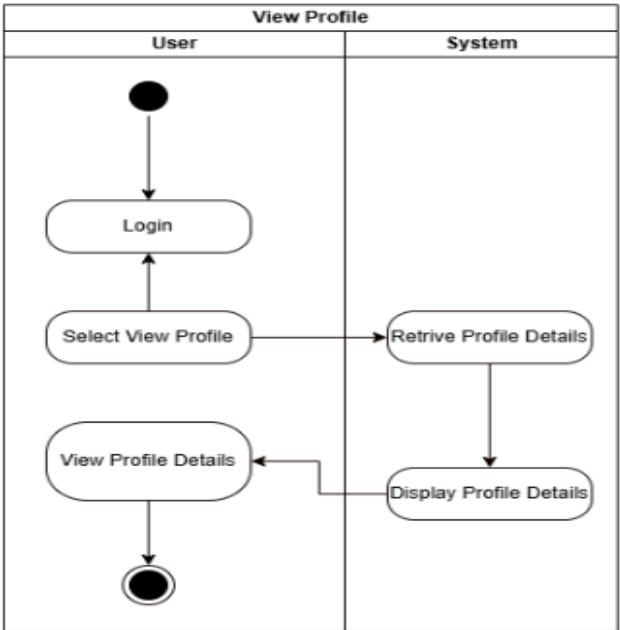


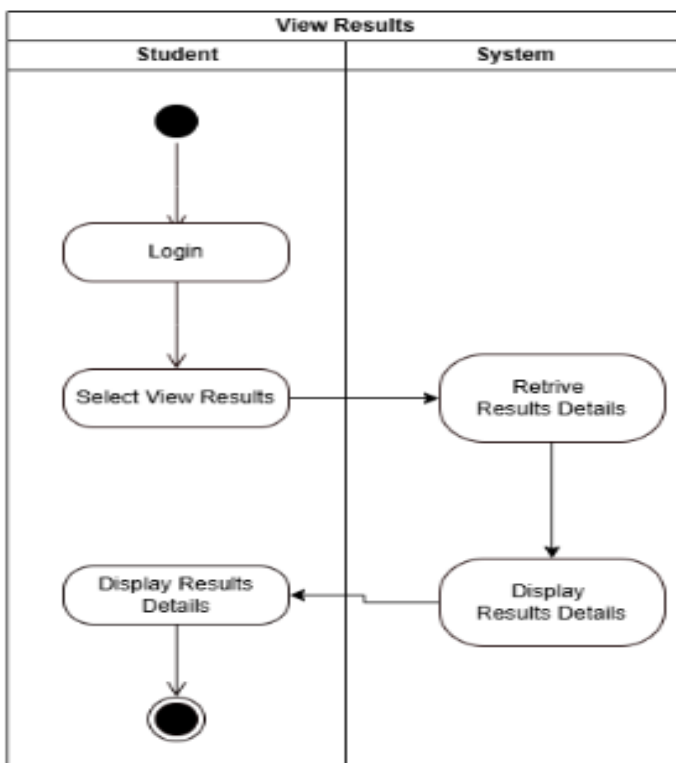
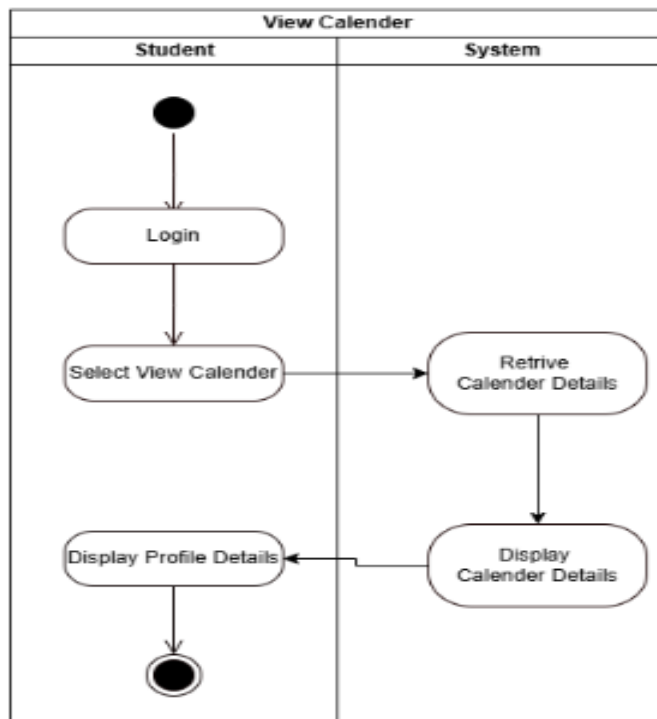
Add Student

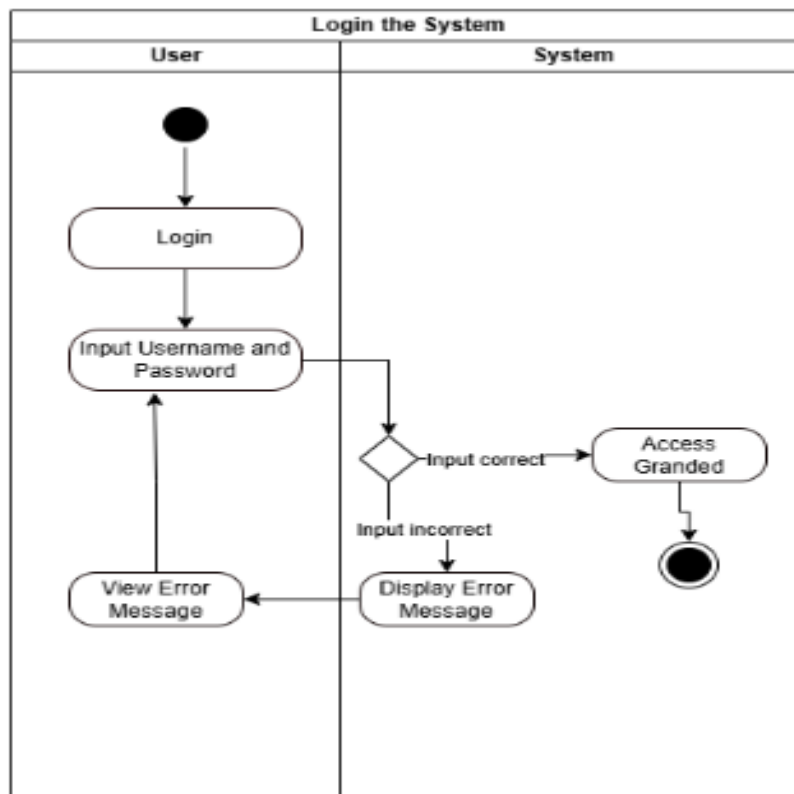
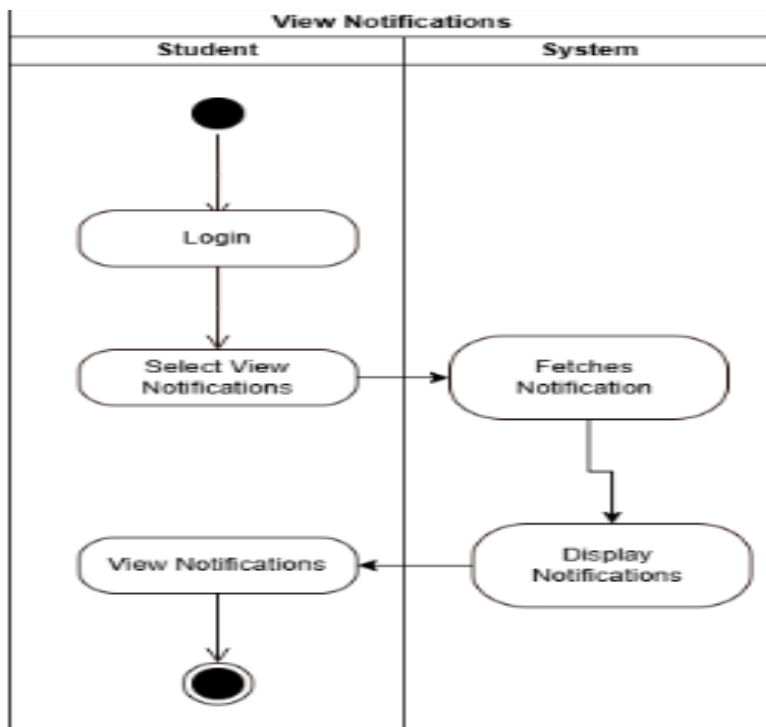


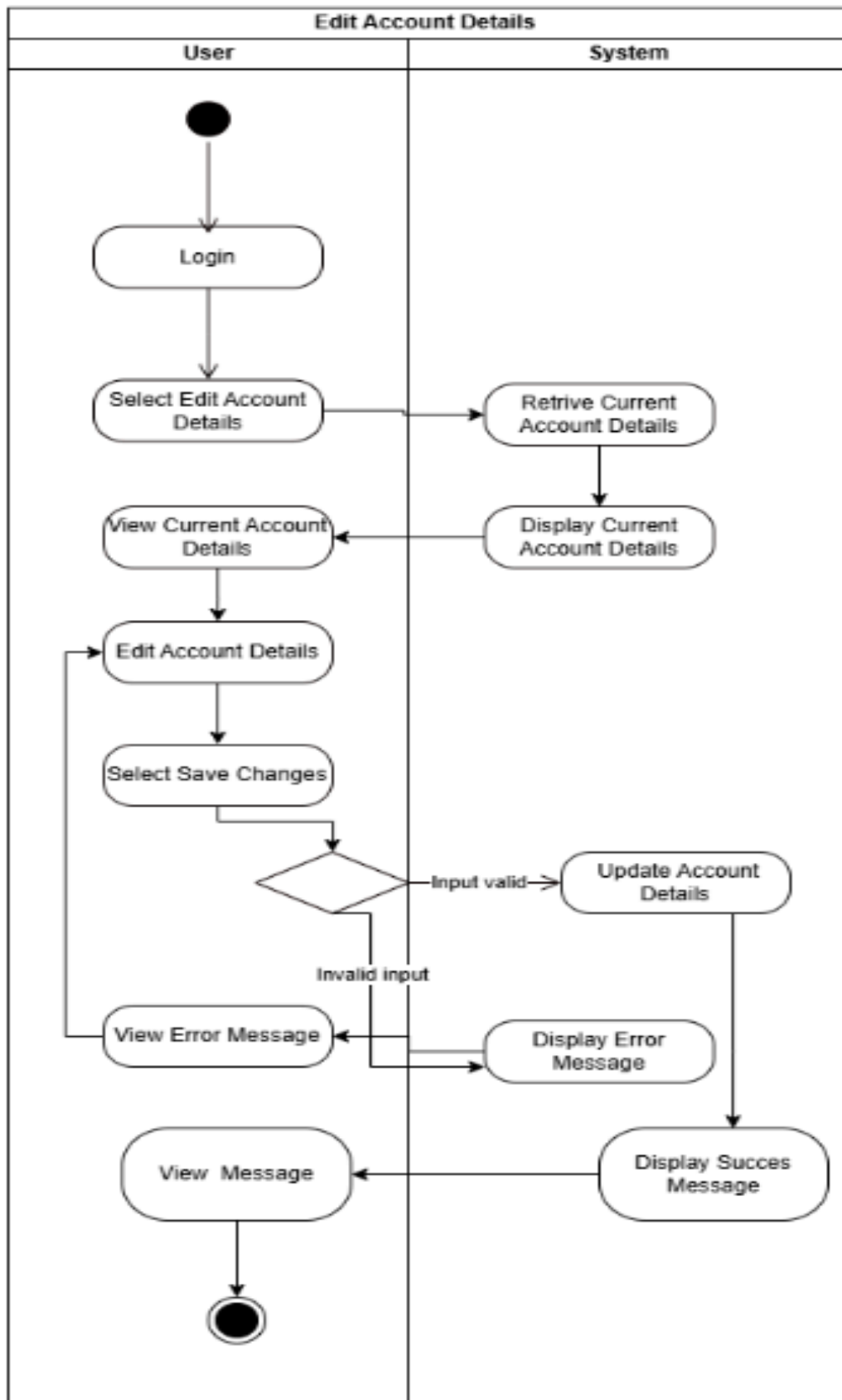
Admin Dashboard

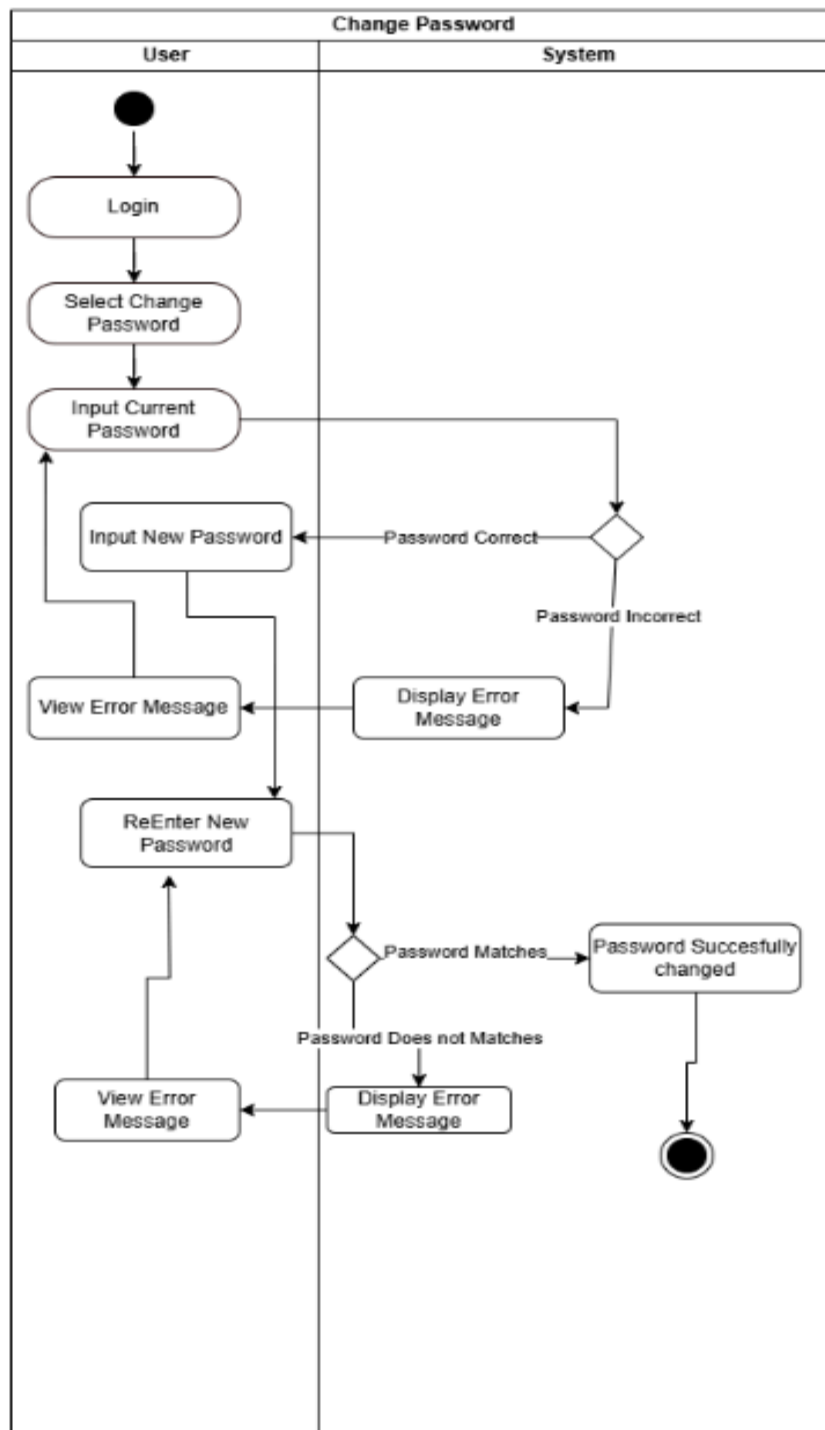
01.

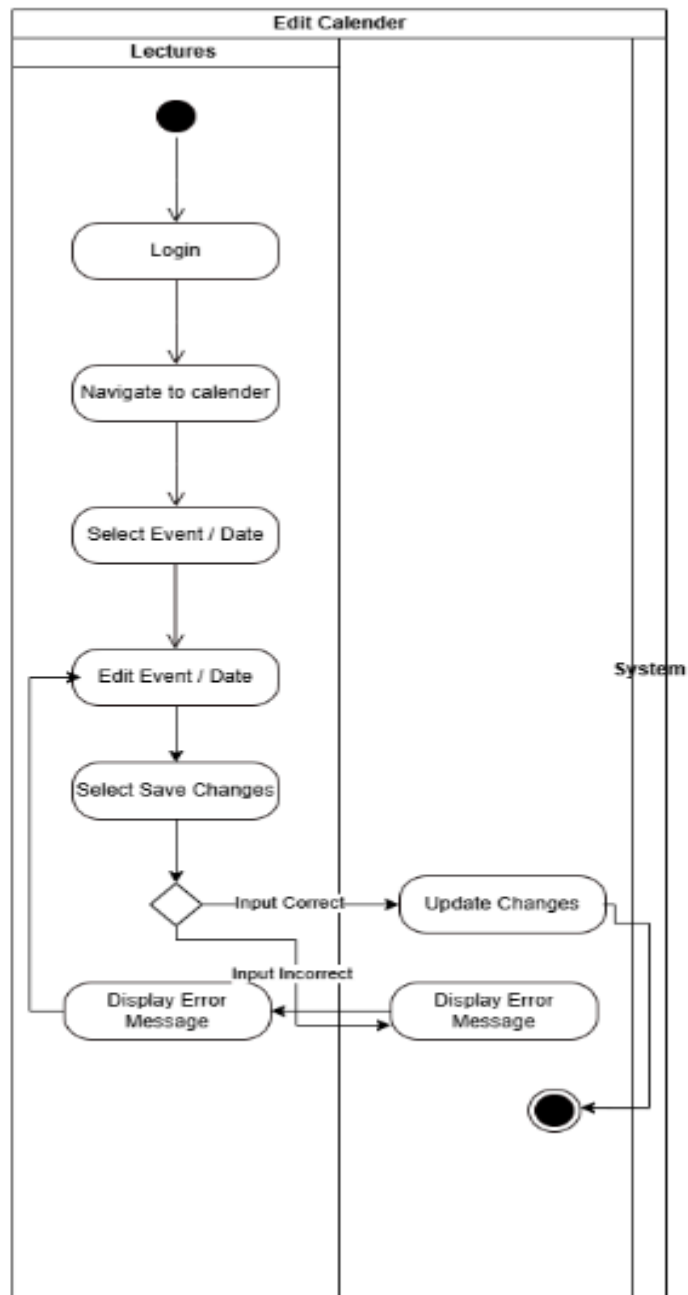






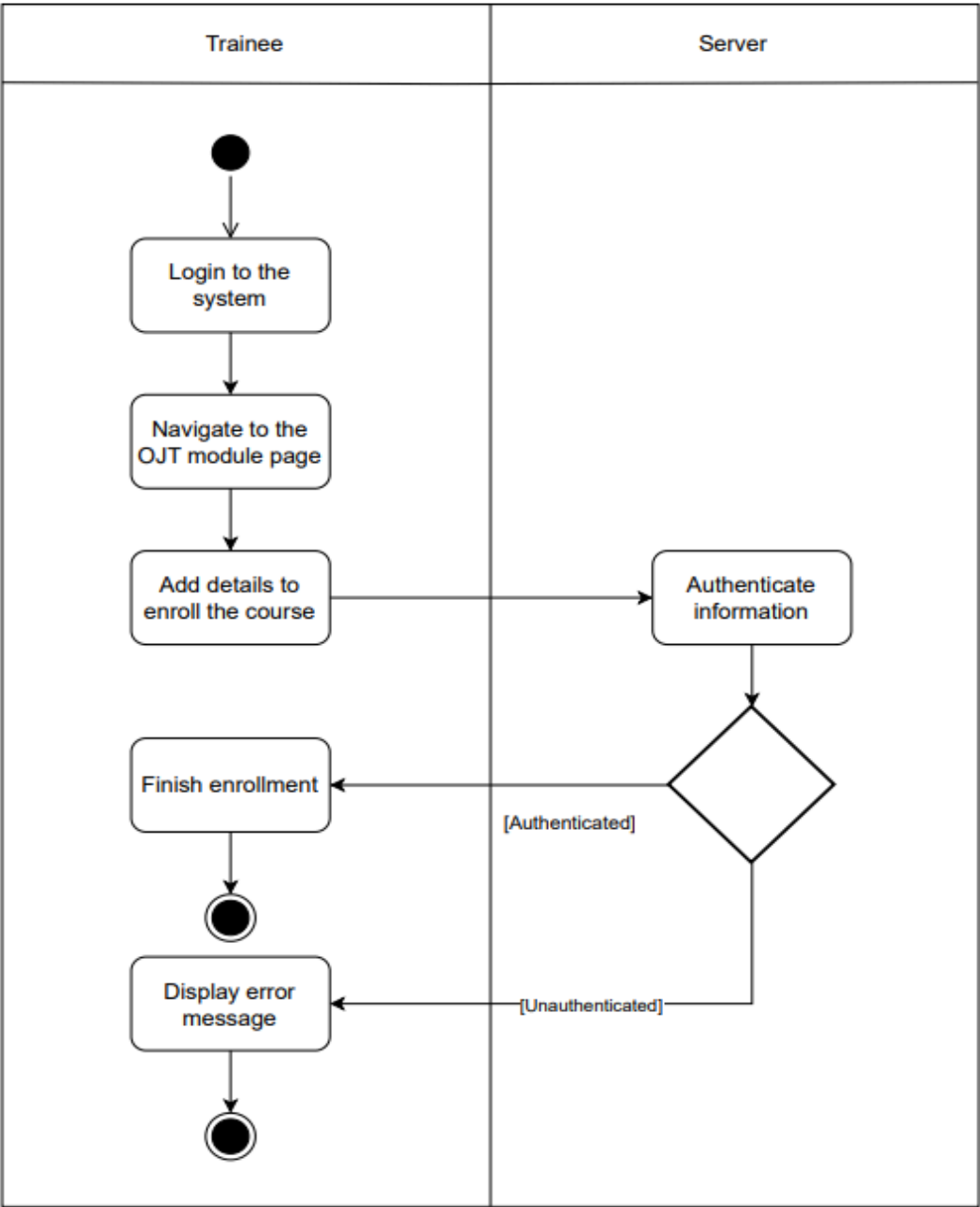




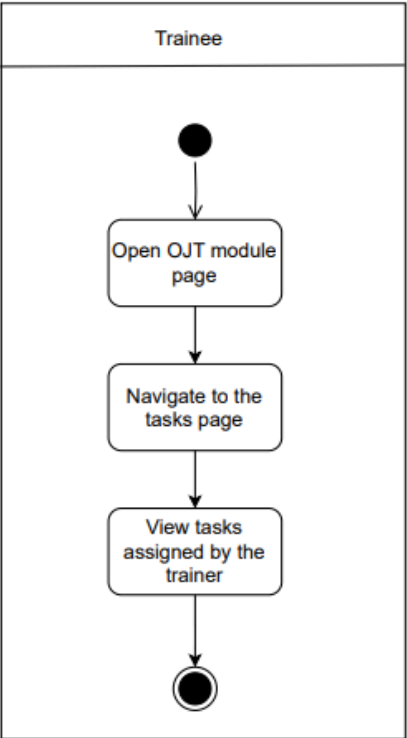


OJT Course Module

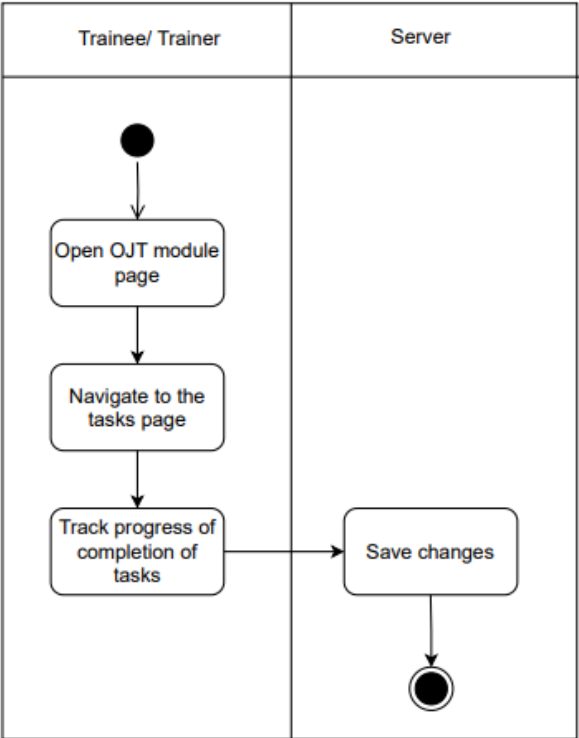
Enroll to OJT Course



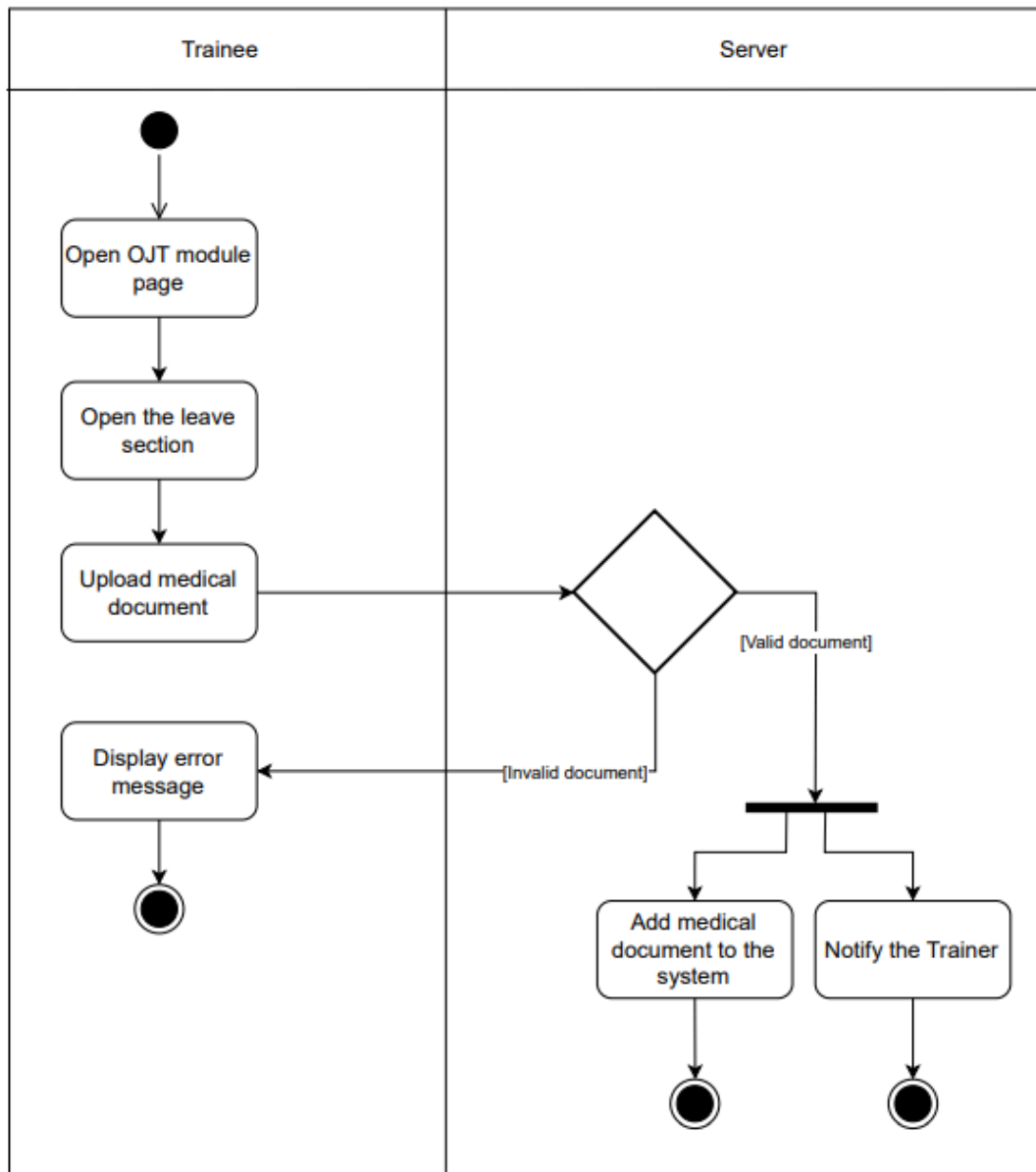
View Tasks



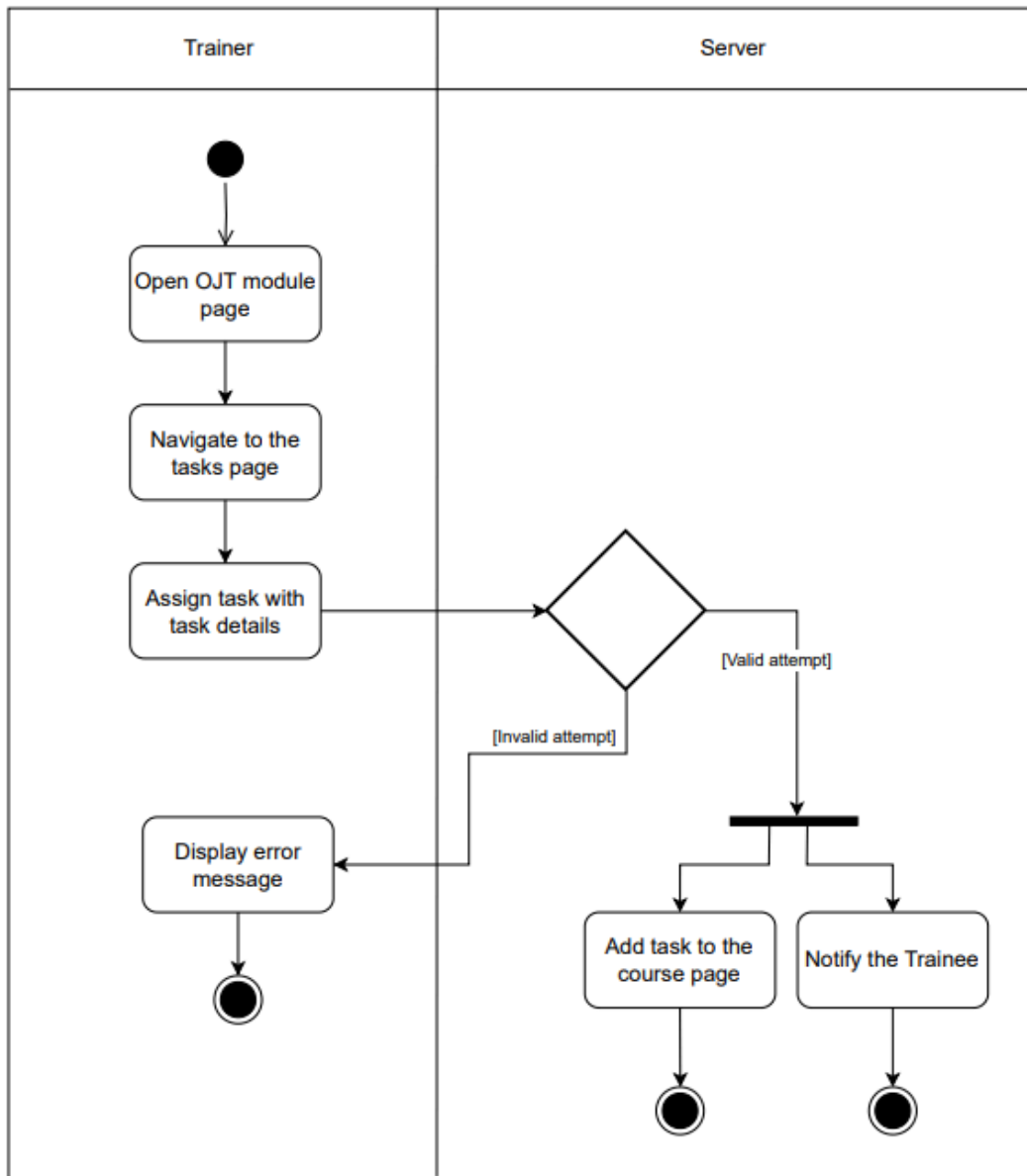
Track Task Progress



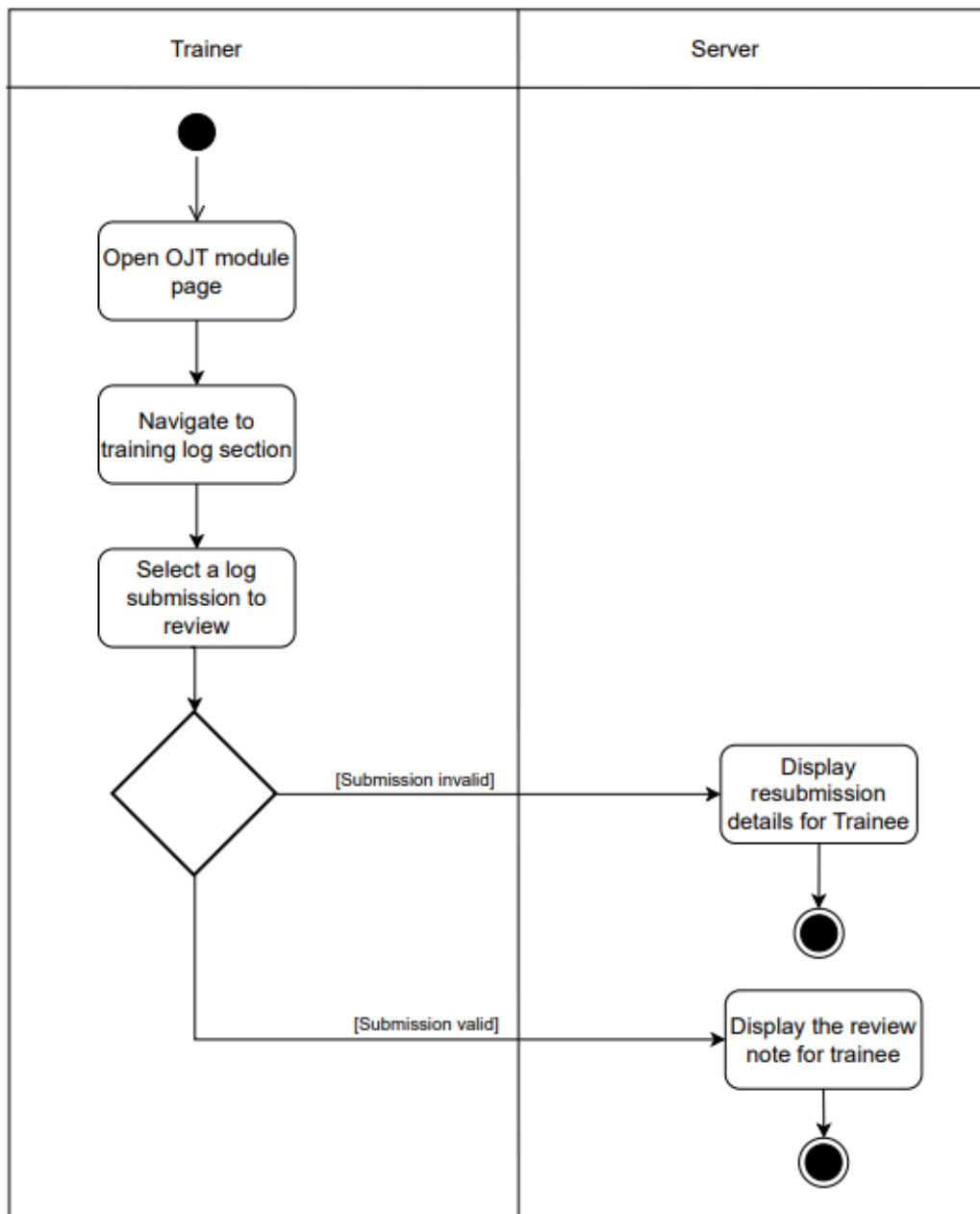
Upload Medical Documents



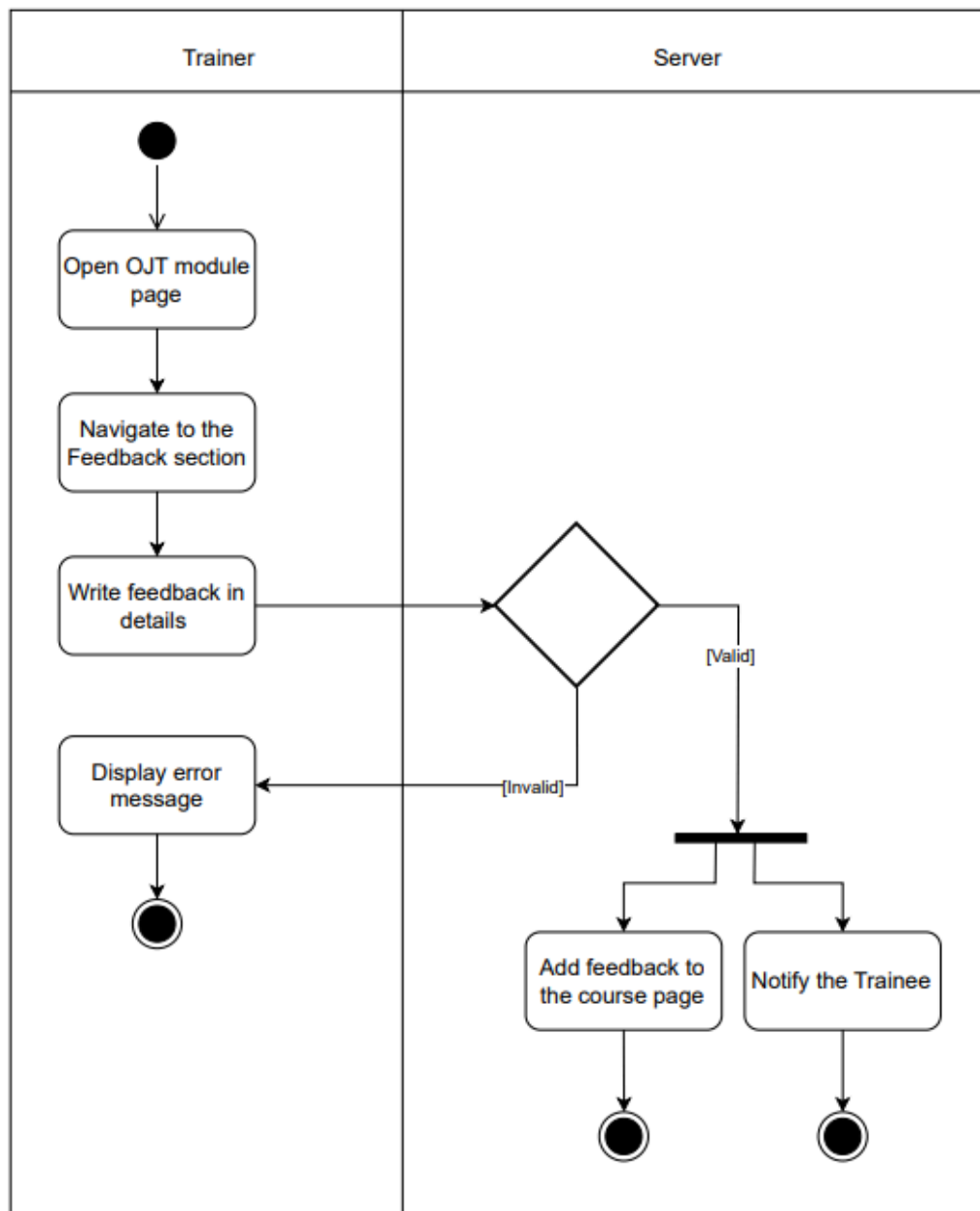
Assign Tasks



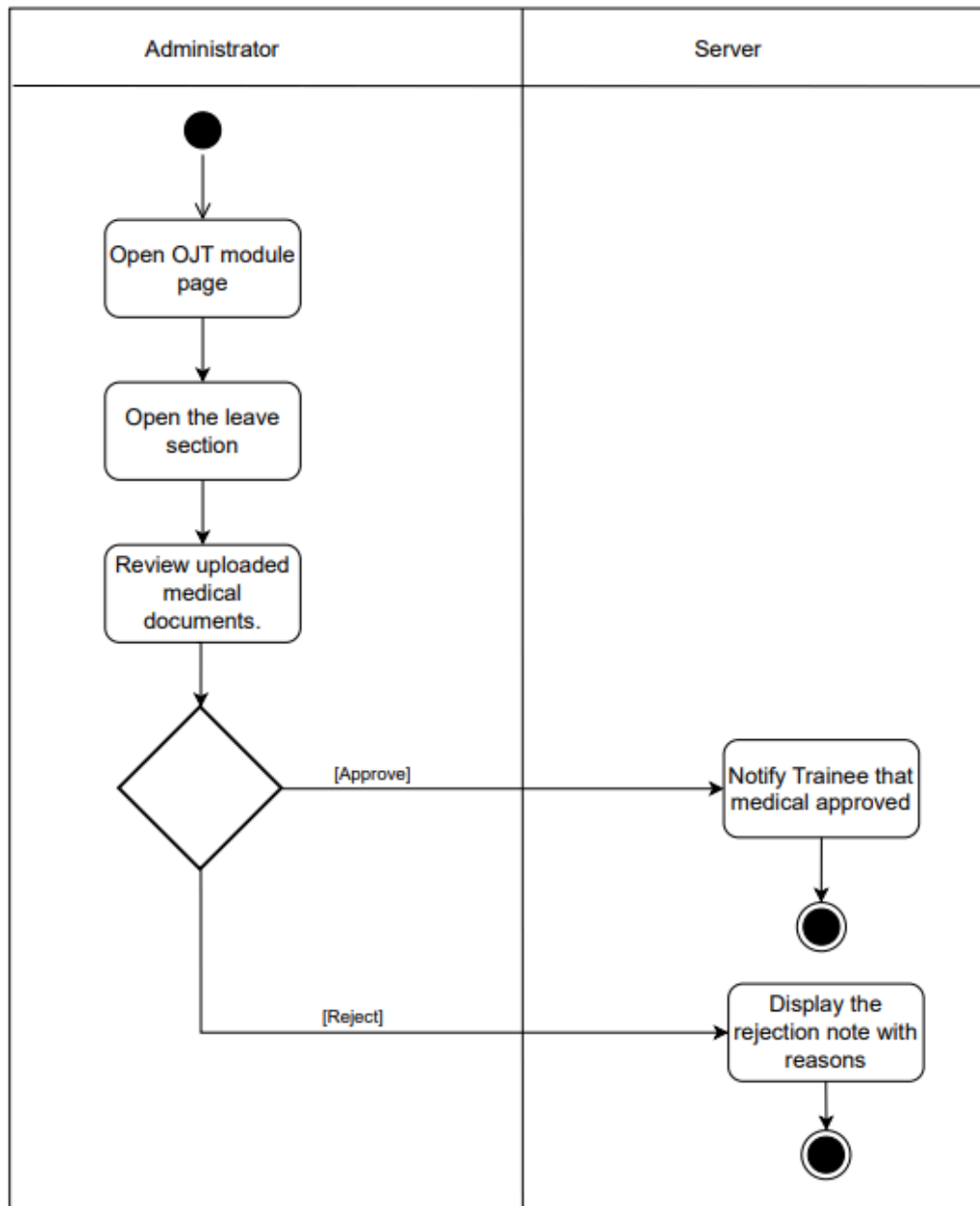
Review Log Submissions



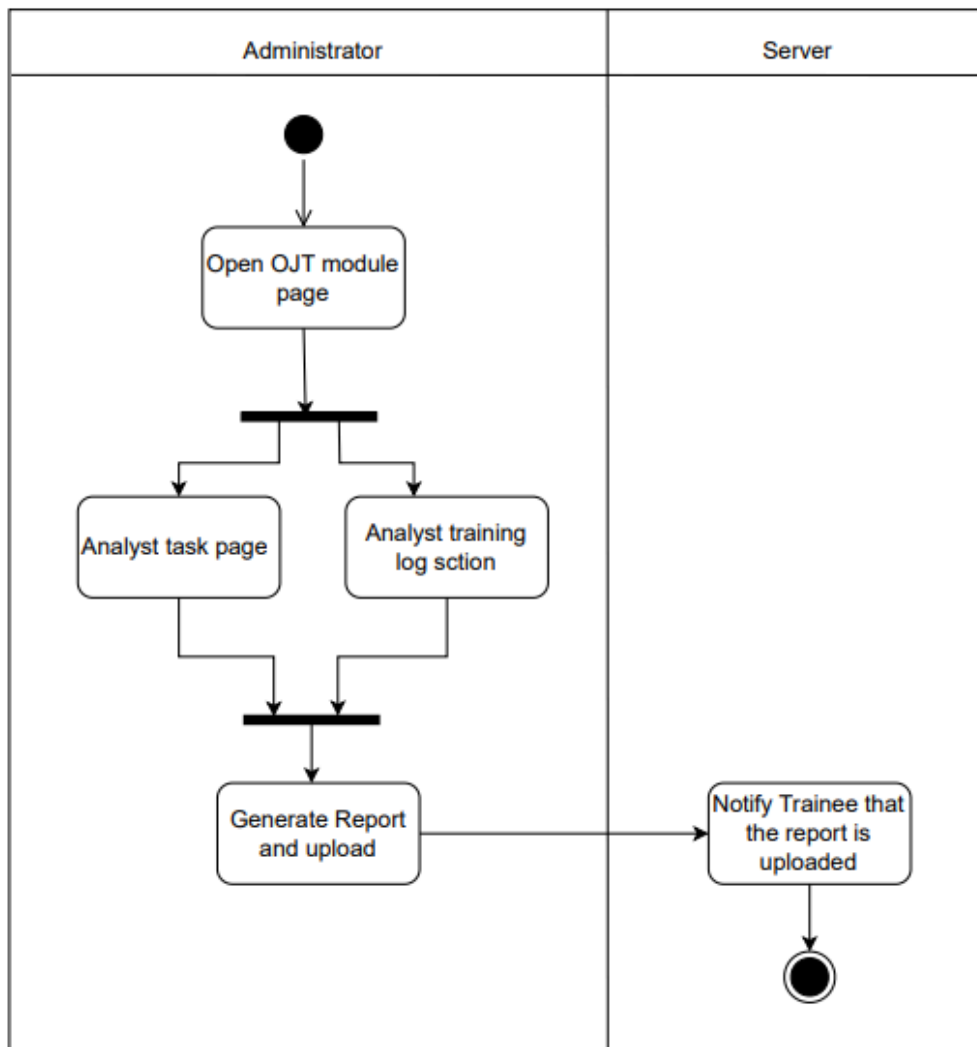
Provide Feedback



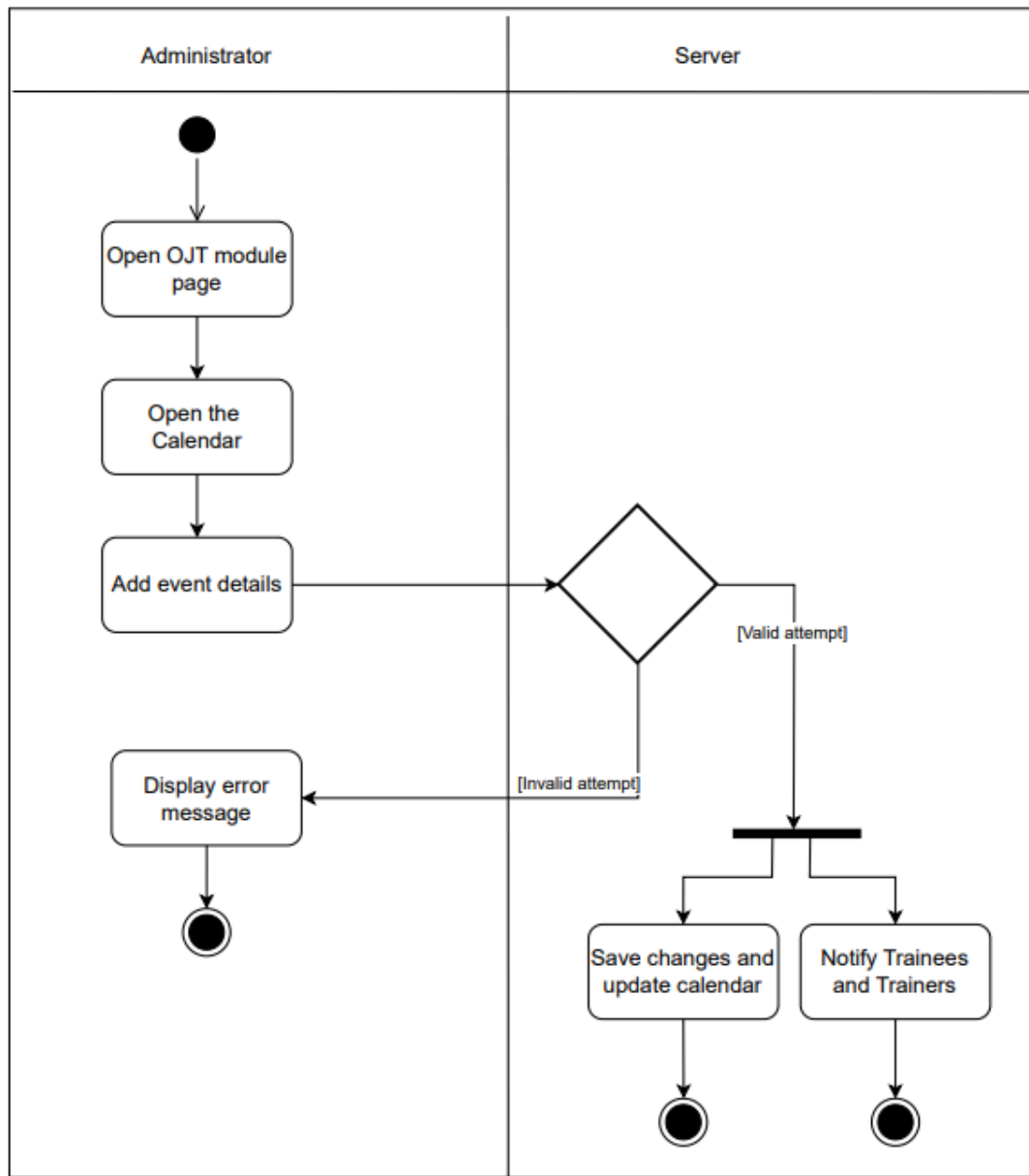
Approve/Reject Medical Report



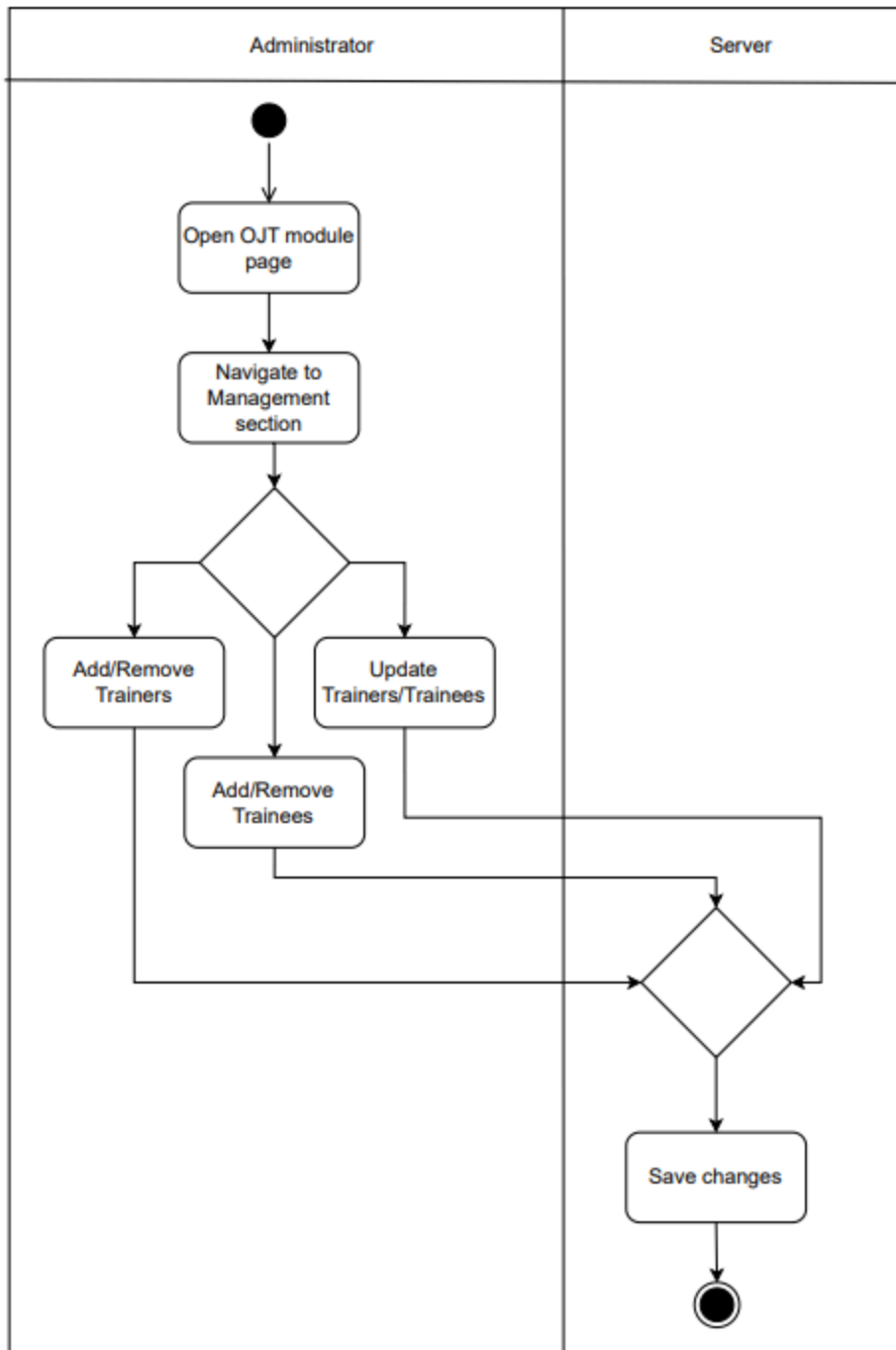
Generate Report



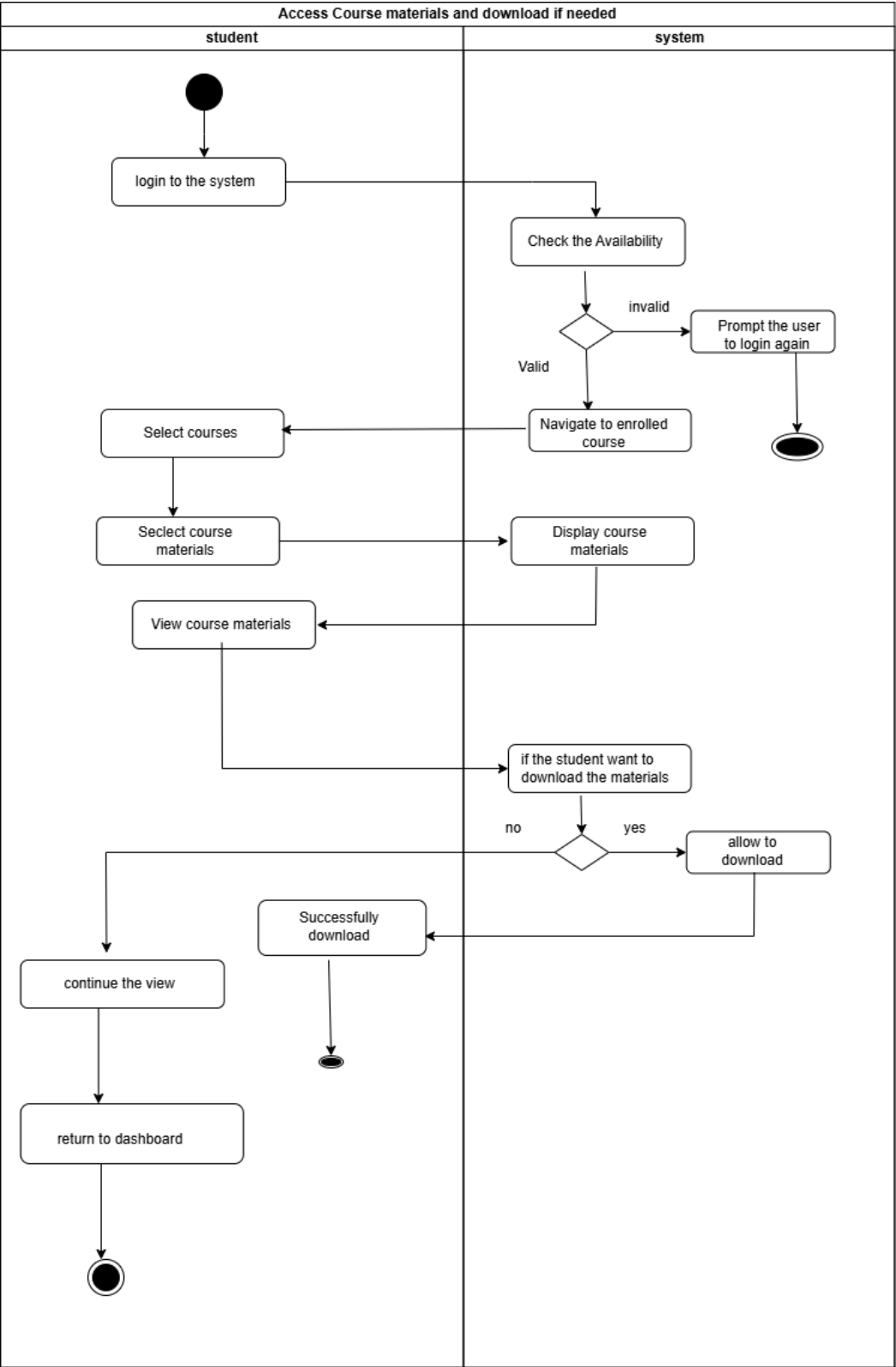
Schedule Calendar Events

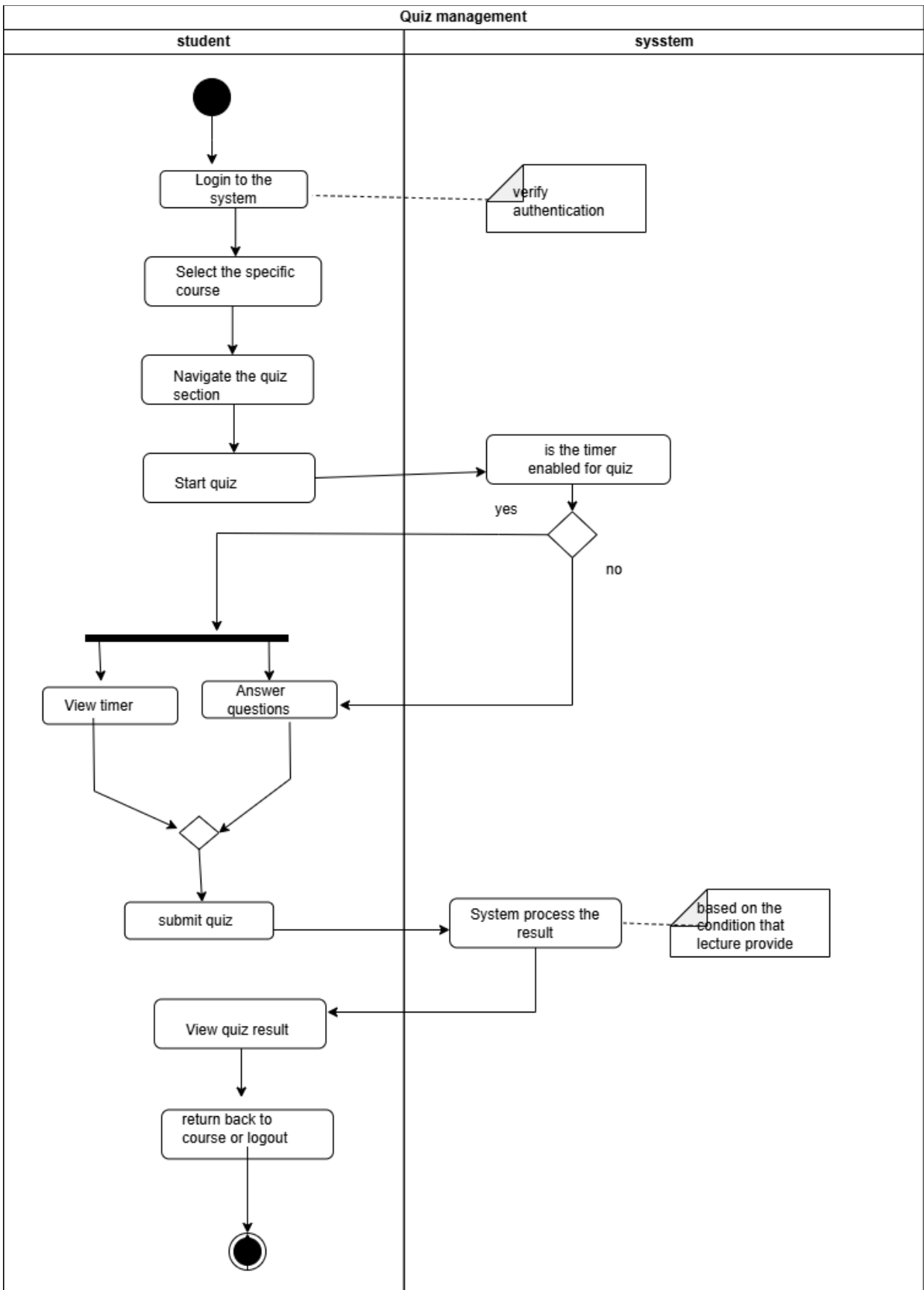


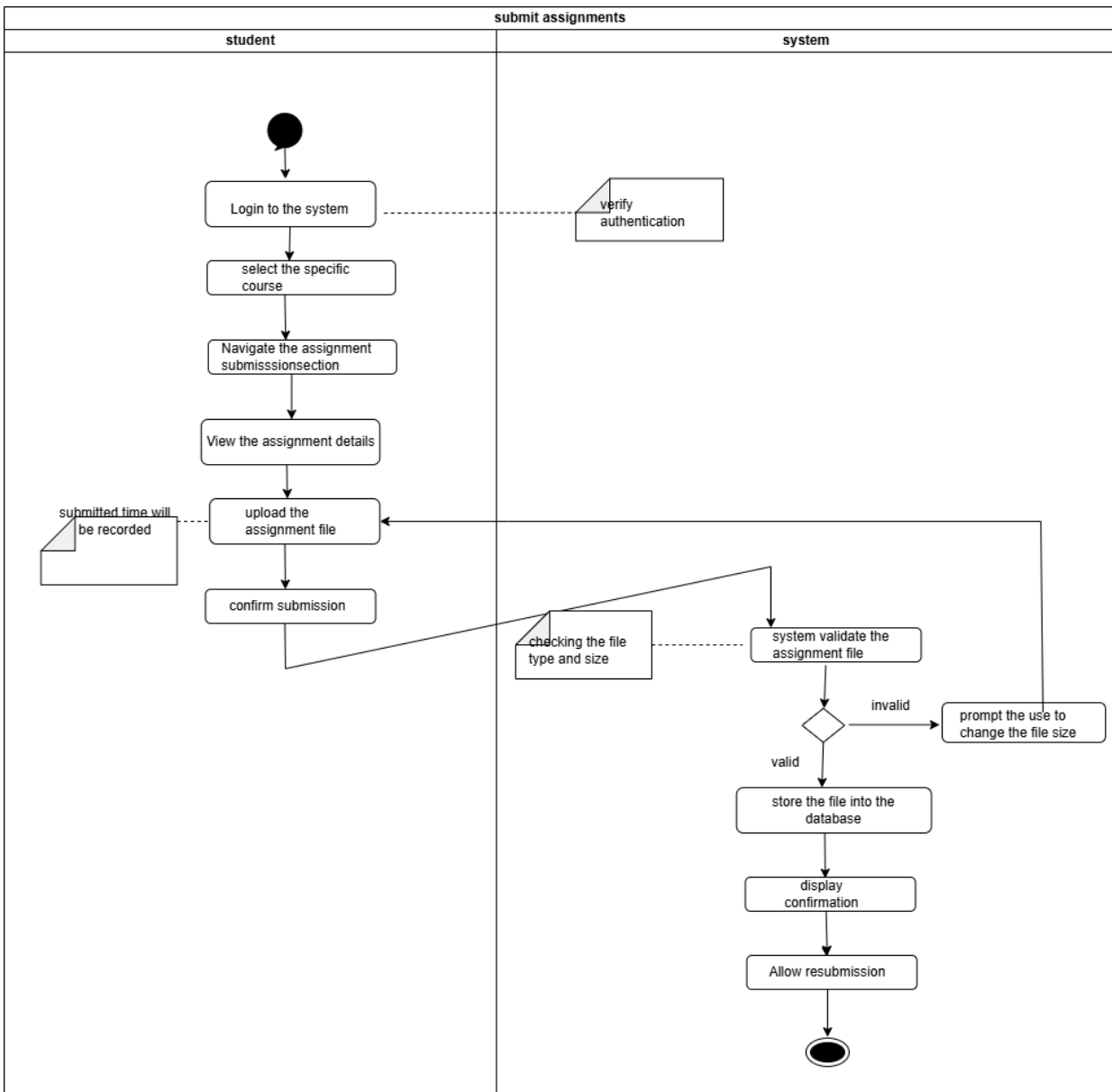
Manage Trainees and Trainers

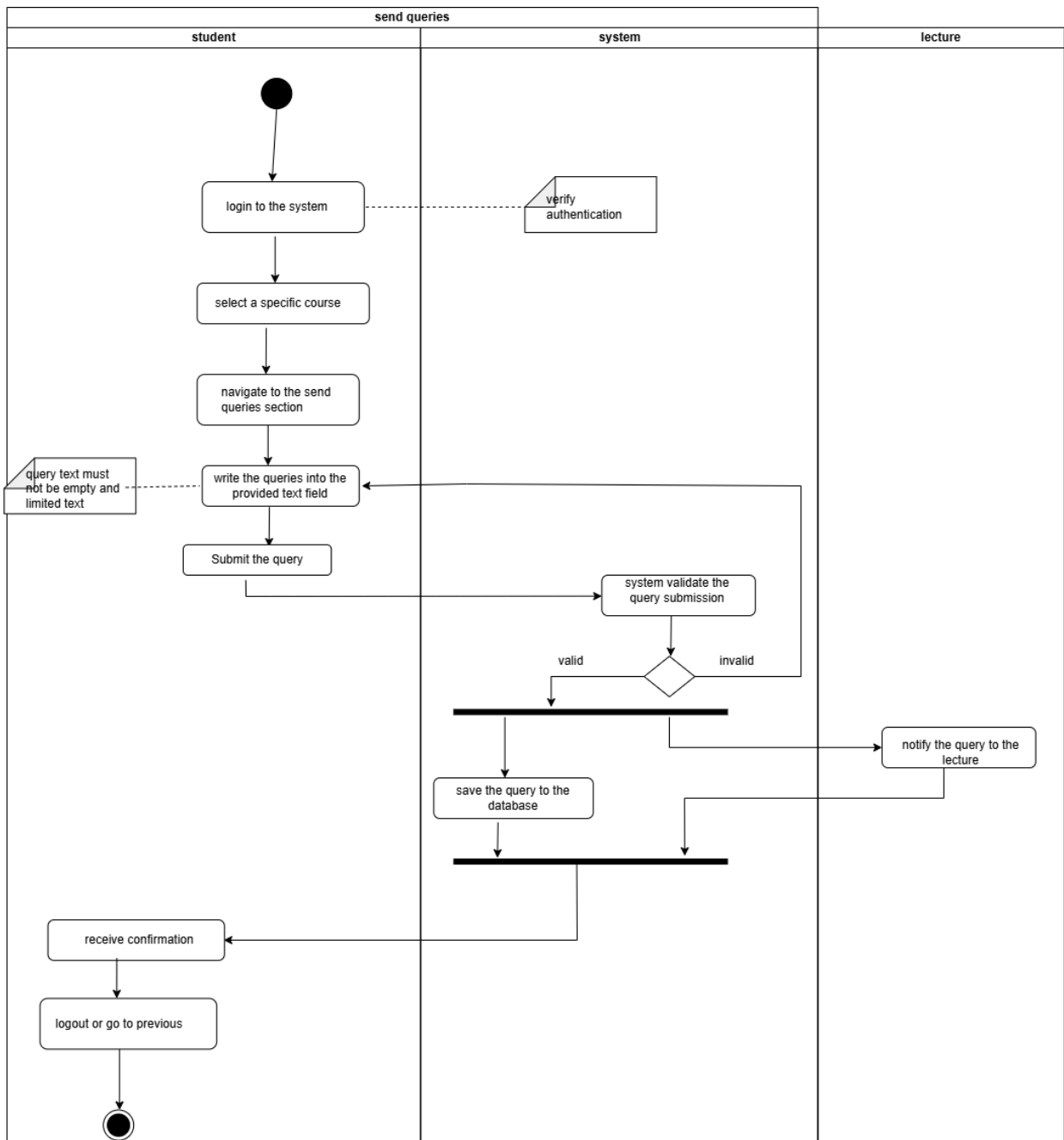


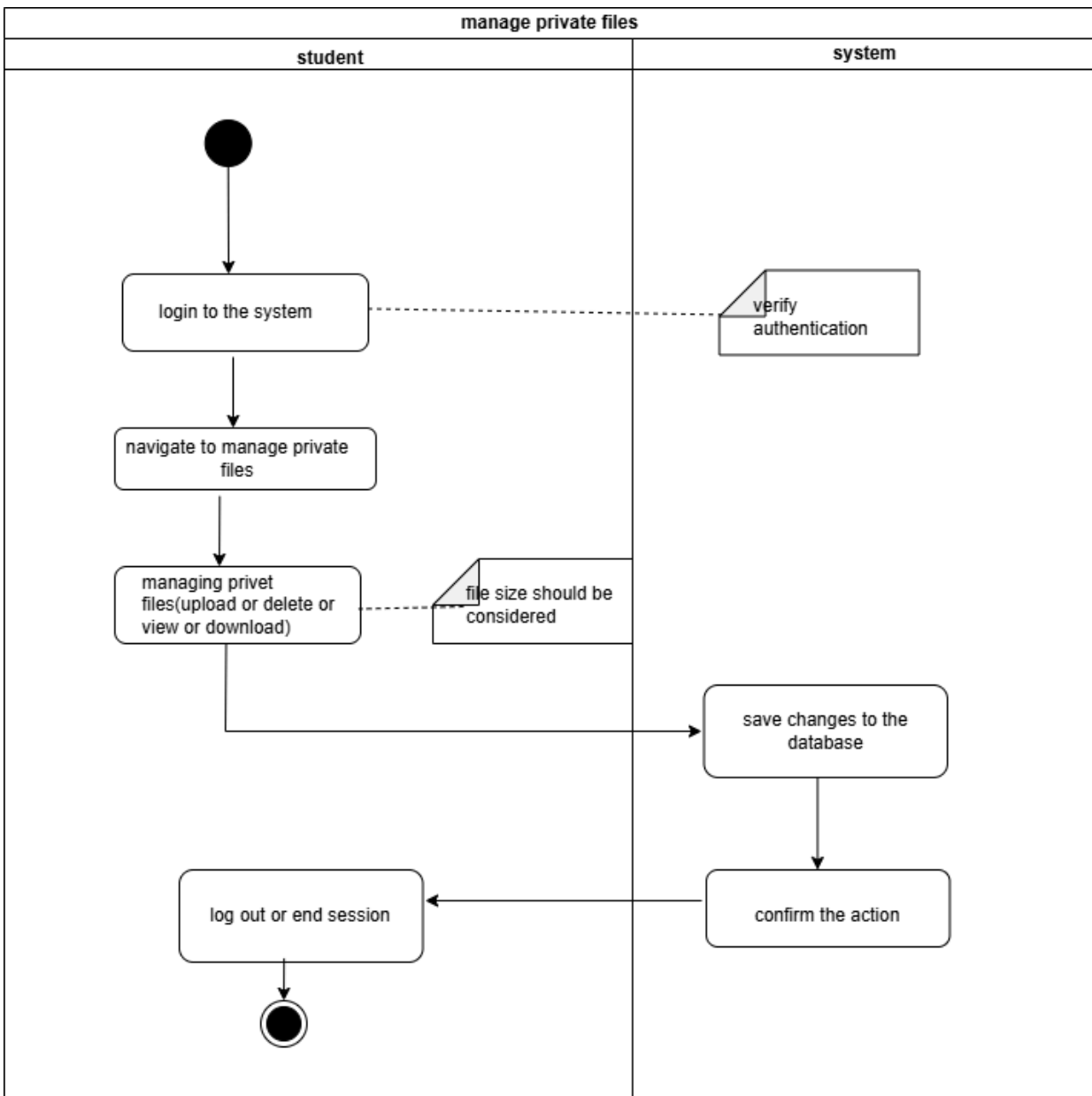
Student Interface

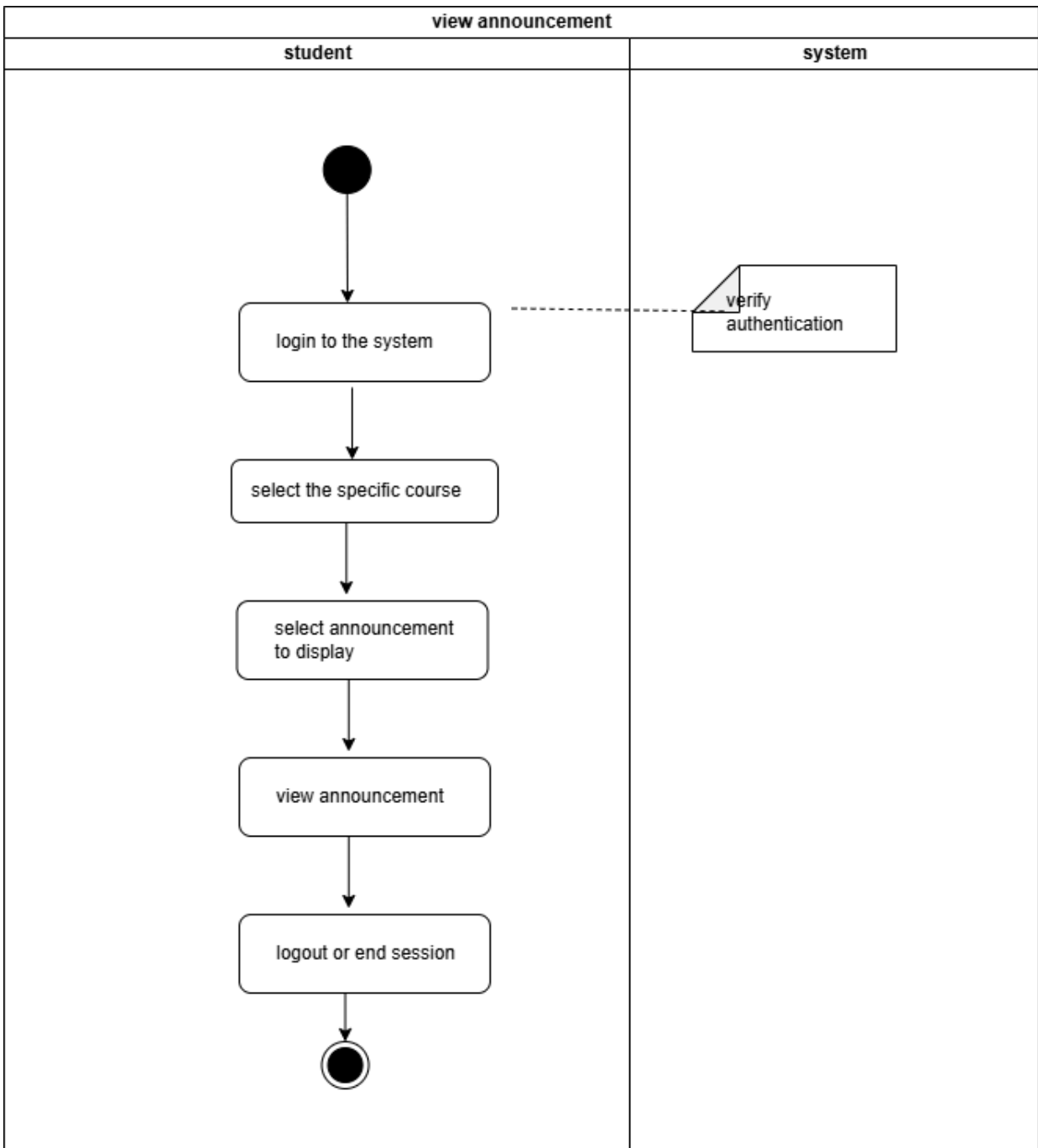


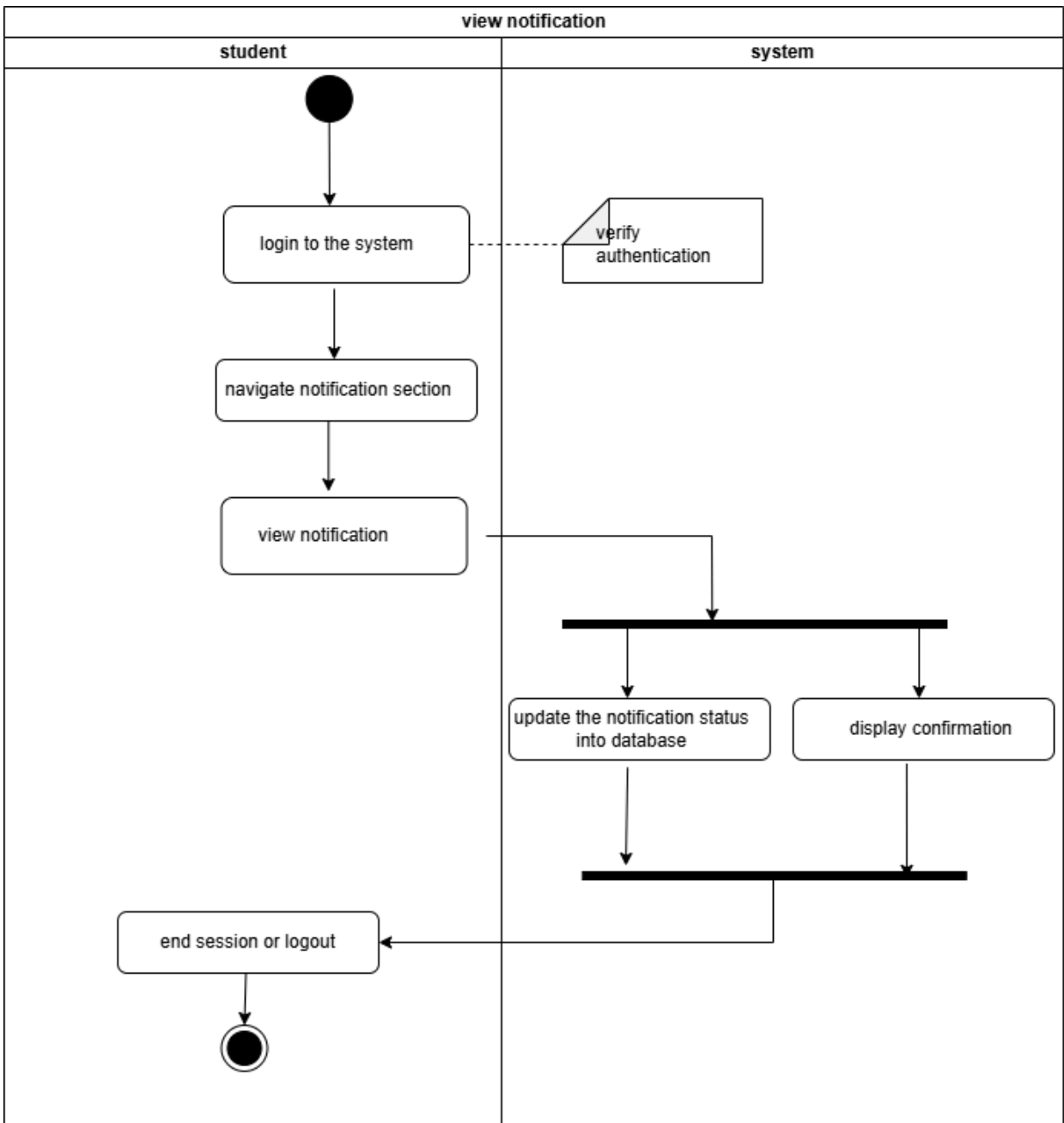




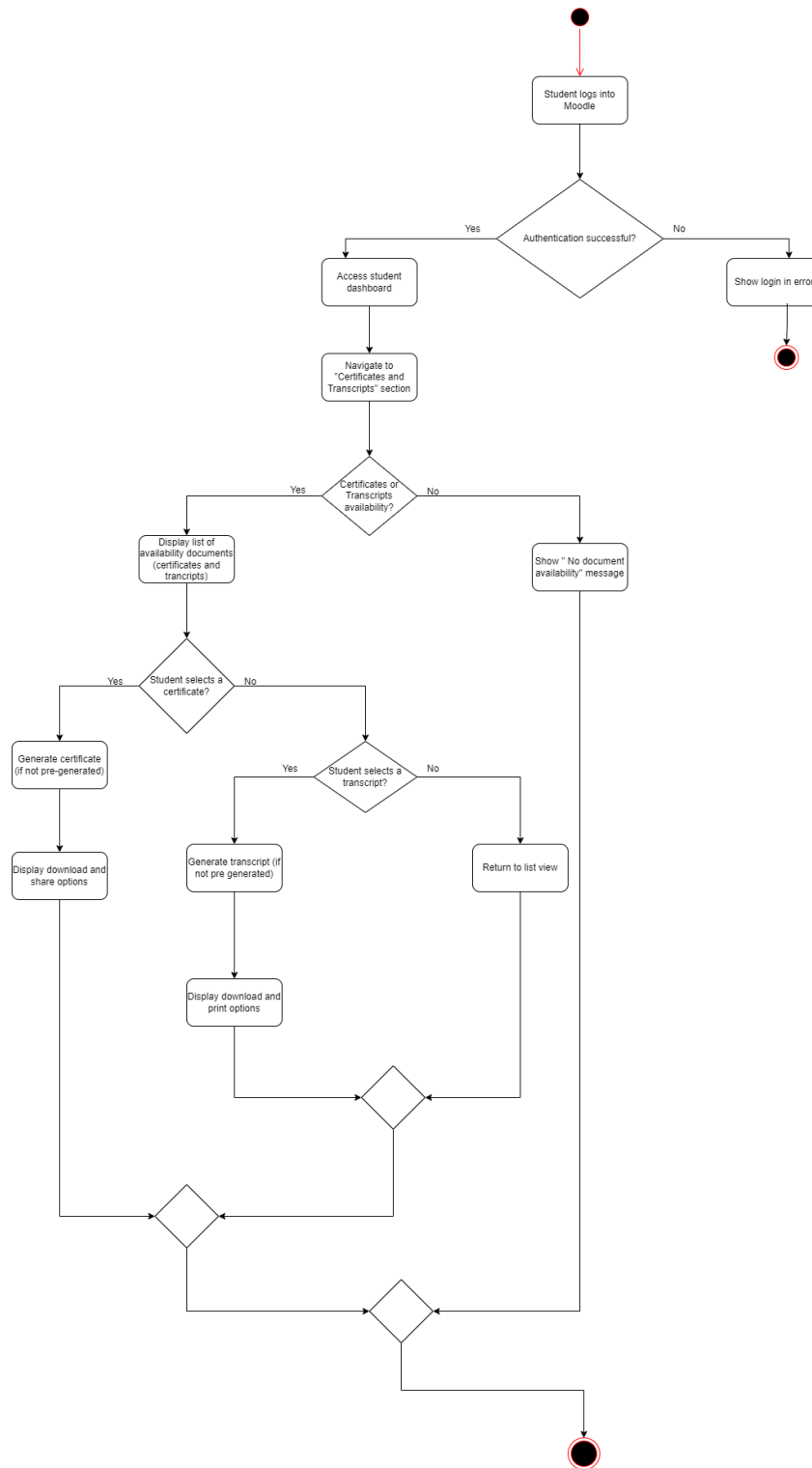








Certificate management and data management system



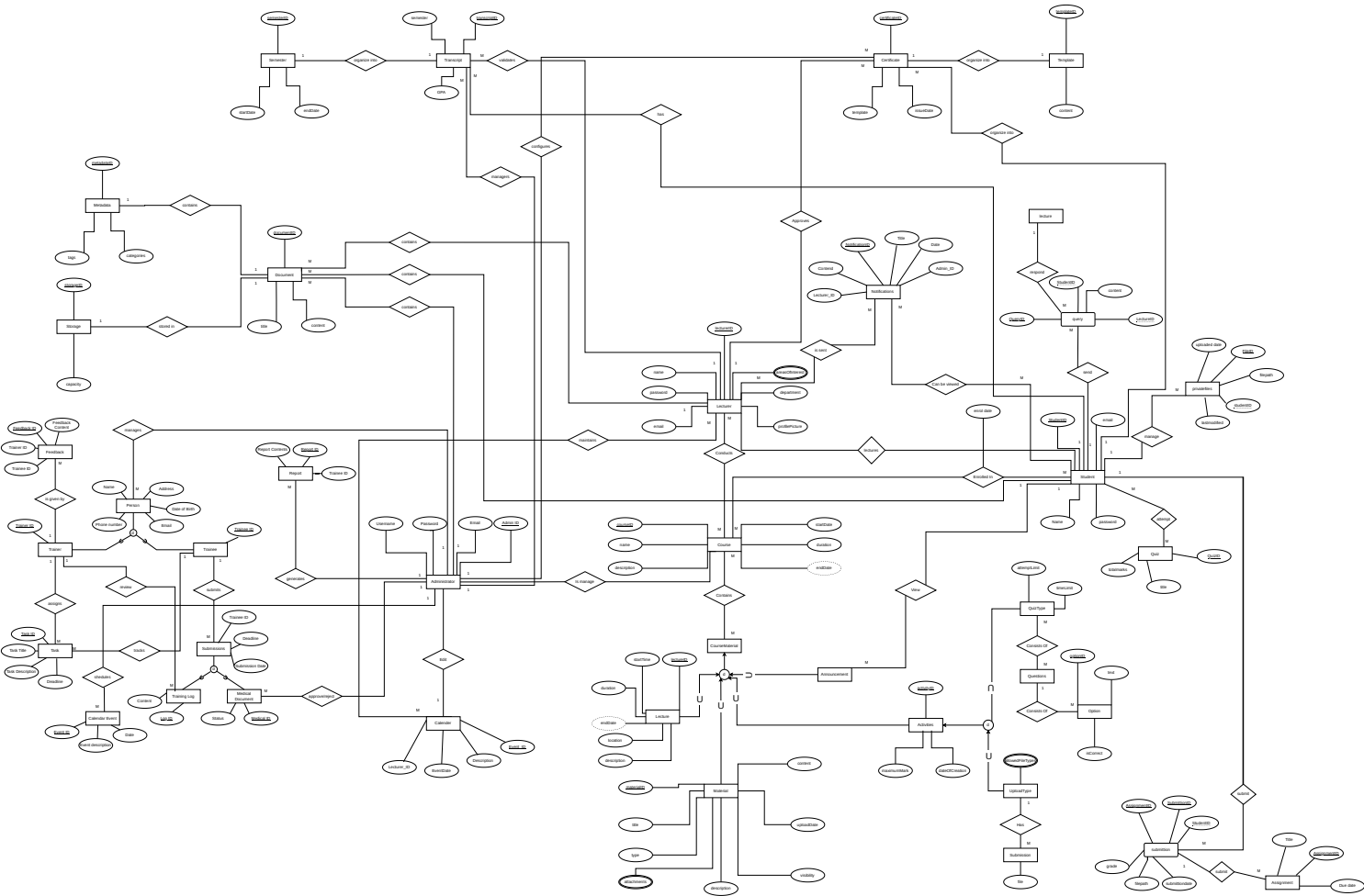
The following diagrams have been added in the next pages in order

5.3 Use Case Diagram

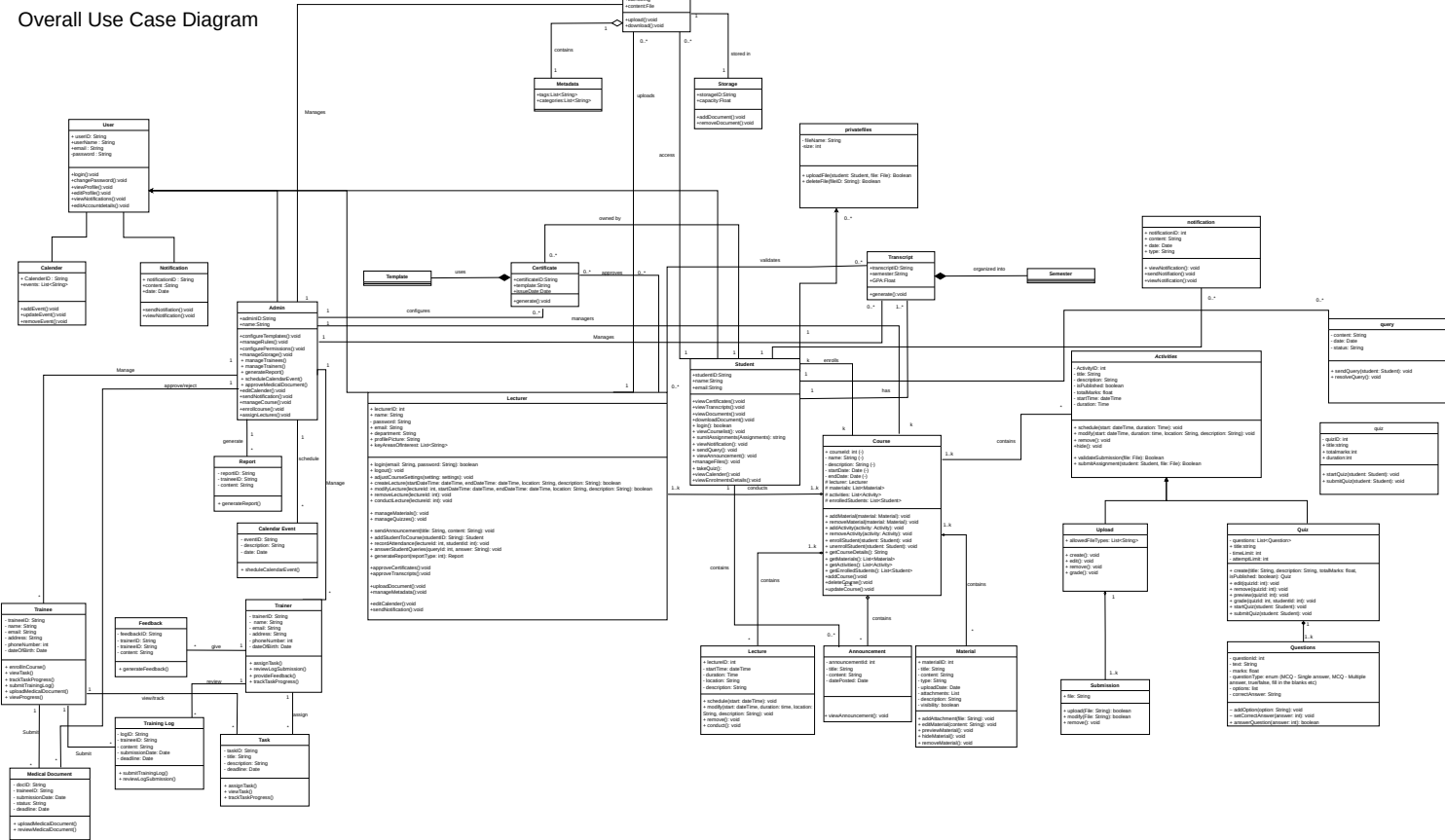
5.4 Class Diagram

5.5 ER Diagram

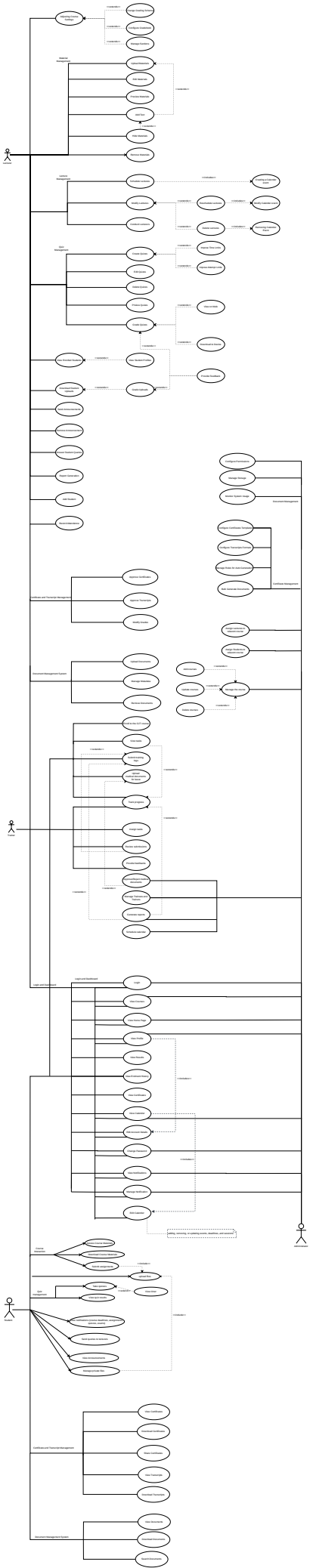
Overall Enhanced Entity Relationship Diagram

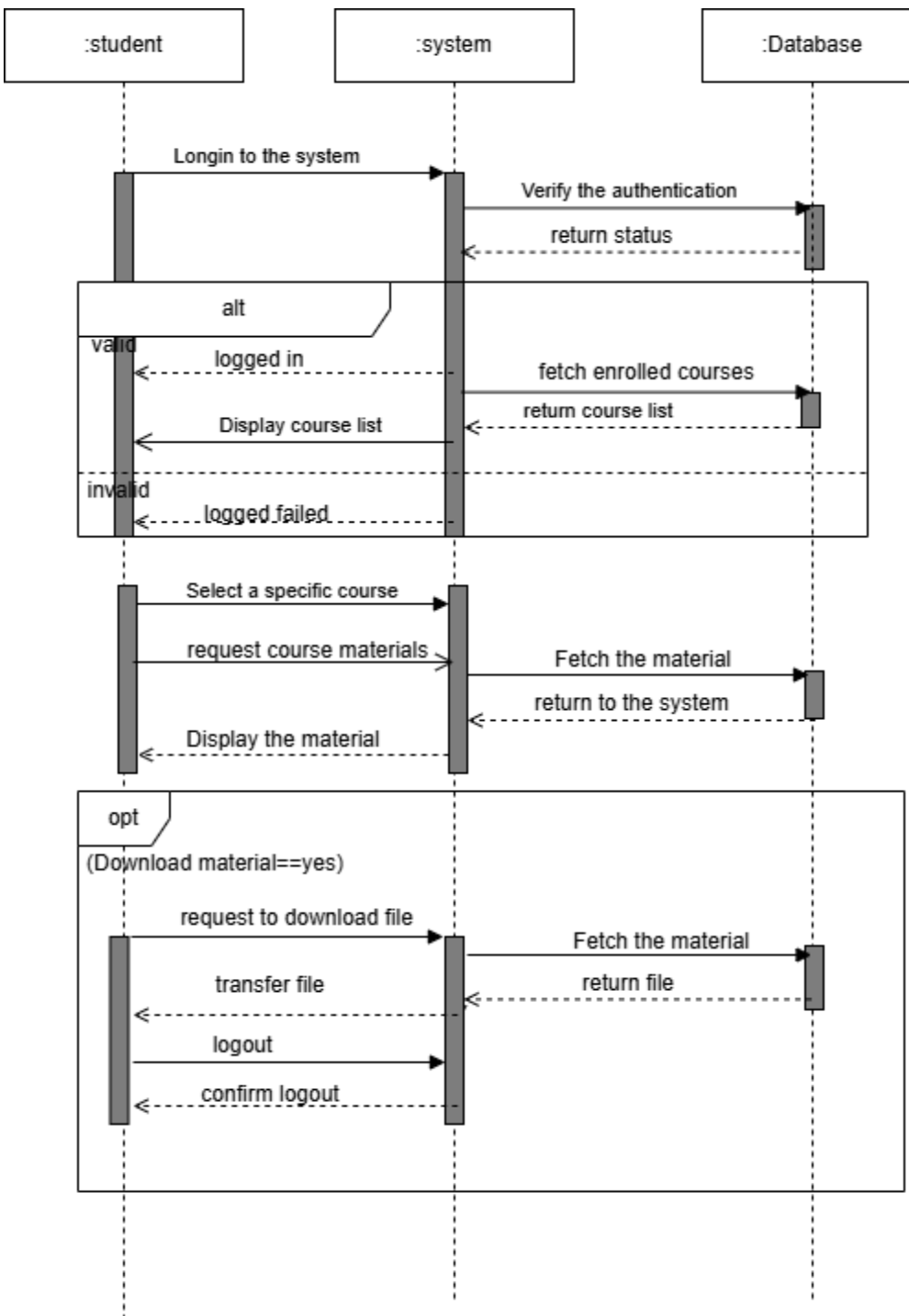


Overall Use Case Diagram

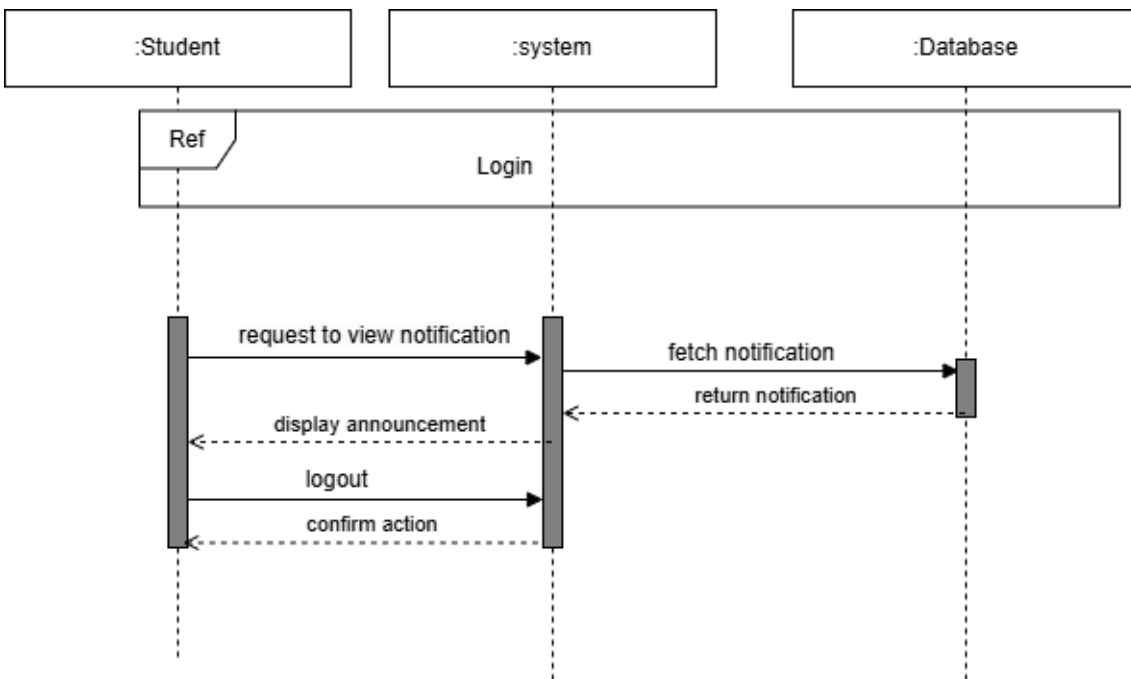


Overall Use Case Diagram

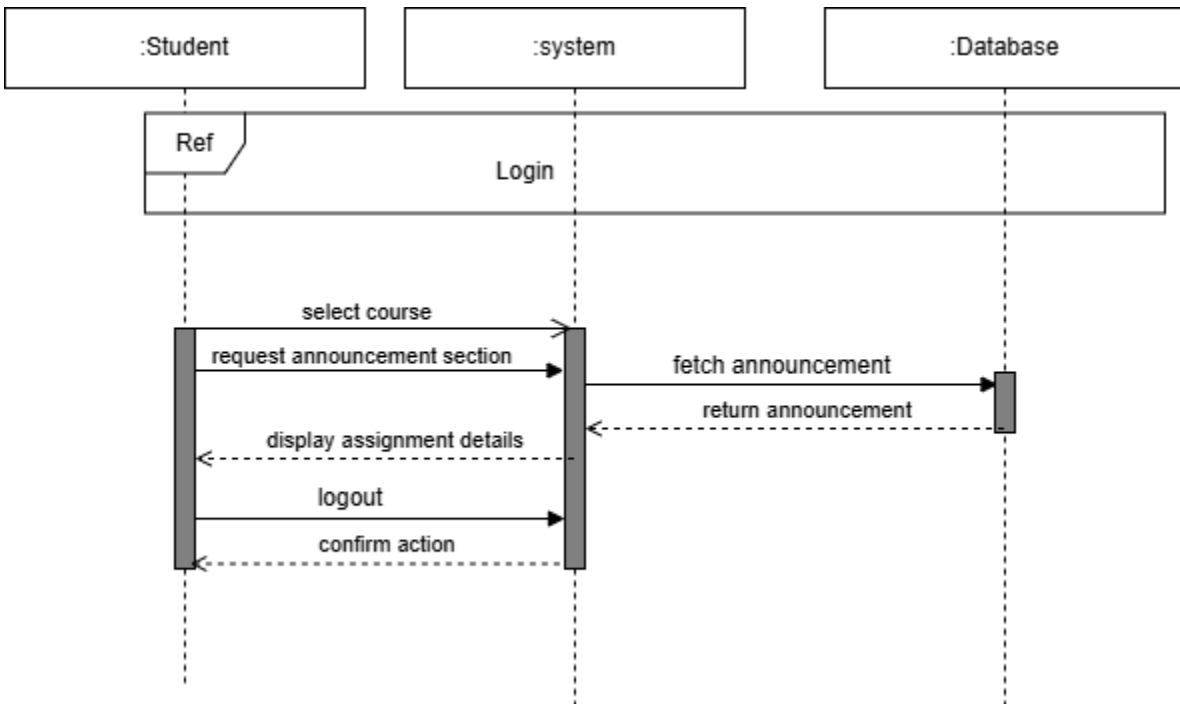




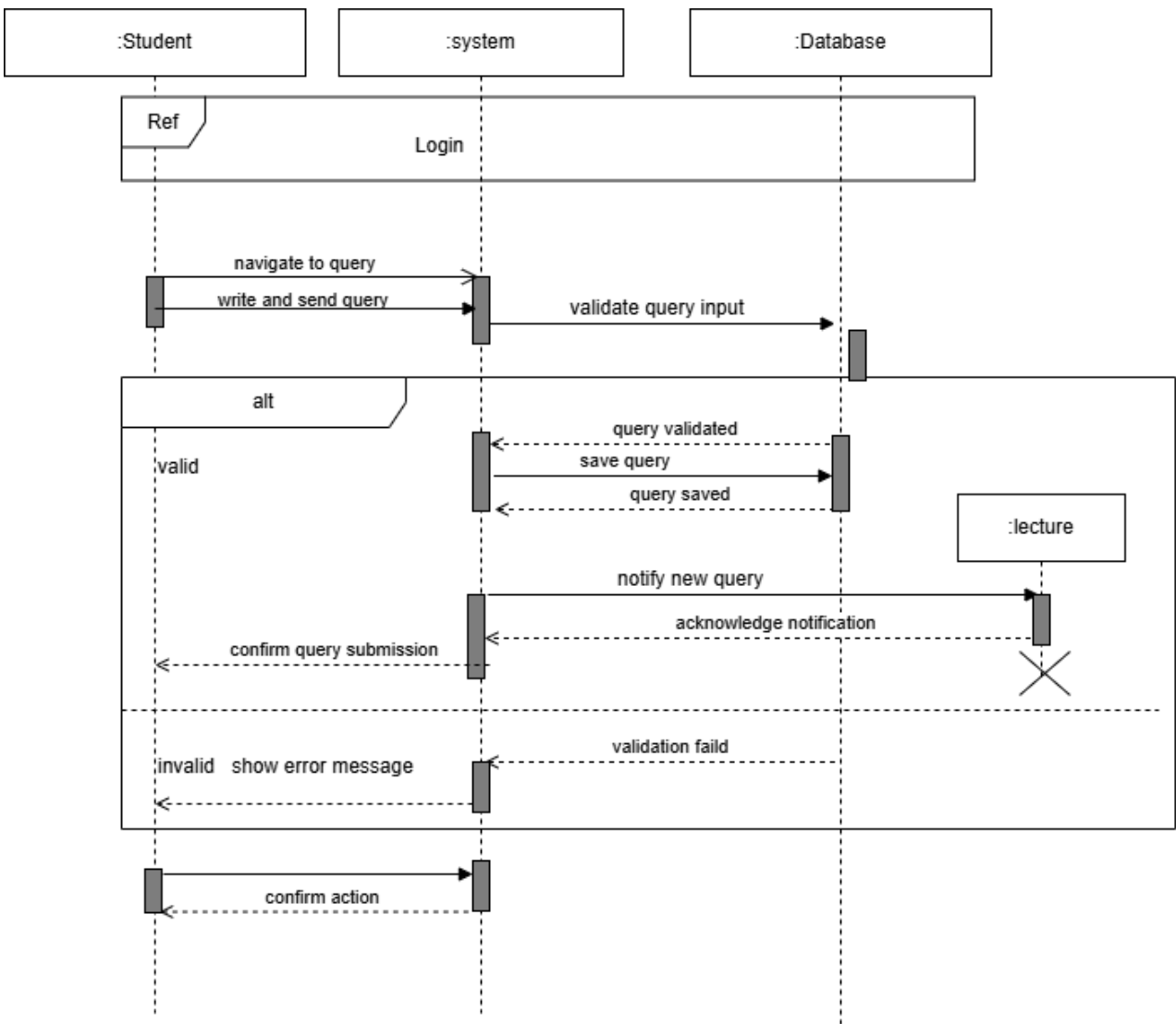
02.View notifications

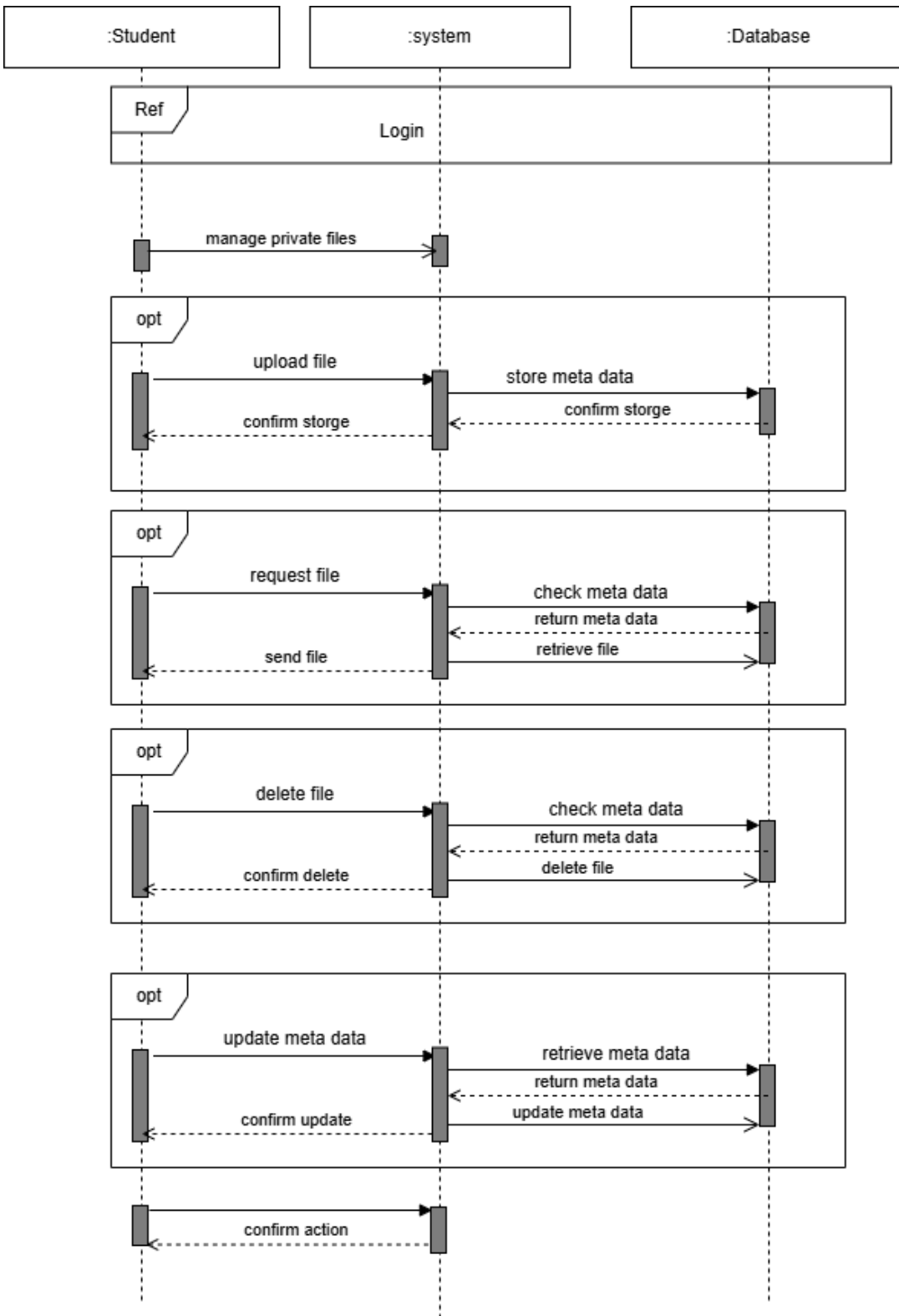


03. View announcement

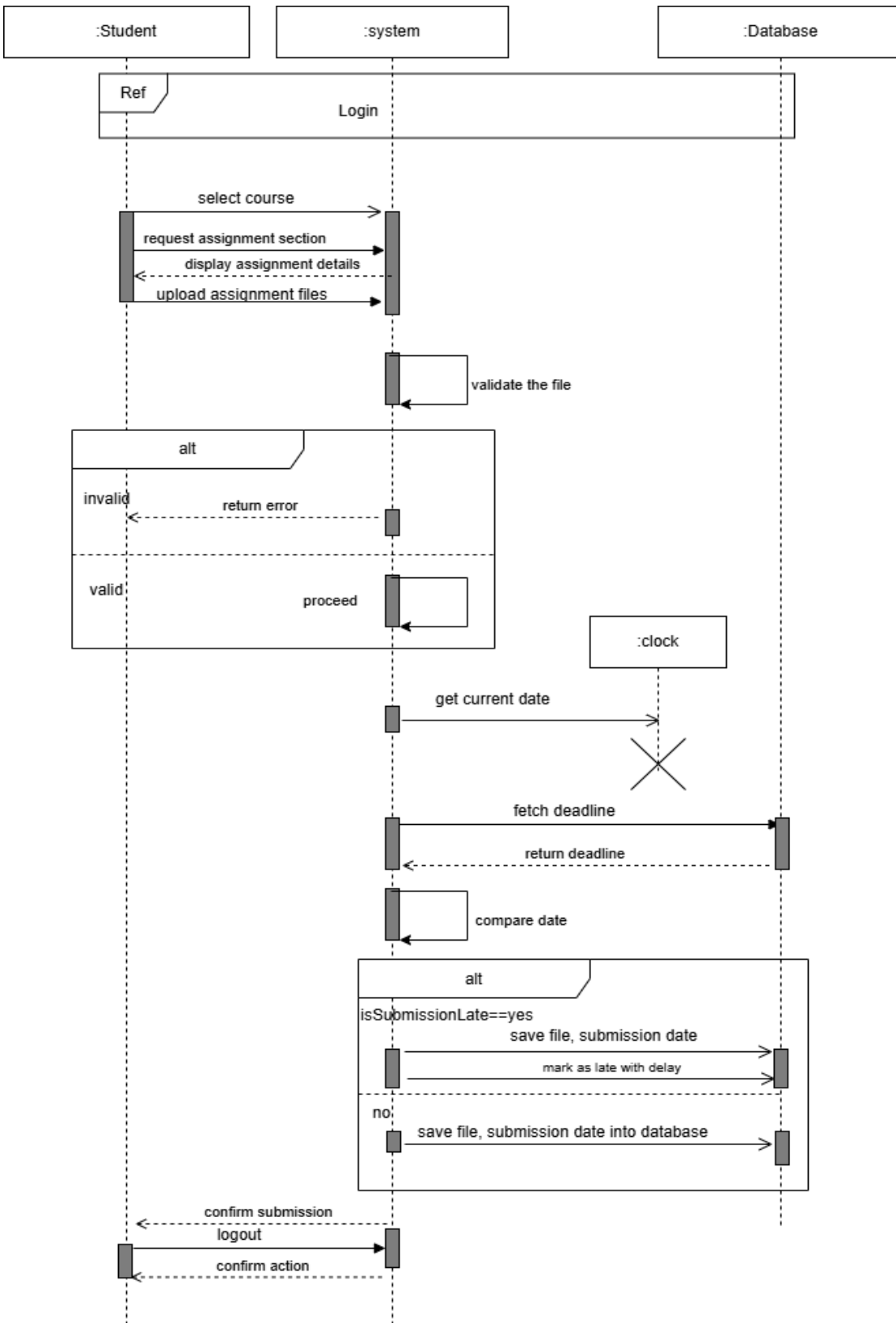


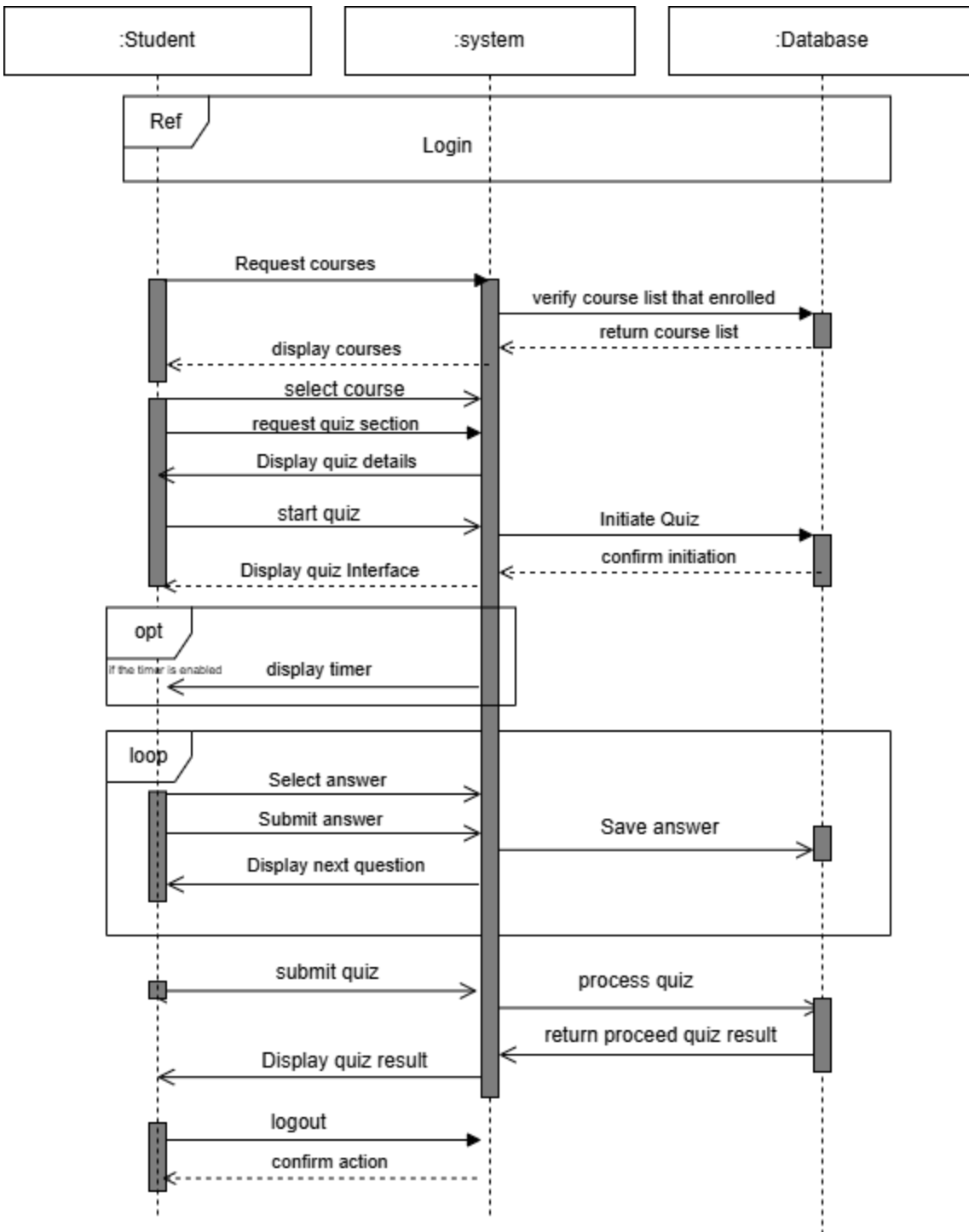
04. Send queries to the lectures





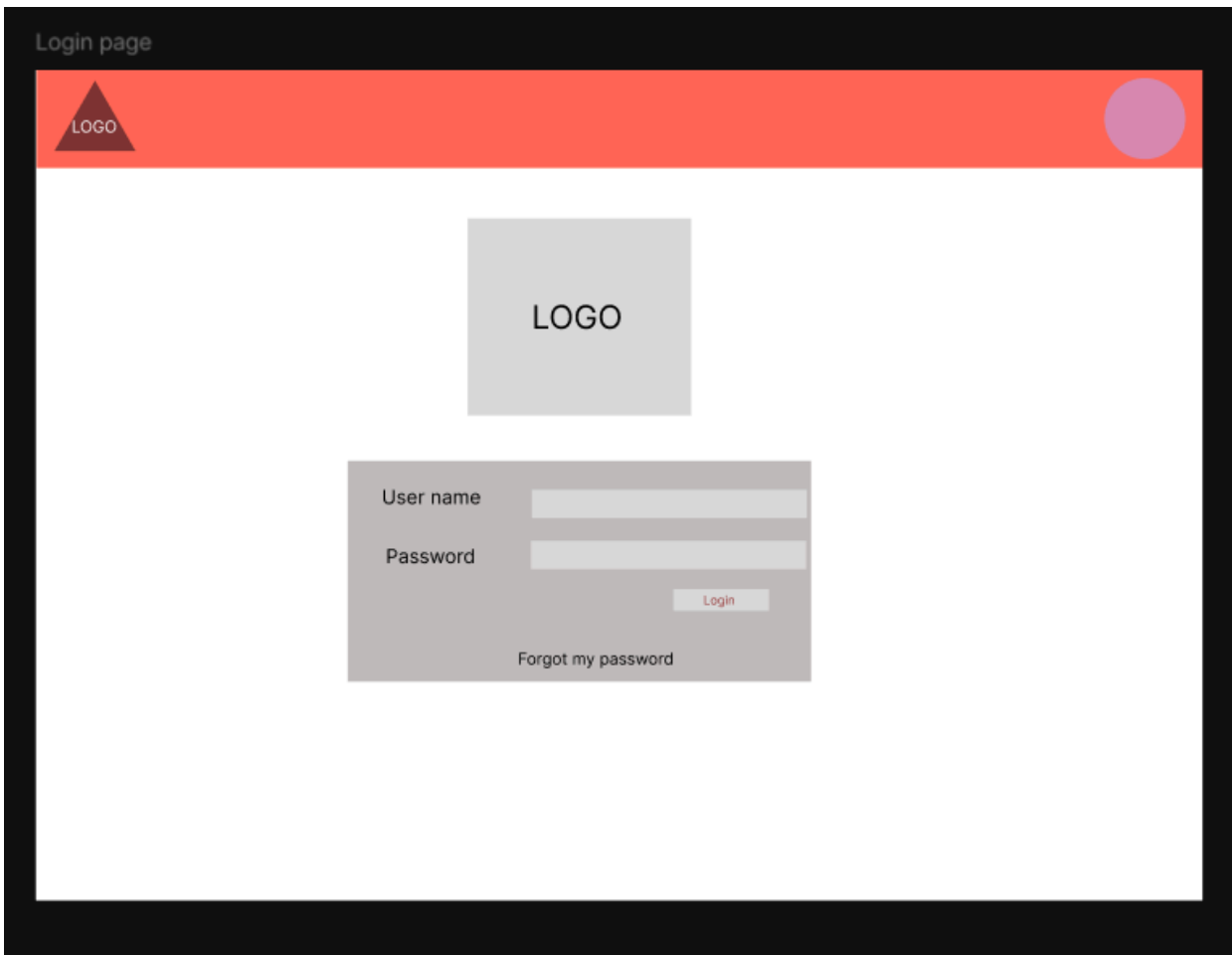
06.upload assignments



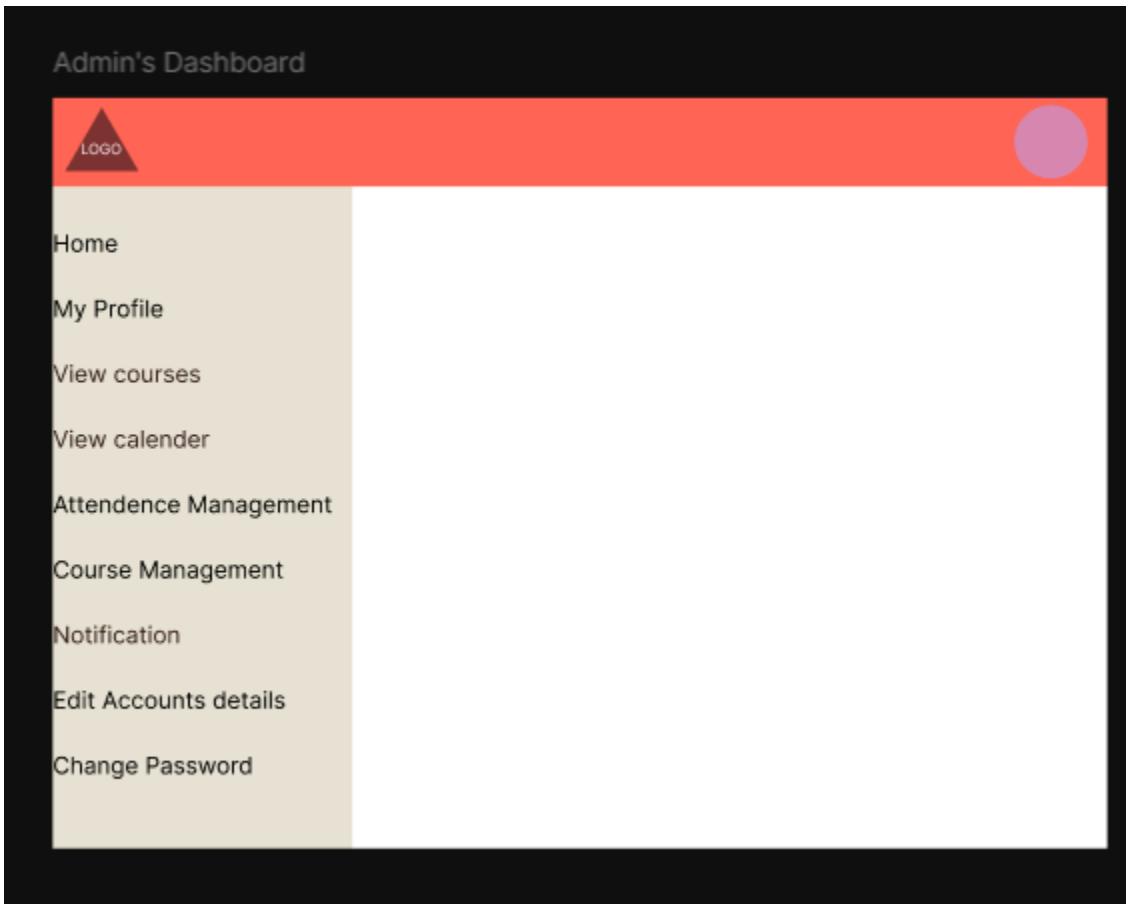


Chapter 6 – Implementation

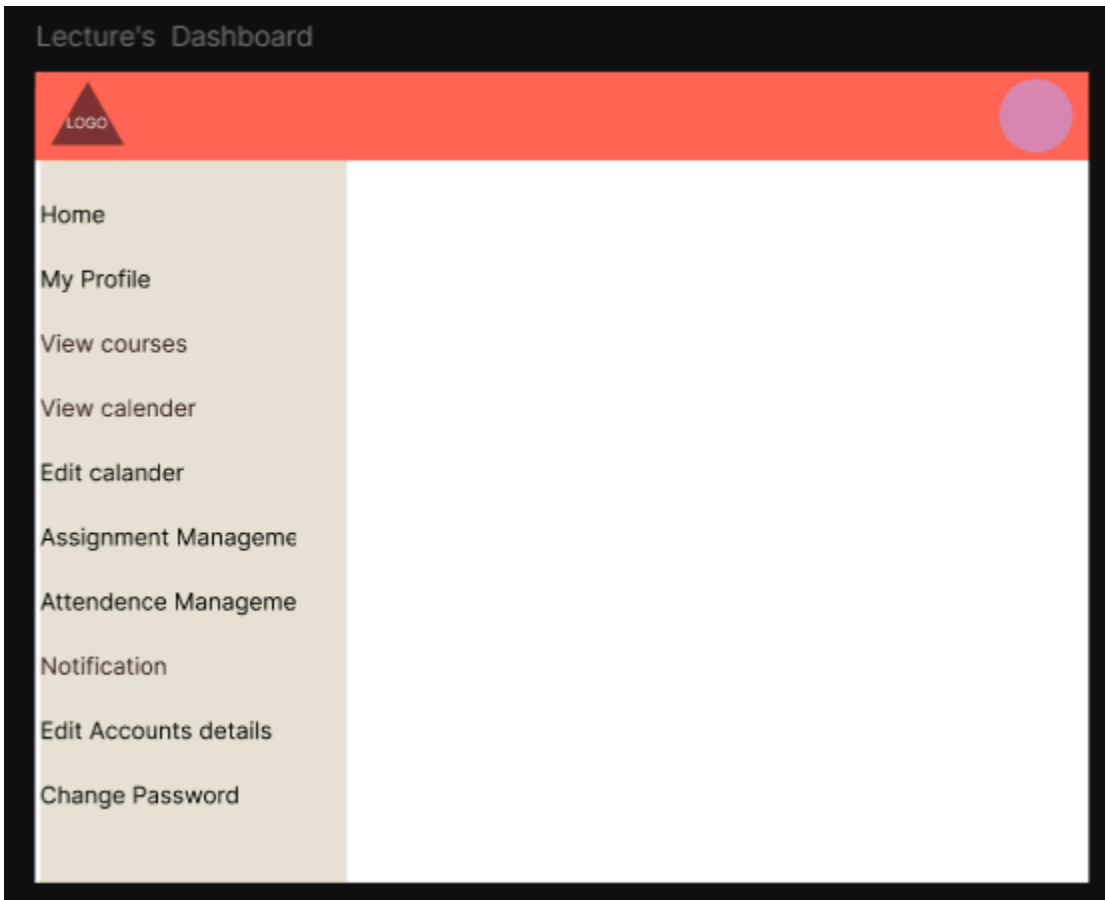
6.1 Login Page



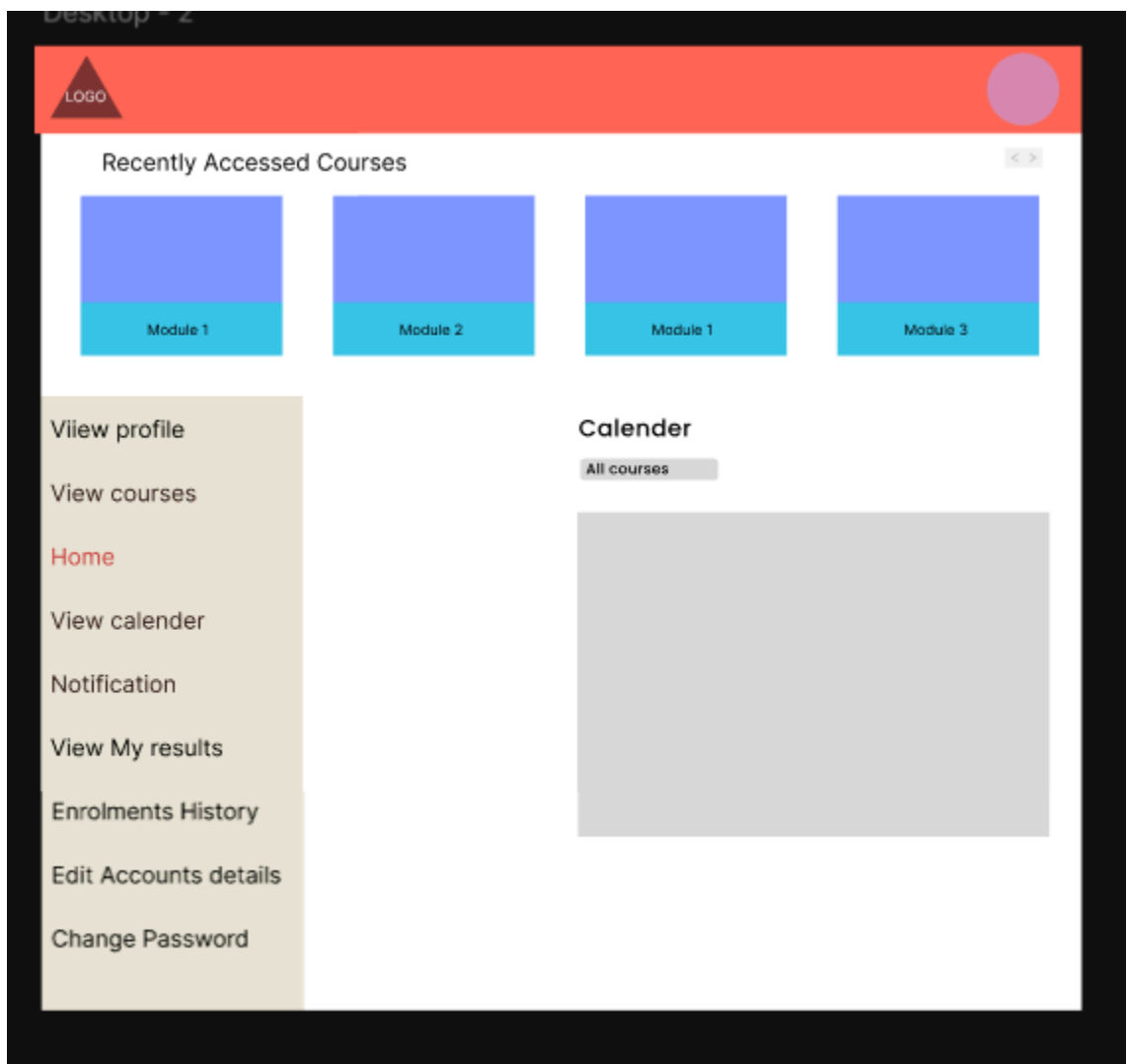
6.2 Administrator's Dashboard



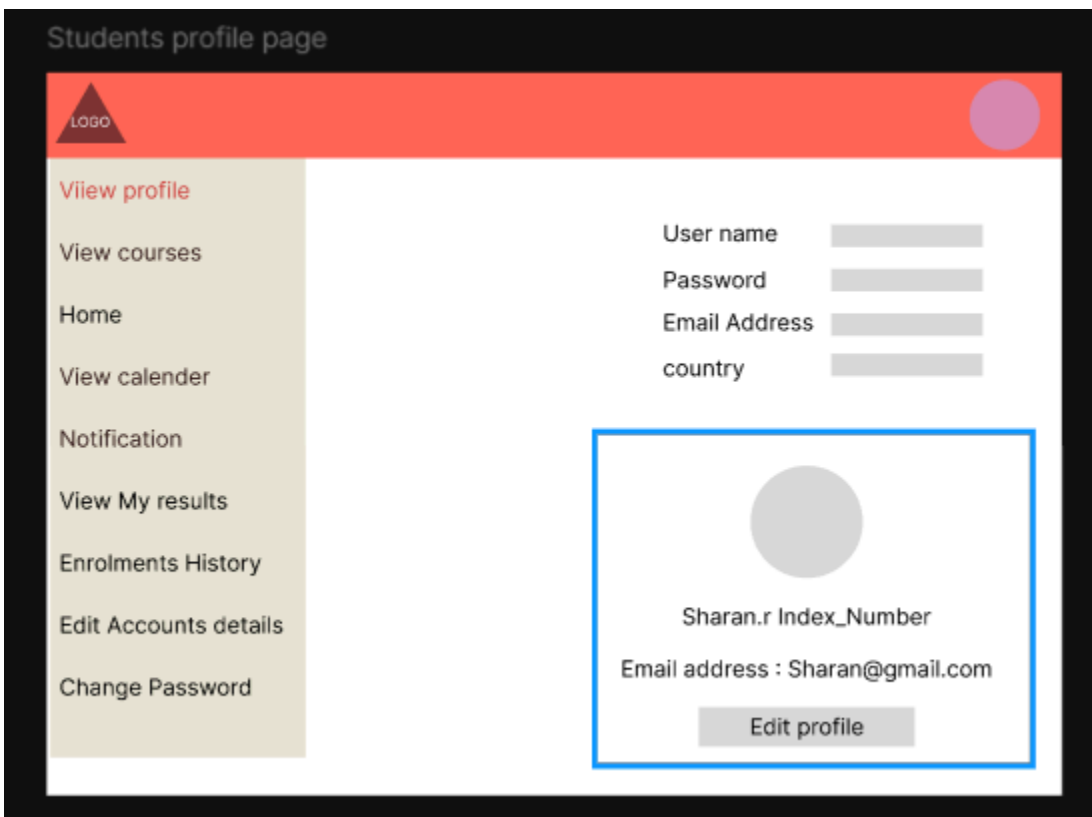
6.3 Lecturer's Dashboard



6.4 Student's Dashboard



6.5 Student's Profile Page View



The screenshot shows a web application titled "Students profile page". It features a red header bar with a "LOGO" icon on the left and a circular profile picture placeholder on the right. A left sidebar contains a list of navigation links: "View profile" (highlighted in red), "View courses", "Home", "View calender", "Notification", "View My results", "Enrolments History", "Edit Accounts details", and "Change Password". The main content area displays a form for profile information with fields for "User name", "Password", "Email Address", and "country". Below these fields is a profile summary box with a circular placeholder, the text "Sharan.r Index_Number", "Email address : Sharan@gmail.com", and an "Edit profile" button.

Students profile page

LOGO

View profile

View courses

Home

View calender

Notification

View My results

Enrolments History

Edit Accounts details

Change Password

User name

Password

Email Address

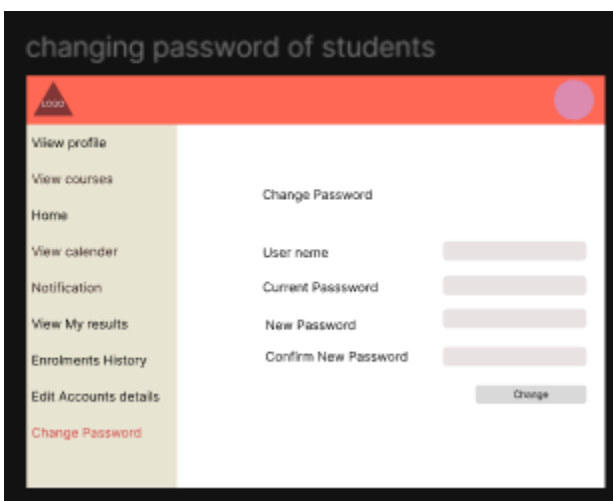
country

Sharan.r Index_Number

Email address : Sharan@gmail.com

Edit profile

6.6 Student's Password Changing Page View



The screenshot shows a web application titled "changing password of students". It features a red header bar with a "LOGO" icon on the left and a circular profile picture placeholder on the right. A left sidebar contains a list of navigation links: "View profile", "View courses", "Home", "View calender", "Notification", "View My results", "Enrolments History", "Edit Accounts details", and "Change Password" (highlighted in red). The main content area displays a form for changing the password with fields for "User name", "Current Password", "New Password", and "Confirm New Password", followed by a "Change" button.

changing password of students

LOGO

View profile

View courses

Home

View calender

Notification

View My results

Enrolments History

Edit Accounts details

Change Password

Change Password

User name

Current Password

New Password

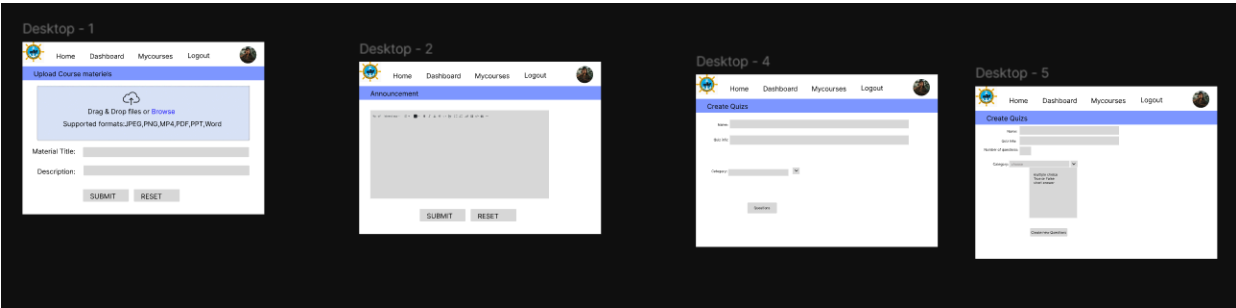
Confirm New Password

Change

6.7 Student's Interface



6.8 Lecturer's Interface



Appendix A – Individual Contribution

224112A - Kulasekera B.H.A.R

My contribution for the project is the Lecturer interface feature, and it will be having a the following features, to make the lecturing experience better than the current implementation, at the MPMA (Mahapola Ports and Maritime Academy).

- Adjusting Course Settings:
 - Grading Schema:
 - Defining weighting for different assessment types (e.g., quizzes, assignments, participation).
 - Grade-book:
 - Viewing the gradebook with student scores and grades. Manually entering or editing grades.
 - Exporting the gradebook.
 - Calculating and displaying statistics.
 - Sections in the Course:
 - Creating and managing different sections or groups within a course. Allowing tailoring content or activities to specific sections.
- Material Management:
 - Uploading material:
 - Uploading various file types (documents, PDFs, presentations).
 - Organizing materials into folders or modules.
 - Setting visibility for materials (visible or hidden).
 - Editing material:
 - Editing existing uploaded files , modifying descriptions, updating visibility settings and inline editing of text-based materials.
 - Previewing material:
 - Viewing materials as students would see them.
 - Hiding material:
 - Making materials temporarily unavailable to students.
- Lecture Management:
 - Schedule Lectures:
 - Setting date, time, duration, and location (physical (then mention room number)

- or virtual) for lectures.
 - Modify Lectures:
 - Changing the date, time, location, description, or linked resources of existing lectures.
 - Conduct Lectures:
 - Adding meeting links and linking recordings
- Quiz Management:
 - Creating Quizzes:
 - Adding different question types (multiple choice, true/false, fill-in-the-blanks, essay). Setting question weights and point values. Select correct answers and feedback for each question. Setting overall quiz parameters (time limit, attempts allowed, open/close dates).
 - Editing Quizzes:
 - Modifying existing questions, answers, settings, and parameters of a quiz.
 - Delete Quizzes:
 - Removing quizzes entirely from the course.
 - Preview Quizzes:
 - Viewing the quiz as a student would to verify the questions and settings.
 - Grading Quizzes:
 - Automatic grading for objective question types. Manual grading for subjective questions (essays). Providing feedback on individual questions or the overall quiz. Releasing grades to students.
- Lecturer functions including works with Students:
 - Viewing student profiles. Communicating with individual students or groups (announcements, direct messages).
 - Answering student questions.
 - Managing student enrollment (adding/removing students – potentially), with Admin approval.
 - Grading Student Submissions and Quizzes:
 - Accessing submitted assignments.
 - Downloading or viewing submitted files.
 - Providing feedback (textual, file annotations).
 - Assigning grades or scores.
- Report Generation:
 - Generating reports on student performance (overall grades, individual assignment scores). Attendance reports. Quiz statistics. Potentially custom report generation based on specific criteria.
- Managing attendance:
 - Manually marking attendance for lectures or sessions.

- Viewing attendance records.
 - Generating attendance reports.
- OOAD, Figma Mockup and Approval Process:
 - An intensive OOAD has been conducted in these areas by me and draw.io was used as a tool, all design diagrams will be available on the Git repository that will be handed over to the MPMA at the end of the project.
 - A Figma Mockup will be used in the designing phase of the project.
 - The MockUp will be shown to the port authority for confirmation by the MPMA

224074G - T. A. Hettige

1. Certificate and Transcript Management System (CTMS)

- Automate the generation and management of certificates and transcripts.
- Allow students to view, download, and share their certificates and transcripts.
- Provide teachers with the ability to approve certificates and transcripts.
- Enable administrators to configure templates and manage rules for generation.

2. Document Management System (DMS)

- Facilitate the upload, storage, and retrieval of documents within Moodle.
- Allow students to access and download course-related documents.
- Provide teachers with tools to upload and categorize documents.
- Enable administrators to manage permissions and monitor storage usage.

3. Progress Overview

3.1 CTMS Progress

- Requirements Analysis:
 - Identified key features: certificate and transcript generation, approval workflows, and template configuration.
 - Stakeholder interviews conducted with students, teachers, and administrators.
- Design Phase:
 - Use case and activity diagrams completed.
 - Class diagrams and ER diagrams finalized.
- Development Phase:
 - Basic user interface for students and teachers implemented.
 - Template generation for certificates under development.

3.2 DMS Progress

- Requirements Analysis:
 - Identified functionalities: document upload, categorization, and permission management.
 - Conducted usability studies to determine optimal workflows.
- Design Phase:
 - Use case and activity diagrams without swimlanes completed.
 - Class diagrams and ER diagrams finalized.
- Development Phase:
 - Core document upload functionality implemented.
 - Metadata management module under testing.

1. Administrator Dashboard:

- Manage courses
- Assigning lectures and students to relevant courses.
- Give access to edit calendar details
- tracking the overall performance of students and lectures.
- Manage attendance of students
- Monitor notifications
- Review system performance and data analytics.

2. Lecturer Dashboard:

- Manage interact with the relevant course
- Update course content
- Track student progress and performance
- Manage attendance of students
- Upload lecture materials , schedule classes, Grade assignments, providing real-time feedback to students

3. Student Dashboard:

- The Student dashboard offers a personalized view for each student, providing access to enrolled courses, upcoming lectures, assignments, and progress tracking.
- Provide access to enroll courses
- Provide access to view upcoming lectures, assignments and progress tracking
- Students can view their grades, course materials, and deadlines.
- View notifications regarding class updates , assignments and announcements from lectures.

Each dashboard serves a distinct role within the LMS, contributing to an organized, efficient, and user-friendly educational experience. They are tailored to meet the needs of administrators, lecturers, and students, ensuring effective communication, management, and learning.

224256R – J.Dinuja

This report includes my contributions to the development of moodle based learning management system, focusing on the student interface feature, my primary responsibility was to design and implement workflows for seamless interaction between students and the system.

This will mainly include the following features , design to enhance the learning experience and improve the efficiency of the learning process compared with current system that is using at MPMA(Mahapola Ports and Maritime Academy) ,

- accessing course materials, and downloading them if needed,
- submitting assignments
- quiz management
- sending academic queries to the lectures
- managing private files (upload, delete, add, files).
- View Announcement and notifications

The student interface was modeled using use case ,activity,sequence,EER,class UML diagrams to ensure a robust design by using draw.io .These diagrams comprehensively map out workflows incorporating elements like authentication, real time feedback, automated notifications.In addition i have contribute to preparing interim report work with my group members as well and looked some similar products to get some ideas of the student interactions with the system .

224236G - M.B.F. Bushra

This report highlights my contributions to developing the Moodle-based LMS, specifically the On-the-Job Training (OJT) module. The module enables administrators to track trainee progress and manage documents, trainers to review logs and provide feedback, and trainees to submit logs, upload medical documents, and monitor their progress. The streamlined design ensures efficient management of OJT activities for all users.

Administrator View- The administrator's interface allows for comprehensive oversight and management of the OJT module. Administrators can:

- Oversee the submission and approval of medical documents.
- Access analytics and generate reports on OJT performance.
- Manages Trainers and Trainees.
- Schedule calendar events.

Trainer View- The trainer interface is equipped to support the evaluation and feedback process for trainees. Trainers can:

- Review and assess training logs submitted by trainees.
- Assign tasks and track task progress.
- Provide detailed feedback.

Trainee View- The trainee interface offers a user-friendly platform for managing all aspects of their OJT journey. Trainees can:

- Submit detailed training logs documenting daily activities, achievements, and challenges during their OJT.
- Upload medical documents for leave requests and check their approval status.
- View their progress, including completed tasks, reports and feedback.

Each role-based interface within the OJT module is carefully designed to cater to the specific needs of its users. The intuitive design and seamless integration with the LMS ensure efficient communication, tracking, and management of On-the-Job Training activities at MPMA.

Appendix B – Action Plan

Time Duration	Nov				Dec				Jan				Feb				Mar				Apr				May				Jun			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Project proposal preparation																																
Design structural diagram																																
Design behavior diagram																																
Design wireframes																																
Implement UI, Interim Report & Presentation																																
Implement basic interactions																																
Connecting database																																
Completing backend																																
Completing functional Requirements																																
UX/UI tuning & Draft Final Report																																
Final product testing																																
Final submission Final Report & Presentatio																																

Chapter 7 – Reference

[1]	UML Diagrams Resource: www.visual-paradigm.com
[2]	Lucidchart UML Examples: https://www.lucidchart.com/pages/uml-use-case-diagram
[3]	SRS Document : Moodle https://moodle.org
[4]	<i>draw.io</i> <i>canva</i>
[5]	Tutorials on Class Diagrams: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/
[6]	Database Schema Design: https://www.geeksforgeeks.org/enhanced-er-model/
[7]	Official Figma Getting Started Guide: https://www.figma.com/community/file/1019367855044645938/figma-resources
[8]	Database Normalization Basics: https://guides.visual-paradigm.com/a-comprehensive-guide-to-database-normalization-with-examples/

[9]	Comprehensive guide for building React-based applications with Next.js.: https://nextjs.org/docs
[8]	The official PostgreSQL documentation for all database-related needs.: https://www.postgresql.org/docs/
[9]	Step-by-step guide for integrating PostgreSQL with Spring Boot.: https://talesofdancingcurls.medium.com/spring-boot-with-postgresql-a-step-by-step-guide-c451848f0184

Software Requirements Specification

for

Learning Management System for the Mahapola Ports and Maritime Academy

Version 1.0 approved

Prepared by Team Laser

University Of Moratuwa

23/12/2024

Revision History	ii
1. Introduction.....	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Product Scope.....	1
1.5 References	2
2. Overall Description	2
2.1 Product Perspective	2
2.2 Product Functions.....	2
2.3 User Classes and Characteristics	3
2.4 Operating Environment	3
2.5 Design and Implementation Constraints	3
2.6 User Documentation.....	4
2.7 Assumptions and Dependencies.....	4
3. External Interface Requirements	4
3.1 User Interfaces.....	4
3.2 Hardware Interfaces	4
3.3 Software Interfaces.....	4
3.4 Communications Interfaces.....	5
4. System Features	5
4.1 User Management	5
4.2 Course Management.....	5
4.3 Material Delivery	6
4.4 Assessment and Grading	6
4.5 Communication and Collaboration	7
4.6 Reporting and Analytics.....	7
5. Other Nonfunctional Requirements	8
5.1 Performance Requirements	8
5.2 Safety Requirements	8
5.3 Security Requirements	8
5.4 Software Quality Attributes	8
5.5 Business Rules.....	8
6. Other Requirements	9
6.1 Database Requirements	9
6.2 Future Considerations for Mobile Application	9
Appendix A: Glossary.....	10
Appendix B: Analysis Models.....	11
Appendix C: To Be Determined List.....	12

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This document details the proposed Learning Management System for the Mahapola Ports and Maritime Academy. This will be the initial release of the System. This document outlines the system's functionalities, performance expectations, design constraints, and other relevant factors necessary for the development and deployment of the said Learning Management System.

This document will be a guide for the developers, the users of the Port Authority and the supervisors from the Faculty of Information Technology, University of Moratuwa.

1.2 Document Conventions

The abbreviations needed for the interpretation of the document are available in Appendix A.

The UML and ER Diagrams for the project have been created using draw.io (diagrams.net)

The UI/UX design for the system has been developed using Figma.

The priority for each use case will be considered the same unless specifically mentioned.

1.3 Intended Audience and Reading Suggestions

The document is intended for the following audiences.

- Developers
- Stakeholders from Mahapola Ports and Maritime Academy
- Faculty members from the Faculty of Information Technology, University of Moratuwa

It is recommended to read the document sequentially, starting with the introduction for context, then Appendix A for abbreviations.

Sections 4 and 5 contain the core functional and non-functional requirements, hence it is suggested to have special attention to those sections.

1.4 Product Scope

This LMS will provide a centralized platform for the MPMA to manage and deliver educational content, hold quizzes, track student progress, generate certificates, manage OJT, and facilitate communication between instructors and students. The initial scope focuses on a web-based application.

Key features included in this scope are:

- User management (administrators, lecturers, students).
- Overview Dashboards
- Course creation and management.
- Content uploading and delivery (documents, quizzes, etc.).

- Assessment creation and grading.
- Communication tools (announcements, chat).
- Reporting and analytics on student activity and course performance.

Future expansion, such as a mobile application, is outside the scope of this initial version but will be considered in the architectural design of the backend.

1.5 References

- The IEEE Standard SRS document template.
- Interactive Figma project for UI/UX designs:
 - <https://www.figma.com/proto/3koUUP4cs8PWZx3Njcrf2J/Low-Fidelity-Diagram?node-id=0-1&t=WyHQ0IoD9BQ7kD81-1>
 - <https://www.figma.com/proto/3koUUP4cs8PWZx3Njcrf2J/Low-Fidelity-Diagram?node-id=180-667&t=WyHQ0IoD9BQ7kD81-1>
 - <https://www.figma.com/proto/3koUUP4cs8PWZx3Njcrf2J/Low-Fidelity-Diagram?node-id=5-2&t=WyHQ0IoD9BQ7kD81-1>
- Draw.io project for diagrams:
 - Available on the Private Git Repository
 - <https://github.com/RandithaK/mpma>
- Material Toolkit for UI design

2. Overall Description

2.1 Product Perspective

This LMS is a new web application designed to replace existing manual processes used for learning management at the MPMA. It will be a central platform accessible through standard web browsers. While currently only accessible using a web browser, the backend architecture will be designed with potential future integration with other academy systems in mind such as the MPMA's existing attendance system. The backend will be built using Java Spring Boot, providing APIs that can potentially be connected to other applications, including a future mobile app.

2.2 Product Functions

The LMS will provide the following core functions:

- Initiation of the System with Super Administrator
- User Management: Creation, modification, and deletion of user accounts (administrators, lecturers, students), role assignment, and permission management.
- Course Management: Creation, modification, enrollment management, and lecturer and student assignment.
- Content Delivery: Uploading, organizing, and delivering several types of learning materials (documents, videos, presentations, links). Content access control based on user roles and course enrollment.
- Assessment and Grading: Creation of quizzes, assignments, and exams, automated grading for certain assessment types, manual grading capabilities, and gradebook management.

- Communication and Collaboration: Announcements, and potentially direct messaging between students and lecturers.
- Reporting and Analytics: Generation of reports on student progress, course completion rates, assessment scores, and user activity.
- Administrative dashboards for system overview.



2.3 User Classes and Characteristics

- **Administrator:**
 - Technical proficiency: Medium to High.
 - Responsible for system configuration, user management, course creation (potentially), and generating reports. Needs a comprehensive view of the system.
- **Lecturer:**
 - Technical proficiency: Medium.
 - Responsible for creating and managing course content, conducting assessments, grading student work, and communicating with students. Needs tools for content upload and interaction.
- **Student:**
 - Technical proficiency: Low to Medium.
 - Responsible for accessing course materials, submitting assignments, taking assessments, and communicating with lecturers. Needs a user-friendly interface for learning.

2.4 Operating Environment

- **Backend Server:** Linux Server
- **Backend Technology:** Java Spring Boot
- **Frontend Framework:** NextJS
- **Database:** PostgreSQL
- **Containerization:** Docker
- **Web Browsers:** Compatible with modern web browsers, both desktop and mobile (Chromium based, Safari and Firefox).
- **Hardware Platform:** VMs provided by the MPMA, all costs borne by MPMA

2.5 Design and Implementation Constraints

- **Technology Stack:** The system must be built using the specified technologies: Java Spring Boot (backend), NextJS (frontend), PostgreSQL (database), and Docker for deployment.
- **Database Choice:** PostgreSQL is the designated database system.
- **Operating System:** The backend server will run on a Linux environment.
- **UI/UX Design:** The user interface will have the functionalities drafted in the Figma design.
- **Diagramming:** System architecture and other relevant diagrams will be created using Draw.io considering long term support and the project being available for free.
- **Scalability:**
 - **Backend:** The backend architecture should be designed to accommodate potential future expansion, including the possibility of a mobile application. Implementing using a containerizing mechanism such as docker will make the system easily scalable.
 - **Frontend:** NextJS with React will be used as a future mobile app that can be implemented using React Native easily.

- **Security:** Appropriate security measures must be implemented to protect user data and prevent unauthorized access with protections against common attack vectors.
- **Languages:** English language will be supported

2.6 User Documentation

In-app tooltips will be provided as the system will inherently be self-explanatory.

2.7 Assumptions and Dependencies

- It is assumed that the necessary server infrastructure (Linux server) will be available for deployment as agreed in the initial meetings.
- The development environment (including necessary software and libraries) will be available for the development team.
- It is assumed that the MPMA will need a mobile application at some point and the system backend should be able to handle such an application with minimal change.

3. External Interface Requirements

3.1 User Interfaces

The user interface will be a web-based application accessed through standard web browsers mentioned earlier in section 2.4. The UI/UX design will be based on the mockups and prototypes created in Figma.

Key considerations include:

- **Intuitive Navigation:** Easy-to-understand menus and navigation patterns for all user roles.
- **Responsive Design:** The interface should be responsive and adaptable to different screen sizes and devices (desktops, laptops, tablets, and smart phones).
- **Accessibility:** Adherence to accessibility guidelines (WCAG2) to ensure usability for most users.
- **Consistent Design Language:** Maintaining a consistent look and feel throughout the application.

3.2 Hardware Interfaces

- **Client-side:** Standard desktop, laptop computers, tablets, smartphones, with a compatible web browser and stable internet connection.
- **Server-side:** The application will reside in Docker Containers on a compatible Linux server(s). Specific hardware specifications will depend on anticipated user load and data storage requirements.

3.3 Software Interfaces

- **Backend API:** Java Spring Boot will provide RESTful APIs for communication with the NextJS frontend. API documentation will be generated using tools like Swagger.
- **Database Interface:** PostgreSQL will be accessed through Java within the Spring Boot application.

- **External Integrations** (Future): Potential future integrations with other academy systems will be considered during API design.

3.4 Communications Interfaces

- **Protocol:** HTTP/HTTPS for web communication between the client and server.
- **Data Format:** JSON for data exchange between the frontend and backend APIs.
- **Security:** Secure communication over HTTPS will be enforced.
- **Notification Integration:** The system may need to push notifications (e.g., for assignment submissions, announcements).

4. System Features

4.1 User Management

4.1.1 Description and Priority:

- Allows administrators to manage user accounts, roles, and permissions. [High]

4.1.2 Stimulus/Response Sequences:

- Admin navigates to the user management page.
- Admin clicks "Add User," fills in user details including the account type, and clicks "Save." -> System creates a new user account.
- Admin selects a user and clicks "Edit," modifies user details, and clicks "Save." -> System updates user information.
- Admin selects a user and clicks "Delete" -> System prompts for confirmation and deletes the user account.

4.1.3 Functional Requirements:

- REQ-UM-1: The system shall allow administrators to create new user accounts with specified roles (Administrator, Lecturer, Student).
- REQ-UM-2: The system shall allow administrators to edit existing user account information (name, email, reset password, etc.).
- REQ-UM-3: The system shall allow administrators to delete user accounts.
- REQ-UM-4: The system shall enforce password complexity requirements.

4.2 Course Management

4.2.1 Description and Priority:

- Enables administrators to create and manage courses. [High]

4.2.2 Stimulus/Response Sequences:

- Administrator navigates to the course management page.
- Instructor clicks "Create Course," enters course details (name, description), and clicks "Save." -> System creates a new course.
- Admin or Lecturer (with Admin approval) can enroll/unenroll students in a course.

4.2.3 Functional Requirements:

- REQ-CM-1: The system shall allow administrators to create new courses.
- REQ-CM-2: The system shall allow administrators to edit course details (name, description, syllabus, start/date, duration).
- REQ-CM-3: The system shall allow administrators to allocate/deallocate lecturers to courses.
- REQ-CM-4: The system shall allow administrators and authorized lecturers to enroll and unenroll students in courses.

4.3 Material Delivery

4.3.1 Description and Priority:

- Provides a platform for lecturers to upload and manage learning materials for students. [High]

4.3.2 Stimulus/Response Sequences:

- Instructor navigates to a specific course.
- Instructor clicks "Add Content," selects a file (document, video, etc.), and uploads it. -> System stores the content.
- Student navigates to a course and views available content.

4.3.3 Functional Requirements:

- REQ-CD-1: The system shall allow lecturers to upload various file types (such as PDF).
- REQ-CD-2: The system shall allow lecturers to organize content into modules or sections within a course.
- REQ-CD-3: The system shall allow lecturers to set visibility for content.
- REQ-CD-4: The system shall provide students with a user-friendly interface to access and download course content.
- REQ-CD-5: The system shall provide mechanisms for embedding external content (such as links).

4.4 Assessment and Grading

4.4.1 Description and Priority:

- Enables lecturers to create and manage assessments (quizzes, and upload activities), and grade student submissions. [High]

4.4.2 Stimulus/Response Sequences:

- Instructor navigates to a course and clicks "Create Assessment."

- Instructor selects assessment type (quiz, and upload activities), defines questions, points, and due dates. -> System creates the assessment.
- Student accesses an assessment, answers questions, and submits.
- The lecturer reviews submissions and assigns grades.

4.4.3 Functional Requirements:

- REQ-AG-1: The system shall allow lecturers to create quizzes with various question types (multiple choice, true/false, short answer, etc.).
- REQ-AG-2: The system shall allow lecturers to create assignments with submission guidelines and deadlines.
- REQ-AG-3: The system shall allow for automated grading of objective quiz questions.
- REQ-AG-4: The system shall provide lecturer with tools to manually grade assignments and provide feedback. (online or download options)
- REQ-AG-5: The system shall maintain a gradebook for each course, displaying student scores and overall grades.

4.5 Communication and Collaboration

4.5.1 Description and Priority:

- Facilitates communication between lecturer and students within the LMS. [Medium]

4.5.2 Stimulus/Response Sequences:

- A lecturer creates an announcement for a course. -> Students enrolled in the course receive the announcement.
- A student posts a question to the lecturer and the lecturer can respond.

4.5.3 Functional Requirements:

- REQ-CC-1: The system shall allow lecturers to create and post announcements for specific courses.
- REQ-CC-2: The system may (future consideration) include a direct messaging feature between users.

4.6 Reporting and Analytics

4.6.1 Description and Priority:

- Provides administrators and instructors with insights into user activity and course performance. [Medium]

4.6.2 Stimulus/Response Sequences:

- Administrator navigates to the reporting dashboard.
- Administrator selects a report type (e.g., course completion rates) and specifies parameters. -> System generates the report.

4.6.3 Functional Requirements:

- REQ-RA-1: The system shall provide reports on student enrollment and course completion rates.
- REQ-RA-2: The system shall provide reports on student performance in assessments.
- REQ-RA-3: The system shall provide administrators with an overview dashboard of system usage and key metrics.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system should respond to user actions within a reasonable time, without timeout.
- The system should be able to handle concurrent users without significant performance degradation.
- The system should be able to store and manage learning materials and user data.

Note: The number of concurrent users for which the system should be load tested has not been communicated yet.

5.2 Safety Requirements

- Data integrity must be maintained to prevent data loss or corruption.
- User authentication and authorization mechanisms must be robust to prevent unauthorized access.

5.3 Security Requirements

- User authentication will be required for access to the LMS, JWT will be implemented.
- User access levels and permissions will be strictly enforced to control access to specific features and data.
- Sensitive data (e.g., passwords) will be encrypted both in transit and at rest.
- The system will be protected against common web vulnerabilities.

5.4 Software Quality Attributes

- **Usability:** The system should be user-friendly and easy to navigate for all user roles.
- **Maintainability:** The codebase should be well-structured to facilitate future maintenance and updates.
- **Reliability:** The system should be stable and operate without frequent errors or failures.
- **Scalability:** The backend architecture should be scalable to accommodate future growth in users and data.
- **Testability:** The system should be designed to be easily tested at various levels (unit, integration, system).

5.5 Business Rules

- Only registered users can access the LMS, and access level is based on their role.

- Students cannot enroll in courses themselves and should be enrolled by the administrator.
- Lecturers can only manage courses they are assigned to.
- Administrators have full access to assign lectures and students and view analytics of the system, including the overall academic calendar.

6. Other Requirements

6.1 Database Requirements

- PostgreSQL database will be used to store all application data. Database schema will be designed to ensure data integrity and efficient querying. Regular database backups will be performed.
- The database will be shared with the proposed ERP system of the MPMA.

6.2 Future Considerations for Mobile Application

The backend API (developed with Java Spring Boot) should be designed with RESTful principles to facilitate communication with potential future mobile applications (e.g., using React Native).

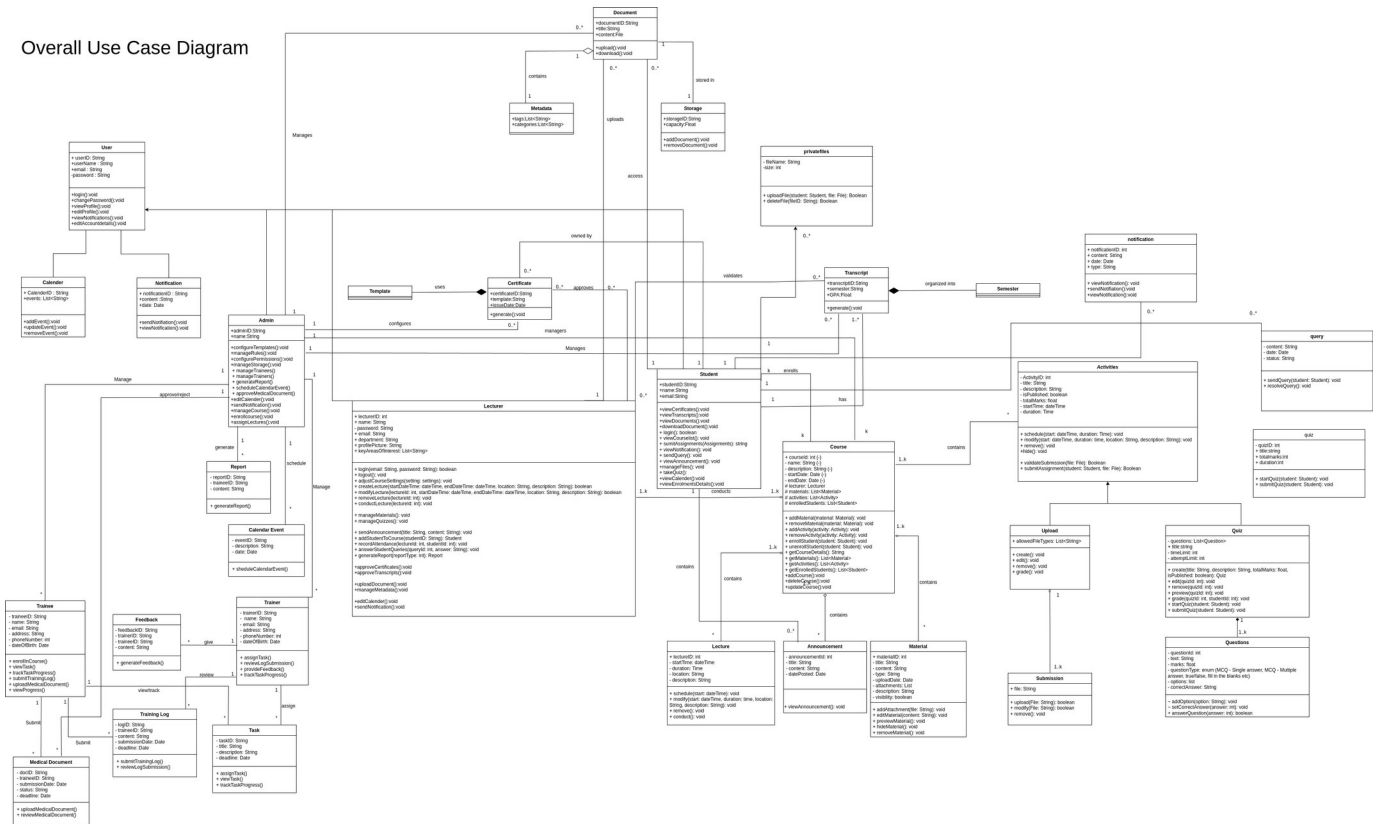
Appendix A: Glossary

Abbreviations needed for interpretation of the Project

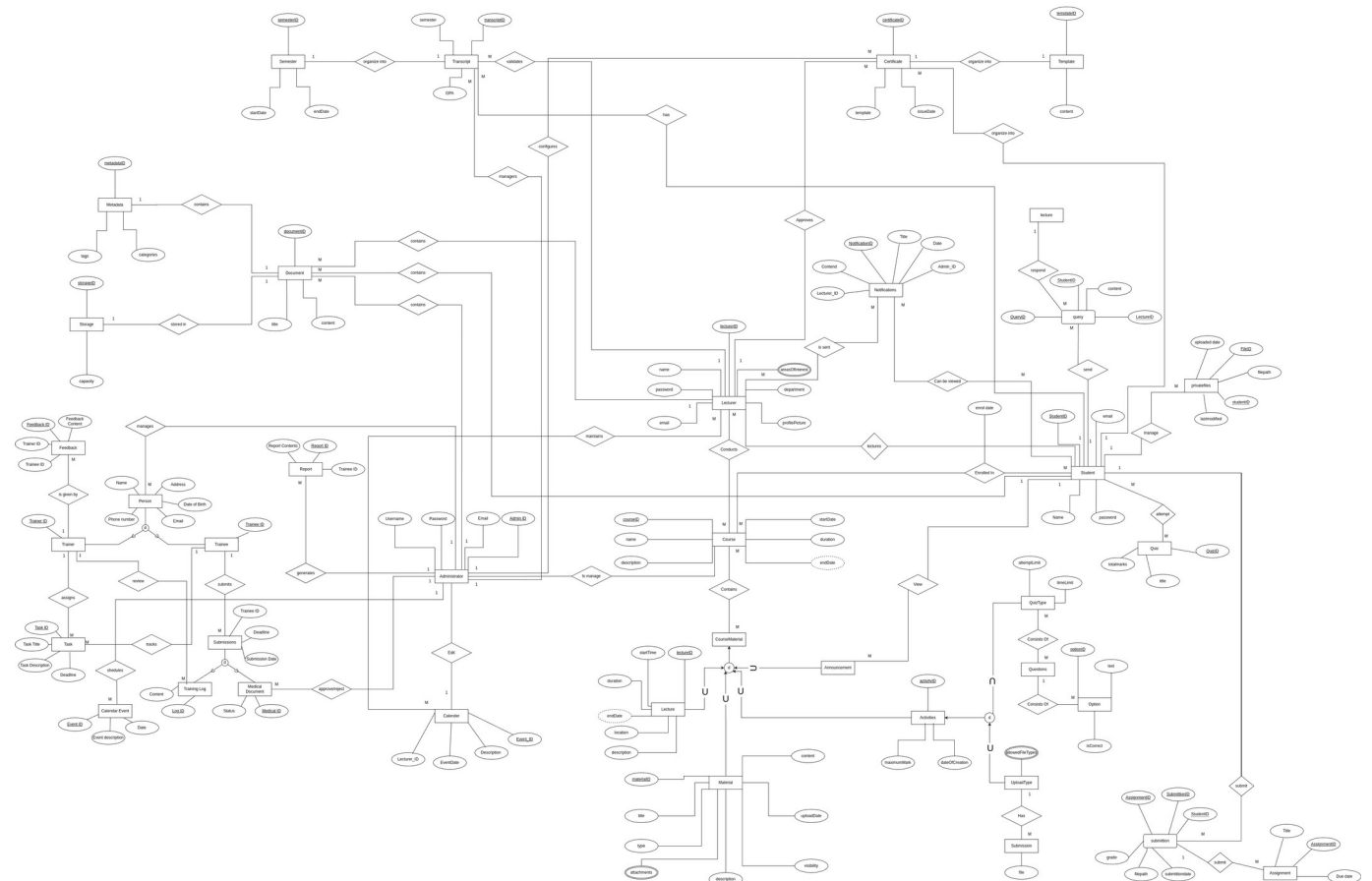
- MPMA – Mahapola Port & Maritime Academy
- SLPA – Sri Lanka Ports Authority
- OJT – On-the-Job Training
- VM – Virtual Machine
- UML – Unified Modeling Language
- EER – Enhanced Entity Relationship
- ERP – Enterprise Resource Planning
- LMS – Learning Management System
- API – Application Programming Interface
- UI – User Interface
- UX – User Experience
- JSON – JavaScript Object Notation
- HTTP – Hypertext Transfer Protocol
- HTTPS – Hypertext Transfer Protocol Secure

Appendix B: Analysis Models

Overall Use Case Diagram



Overall Enhanced Entity Relationship Diagram



Appendix C: To Be Determined List

- The system load was not mentioned in the initial discussion.
- Color preferences of the MPMA were not mentioned to us.