Cairo University
Faculty of Computers and Artificial Intelligence
Department of Computer Sciences

Supervised By

Dr. Desoky Abd El Qawy
TA. Aya Shehata

Implemented By

20206034	Abdalrhman Ahmed Mohamed Mansour
20206003	Ahmed Hamdy Hamed Elgohary
20206061	Mohamed Ashraf Abd El Azim
20206117	Youssuf Mohamed Talaat El Tahan
20206065	Mohamed Ashraf Mohamed Ali

Graduation Project
Academic Year 2023-2024
Final Documentation

Table of Contents:

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Motivation (Abstract)

In higher education, accessing and utilizing organized educational resources remains a persistent challenge for students. UniHub aims to revolutionize this experience by not only providing centralized access to course materials but also by integrating innovative tools such as a comprehensive question bank and an interactive chatbot. These features are designed to enhance learning efficiency, foster collaborative engagement, and support academic success within university communities.

The goal of our project, UniHub, is to create a centralized digital platform that streamlines access to educational resources for university students. Recognizing the common challenges students face in navigating fragmented and disorganized information, UniHub aims to enhance the learning experience through a well-structured repository of textbooks, lecture notes, videos, and supplementary materials curated by both students and instructors.

UniHub is designed to promote collaborative learning and community engagement within academic environments. Key features include efficient navigation through structured organization and an integrated question bank compiled from student inquiries.

The Technologies that will be used in our project are:

- Front-end: React.js
- Back-end: Node.js
- Database Management: MySQL
- Authentication and Authorization: JSON Web Tokens (JWT)
- File Upload and Storage: Firebase

1.2 Problem Definition

Traditional educational platforms often lack integrated tools that cater to the diverse needs of students and instructors. Fragmented resources and inefficient communication channels hinder effective learning and teaching experiences. Additionally, the absence of a structured system for managing and accessing course-related queries further complicates the educational journey.

UniHub addresses these issues by implementing a centralized platform equipped with a dynamic question bank and an intelligent chatbot. These innovations aim to streamline information retrieval, improve student-instructor interaction, and foster a cohesive academic environment.

Moreover, the lack of a centralized resource repository limits opportunities for collaborative learning and peer engagement. Students often struggle to find relevant textbooks, lecture notes, videos, and supplementary materials, resulting in wasted time and reduced academic performance. Instructors also face challenges in distributing materials and fostering an interactive learning environment.

Additionally, the absence of a structured system for managing educational resources exacerbates these issues, as students and Instructors alike lack a cohesive platform for sharing and accessing information. This situation contributes to a disjointed learning experience and hampers the potential for academic success.

Therefore, there is a critical need for a centralized digital platform that aggregates educational materials, promotes collaborative learning, and facilitates efficient resource access. Addressing these challenges through such a platform can enhance the overall learning experience, support academic success, and foster a more engaged academic community.

1.3 Project Objectives

The objective of our project, UniHub, is to develop a centralized digital platform that addresses the challenges faced by students and educators in accessing and utilizing educational resources effectively. Our proposed solution aims to achieve the following goals:

1. **Centralized Resource Access:** Create a unified platform where students can easily find and access educational resources. Integrate a question bank sourced from previous student inquiries to provide quick answers and comprehensive insights into course topics.
2. **Enhanced Collaboration:** Facilitate knowledge-sharing and collaborative learning by enabling students and instructors to interact seamlessly through integrated discussion forums, announcements, and a chatbot capable of answering common queries.
3. **Structured Organization:** Implement a systematic approach to organizing educational materials. Enhance navigation with intuitive features that guide users through courses and facilitate efficient learning.
4. **User Engagement:** Promote active participation through features like interactive discussions, personalized task management, and feedback mechanisms integrated with the question bank to improve content quality.
5. **Technological Integration:** Utilize innovative technologies like Node.js, React.js, Firebase, and AI-driven chatbots to ensure a robust, scalable, and user-friendly platform that meets the diverse needs of students, instructors, and administrators.
6. **Evaluation and Improvement:** Continuously assess the platform's impact on learning outcomes and user satisfaction. Use feedback from users and data analytics to iteratively enhance functionality, content relevance, and overall user experience.

By integrating these advanced features into UniHub, the platform aims to redefine the educational experience, empower users with comprehensive resources, and foster a collaborative learning environment that supports academic success and community engagement.

1.4 Gantt Chart

- Project Gantt chart.

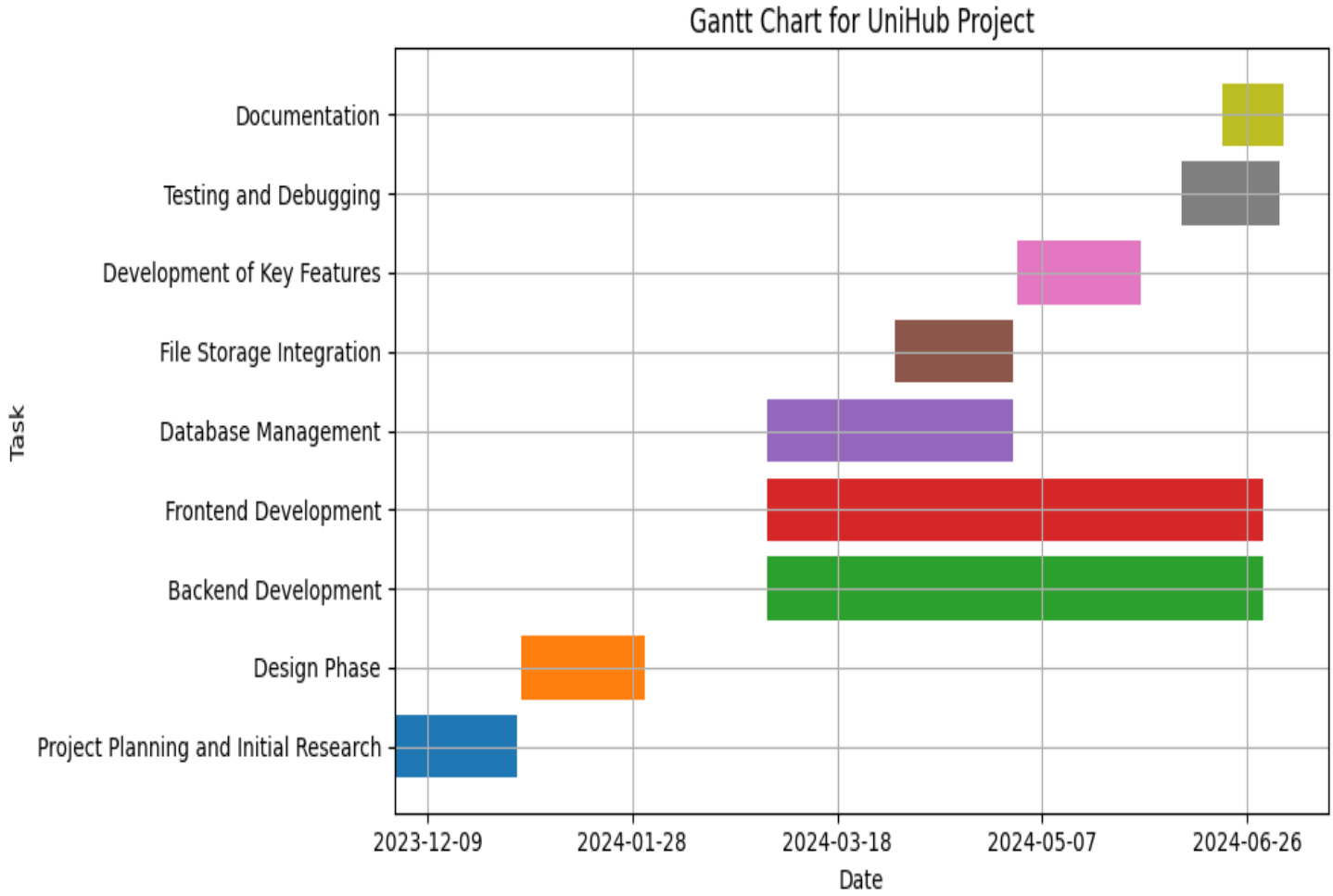

Figure 1: Gantt Chart

1.5 Project Development Methodology

Waterfall, The Waterfall methodology is a linear and sequential approach to software development. It is characterized by a series of distinct phases that are followed in astrict order, with each phase building upon the results of the previous one. The phases typically include requirements gathering and analysis, system design, implementation, testing, deployment, and maintenance.

Requirements

In the requirements phase of the software development process, the main goal is togather a thorough understanding of the user's needs and define the specifications ofthe software product. This phase is crucial as it lays the foundation for the entire development process.

Design

In the design phase of software development, the goal is to translate the requirements gathered in the previous phase into a detailed design that will guidethe implementation of the software. The design phase focuses on defining the structure, architecture, and components of the final solution.

Implementation

In the implementation phase of software development, the focus is on translating the design specifications into actual working code. This is where the developmentteam takes the detailed design and builds the software system.

Testing

Is a critical phase in the software development process, aimed at identifying andfixing any defects or issues in the system before it is deployed to the users. A testplan will be followed. Different test cases will be executed according to three types: unit testing, integration testing, and usability testing.

Deployment and Maintenance

are two essential stages in the software development process. Deployment involves releasing the developed software system and making it available for use by the intended users. It includes activities such as configuring the system, setting up servers, and ensuring a smooth transition to the production environment. On the other hand, Maintenance focuses on ensuring the ongoing operation and performance of the software system. It involves monitoring, bug fixing, applying updates, and providing technical support to users. Together, these stages ensure that the software system is successfully deployed and maintained to deliver reliableand efficient functionality to its users over time.

1.6 The Used Tools

Frontend

- **React.js**
  Our main technology for the front-end of UniHub is React.js. We chose React.js for several reasons:

  - React.js facilitates a fast and efficient development cycle, enabling rapid iteration and deployment of features.
  - It provides a component-based architecture that enhances code reusability and maintainability.
  - React.js supports server-side rendering for improved performance and SEO optimization.

Backend

- **Node.js**
  Our main technology for the back end of our application was Node.js,we choose Node.js because it offers the following benefits:

  - It allows for asynchronous and event-driven programming, making it well-suited for handling multiple simultaneous requests.
  - Node.js has a vibrant ecosystem of libraries and frameworks, which accelerates development and deployment.
  - It offers scalability and performance, crucial for handling the expected load of concurrent users on the platform.

Database Management

- **MySQL**
  MySQL is our chosen database management system. It was selected due to:

  - Its reliability, scalability, and robust performance for managing structured data.
  - Wide support within the development community and compatibility with other technologies.
  - Advanced features such as transactions, indexing, and querying capabilities that meet the needs of a centralized educational platform.

- Authentication and Authorization

  JSON Web Tokens (JWT)

  JWT is employed for authentication and authorization. We opted for JWT because:

  - It provides a secure method for transmitting information between parties as JSON objects.
  - JWTs are compact, making them suitable for use within HTTP headers and URL parameters.
  - They support stateless authentication, enhancing scalability and performance in distributed systems.

- File Upload and Storage

  Firebase

  Firebase serves as our platform for file upload and storage. It was chosen because:

  - It offers a scalable and secure cloud storage solution with built-in file encryption and access controls.
  - Firebase integrates seamlessly with other Google Cloud services and provides real-time database capabilities.
  - Its SDKs and APIs simplify client-side integration and enable efficient management of uploaded files.

- Technological

  AI-driven Chatbot

  To enhance user interaction and support, we are integrating an AI-driven chatbot. This component leverages:

  - Natural Language Processing (NLP) techniques to understand and respond to user queries.
  - Machine learning models for continuous improvement in response accuracy and user engagement.
  - Integration with Node.js for seamless deployment and scalability within our platform architecture.

1.7 Report Organization

This report is structured to provide a comprehensive exploration of the UniHub project, covering the following key aspects:

Chapter 2
Some Similar Solutions to the Same Problem and the Difference from our Project

Chapter 3
Details of the Requirements in the System and Use Cases

Chapter 4
More About the Design of **UniHub** in Details:

- System Component Design
- System Classes UML
- Sequence Diagrams
- System ERD
- System GUI

Chapter 5
Implementation and Testing of the Application

Chapter 2: Related Work

Existing Similar targeted apps

This chapter explores existing university web applications and chatbot solutions using APIs to understand their functionalities and identify how UniHub offers a distinct value proposition.

1. **Blackboard Learn**
    - Overview: Blackboard Learn is a popular learning management system (LMS) used in many educational institutions worldwide. It provides tools for course management, communication, and collaboration.
    - Strengths: Robust course management features, including content organization, assessments, and grading tools. It supports integration with various third-party applications.
    - Weaknesses: User interface can be complex and less intuitive for some users. Customization options may require technical expertise.
    - Comparison: UniHub could differentiate itself by offering a more streamlined and user-friendly interface, enhanced collaboration tools, and a dynamic question bank.
2. **Stack overflow**
    - Overview: Stack Overflow is a community-driven question and answer site focused on programming and software development topics. It serves as a knowledge-sharing platform where developers can ask questions, provide answers, and collaborate on technical issues.
    - Strengths: Extensive database of questions and answers covering a wide range of programming languages, frameworks, and technologies. Community-driven moderation ensures high-quality content.
    - Weaknesses: While excellent for technical Q&A, Stack Overflow lacks structured educational resources such as curated courses, textbooks, and organized learning paths. It may not provide a cohesive learning experience for beginners or structured course materials.
    - Comparison: UniHub could differentiate itself by offering a curated repository of educational resources tailored specifically for university students. This includes structured access to textbooks, lecture notes, videos, and supplementary materials, all organized in a centralized platform. Integration of an AI-driven chatbot for academic support and collaborative tools like discussion forums could further enhance UniHub's appeal in fostering interactive and community-driven learning experiences.

3. **Google Classroom**
   - Overview: Google Classroom is a free web service developed by Google for schools that aims to simplify creating, distributing, and grading assignments in a paperless way.
   - Strengths: Seamless integration with Google Workspace (formerly G Suite), including Google Drive, Docs, and Meet. Easy-to-use interface and strong collaboration features.
   - Weaknesses: Limited customization options compared to other LMS platforms. Requires users to have a Google account.
   - Comparison: UniHub could differentiate itself by offering a more comprehensive approach to resource management, personalized learning experiences through AI-driven recommendations, and robust backend support with Node.js and MySQL.

Chapter 3: System Analysis

3.1 Project Specification

System Architecture

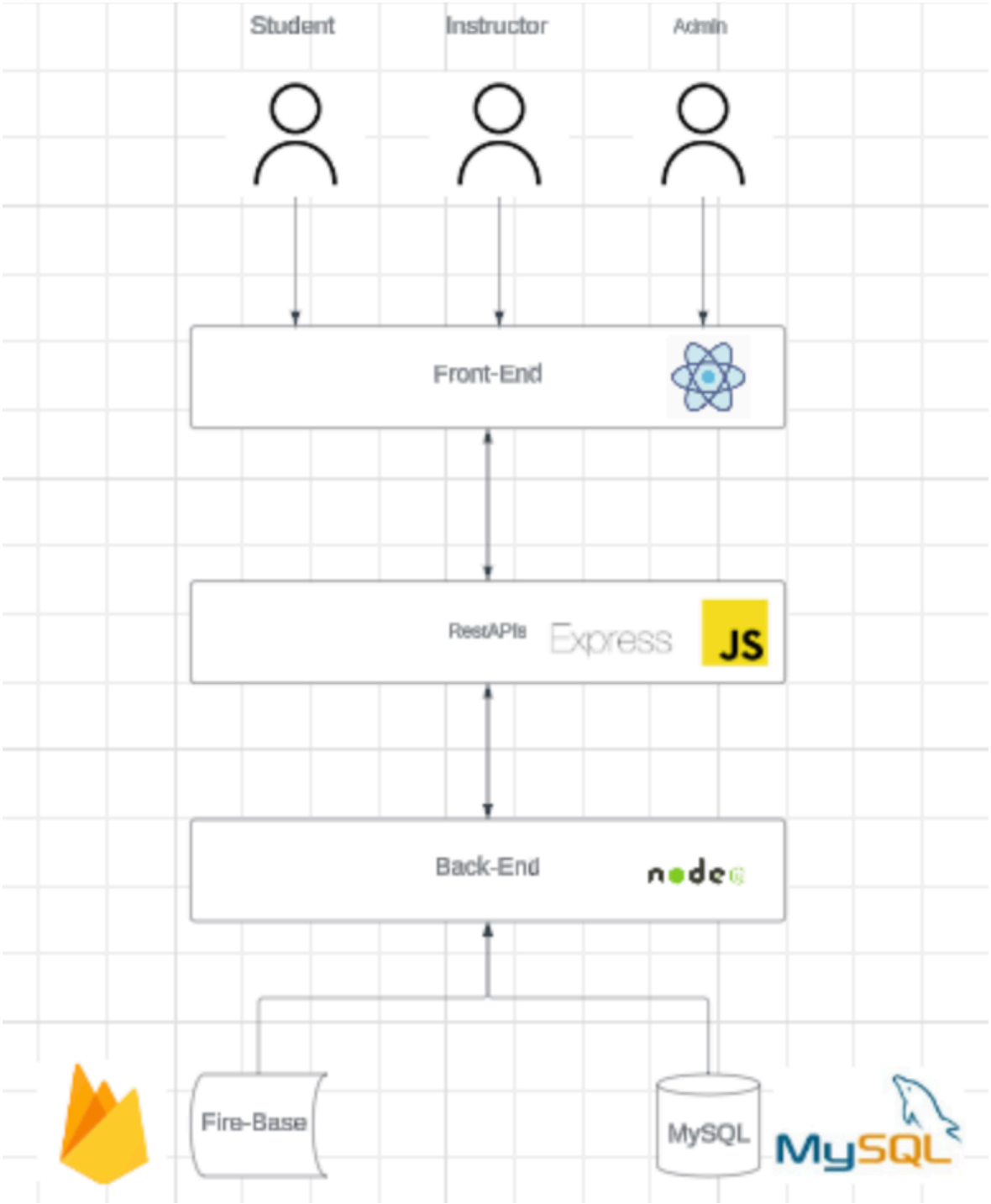

Figure 2: System Architecture

Stakeholders

Students:

End-users who access and contribute to educational materials, engage in collaborative learning, and utilize platform features.

Instructors:

Contributors who curate and share educational resources, collaborate with colleagues, and contribute to the knowledge base.

Administrators:

Oversee platform operations, manage user accounts, ensure compliance with policies and regulations, and ensure the overall functionality and security of the system.

Developers:

Technical experts responsible for designing, developing, and maintaining the platform's software infrastructure, implementing new features, fixing bugs, and ensuring the platform's scalability, performance, and reliability. Developers work closely with administrators, SMEs, students, and professors/instructors to address user needs and enhance the platform's functionality.

3.2 Functional requirements

Student:

- Users should be able to create accounts or log in using their credentials (e.g., email address and password).
- The system should enforce secure authentication practices, such as password hashing and encryption.
- Students should be able to search for courses based on subject, course code, or instructor name.
- Upon finding a course of interest, students should be able to enroll in the course using a provided passkey or course ID.
- Students can check the course's materials and posts.
- Upon enrollment, students should have access to course materials, posts, and discussions.
- They should be able to engage in discussions, ask questions, and provide answers within the course community.
- Students should have the ability to view and download course materials uploaded by instructors.
- Students should be able to provide rating on course materials.
- Students should have a task panel where they can add tasks related to course assignments, readings, or deadlines.
- They should be able to mark tasks as completed and track their progress within the platform.
- Students should have the option to move completed courses or courses they no longer wish to participate in to an archive section for organizational purposes.
- Students can communicate with a chatbot.

Instructor:

- Instructors should be able to post announcements visible to all students enrolled in their courses.
- They should have the ability to communicate important updates, deadlines, and reminders effectively.
- Instructors should be able to upload course materials, including lecture slides, documents, and multimedia files.

- They should have the capability to organize and structure course content for easy navigation by students.
- Instructors should be able to actively engage in discussions initiated by students within the course community.
- They should have the ability to provide guidance, clarify concepts, and facilitate meaningful interactions.
- They should be able to delete or edit content that violates community guidelines or academic integrity policies in their course.

Admin:

- Administrators should have the authority to add new courses to the system.
- They should be able to assign instructors to courses and update course information as needed.
- Administrators should have the ability to moderate user-generated content, including posts, comments, and uploaded materials.
- They should be able to delete or edit content that violates community guidelines or academic integrity policies.
- Administrators should have the ability to post important announcements that are tagged and highlighted for visibility across the platform.

System:

- The system should display recent posts, announcements, and course updates on the homepage for easy access by users.
- Users should have a personalized dashboard displaying relevant information based on their enrolled courses and preferences.
- The system should send notifications of important announcements.
- The system should integrate a question bank compiled from student inquiries.

3.3 Non-functional requirement

Performance

- The system should provide fast response times for user interactions, such as loading pages, accessing course materials, and posting comments. Response times should be within acceptable limits, typically under 2 seconds
- The system should optimize resource utilization, including CPU, memory, and bandwidth, to ensure smooth operation even during peak usage periods.

Security

- User data, including login credentials, personal information, and communication, should be encrypted during transmission and storage using strong encryption algorithms.
- Implement secure authentication mechanisms to verify user identities and authorize access to system features based on user roles and permissions. Use techniques such as JSON Web Tokens (JWT) for session management and access control.
- Apply data protection measures to safeguard sensitive information against unauthorized access, modification, or disclosure.

Usability

- Design the user interface (UI) with simplicity, consistency, and intuitiveness in mind to enhance user experience (UX). Ensure that navigation flows are logical, and interactive elements are accessible and user-friendly.
- Ensure that the application is responsive and compatible with a wide range of devices and screen sizes, including desktops, laptops, tablets, and smartphones. Conduct thorough testing across different browsers and platforms to ensure consistent performance and usability.

Reliability

- Implement mechanisms for error handling, fault tolerance, and graceful degradation to minimize service disruptions and downtime. Use redundant components and failover strategies to ensure continuous availability and reliability.

Maintainability

- Design the application architecture using modular and reusable components to facilitate code maintenance, updates, and enhancements. Encapsulate business logic and separate concerns to minimize dependencies and promote code scalability.
- Provide comprehensive documentation covering system architecture, codebase, APIs, configuration settings, and deployment procedures. Document coding standards, best practices, and development guidelines to facilitate collaboration among development teams and future maintenance efforts.
- Use version control systems (e.g., Git) to manage source code repositories, track changes, and collaborate effectively with development teams. Adopt branching and merging strategies to streamline code management and code review processes.

3.4 Use Case Diagram

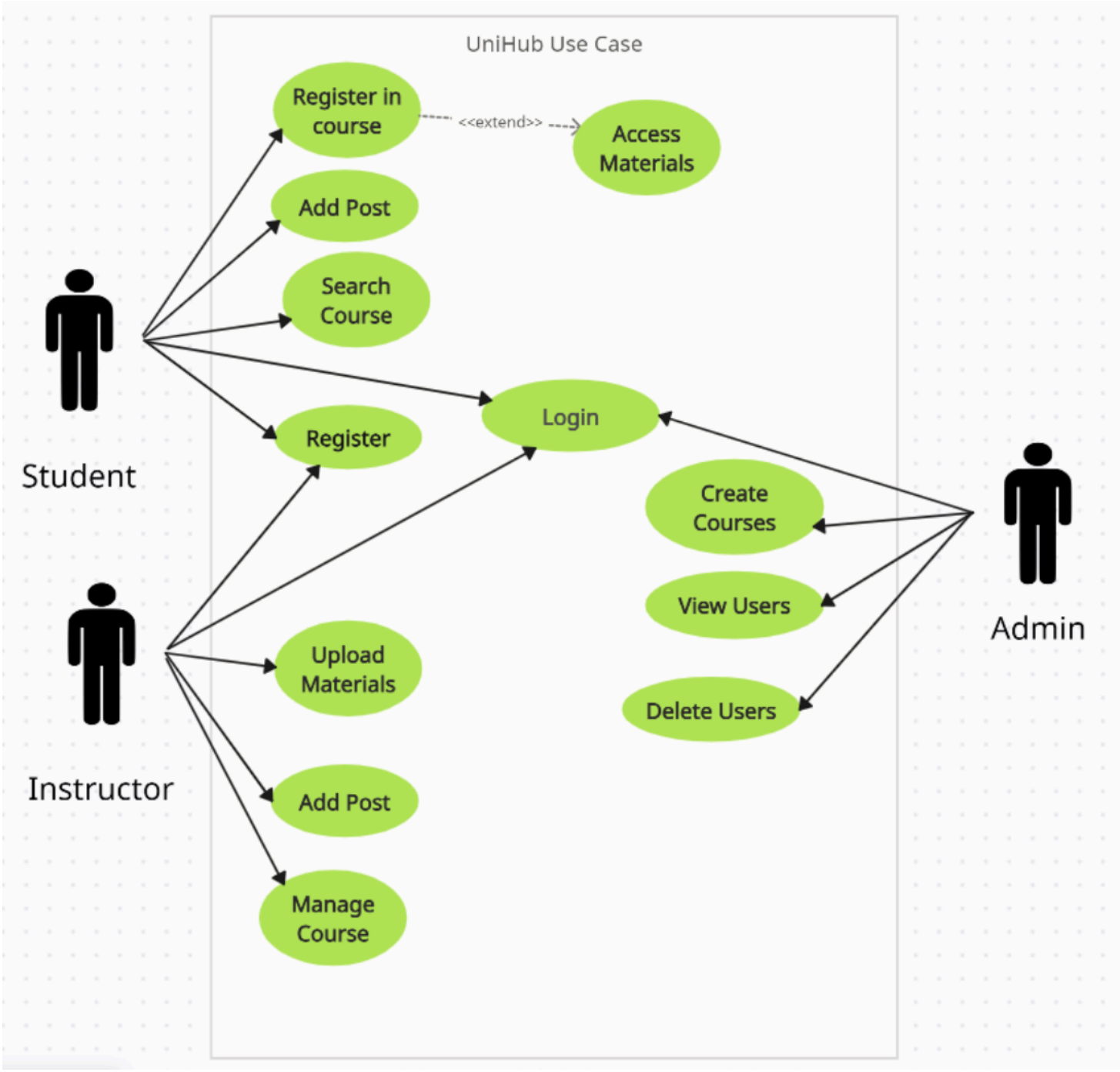

Figure 3: Use Case Diagram

Use Case Tables (User Stories)

Use Case ID	Case #1
Use Case Name:	Register
Actors:	Student, Instructor
Description:	**As** a user. **I** like to be able to sign up in the system. **So** that I create my profile.
Pre-conditions:	The user does not have an existing account
Post-conditions:	The user has a new account and can log in.
Flow of Events:	1-User enters his first name, last name, email address, password, birthdate, and choose student or admin. 2- System checks the validation of inputs. 3- User is redirected to the login page.

Table 1: Use Case 1 --> Register

Use Case ID	case #2
Use Case Name:	Login
Actors:	Student, Instructor
Description:	**As** a user. **I** like to be able to login into the system. **So** that I enter the home page.
Pre-conditions:	User is Signed Up.
Post-conditions:	User can register in course, edit profile, addpost, comment, vote, etc.
Flow of Events:	1- User Types his email 2- User Types his Password. 3- User clicks the "Login" button. 4- If the Credentials are wrong, the user is told so. 5- If the Credentials are Corrected the user will be redirected to the Home page

Table 2: Use Case 2 -->LogIn

Use Case ID	case #3
Use Case Name:	Register In Course
Actors:	Student
Description:	**As** a student. **I** like to be able to register in course **To access course material and community**
Pre-conditions:	Student is registered in the system and have the course passkey
Post-conditions:	Student can access course material and community
Flow of Events:	1-Student navigates to the specific course page. 2-Student selects enroll button. 3-Student provides the correct passkey.

Table 3: Use Case 3 -->Register in Course

Use Case ID	case #4
Use Case Name:	Access Course Materials
Actors:	Student
Description:	**As** a student. **I** like to be able to access material in course. **To view and download course materials.**
Pre-conditions:	Student is enrolled in the course
Post-conditions:	Student views/downloads materials
Flow of Events:	1-Student navigates to the specific course page. 2-Student selects the materials section. 3-System displays available materials. 4-Student views or downloads the desired materials.

Table 4: Use Case 4 -->Access Course Materials

Use Case ID	case #5
Use Case Name:	Add Post
Actors:	Student, Instructor, Admin
Description:	**As** a user. **I** like to be able to post in course community. **To engage in course-related discussions**
Pre-conditions:	Student is enrolled in the course; Instructor is assigned to the course
Post-conditions:	Student/Instructor/Admin add post in the course community
Flow of Events:	1-Student/Instructor/Admin navigates to the specific course page. 2-Student/Instructor/Admin selects the community section. 3-System displays recent posts. 4-Student/Instructor/Admin fill the post form 6-Student/Instructor/Admin click on post button

Table 5: Use Case 5 -->Add Post

Use Case ID	case #6
Use Case Name:	Create Courses
Actors:	Admin
Description:	**As** an admin. **I** like to be able to manage courses in the system. To create course offerings
Pre-conditions:	Admin has administrative privileges
Post-conditions:	Course is created, information is modified, is deleted.
Flow of Events:	1-Admin logs In. 2-Navigate on create course in sidebar 3-System prompts form 4-Admin fill the form 6-Admin clicks on create course button

Chapter 4: System Design

4.1 System Components Diagram

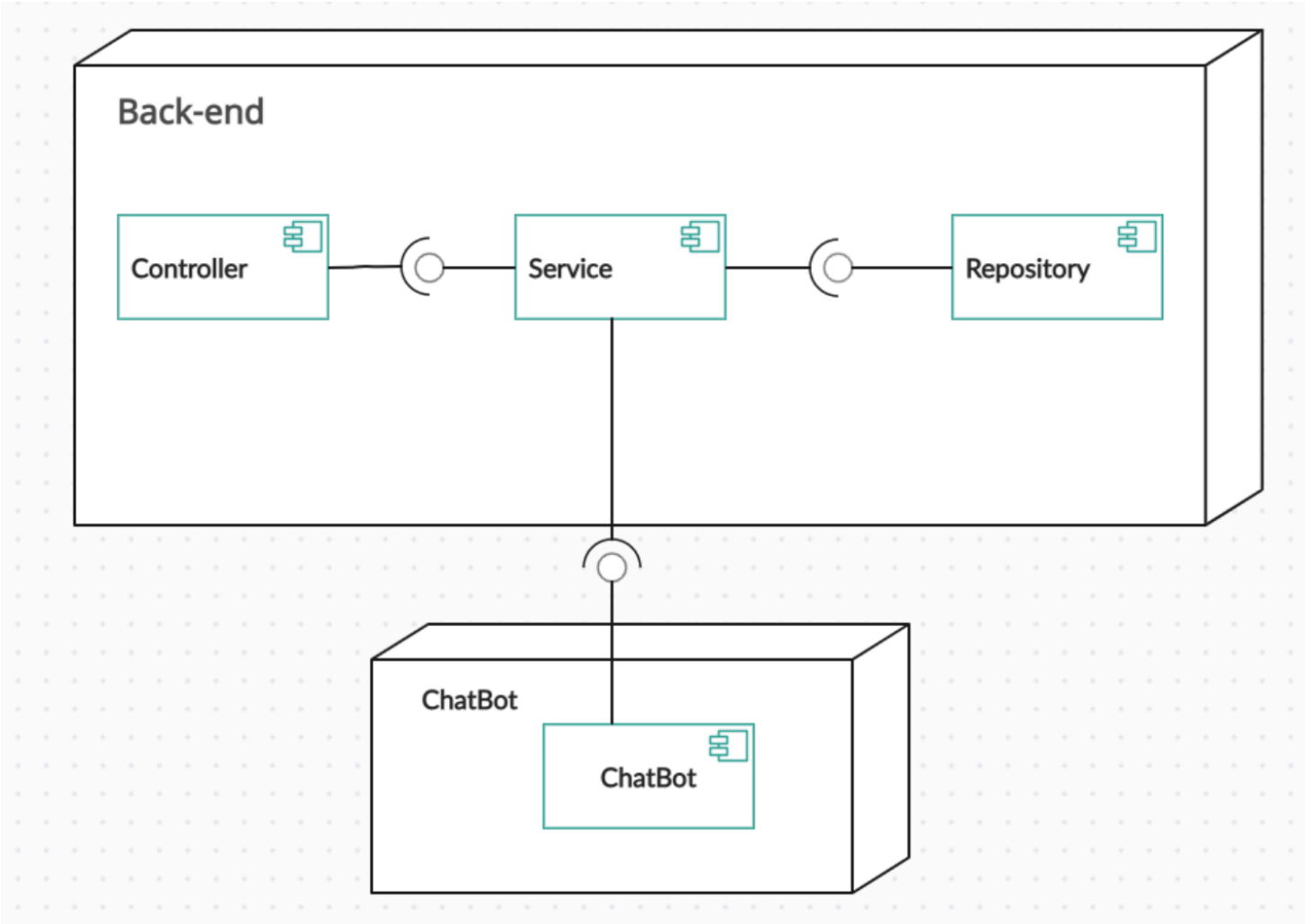

Figure 4: System Component Diagram

4.2 System Class Diagram

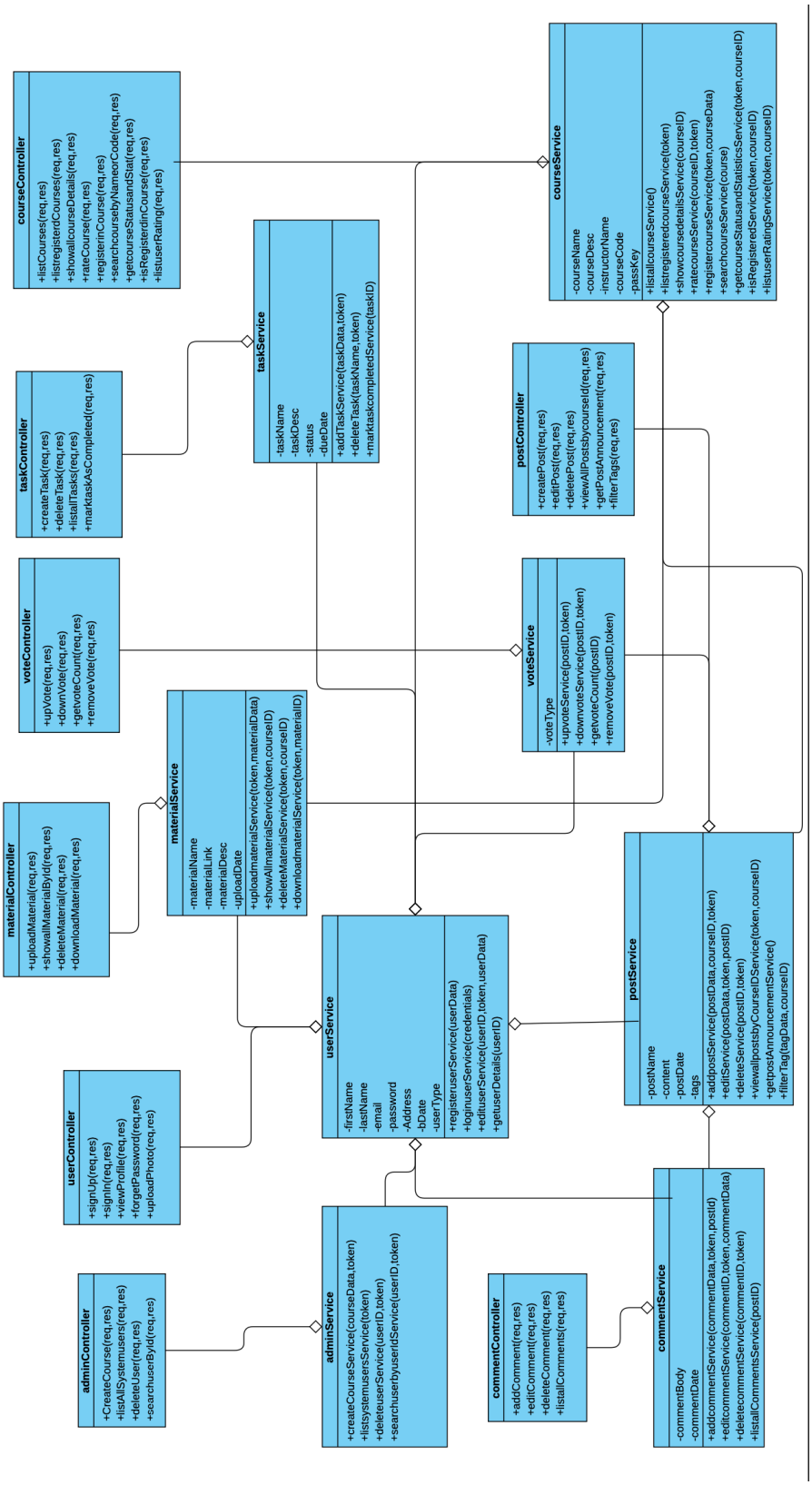

Figure 5: System Class Diagram

Sequence Diagrams

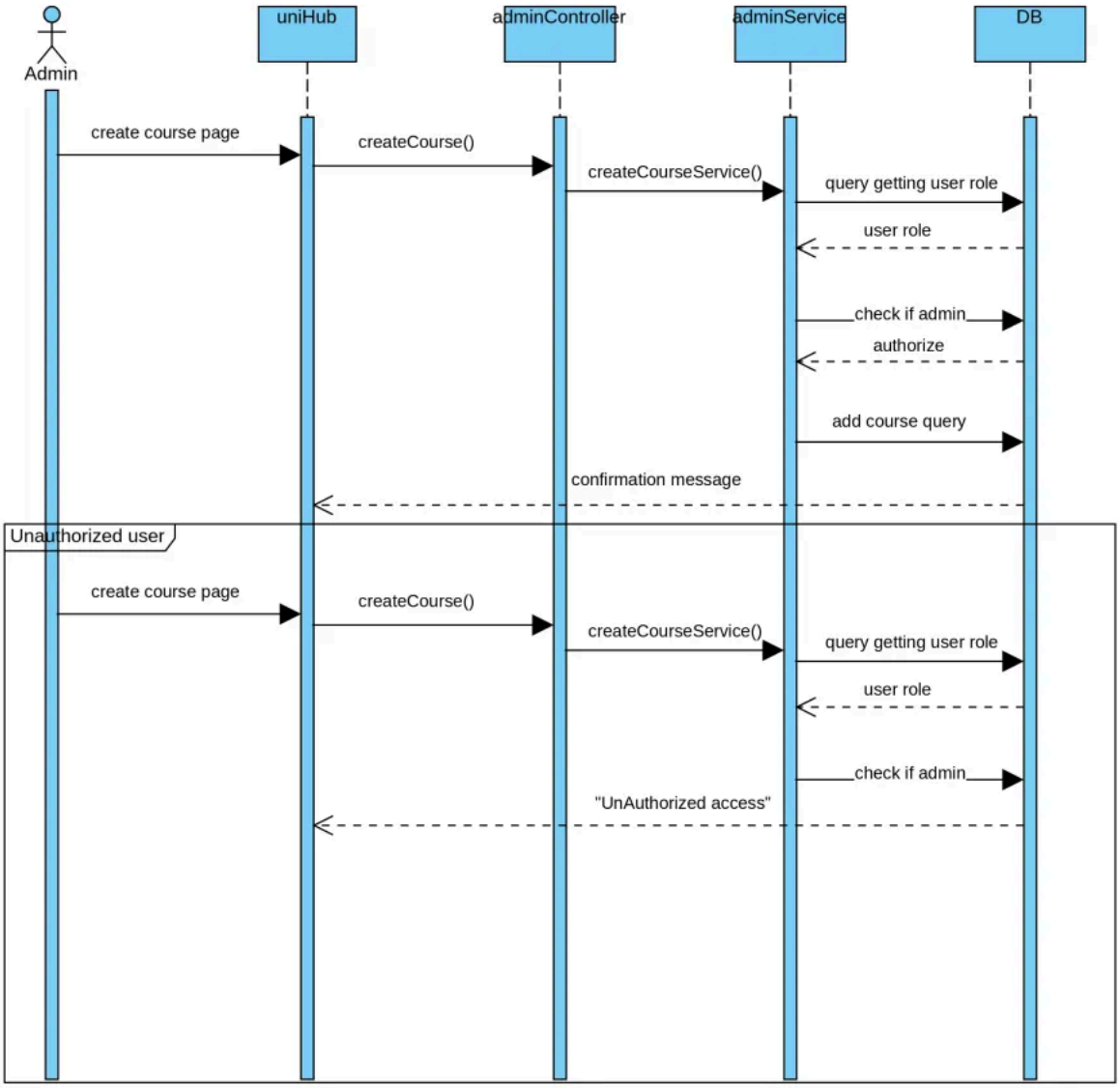

Figure 6: Sequence Diagram (Create Course)

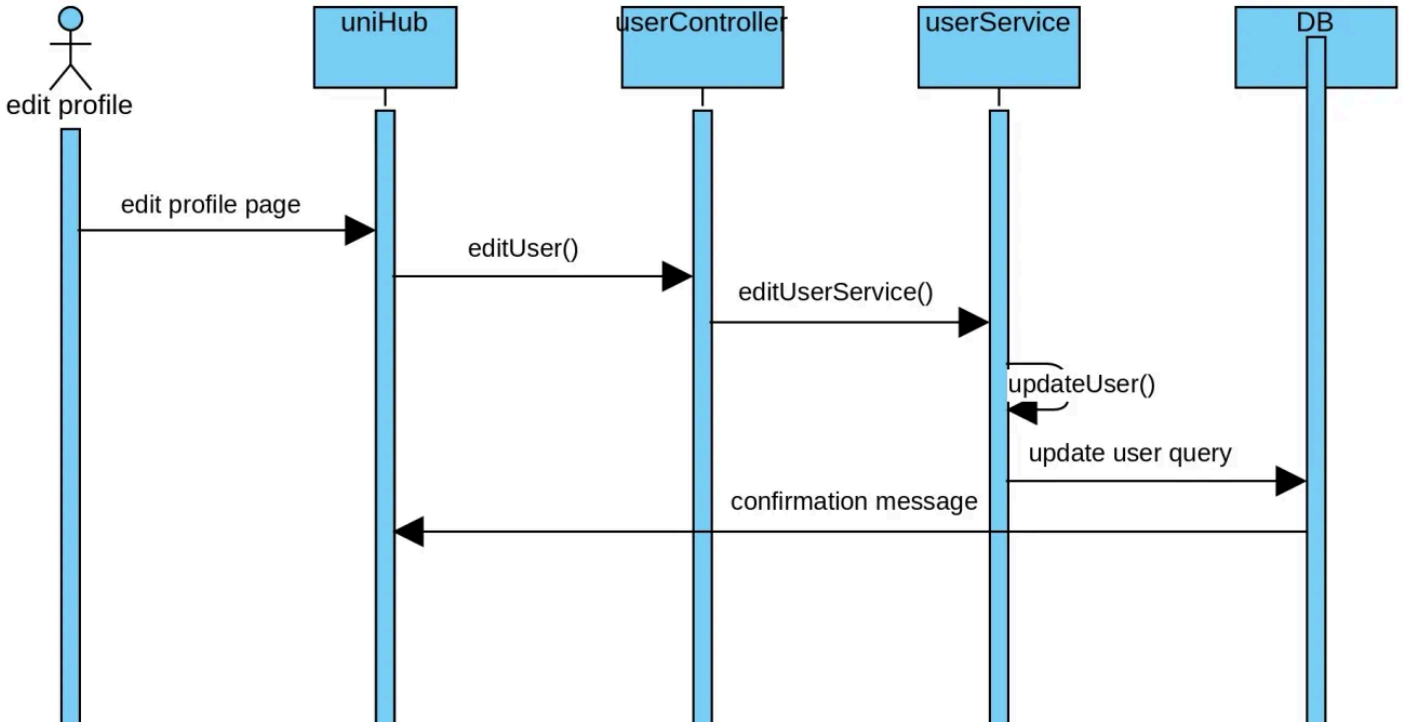

Figure 7: Sequence Diagram (Edit Profile)

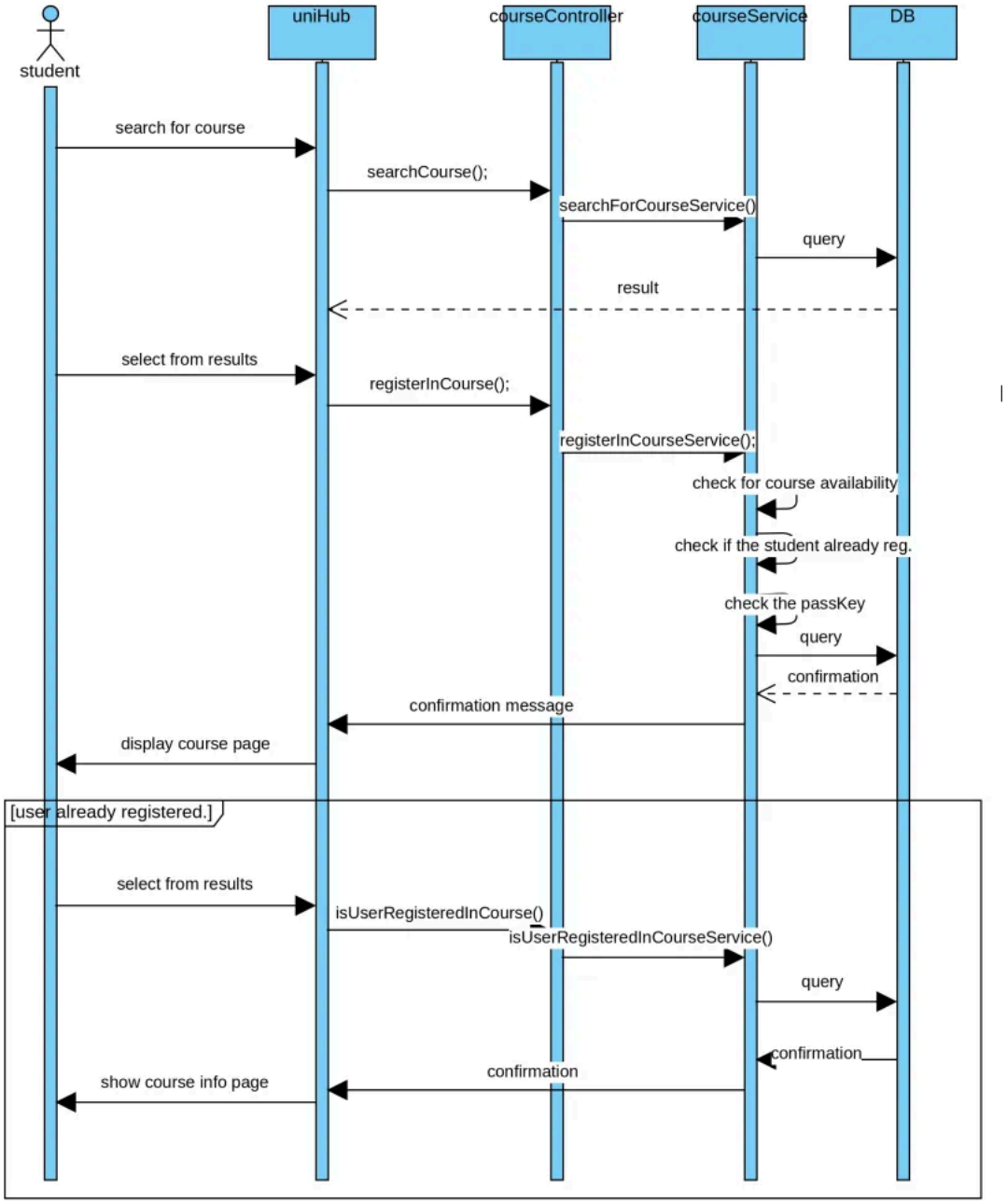

Figure 8: Sequence Diagram (Register in Course)

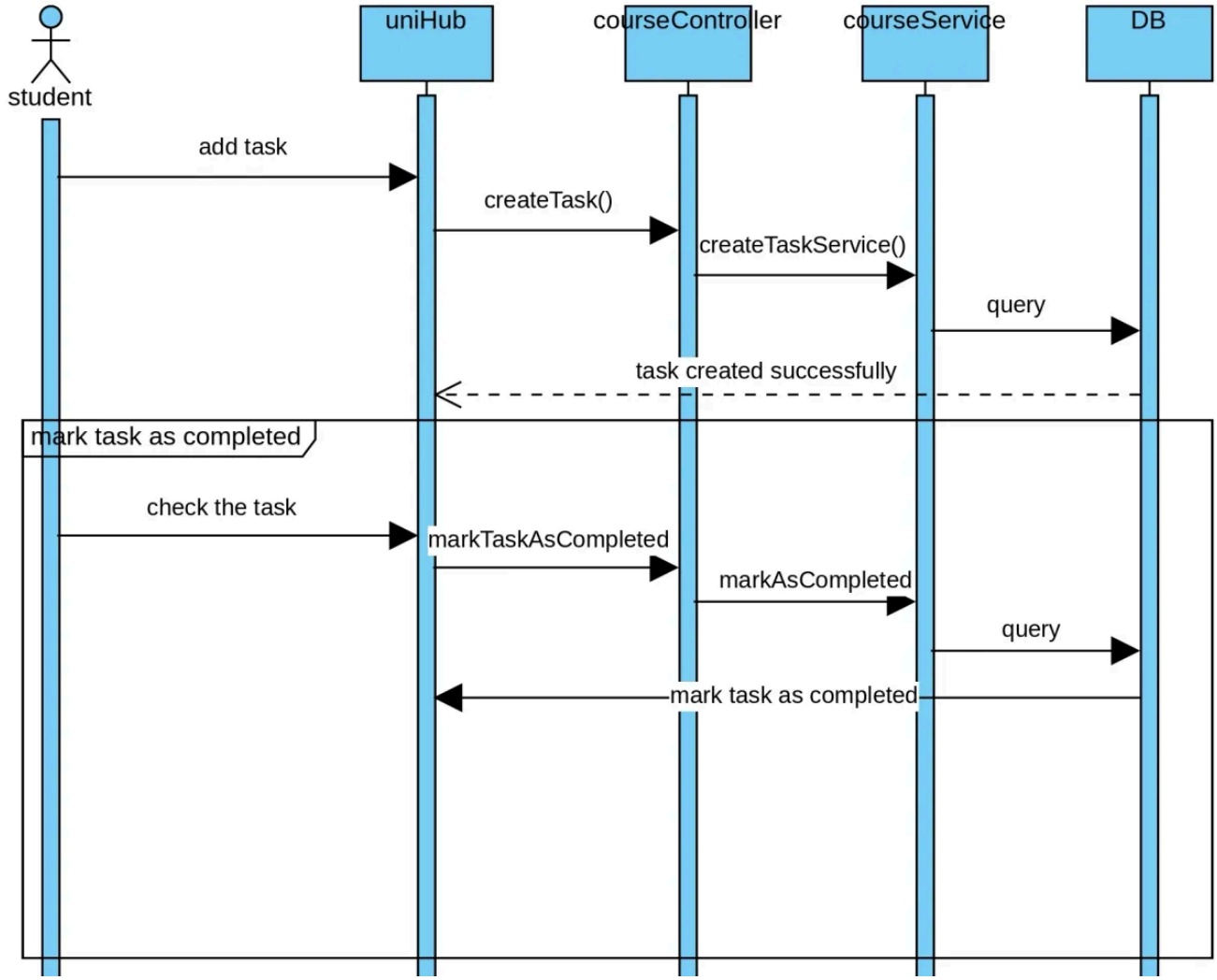

Figure 9: Sequence Diagram (Create Task)

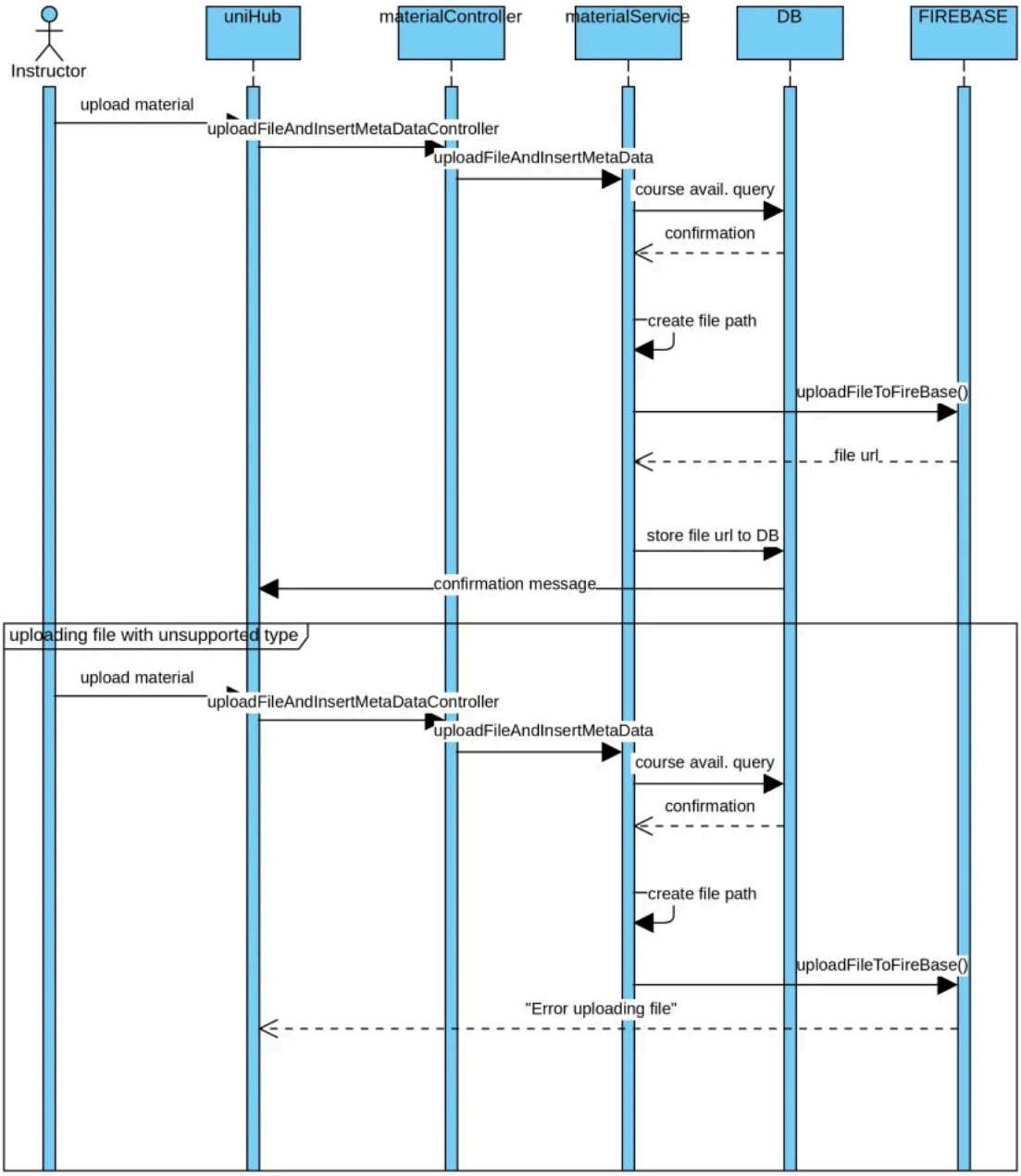

Figure 10: Sequence Diagram (Upload Material)

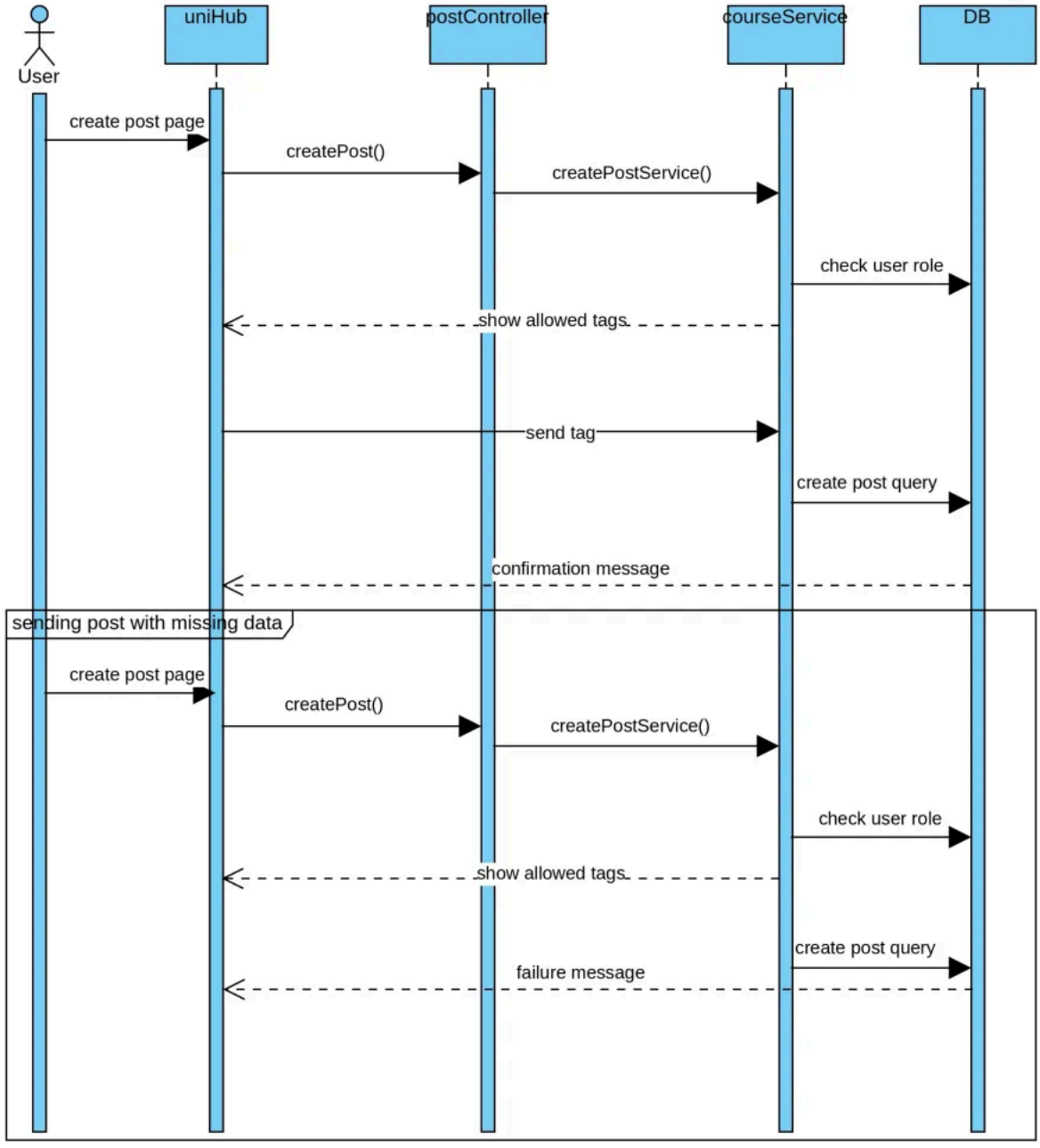

Figure 11: Sequence Diagram (Create Post)

Project ERD

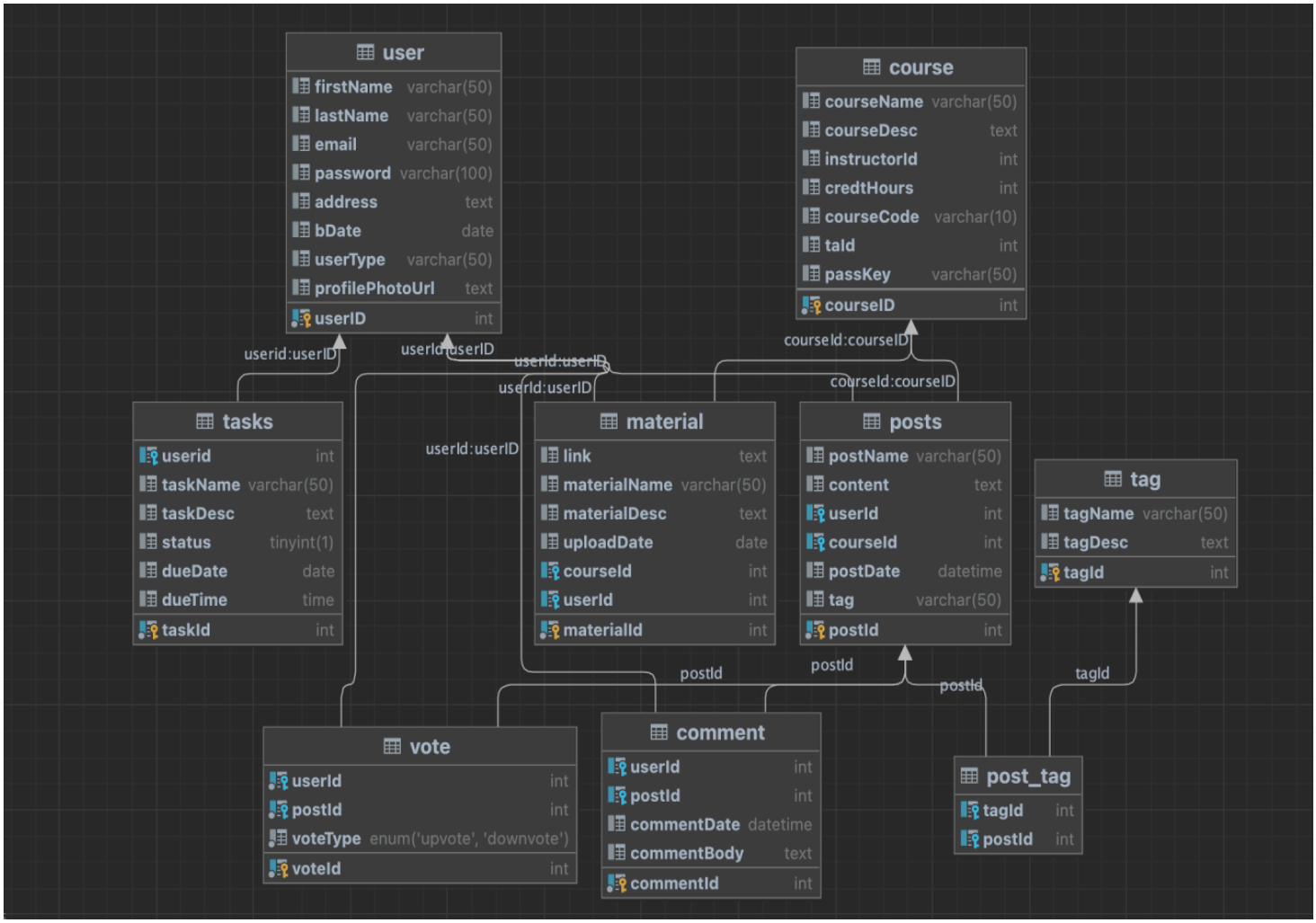

Figure 12: Project ERD

System GUI Design

Login page:

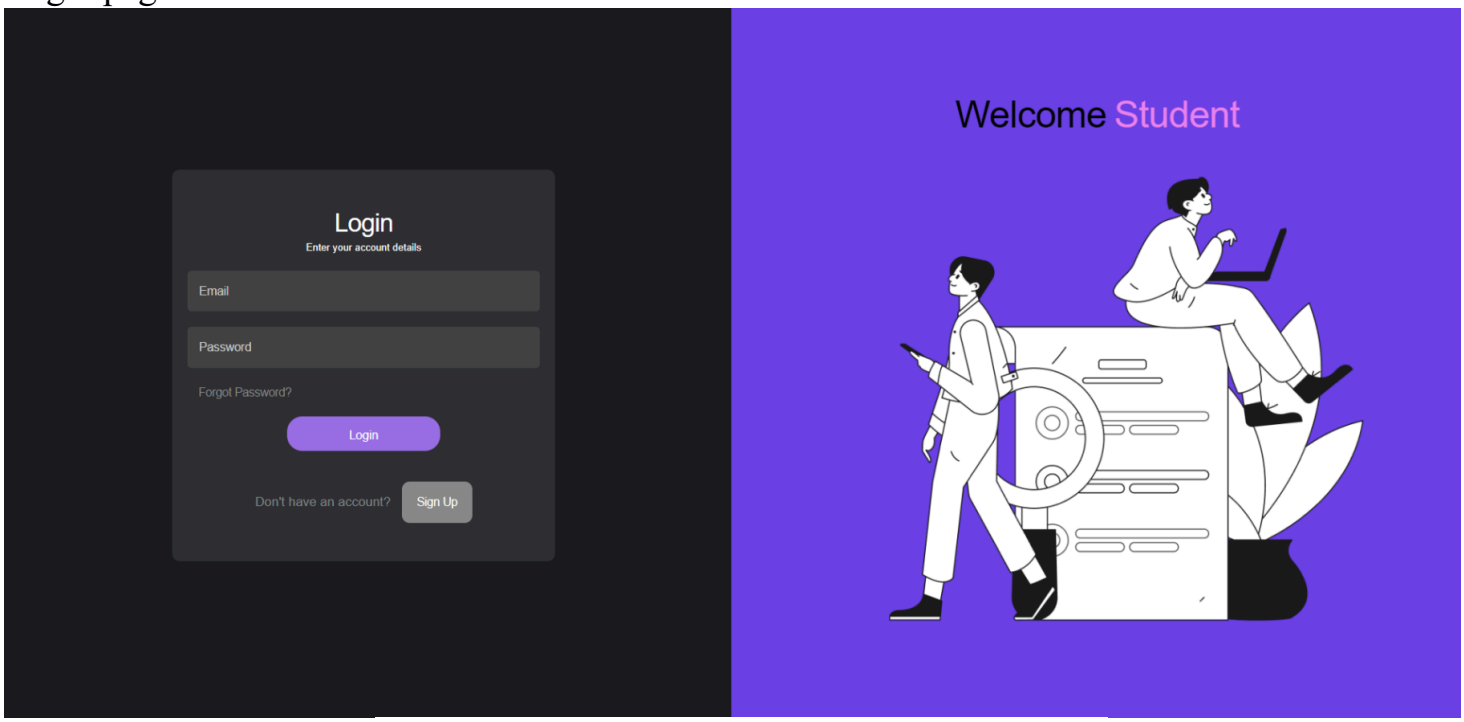

Figure 13: System GUI Design (Login Page)

Register Page:

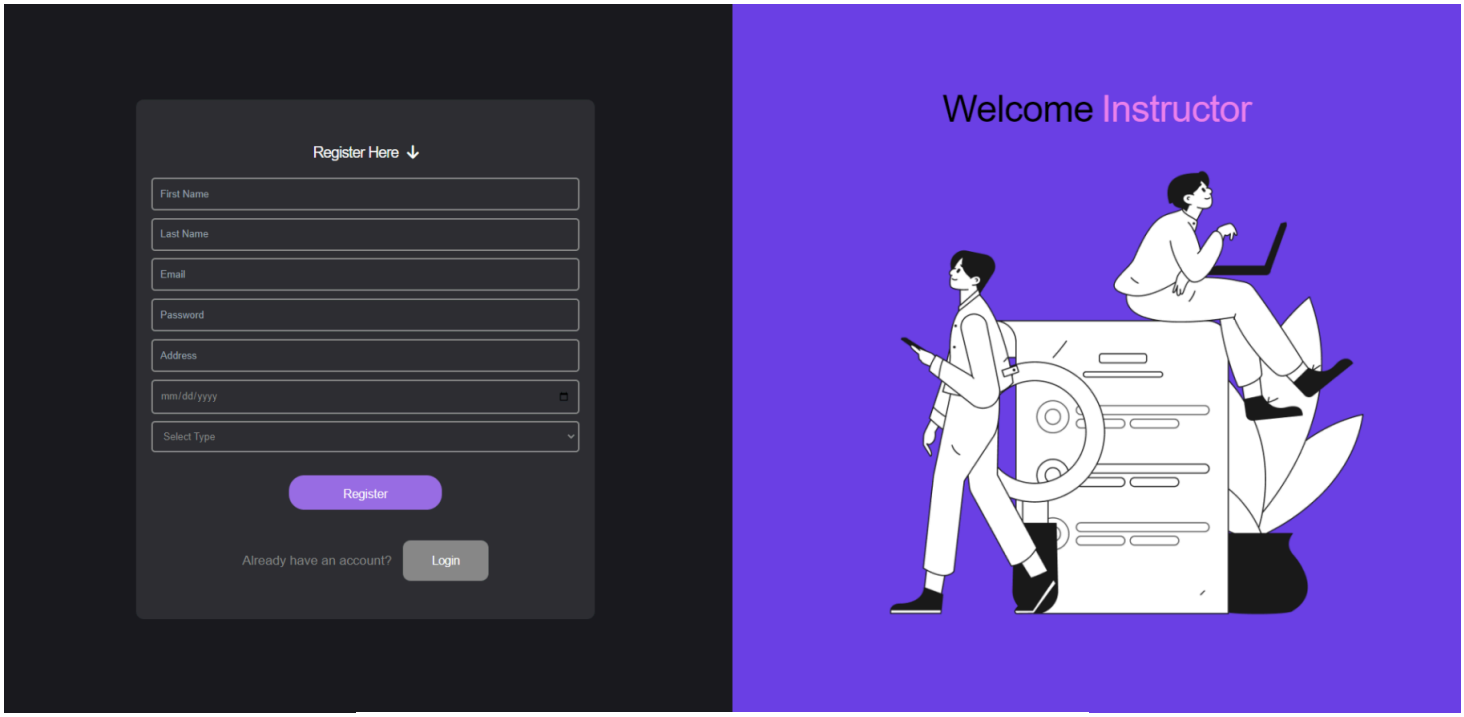

Figure 14: System GUI Design (Register Page)

Student Home Page:

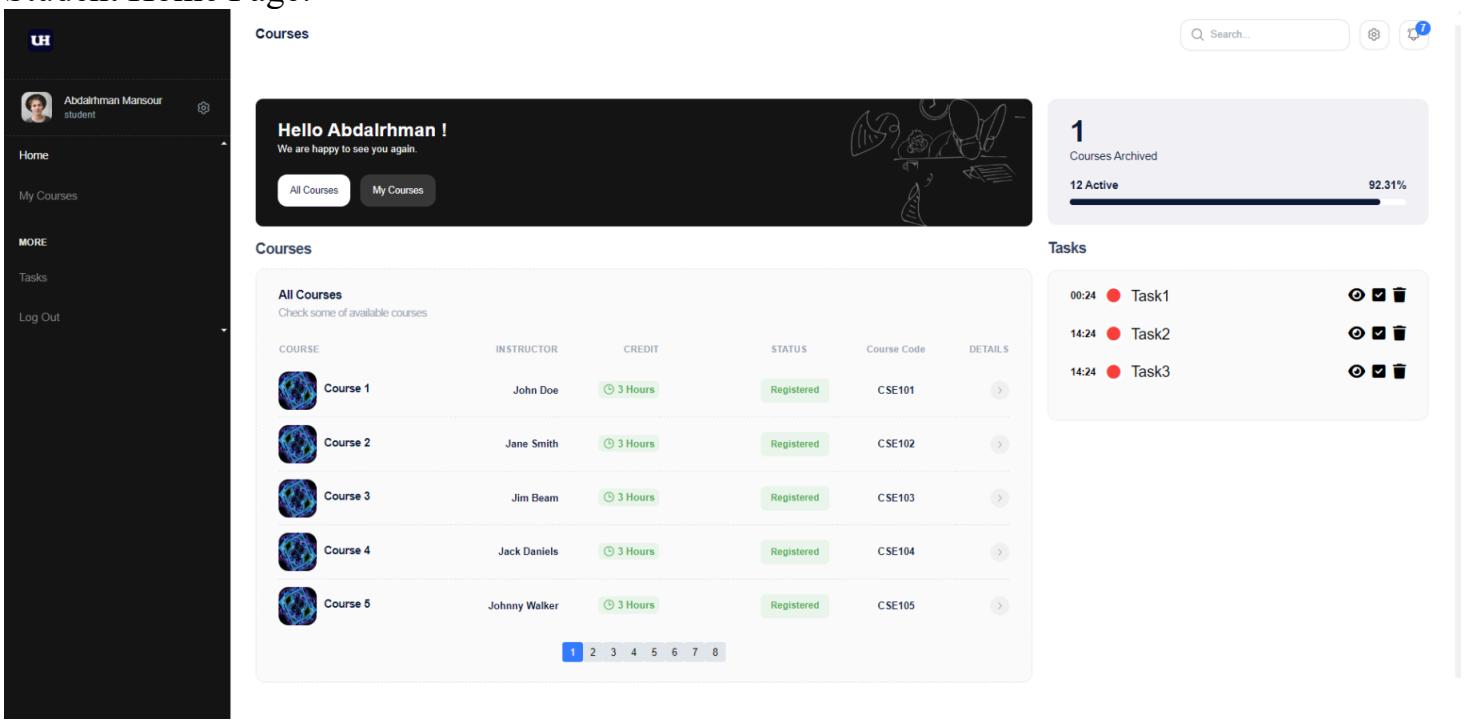

Figure 15: System GUI Design (Student Home Page)

Student Material Page:

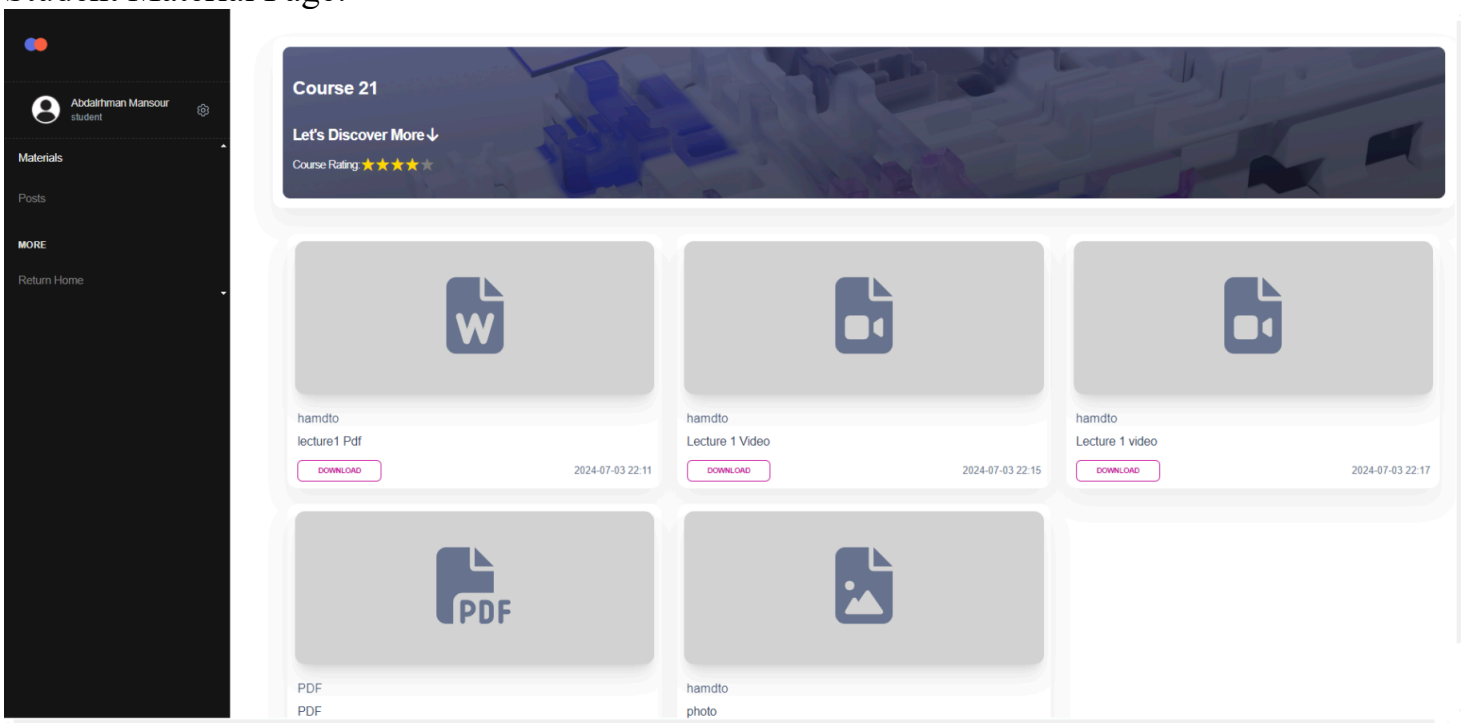

Figure 16: System GUI Design (Student Material Page)

Post Page:

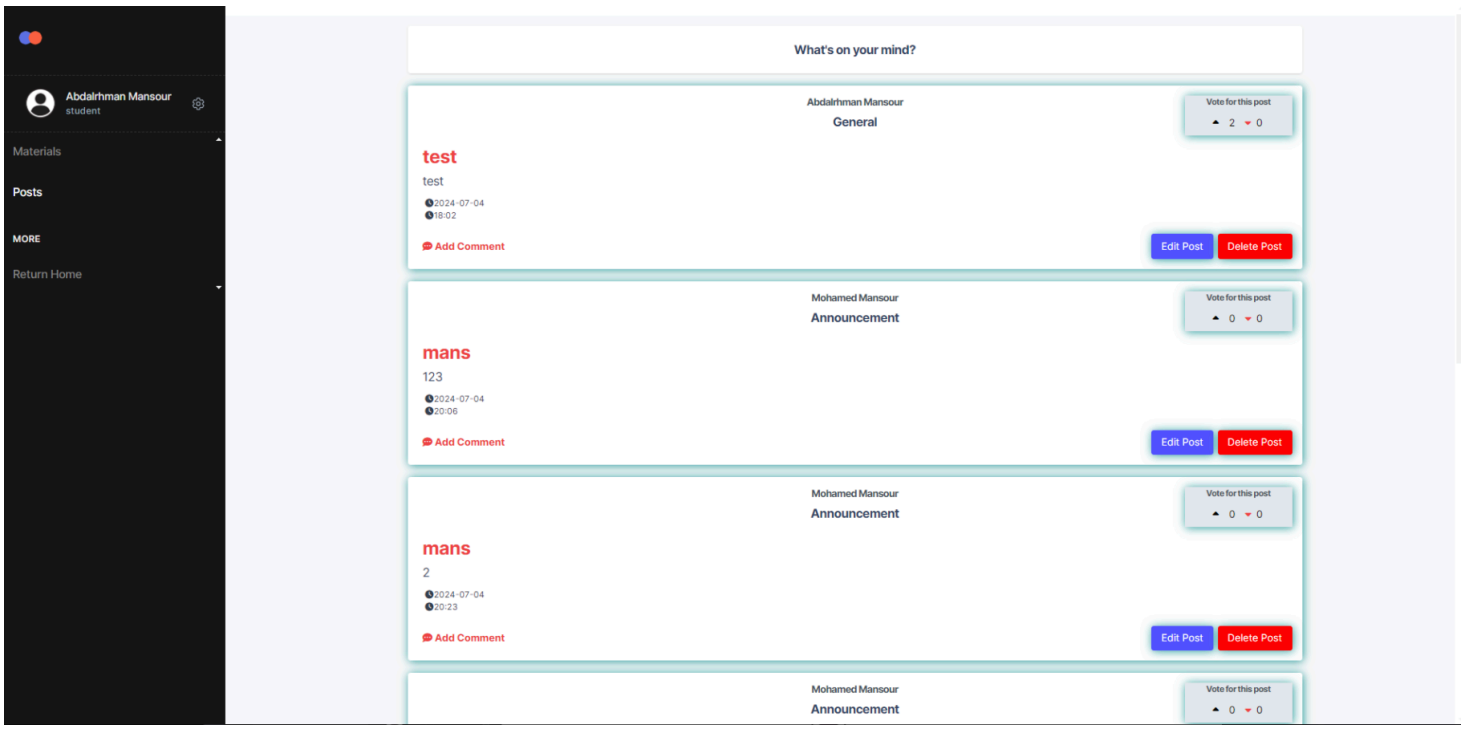

Figure 17: System GUI Design (Post Page)

Post Page with Comments:

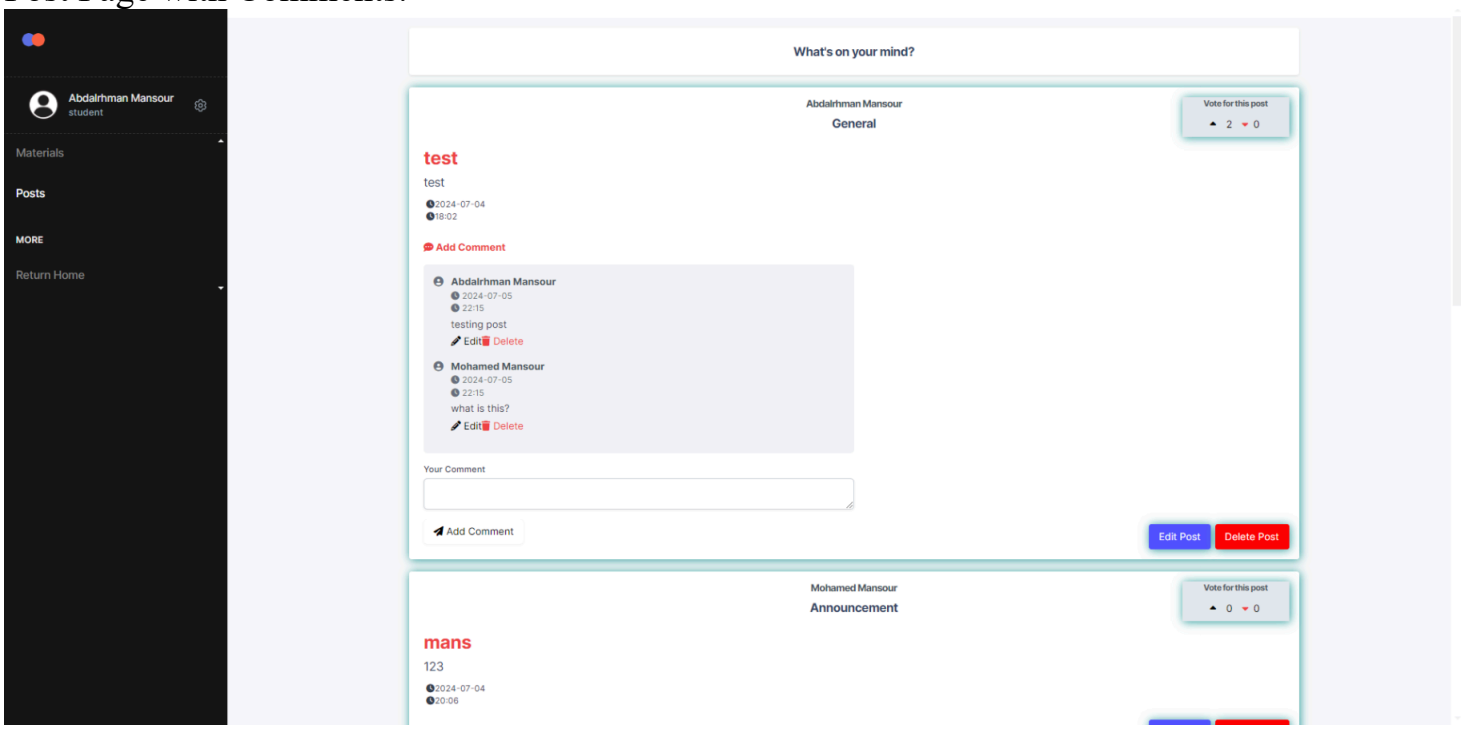

Figure 18: System GUI Design (Post Page with Comments)

Personal Info:

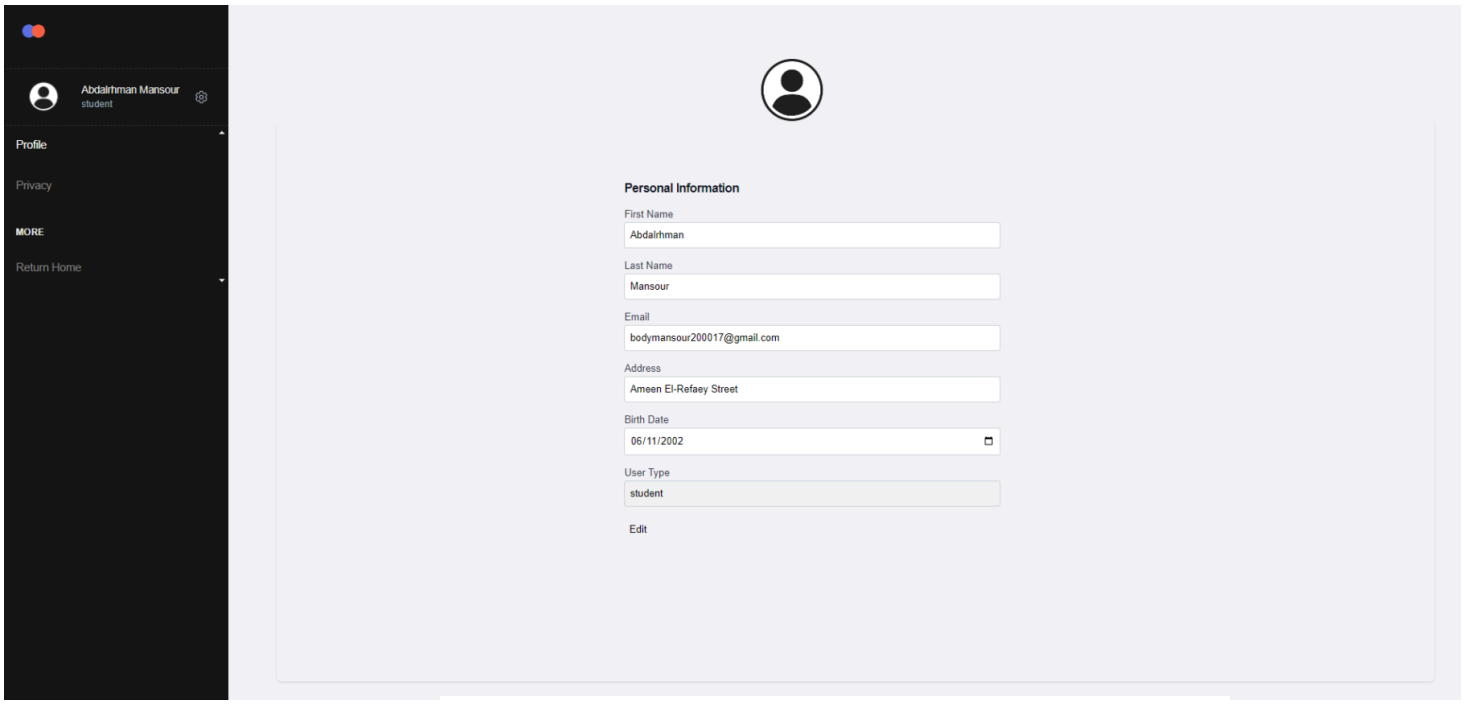

Figure 19: System GUI Design (Personal Info)

Privacy (change password):

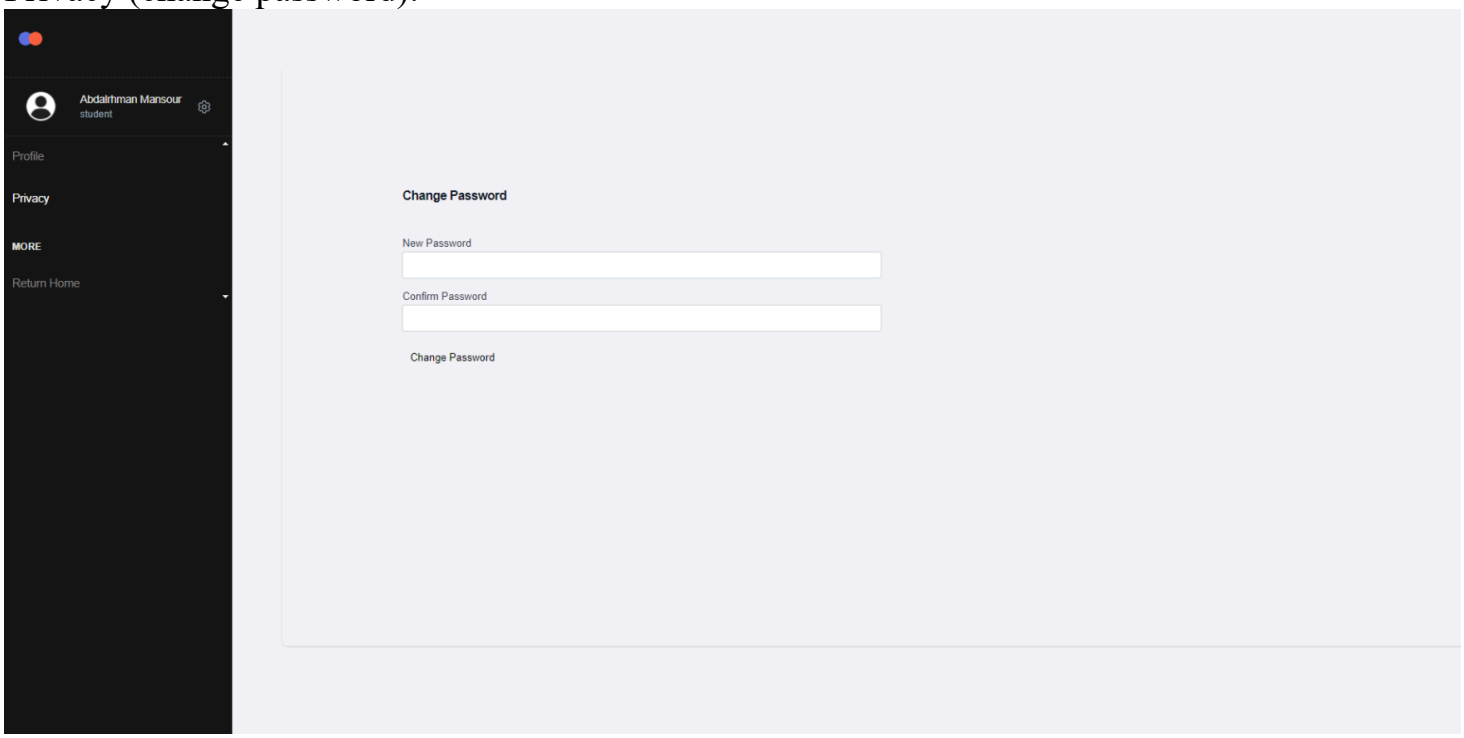

Figure 20: System GUI Design (Privacy: Change Password)

Instructor Material:

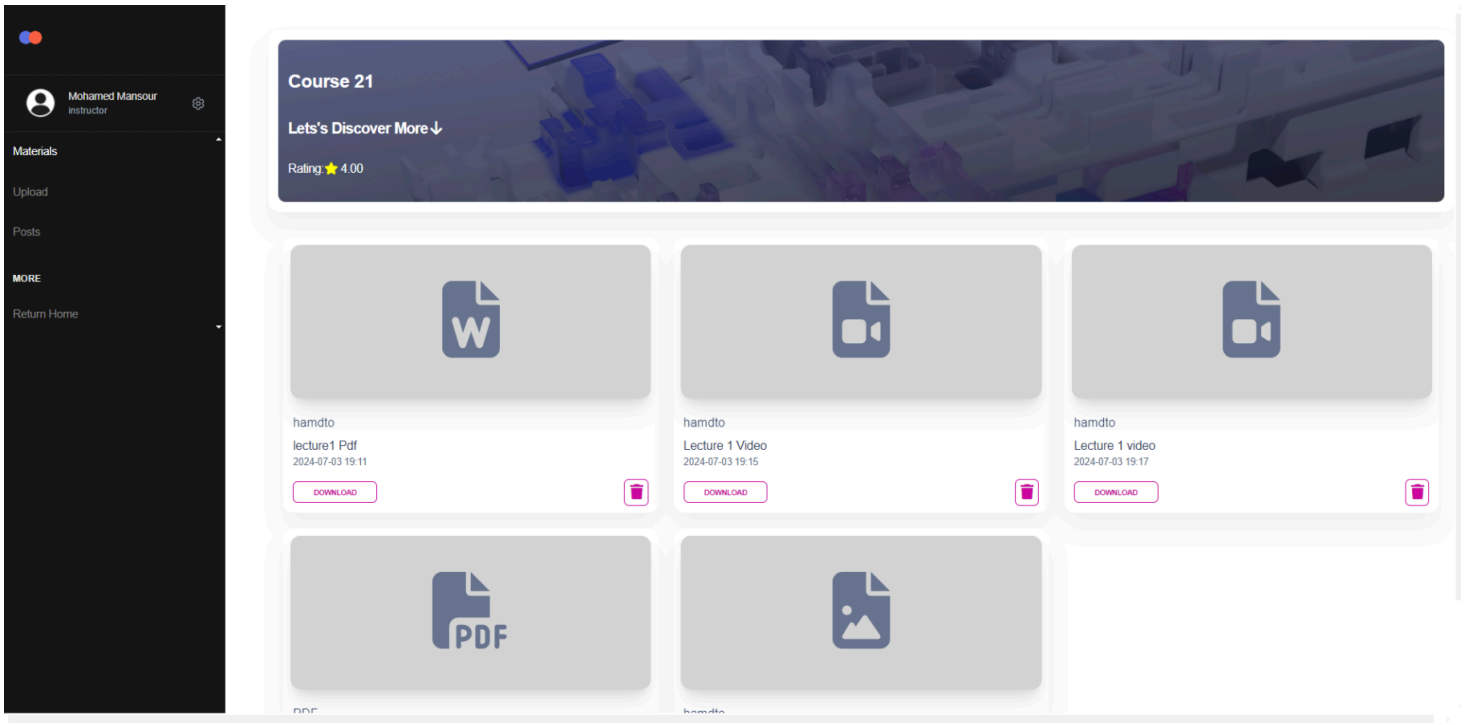

Figure 21: System GUI Design (Instructor Material)

Instructor Upload Material:

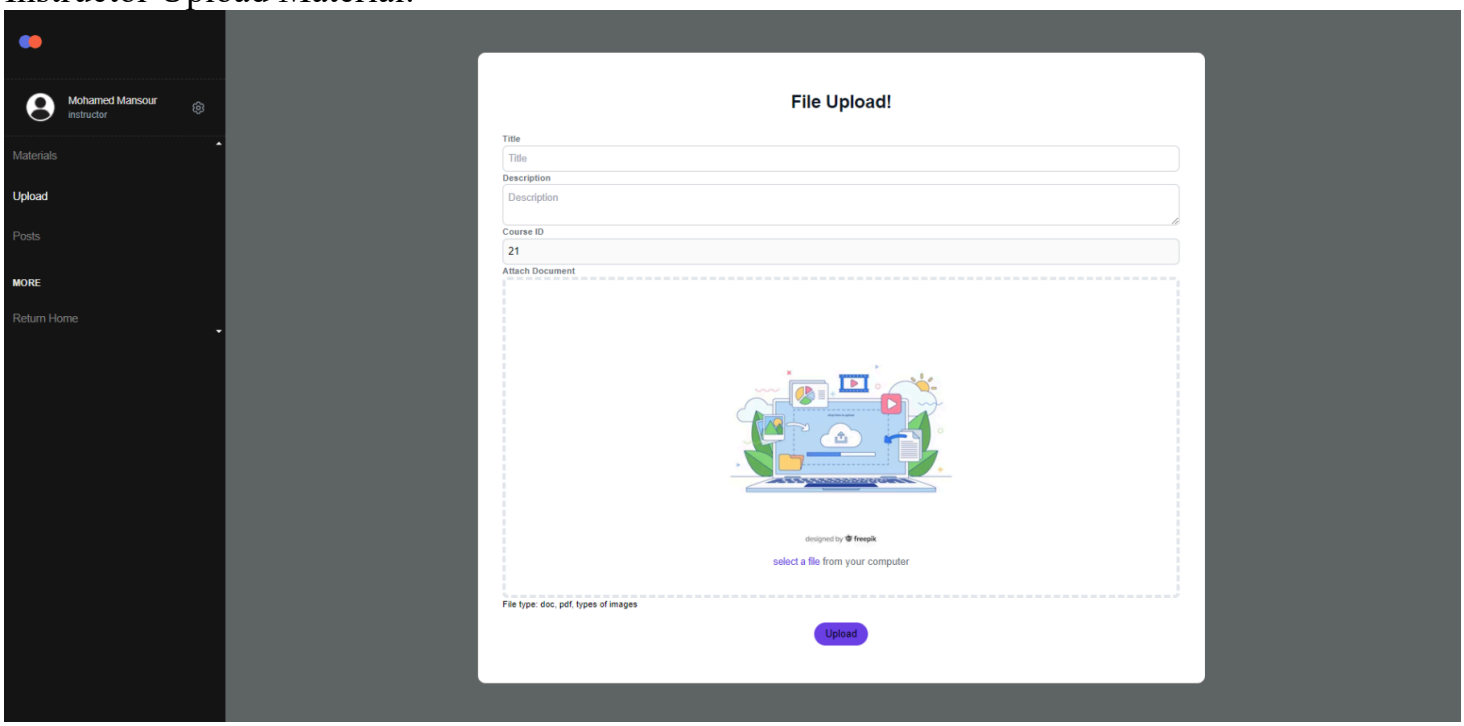

Figure 22: System GUI Design (Upload Material)

Create Course(Admin):

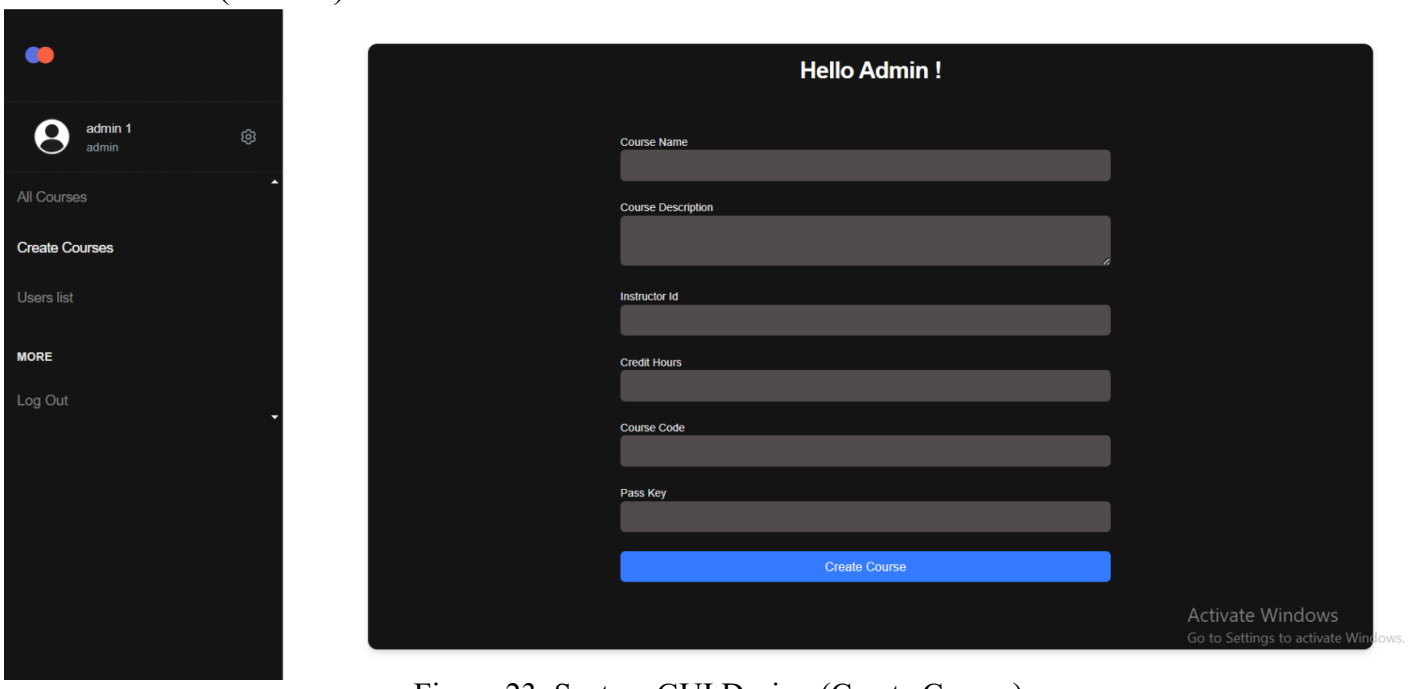

Figure 23: System GUI Design (Create Course)

User Management (Admin):

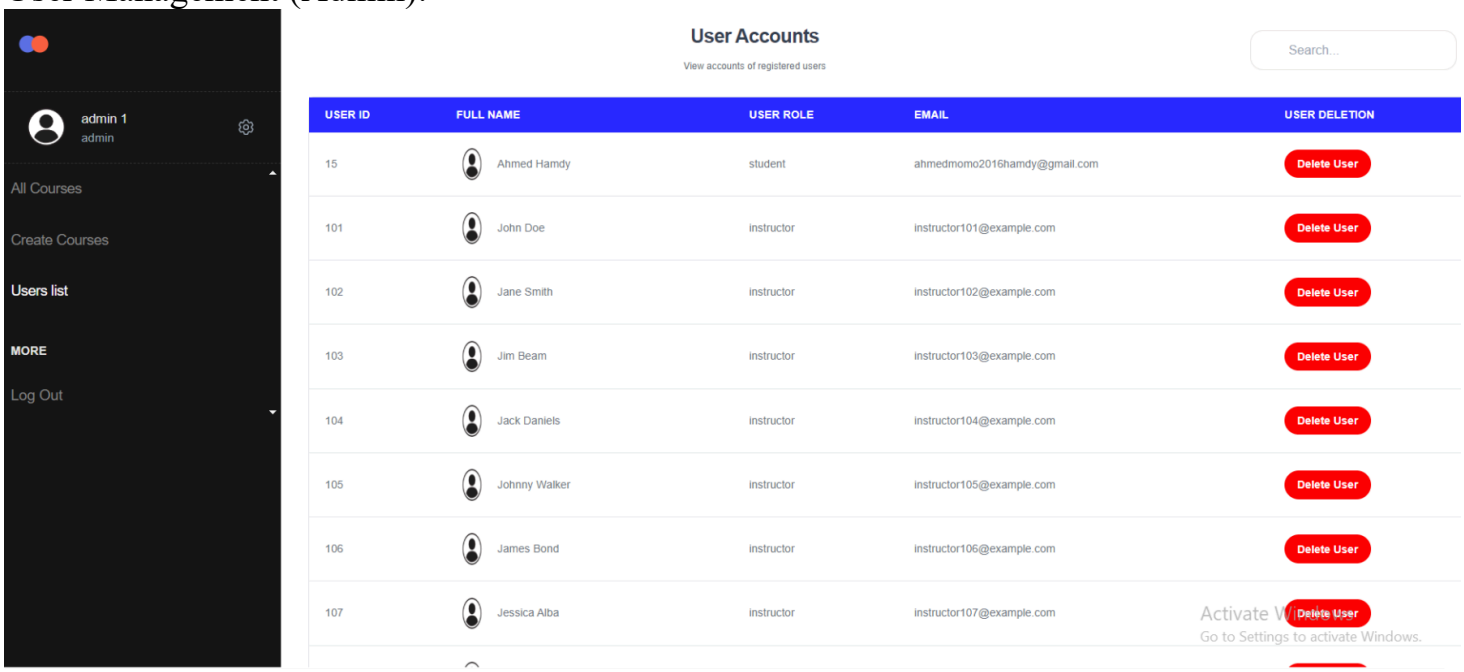

Figure 23: System GUI Design (User Management)

Chapter 5: Implementation and Testing

5.1 Implementation

The implementation of the UniHub project involved developing a digital educational platform using various technologies for the front-end, back-end, database management, and additional functionalities like authentication, file storage, and an AI-driven chatbot. The main technologies used include React.js for the front-end, Node.js for the back-end, MySQL for database management, JWT for authentication, Firebase for file storage, and an AI-driven chatbot integrated with GPT.

The implementation process followed the waterfall methodology, with each phase carefully planned and executed in a sequential manner. Below are the key components and their implementation details:

5.1.1 Front-End (React.js)

The front-end was developed using React.js to create a dynamic and responsive user interface.

5.1.2 Back-End (Node.js)

The back-end was built using Node.js to handle server-side logic and API endpoints. Key features implemented include:

RESTful APIs for user authentication, course management.
Middleware for JWT authentication.
File upload and storage using Firebase.
Integration with the MySQL database for data storage and retrieval.

5.1.3 Database Management (MySQL)

MySQL was used to manage the relational database for UniHub.

5.1.4 Authentication and Authorization (JWT)

JWT was used to implement secure user authentication and authorization. Key features include:

Generation and validation of JWT tokens for user sessions.
Role-based access control to restrict access to certain functionalities based on user roles.

5.1.5 File Storage (Firebase)

Firebase was integrated for file storage to handle the uploading and storing of course materials and assignment submissions. Key features include:

Secure file upload and download functionalities.
Integration with the front-end to provide seamless access to uploaded files.

5.1.6 AI Chatbot (GPT Integration)

An AI-driven chatbot was integrated to provide users with quick answers to common queries. Key features include:

Integration with GPT (Generative Pre-trained Transformer) for Natural Language Processing (NLP) to understand user queries.
Connection to the question bank to provide relevant answers.
Accessibility from the main interface for easy user interaction.

5.2 Testing

Functionality	***Input***	***Expected Output***
Sign Up	Empty	"Sign Up" button is disabled until the form is filled completely
	Full Credentials but with wrong email format First Name: medhat Last Name: fahmy Email: medhat.com Password: Medhat2001! Adress: madenaty al rehab Birth Date: 12-15-2000 User Type: student	Invalid email format error message
	Full Credentials but with wrong email format First Name: medhat Last Name: fahmy Email: medhat@gmail.com Password: Medhat Adress: madenaty al rehab Birth Date: 12-15-2000 User Type: student	Invalid password formatPassword should contain both letters and numbers Uppercase letters and special characters
	Full correct credentials First Name: Mohamed Last Name: Ashraf Email: Mohamed@gmail.com Password: Mohamed2001! Adress: mirag-maadi Birth Date: 1-1-2001 User Type: student	Login successful
Login	Empty	Error (Must Enter Credentials)
	Wrong Email	Error (Wrong Username or Password)
	Wrong Password	Error (Wrong Username or Password)
	Correct Email and Password Email: mohamed@gmail.com Password: Mohamed2001!	Login in Successfully and redirect to the Home Page

Register in course	Empty	Faild to enroll, please check the passkey and try again
	Wrong passkey Passkey: wrong123	Faild to enroll, please check the passkey and try again
	Correct passkey Passkey: cs101	Registered successfully and redirected to course material
Add Task	Empty	Task Name, task description and due date are needed
	Adding task but with expired token Task Name: assignment1 Task Description: programming assignment Due date: 7-19-2024	Unauthorized
Edit user	Editing wrong formatted email email: moo.ash	Invalid email formaterror message
	Editing wrong formatted password Password: moo200	Invalid password formatPassword should contain both letters and numbers Uppercase letters and special characters
	Editing with correct email Email: moo@gmail.com	Profile updated successfully
	Editing with correct password Password: Moo200!	Password updated successfully
Add comment	Empty	Comment body is required
	Adding comment with expired token Comment body: watch youtube	Unauthorized
	Adding comment Comment body: watch youtube	Comment added

Delete comment	A user trying to delete someone's comment Press on delete	Unauthorized to delete this comment
	Admin or instructor deleting a comment Press on delete	Comment deleted successfully
	User who wrote the comment Press on delete button	Comment deleted successfully

Conclusion

In conclusion, the UniHub platform has been developed with the aim of enhancing the educational experience for university students, instructors, and administrators. The platform is designed to facilitate centralized access to educational resources, promote collaborative learning, and streamline course management.

Throughout the project development process, modern programming techniques and best practices have been employed to ensure high performance, security, and a user-friendly interface. The main objectives of the project have been achieved, including providing a comprehensive digital platform for managing educational resources and supporting academic success.

We believe that the UniHub platform will be a powerful and valuable tool for university communities. The platform enables users to efficiently access course materials, interact seamlessly, and receive timely support through the integrated AI-driven chatbot. By fostering a more connected and engaging learning environment, UniHub aims to improve the overall educational experience.

We look forward to sharing the platform with users and receiving their feedback to continue improving and enhancing UniHub in the future. User feedback will be crucial in refining features and ensuring that the platform meets the evolving needs of the academic community.

Finally, we would like to express our gratitude to all the individuals who contributed to this project, especially our Dr Desoky and TA Aya who provided us with invaluable support and guidance. We hope that UniHub will prove to be a useful and effective tool in promoting academic success and enhancing the learning experience for all users.

References

1. React.js Documentation: React - A JavaScript library for building user interfaces. Available at: [React.js](https://reactjs.org/)

2. Node.js Documentation: Node.js - JavaScript runtime built on Chrome's V8 JavaScript engine. Available at: [Node.js](https://nodejs.org/)

3. MySQL Documentation: MySQL - The world's most popular open-source database. Available at: [MySQL](https://dev.mysql.com/doc/)

4. JWT Documentation: JSON Web Tokens - JWT.IO. Available at: [JWT](https://jwt.io/)

5. Firebase Documentation: Firebase - Build and deploy apps quickly using Firebase. Available at: [Firebase](https://firebase.google.com/)

6. OpenAI GPT-3 Documentation: OpenAI API - Language model capabilities and usage. Available at: [OpenAI](https://beta.openai.com/docs/)

7. Tailwind CSS Documentation: Tailwind CSS - A utility-first CSS framework for rapid UI development. Available at: [Tailwind CSS](https://tailwindcss.com/docs)

8. Codecademy Learn Node.js: Learn Node.js - Comprehensive Node.js tutorials. Available at: [Codecademy](https://www.codecademy.com/learn/learn-node-js)

9. Codecademy React 101: Learn the fundamentals of React. Available at: [Codecademy](https://www.codecademy.com/learn/react-101)