DEENIYATH STUDENT ATTENDANCE TRACKING AND MANAGEMENT SYSTEM

A project work submitted to the Jamal Mohamed College (Autonomous), Tiruchirappalli in partial fulfillment of the requirements for the award of the degree of

## MASTER OF SCIENCE IN COMPUTER SCIENCE

### Submitted by

**S. HANSA THASLEEM**

### (Reg No: 23PCS061)

**Under the guidance of**

### S. PRABAVATHI, M.Sc., M.Phil., B.Ed.



**DEPARTMENT OF COMPUTER SCIENCE**

**JAMAL MOHAMED COLLEGE (AUTONOMOUS)**

Accredited with A++ Grade by NAAC (4th Cycle) with CGPA 3.69 out of 4.0

(Affiliated to Bharathidasan University)

**TIRUCHIRAPPALLI – 620 020**

### APRIL 2025

# JAMAL MOHAMED COLLEGE (AUTONOMOUS)

Accredited with A++ Grade by NAAC (4th Cycle) with CGPA 3.69 out of 4.0

(Affiliated to Bharathidasan University)

**TIRUCHIRAPPALLI – 620 020**



This is to certify that the project work entitled

DEENIYATH STUDENT ATTENDANCE TRACKING AND

MANAGEMENT SYSTEM

is a bonafide record of the work done

by

**S. HANSA THASLEEM**

### (Reg No: 23PCS061)

at Jamal Mohamed College (Autonomous), Tiruchirappalli during the year 2024-2025 in partial fulfillment of the requirements for the award of the degree of

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

S. PRABAVATHI, M.Sc., M.Phil., B.Ed. Head of the Department Internal Guide

Submitted for the Viva Voce examination held at Jamal Mohamed College, Tiruchirappalli – 20 on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Internal Examiner External Examiner

**ACKNOWLEDGEMENTS**

Gratitude is a sense of thankfulness to the hands that stretched out towards us in help. I am grateful to the Lord Almighty who has showered his grace and enlightened us through this project. Words are inadequate to express our heartiest, sincere and grateful thanks to the following persons.

My first teachers and well-wishers, our first guide, I first express our deepest gratitude to our parents who were our strength and for their constant support in our every single step.

I extend my heartfelt appreciation to the Members of the College Management Committee for providing the necessary facilities and a conducive environment for academic pursuits.

I am deeply grateful to our Principal, **Dr. D.I. George Amalarethinam**, for his constant encouragement and invaluable support in facilitating this project work.

High among the kings is the emperor with bright visions and novel ideas. I extend my thanks to **Dr. M. Mohamed Surputheen**, Coordinator-Department of Computer Science and IT and our Member In-charge of the Department **Dr. S. Adbul Saleem**, Associate Professor of Computer Science for their support and encouragement.

I am immensely thankful to my project Guide, **S. Prabavathi**, whose expertise, patience, and constructive feedback have been instrumental in the successful completion of this work. His continuous support and encouragement have been invaluable.

My special thanks to our class teacher **DR. S. VAAHEETHA KFATHEEN**, for assisting me in all the activities during the course of study.

I also wish to extend my gratitude to all the faculty and staff members of the department for their assistance, cooperation, and for sharing their knowledge, which has significantly contributed to the success of this project.

Lastly, I would like to express my appreciation to my friends, family, and all those who have directly or indirectly contributed to the successful completion of this project work.

**S. HANSA THASLEEM**

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page No.** |
|  | Certificates |  |
|  | Acknowledgements |  |
| 1 | INTRODUCTION | 2 |
|  | 1.1 Problem Definition | 2 |
|  | 1.2 Objective of the Project | 2 |
|  | 1.3 Web Application | 3 |
| 2 | SYSTEM ANALYSIS | 5 |
|  | 2.1 Existing System | 5 |
|  | 2.1.1 Existing Disadvantage | 5 |
|  | 2.2 Proposed System | 5 |
|  | 2.2.1 Proposed Disadvantage | **6** |
| 3 | SYSTEM REQUIREMENTS | **7** |
|  | 3.1 Hardware Requirements | **7** |
|  | 3.2 Software Requirements | **7** |
|  | 3.3 Software Description | **7** |
|  | 3.3.1 Java | **7** |
|  | 3.3.2 Java Spring Boot | 10 |
|  | 3.3.3 HTML | 13 |
|  | 3.3.4 MySQL Database | 14 |
| 4 | PROJECT DESCRIPTION | 20 |
|  | 4.1 Portal Functionality Overiew | 21 |
|  | 4.1.1 Admin Portal | 21 |
|  | 4.1.2 Co-Ordinator Portal | 21 |
|  | 4.1.3 Staff Portal | 22 |
| 5 | SYSTEM DESIGN | 23 |
|  | 5.1 Architecture Diagram | 23 |
|  | 5.2 Flow Chart | 23 |
|  | 5.2.1 Admin Flow Chart | 24 |
|  | 5.2.2 Co-Ordinator Flow Chart | 24 |
|  | 5.2.3 Staff Flow Chart | 24 |
|  | 5.3 Database ER Diagram | 25 |
| 6 | IMPLEMENTATION | 26 |
|  | 6.1 Implementation | 26 |
|  | 6.1.1 System Setup and Technology Stack | 26 |
|  | 6.1.2 Backend Development | 26 |
|  | 6.1.3 Frontend Development | 27 |
|  | 6.1.4 Database Integration | 27 |
| 7 | TESTING | 29 |
|  | 7.1 Unit Testing | 29 |
|  | 7.2 Functional Testing | 30 |
|  | 7.3 Functional Testing | 30 |
|  | 7.4 User Interface Testing | 31 |
| 8 | CONCLUSION | 33 |
|  | 8.1 Conclusion | 33 |
|  | 8.2 Future Enhancements | 33 |
|  | BIBLIOGRAPHY | 35 |
| A | SAMPLE CODE | 35 |
| B | SCREEN SHOTS | 40 |

# ABSTRACT

Deeniyath Student Attendance Management System is a web-based solution that digitizes attendance tracking and class management, enhancing accuracy, efficiency, and accessibility. Built with Java (Spring Boot) for the backend, Angular, HTML, and CSS for the frontend, and MySQL for database management, it ensures seamless functionality and real-time synchronization.

The system provides administrators and teachers with up-to-date records, reducing errors and improving collaboration. Its user-friendly interface minimizes manual effort, boosts productivity, and supports better decision making. Designed for scalability, it alows future enhancements like analytics and notifications, making it an essential tool for educational institutions.

# CHAPTER 1

**INTRODUCTION**

## PROBLEM DEFINITION

In educational institutions, tracking and managing student attendance plays a vital role in maintaining discipline and monitoring academic performance. However, the conventional methods of recording attendance, such as maintaining physical registers or manual data entry, are inefficient, error-prone, and difficult to manage, especially when handling large student populations.

To address these challenges, the Deeniyath Student Attendance Management System has been developed as a web-based application that automates the entire attendance process. This system is designed to simplify attendance tracking, ensure data accuracy, and provide quick access to attendance records anytime, anywhere.

The application leverages modern web technologies, including Java (Spring Boot) for backend development, Angular, HTML, and CSS for the user interface, and MySQL as the database for secure data storage. It offers a user-friendly platform where administrators, faculty members, and students can seamlessly interact with the system based on their assigned roles.

One of the key advantages of this system is the real-time synchronization of attendance data, which helps administrators and teachers in making informed decisions. It minimizes manual workload, enhances transparency, and allows for easy generation of attendance reports.

Additionally, the system is built to be scalable and flexible, allowing institutions to incorporate future features like analytics dashboards, mobile app integration, and automated notifications. By digitizing the attendance process, the system contributes to better academic management and promotes a paperless, efficient environment.

## OBJECTIVE OF THE PROJECT

The Deeniyath Student Attendance Management System is developed to improve the accuracy and efficiency of attendance tracking within educational institutions. The key objectives of this project are:

1. **To Automate Attendance Recording:** Replace the traditional manual attendance system with a digital platform that allows teachers to mark student attendance quickly and efficiently.
2. **To Improve Accuracy and Minimize Errors:** Reduce the chances of human errors in attendance management by securely storing accurate attendance records in a centralized database.
3. **To Provide Real-Time Access to Attendance Data:** Enable administrators, teachers, and students to access attendance records instantly, ensuring transparency and timely information.
4. **To Enhance Communication and Accountability:** Ensure that attendance records are accessible to all authorized users, promoting accountability among students and faculty.
5. **To Facilitate Easy Retrieval and Management of Attendance Records**: Allow administrators and teachers to search, filter, and generate attendance reports effortlessly for academic and administrative purposes.
6. **To Reduce Manual Workload and Paper Usage:** Eliminate the need for physical registers, thereby reducing paperwork and simplifying attendance management.
7. **To Generate Attendance Reports for Analysis:** Provide institutions with structured attendance reports that help in analyzing student participation and academic discipline.
8. **To Develop a User-Friendly Web Application:** Ensure that the system has an intuitive and accessible interface, allowing users to operate it without requiring advanced technical knowledge.

By achieving these objectives, the Deeniyath Student Attendance Management System aims to enhance the overall efficiency, transparency, and reliability of attendance tracking in educational institutions.

## 1.3 WEB APPLICATION

A Web Application is a software application that runs on a web server and can be accessed by users through a web browser over the Internet or an intranet. Web applications are designed to provide interactive services to users, allowing them to perform specific tasks or access information in real time without the need to install dedicated software on their devices.

Web applications can range from simple applications, such as online forms and surveys, to complex platforms like e-commerce portals, learning management systems, and data-driven management systems. They typically involve a combination of client-side and server-side programming to deliver dynamic and responsive user experiences.

In the case of the Deeniyath Student Attendance Management System, the web application is developed to facilitate the automated tracking and management of student attendance. It uses modern web technologies such as Java (Spring Boot) for server-side development, Angular, HTML, and CSS for client-side interface design, and MySQL for database management.

Web applications are preferred in academic and business environments due to their accessibility, scalability, and ease of maintenance. Users can access the application from any device with an internet connection, making the system highly flexible and user-friendly.

Web application development typically involves the following components:

* **Front-End Development:** Responsible for the visual layout and user interface that runs in the user's browser.
* **Back-End Development:** Handles the server-side logic, database interactions, and core functionalities.
* **Database Management:** Stores and retrieves data efficiently for use within the application.
* **Security and Maintenance:** Ensures secure access to sensitive information and provides regular updates for smooth operation.

The development of the Deeniyath Student Attendance Management System as a web application ensures that attendance records can be managed effectively, accessed conveniently, and maintained securely by all stakeholders of the institution.

# CHAPTER 2

**SYSTEM ANALYSIS**

## 2.1 EXISTING SYSTEM

In many educational institutions, student attendance is managed manually using physical registers or paper-based records, where faculty members mark each student’s presence or absence during class sessions. This traditional method is inefficient, time-consuming, and prone to human errors, particularly when handling large numbers of students. Maintaining and retrieving past attendance records becomes challenging and labor-intensive. Additionally, the manual system lacks real-time accessibility and transparency, making it difficult for administrators, teachers, and students to track attendance efficiently. Due to these limitations, there is an increasing need for a web-based automated system that simplifies attendance management, improves accuracy, and provides quick access to records.

## 2.1.1 EXISTING DISADVANTAGES

* Time-Consuming and Tedious Process
* Prone to Human Errors and Data Inaccuracy
* Lack of Real-Time Access to Attendance Records
* Risk of Data Loss or Misplacement
* Limited Security and Confidentiality of Data
* Difficulty in Generating Attendance Reports
* Increased Administrative Workload

## 2.2 PROPOSED SYSTEM

The proposed Deeniyath Student Attendance Management System is a web-based application designed to automate and simplify the process of recording, managing, and monitoring student attendance. This system eliminates the drawbacks of the traditional manual method by providing a digital platform where administrators, teachers, and students can access attendance data in real-time. The application is built using Java (Spring Boot) for backend development, Angular, HTML, and CSS for the front-end interface, and MySQL for secure data storage.

The system allows teachers to mark attendance quickly and accurately, while administrators can manage student and faculty records efficiently. Students can also view their attendance status through the application. The proposed system ensures better accuracy, reduces manual workload, and enhances transparency in attendance tracking.

## 2.2.1 PROPOSED DISADVANTAGES

* Time-Consuming and Tedious Process
* Prone to Human Errors and Data Inaccuracy
* Lack of Real-Time Access to Attendance Records
* Risk of Data Loss or Misplacement
* Limited Security and Confidentiality of Data
* Difficulty in Generating Attendance Reports
* Increased Administrative Workload

# CHAPTER 3

**SYSTEM REQUIREMENTS**

## 3.1 HARDWARE REQUIREMENTS

* Processor: Intel Core i3 or higher
* RAM: Minimum 4 GB (8 GB recommended)
* Hard Disk: Minimum 100 GB free space
* Display: 1024 × 768 resolution or higher
* Network: Internet connectivity for web access

## 3.2 SOFTWARE REQUIREMENTS

* Operating System: Windows 10 / Linux / Mac OS
* Backend Technology: Java (Spring Boot Framework)
* Frontend Technology: Angular, HTML, CSS
* Database: MySQL Server
* Web Browser: Google Chrome / Microsoft Edge
* IDE/Editor: IntelliJ IDEA / Visual Studio Code / Eclipse
* Web Server: Apache Tomcat / Embedded Spring Boot Server
* Other Tools: Node.js and npm (for Angular).

## 3.3 SOFTWARE DESCRIPTIONS

## 3.3.1 JAVA

**What is Java?**

Java is a high-level, object-oriented, class-based programming language designed to have as few implementation dependencies as possible. It was originally developed by James Gosling at Sun Microsystems and released in 1995. Java is widely known for its "Write Once, Run Anywhere" (WORA) capability, meaning that compiled Java code can run on any platform that supports Java without the need for recompilation.

Java is an interpreted and compiled language. It first compiles source code into bytecode, which is then interpreted and executed by the Java Virtual Machine (JVM), making it platform-independent. Java follows a strongly-typed, statically-typed structure, meaning variables must be declared before use, and type checking is performed at compile-time.

Java is used for developing a wide variety of applications, including web-based applications, desktop software, enterprise-level solutions, Android mobile applications, and embedded systems. It is one of the most popular programming languages in the world due to its robustness, security features, and scalability.

Java offers several key features such as automatic memory management (Garbage Collection), multithreading support, platform independence, high performance, and strong community support. Its syntax is simple, easy to learn, and similar to C++, which makes it beginner-friendly yet powerful enough for complex application development. Java's rich set of APIs and libraries enables developers to build modular, reusable, and efficient applications.

Being an open-source language, Java has a vast ecosystem maintained by contributors worldwide, along with continuous updates and enhancements by Oracle Corporation.

**Java Use Cases**

* Developing cross-platform desktop applications
* Building enterprise-level web applications
* Creating Android mobile applications
* Developing server-side applications
* Connecting and interacting with databases
* Implementing distributed systems and cloud-based solutions
* Building embedded systems and IoT devices
* Developing games and interactive applications
* Creating large-scale enterprise software solutions
* Building APIs and backend services

Professionally, Java is widely used for enterprise application development, Android app development, cloud computing, web-based applications, and financial services software. It is preferred by developers for building secure, scalable, and high-performance applications across various domains such as banking, e-commerce, healthcare, and education.

**Features and Benefits of Java**

* Platform Independent: Java code can run on any operating system that has a Java Virtual Machine (JVM), making it highly portable across Windows, Mac, Linux, and other platforms.
* Simple and Familiar Syntax: Java has a clean and easy-to-understand syntax, influenced by C++, making it accessible for beginners and experienced programmers alike.
* Object-Oriented Programming: Supports principles like inheritance, encapsulation, polymorphism, and abstraction, enabling modular, reusable, and maintainable code.
* Robust and Secure: Provides strong memory management, automatic garbage collection, and built-in security features, reducing the risk of memory leaks and vulnerabilities.
* High Performance: Java bytecode is optimized and executed by the JVM, ensuring faster performance compared to interpreted languages.
* Supports Multithreading: Allows concurrent execution of multiple threads, making it suitable for high-performance and real-time applications.
* Rich API and Library Support: Offers an extensive set of libraries and frameworks for networking, data structures, database connectivity, and more.
* Scalable and Reliable: Suitable for building large-scale applications that require scalability, stability, and long-term support.

**Java Flexibility**

Java, being a statically typed language, offers flexibility in building large-scale and complex applications with a strong structure and clear rules. Its platform independence through the Java Virtual Machine (JVM) allows developers to create programs that can run on any device without modification. Java also supports object-oriented principles, enabling flexible design patterns and modular development. Additionally, Java provides robust exception handling and multithreading capabilities, allowing developers to manage complex workflows and processes efficiently while maintaining stability and scalability.

**The Less Great Parts of Java**

On the downside, Java can be verbose and complex to write and maintain. Its strict syntax rules and object-oriented structure often require developers to write longer lines of code compared to other modern languages, which can make simple tasks feel unnecessarily complicated. This verbosity can lead to slower development times, especially for small-scale or quick prototype projects.

Additionally, Java applications tend to consume more memory and require more processing power, as the Java Virtual Machine (JVM) adds an extra layer of abstraction. This can affect performance and speed, particularly in systems with limited resources.

Java's slow startup time is another limitation, especially in desktop applications or small utility programs, where faster response times are expected. While Java is highly scalable and robust for large applications, it can feel heavy and over-engineered for lightweight solutions.

Despite these drawbacks, Java continues to be popular in enterprise environments, Android development, and large-scale systems due to its reliability, strong community support, and rich ecosystem of libraries and frameworks.

**Good to Know**

The Java Development Kit (JDK) is the core component required to develop Java applications. It includes the Java Runtime Environment (JRE), Java Compiler (javac), and various development tools. The JDK is regularly updated and maintained by Oracle Corporation and the OpenJDK community, providing developers with the latest features, security updates, and performance improvements. Additionally, Maven and Gradle are popular build automation tools used in the Java ecosystem to manage project dependencies and simplify the development process.

## 3.3.2 JAVA SPRING BOOT (WEB FRAMEWORK)

**What is Spring boot?**

Spring Boot is an open-source Java-based framework used to create standalone, production-ready Spring applications with minimal configuration. It is built on top of the Spring Framework and simplifies Java application development by reducing boilerplate code, providing embedded servers, and enabling easy dependency management.

Spring Boot was developed by Pivotal Software to allow developers to quickly build, test, and deploy applications without dealing with complex configurations. It follows a "convention over configuration" approach, meaning it automatically applies sensible defaults while still allowing customization when needed.

Spring Boot is widely used for building microservices, RESTful APIs, web applications, and enterprise-level software due to its scalability, flexibility, and integration with various databases, cloud platforms, and messaging systems.

**Common Uses of Spring Boot**

* Building Microservices – Spring Boot simplifies the development of microservices architecture by providing built-in tools for service discovery, configuration, and distributed tracing.
* Developing RESTful APIs – Used to create lightweight and scalable REST APIs for web and mobile applications.
* Enterprise Application Development – Supports the development of large-scale enterprise applications with security, transaction management, and integration features.
* Database-Driven Applications – Works seamlessly with SQL (MySQL, PostgreSQL) and NoSQL (MongoDB, Cassandra) databases using Spring Data.
* Cloud-Based Applications – Easily integrates with AWS, Azure, Google Cloud, and Kubernetes for cloud deployment.
* Real-Time Applications – Supports WebSockets and messaging queues (RabbitMQ, Kafka) for real-time data processing.
* Batch Processing & Scheduling – Facilitates scheduled tasks and large-scale batch processing with Spring Batch.
* Embedded Web Applications – Comes with built-in servers (Tomcat, Jetty, Undertow), making deployment easier without external configurations.

**What is a Framework?**

The framework is the basis upon which software programs are built. It serves as a foundation for software developers, allowing them to create a variety of applications for certain platforms. A set of functions and predefined classes used to connect with the system software and handle inputs and outputs. The life of a developer while giving them the ability to use certain extensions and makes the online applications scalable and maintainable.

**What are the key features of Spring Boot?**

* Auto-Configuration – Automatically configures components based on application dependencies, reducing manual setup.
* Embedded Web Server – Comes with built-in servers like Tomcat, Jetty, and Undertow, allowing applications to run without external servers.
* Microservices Support – Provides tools for building, deploying, and managing microservices efficiently.
* Spring Boot Starters – Pre-configured dependency modules that simplify project setup (e.g., spring-boot-starter-web, spring-boot-starter-data-jpa).
* Production-Ready Monitoring – Includes Spring Actuator for health checks, metrics, and application insights.
* Spring Boot CLI – Allows developers to quickly prototype applications using Groovy scripts.
* Externalized Configuration – Supports YAML, properties files, environment variables, and command-line arguments for flexible configuration management.
* Security Integration – Works seamlessly with Spring Security to enable authentication, authorization, and encryption.
* Simplified Database Access – Uses Spring Data JPA, JDBC, and Hibernate for efficient database connectivity.
* Cloud Deployment – Easily integrates with Docker, Kubernetes, and cloud platforms like AWS, Azure, and Google Cloud.

**Frontend Development vs Backend Development**

The front end of the Deeniyath Student Attendance Management System is the user interface (UI) where students, teachers, and administrators interact with the system. It includes features such as dashboards, attendance forms, reports, and notifications. The frontend is developed using Angular, HTML, CSS, and TypeScript, providing a responsive, interactive, and user-friendly experience.

Some of the frontend frameworks and technologies used are:

* **Angular** – A powerful framework for building dynamic web applications.
* **Bootstrap** – Ensures responsiveness and an aesthetically appealing UI.
* **CSS3** – Enhances design and styling for better user experience.

The backend of the system is responsible for data processing, authentication, and communication between the database and the frontend. It ensures secure and efficient handling of attendance records, user authentication, and reporting functionalities. The backend is developed using Java (Spring Boot) and interacts with a MySQL database to store and retrieve attendance-related data.

Some of the backend frameworks and technologies used are:

* **Spring Boot** – A Java-based framework that simplifies backend development.
* **Hibernate** – Helps in database management and ORM (Object-Relational Mapping).
* **Spring Security** – Ensures secure authentication and authorization.

## 3.3.3 HTML (HYPER TEXT MARKUP LANGUAGE)

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript).

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML uses "markup" to annotate text, images, and other content for display in a Web browser. HTML markup includes special "elements" such as <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, <div>, <span>, <img>, <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <search>, <output>, <progress>, <video>, <ul>, <ol>, <li> and many others.

An HTML element is set off from other text in a document by "tags", which consist of the element name surrounded by < and >. The name of an element inside a tag is case-insensitive. That is, it can be written in uppercase, lowercase, or a mixture. For example, the <title> tag can be written as <Title>, <TITLE>, or in any other way. However, the convention and recommended practice is to write tags in lowercase.

**Understanding HTML**

HyperText Markup Language is the computer language that facilitates website creation. The language, which has code words and syntax just like any other language, is relatively easy to comprehend and, as time goes on, is increasingly powerful in what it allows someone to create. HTML continues to evolve to meet the demands and requirements of the internet under the guise of the World Wide Web Consortium, the organization that designs and maintains the language—for instance, with the transition to Web 2.0.

HyperText is the method by which internet users navigate the web. By clicking on special text called hyperlinks, users are brought to new pages. The use of hyper means it is not linear, so users can go anywhere on the internet simply by clicking on the available links. Markup is what HTML tags do to the text inside of them; they mark it as a specific type of text. For example, markup text could come in the form of boldface or italicized type to draw specific attention to a word or phrase.

**How Can HTML Be Used?**

Uses for HTML include:

* Webpage development
* Internet navigation
* Browser storage function
* Web document creation
* Game development

## 3.3.4 MySQL DATA BASE

MySQL is the world's most popular open-source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo! and many more.

**MySQL Explained**

MySQL is an open-source RDBMS that uses SQL to create and manage databases. As a relational database, MySQL stores data in tables of rows and columns organized into schemas. A schema defines how data is organized and stored and describes the relationship among various tables. With this format, developers can easily store, retrieve, and analyse many data types, including simple text, numbers, dates, times, and, more recently, JSON and vectors.

Because MySQL is open source, it includes numerous features developed in close cooperation with a community of users over almost 30 years. Two capabilities that developers rely on are MySQL’s support for ACID transactions and MySQL’s ability to scale. ACID stands for “atomicity, consistency, isolation, and durability,” the four properties that ensure database transactions are processed dependably and accurately. With ACID transactions, MySQL can guarantee that all data modifications are made in a consistent and reliable way, even in the event of a system failure. MySQL can be scaled out to support very large databases, and it can handle a high volume of concurrent connections.

MySQL’s performance, ease of use, and low cost combined with its ability to reliably scale as a business grows have made it the world’s most popular open source database.

MySQL: Distinguishing It from SQL

The acronym “SQL” stands for Structured Query Language, a type of programming language that’s used for manipulating data in a database. MySQL uses the SQL language to manage and query data in databases and, hence, uses the acronym as part of its name. If you’ve got data stored in a MySQL RDBMS, then you can write simple SQL prompts to add, search, analyse, and retrieve data.

**Understanding MySQL: Features and Popularity**

MySQL’s ability to efficiently store and analyse vast quantities of data means it can help with tasks as varied as informing complex business decisions and finding a local restaurant for a date night. Here’s a look at the top functionality that makes MySQL so pervasive in today’s tech landscape.

A Comprehensive Relational Database System

MySQL is known as a flexible, easy-to-use database management system. You’ll find it used by lone developers grabbing an open-source database for a small project all the way up to the world’s most visited websites and applications. MySQL has been evolving to keep up with demand for nearly 30 years and offers ACID transactions that ensure data modifications are made in a consistent way-even when supporting a high volume of concurrent connections.

**The Open-Source Advantage of MySQL**

MySQL is open source, which means anyone can download MySQL software from the internet and use it without cost. Organizations can also change its source code to suit their needs. MySQL software uses the GNU General Public License (GPL), which is a common set of rules for defining what may or may not be done with or to the software in various situations. If an organization feels uncomfortable with the GNU GPL or wishes to embed MySQL code into a commercial application, it can buy a commercially licensed version. See the MySQL Legal Policies page for more information about licensing.

**Why Developers Prefer MySQL’s Performance and Flexibility**

MySQL is known for being easy to set up and use, yet reliable and scalable enough for organizations with very large data sets and vast numbers of users. MySQL’s native replication architecture enables organizations such as Facebook to scale applications to support billions of users.

Other key factors in MySQL’s popularity include abundant learning resources and the software’s vibrant global community.

**How Does MySQL Work?**

Each software application needs a repository to store data so the information can be accessed, updated, and analysed in the future. A relational database such as MySQL stores data in separate tables rather than putting all the data in one big storeroom. The database structure is organized into files optimized so data can be accessed quickly. This logical data model, with objects such as data tables, views, rows, and columns, offers developers and database administrators a flexible programming environment. They can set up rules governing the relationships between different data fields, such as one to one, one to many, unique, required, or optional, and add “pointers” among different tables. The system enforces these rules so that, with a well-designed database, an application never sees data that’s inconsistent, duplicated, orphaned, or out of date.

MySQL Database is a client/server system that consists of a multithreaded SQL server that supports different back ends, several client programs and libraries, a choice of administrative tools, and a wide variety of application programming interfaces (APIs). MySQL is available as an embedded multithreaded library that developers can link into applications to get a smaller, faster, easier-to-manage standalone product.

SQL is the most common standardized programming language used to access databases. Depending on the programming environment, a developer might enter SQL directly—for example, to generate reports. It’s also possible to embed SQL statements into code written in another programming language or use a language-specific API that hides the SQL syntax.

**Why Is MySQL Important?**

MySQL is important because of its ubiquitousness and the fundamental role of databases as the amount of data both grows exponentially and fuels AI. MySQL underpins a vast array of websites and applications and helps businesses worldwide organize, analyze, and protect their data.

Other factors also help maintain MySQL’s enduring popularity.

Open source with strong community support

During MySQL’s nearly three decades as the leading open source RDBMS, a vibrant global community has grown up around it. That’s important because the community provides a wealth of expertise and resources, such as tutorials, tips in forums, and more. By testing the software in multiple use case scenarios, the community also has helped discover and fix bugs, making MySQL highly reliable.

The open-source community’s knowledge sharing, problem-solving, and continuous innovation keep MySQL users at the forefront of technological advancements.

**High performance and reliability**

MySQL is at home in many different environments, including individual developer projects and mission-critical applications that demand unwavering stability. The open-source RDBMS can handle high volumes of data and concurrent connections and provide uninterrupted operations under demanding circumstances. This is partly due to MySQL’s robust replication and failover mechanisms, which help minimize the risk of data loss.

**Ease of use and compatibility**

MySQL is often praised for being easy to use and for offering broad compatibility with technology platforms and programming languages, including Java, Python, PHP, and JavaScript. MySQL also supports replication from one release to the next, so an application running MySQL 5.7 can easily replicate to MySQL 8.0.

In addition, MySQL offers flexibility in developing both traditional SQL and NoSQL schema-free database applications. This means developers can mix and match relational data and JSON documents in the same database and application.

**Cost-effectiveness and scalability**

Because MySQL is open source, it’s freely available to use at no cost, beyond the on-premises hardware it runs on and training on how to use it. For the latter, a global community of MySQL users provide cost-effective access to learning resources and troubleshooting expertise. Oracle also offers a wide range of training courses.

When it’s time to scale out, MySQL supports multithreading to handle large amounts of data efficiently. Automated failover features help reduce the potential costs of unplanned downtime.

**Benefits of MySQL**

MySQL is fast, reliable, scalable, and easy to use. It was originally developed to handle large databases quickly and has been used in highly demanding production environments for many years. MySQL offers a rich and useful set of functions, and it’s under constant development by Oracle, so it keeps up with new technological and business demands. MySQL’s connectivity, speed, and security make it highly suited for accessing databases on the internet.

**MySQL’s key benefits include the following:**

* Ease of use. Developers can install MySQL in minutes, and the database is easy to manage.
* Reliability. MySQL is one of the most mature and widely used databases. It has been tested in a wide variety of scenarios for nearly 30 years, including by many of the world’s largest companies. Organizations depend on MySQL to run business-critical applications because of its reliability.
* Scalability. MySQL scales to meet the demands of the most accessed applications. MySQL’s native replication architecture enables organizations, including Facebook, Netflix, and Uber, to scale applications to support tens of millions of users or more.
* Performance. MySQL is a proven high performance, zero-administration database system and comes in a range of editions to meet nearly any demand. Cloud-based HeatWave MySQL provides unmatched performance and price-performance, according to industry benchmarks including TPC-H, TPC-DS, and CH-benCHmark.
* High availability. MySQL delivers a complete set of native, fully integrated replication technologies for high availability and disaster recovery. For business-critical applications and service level agreement commitments, customers can achieve recovery point objective zero (zero data loss) and recovery time objective zero seconds (automatic failover).
* Security. Data security entails both data protection and compliance with industry and government regulations, including the European Union General Data Protection Regulation, the Payment Card Industry Data Security Standard, the Health Insurance Portability and Accountability Act, and the Defense Information Systems Agency’s Security Technical Implementation Guides. MySQL Enterprise Edition provides advanced security features, including authentication/authorization, transparent data encryption, auditing, data masking, and a database firewall.
* Flexibility. The MySQL Document Store gives users maximum flexibility in developing traditional SQL and NoSQL schema-free database applications. Developers can mix and match relational data and JSON documents in the same database and application.

# CHAPTER 4

**PROJECT DESCRIPTION**

The Deeniyath Student Attendance Management System is a web-based application designed to assist educational institutions in tracking and managing student attendance efficiently. Traditional manual attendance methods, such as paper-based registers, are often time-consuming, error-prone, and difficult to maintain, leading to inefficiencies in student monitoring. This project aims to provide an automated, digital solution that allows teachers and administrators to record student attendance in real-time, ensuring accurate and secure data management.

The system is developed using Spring Boot (Java) as the backend framework, enabling fast, scalable, and secure web application development. The frontend is built using Angular, HTML, and CSS, ensuring a responsive and user-friendly interface. For data storage, the system utilizes MySQL, allowing staff members to efficiently store, retrieve, and manage attendance records. The system operates through a secure login portal, where authorized users can mark attendance, view reports, and manage student records.

One of the key advantages of this system is its ability to provide real-time attendance tracking. This helps institutions monitor student participation, identify absentees, and analyze attendance trends. The stored data can be accessed for future analysis, making it easier for administrators to generate reports based on class, date, or individual student records. Additionally, automated data storage eliminates paperwork, reducing administrative workload and the risk of lost or damaged records.

Security is a major concern in educational institutions, and this system enhances it by ensuring only authorized users can access attendance records, preventing unauthorized modifications. The system can also be extended in the future by integrating biometric authentication, RFID-based attendance tracking, or facial recognition for advanced monitoring capabilities. With database backups and secure authentication mechanisms, institutions can ensure that attendance records remain safe and easily retrievable, even in case of system failures.

The Deeniyath Student Attendance Management System is a cost-effective, scalable, and efficient solution that addresses the shortcomings of traditional manual attendance tracking methods. By automating the attendance process, institutions can improve organization, enhance monitoring, and increase accuracy in student attendance tracking. This project serves as an essential tool for modernizing attendance management in educational institutions, ensuring efficiency, security, and accountability.

## 4.1 PORTAL FUNCTIONALITY OVERVIEW

The Deeniyath Student Attendance Management System consists of three distinct portals: Admin Portal, Coordinator Portal, and Staff Portal. Each portal has specific functionalities that cater to the needs of different users, ensuring efficient attendance tracking, student management, and report generation.

## 4.1.1 ADMIN PORTAL

The admin has the highest level of access and control over the system. Admin functionalities include:

* Full Access: Admin can perform all the functions available to the coordinator.
* User Management: Admin can create, edit, or delete coordinator accounts and manage their login credentials.
* Secure Authentication: Admins can log in securely and access all system features.

## 4.1.2 CO-ORDINATOR PORTAL

The coordinator manages attendance and student distribution across different rooms and classes. Coordinator functionalities include:

**Student & Room Management**

* Creating rooms for each academic year to organize students efficiently.
* Assigning students to specific rooms based on their year and program.
* Assigning teachers to rooms to manage attendance and student records.
* Adding, editing, or deleting rooms, students, and assigned teachers.
* Viewing student lists and class details across multiple years.
* Managing student information updates as needed.

**Attendance Tracking & Reports**

* Marking attendance for students in different rooms.
* Downloading attendance reports for classes or individual students.
* Viewing and downloading overall student attendance reports based on:
  + Individual students
  + Specific classes
  + Year-wise attendance summary
  + Present and absent student lists

**Customized View Based on Gender**

* Male coordinators can access and manage only male student records.
* Female coordinators can access and manage only female student records.

**Access Control**

* Coordinators have secure login functionality with role-based access.
* Coordinators can create login credentials for staff members.

## 4.1.3 STAFF PORTAL

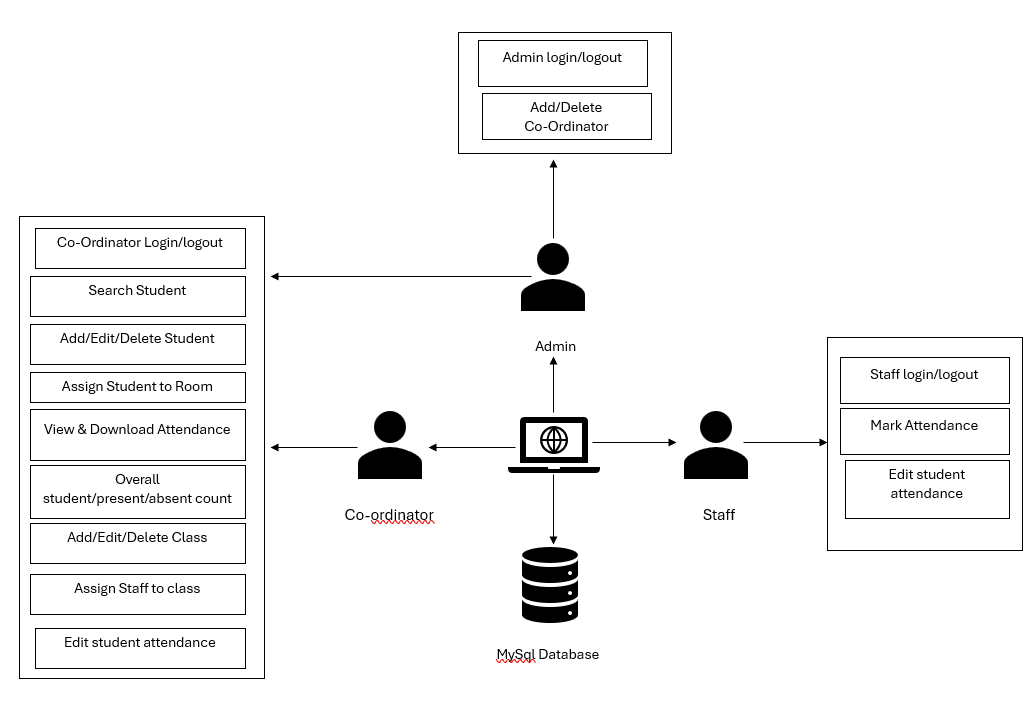
The Staff portal is designed for teachers to mark attendance and view reports for their assigned classes. Staff functionalities include:

* Viewing student lists based on assigned classes.
* Secure login functionality for authenticated access.
* Marking attendance and downloading attendance reports.
* Viewing and downloading individual student attendance reports.
* Accessing total numbers of present and absent students per class.

# CHAPTER 5

**SYSTEM DESIGN**

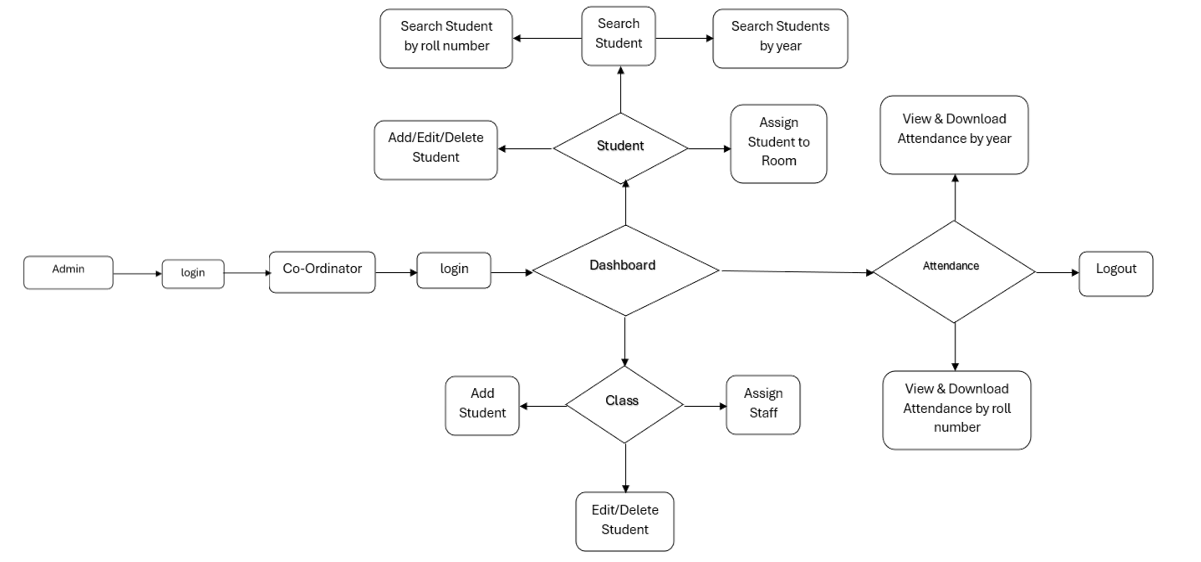
## 5.1 ARCHITECTURE DIAGRAM



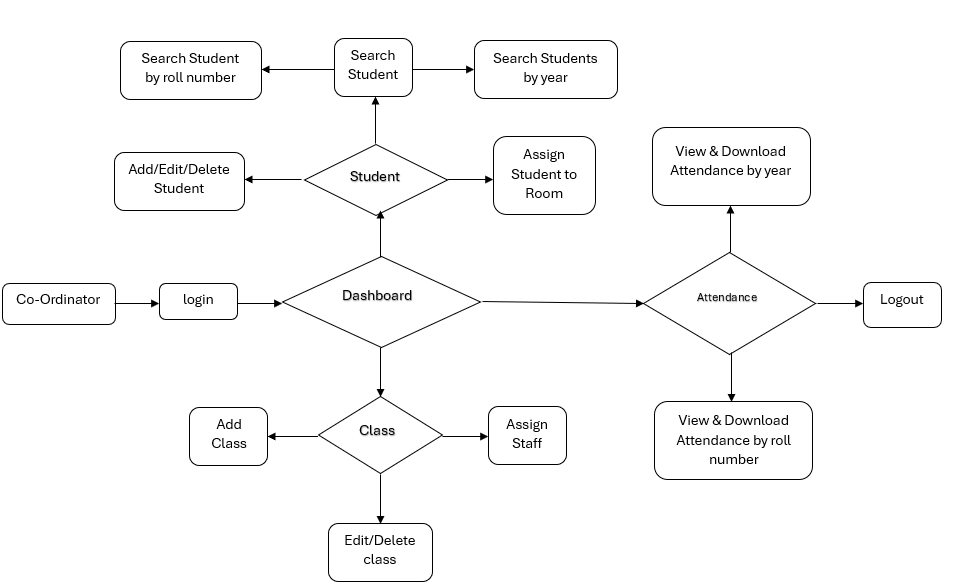
## 5.2 FLOW CHART

A flowchart is a visual representation of a process, system, or algorithm using symbols and arrows to show the flow of steps. It helps in understanding, analyzing, and improving workflows.

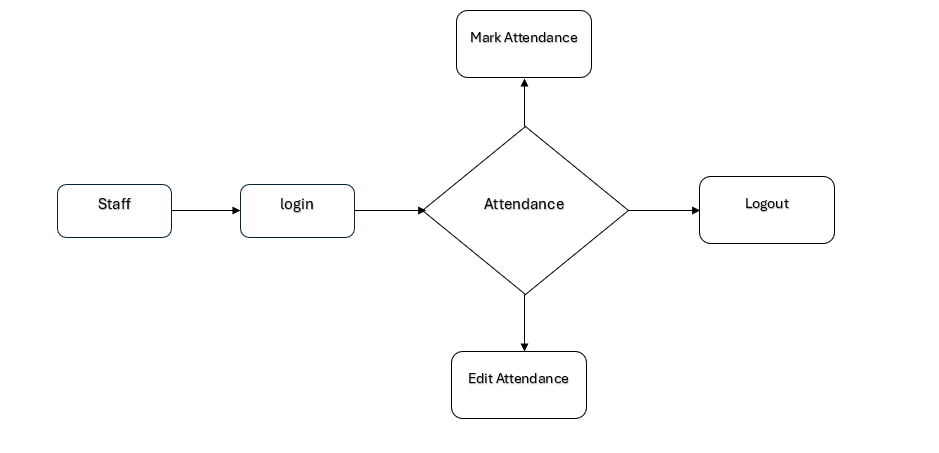
## 5.2.1 ADMIN FLOW CHART



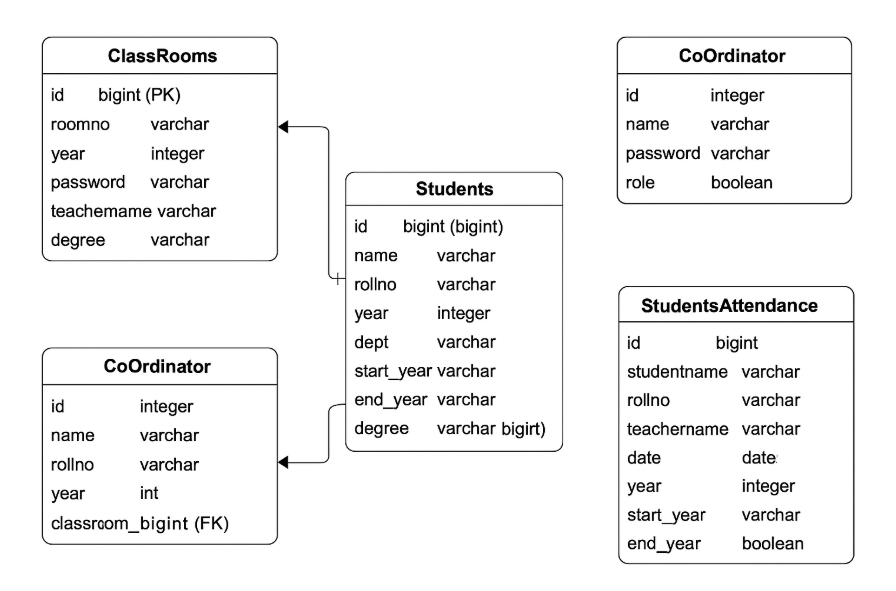
## 5.2.2 CO-ORDINATOR FLOW CHART



## 5.2.3 STAFF FLOW CHART



## 5.3 DATABASE ER DIAGRAM



# CHAPTER 6

**IMPLEMENTATION**

## 6.1 IMPLEMENTATION

The implementation of the Student Attendance Management System involves several key phases, including system setup, backend development, frontend design, database integration, security implementation, testing, and deployment. This project is developed as a web based application using Spring boot, ensuring an efficient and scalable solution for student attendance management system. The implementation process ensures a smooth, secure, and userfriendly experience for staff, co-ordinator and admin.

## 6.1.1 System Setup and Technology Stack

The first step in implementation is setting up the development environment. The system is built using:

* Backend: Java spring boot (for handling RestApi’s, requests, and database interactions).
* Frontend: HTML, CSS, Angular (for user interface and form handling).
* Database: MySQL (for storing student details and attendance records securely).
* Development Tools: Visual Studio Code (VS Code) or Eclipse for coding, MySQL Workbench for database management.

## 6.1.2 Backend Development (Spring boot)

The backend of the system is developed using Spring Boot, a robust and scalable Java framework. Spring Boot handles:

• User authentication: Secure login system for admins, coordinators, and staff.  
• Data handling: Managing student, staff, and class-related data efficiently.  
• Database interaction: Storing and retrieving student, staff, and attendance records using MySQL.  
• API endpoints: Providing RESTful services for real-time data updates, room assignments, and attendance tracking.

The Spring Boot server listens for HTTP requests from the Angular frontend and processes student, staff, and class-related operations.

## 6.1.3 Frontend Development (User Interface Design)

The frontend is designed to be user-friendly, interactive, and responsive. It includes:

* Login Page: Secure access for admin, coordinators, and staff.
* Dashboard: Displays student records, attendance reports, and room allocations.
* Student & Staff Management: Features for adding, editing, and deleting records.
* Attendance System: Enables coordinators and staff to mark, view, and download attendance reports.
* Role-based Access: Coordinators can only view students based on gender restrictions.
* Search & Filter Options: Helps users find student records and attendance reports efficiently.

The frontend is built using Angular, along with HTML, CSS, and TypeScript, ensuring a responsive and seamless user experience across desktops and mobile devices.

## 6.1.4 Database Integration

A relational database (MySQL or SQLite) is implemented to store student entry details. The database includes tables such as:

* Student Table: Stores student’s details.
* Class Room Table: Stores class rooms details.
* Attendance Table: Stores students attendance record.
* Co-Ordinator Table: Stores login credentials for coordinator’s.

The application uses JPA (Java Persistence API) with Spring Data JPA to interact with the MySQL database. JPA simplifies database operations by providing an abstraction over SQL queries. The system handles:

* Insertion: Automatically saves new student, staff, and attendance records using save() method.
* Retrieval: Fetches student lists, attendance details, and room allocations using findById(), findAll(), and custom query methods.
* Updating: Modifies existing records, such as student details and attendance updates, using save() with an existing entity.
* Deletion: Removes records from the database using deleteById() and delete().
* Custom Queries: Uses JPQL (Java Persistence Query Language) and native queries when complex filtering and joins are required.

Spring Data JPA ensures efficient data handling, reducing the need for writing raw SQL queries while maintaining flexibility.

# CHAPTER 7

**TESTING**

To ensure the reliability and efficiency of the Student Entry Monitoring System, various testing methods are conducted. The testing process includes the following:

1. Unit Testing

2. Integration Testing

3. Functional Testing

4. User Interface (UI) Testing

## 7.1 UNIT TESTING

Unit testing is conducted to validate the individual components of the Student Attendance Management System, ensuring each module works as expected. The testing focuses on critical functionalities such as admin login, coordinator login, staff login, student management, attendance marking, and report generation.

Each function, including database operations, authentication mechanisms, and attendance processing, is tested separately using JUnit and Mockito in the Spring Boot framework. For instance:

* Authentication Testing: Valid and invalid credentials are tested to verify secure login for admin, coordinator, and staff.
* Database Testing: Ensures that student records, attendance data, and room allocations are stored, retrieved, updated, and deleted correctly.
* Attendance Module Testing: Confirms that marking attendance reflects correctly in reports and allows accurate data retrieval.
* Controller & Service Layer Testing: API endpoints are tested for correct data processing, ensuring proper integration between frontend and backend.

By implementing unit testing, the system ensures stability, reliability, and error-free functionality, making it robust for real-world use.

## 7.2 INTEGRATION TESTING

Integration testing is performed to verify that different modules of the Student Attendance Management System work together seamlessly. This ensures smooth data flow and interaction between frontend, backend, and database components.

The testing is conducted using Spring Boot Test, JUnit, and Mockito to validate the integration of the following:

* Authentication Flow: Ensures that login credentials from the frontend are correctly processed by the backend and validated against the database.
* Coordinator & Staff Management: Checks that coordinators can assign students and staff to rooms, and staff can access their assigned class lists.
* Attendance Management: Verifies that attendance records marked by staff or coordinators are stored correctly in the database and displayed accurately in reports.
* REST API & Database Interaction: Confirms that API requests from the Angular frontend correctly communicate with the Spring Boot backend and interact with the JPA repository for data retrieval and updates.
* Report Generation: Tests the accuracy of student and class attendance reports by ensuring the correct filtering and retrieval of data.

By conducting integration testing, the system ensures consistent functionality, error-free communication between modules, and smooth end-to-end performance for users.

## 7.3 FUNCTIONAL TESTING

Functional testing is performed to verify that the Student Attendance Management System meets all specified requirements and works as expected. This type of testing ensures that each feature of the system functions correctly according to the defined use cases.

The testing is conducted using JUnit, Selenium, and Postman to validate the following:

* User Authentication: Ensures that admin, coordinator, and staff can log in with valid credentials and receive appropriate access permissions.
* Room & Student Management: Confirms that coordinators can create, assign, edit, and delete rooms, as well as manage student records.
* Attendance Marking & Retrieval: Verifies that staff and coordinators can accurately mark attendance, store records, and retrieve attendance reports.
* Role-Based Access Control: Checks that coordinators can access student lists based on gender-specific restrictions, and staff can view only their assigned classrooms.
* Report Generation & Download: Tests the system’s ability to generate and download individual student reports, class-wise attendance reports, and yearly summaries.
* Data Integrity & Validation: Ensures that invalid or incomplete data (such as empty student names or duplicate records) is prevented from being stored in the database.
* Responsive Frontend Performance: Tests that the Angular-based frontend works smoothly on different devices (desktop, tablet, and mobile) without errors.
* Database Transactions: Confirms that all CRUD operations (Create, Read, Update, Delete) in the Spring Boot backend and JPA repository are executed correctly without data inconsistencies.

By performing functional testing, the system ensures that all user interactions, data processing, and system behaviors align with project requirements, providing a reliable and efficient attendance management solution.

## 7.4 USER INTERFACE TESTING

User Interface (UI) testing is conducted to ensure that the Student Attendance Management System provides a smooth, responsive, and user-friendly experience. This testing verifies the visual elements, layout consistency, and interactive components of the system across different devices and screen sizes.

The testing is performed using Selenium, Browser Developer Tools, and Manual Testing to validate the following aspects:

* Login & Navigation: Ensures that the login page, dashboard, and navigation menus work correctly for admin, coordinators, and staff, with a smooth transition between pages.
* Responsive Design: Checks that the UI adjusts properly across desktops, tablets, and mobile devices, ensuring proper alignment of buttons, text fields, and tables.
* Form Validation: Confirms that all input fields (such as student entry forms, attendance marking, and login) provide appropriate error messages for invalid or missing data.
* Dashboard Layout: Verifies that the student list, attendance reports, and charts are displayed correctly and are easy to read.
* Button & Link Functionality: Ensures that all buttons (such as "Mark Attendance," "Download Report," and "Add Student") respond correctly when clicked.
* Theme & Styling Consistency: Checks that fonts, colors, and design elements are consistent across all pages according to the UI design guidelines.
* Load Time & Performance: Evaluates page load speed and smooth functionality to prevent lag or slow responses during user interactions.

By performing user interface testing, the system ensures an intuitive and visually appealing experience, reducing usability issues and improving user satisfaction.

# CHAPTER 8

**CONCLUSION**

## 8.1 CONCLUSION

The Student Attendance Management System is an efficient and automated solution designed to streamline attendance tracking in educational institutions. By replacing traditional manual methods with a digital platform, the system enhances accuracy, reduces administrative workload, and improves security.

Developed using Spring Boot for the backend and Angular, HTML, CSS, and TypeScript for the frontend, the system ensures seamless data processing, real-time attendance monitoring, and secure user authentication. The JPA repository enables smooth database interactions, making record management faster and more reliable.

With distinct portals for Admin, Coordinators, and Staff, the system provides role-based access control, ensuring that only authorized users can manage student attendance and records. The system also offers report generation, search and filter functionalities, and downloadable attendance reports, making it a valuable tool for educational institutions.

Through unit, integration, functional, and UI testing, the application has been thoroughly evaluated to ensure a stable, user-friendly, and error-free experience. By implementing this system, institutions can reduce paperwork, improve efficiency, and enhance overall student attendance management.

## 8.2 Future Enhancements

To further improve the system and meet evolving needs, several future enhancements can be considered:

* RFID or Barcode Integration: Implementing RFID or barcode scanning for automatic student identification and attendance marking.
* Facial Recognition Technology: Enhancing security by integrating AI-based facial recognition for attendance verification.
* Mobile Application Support: Developing a mobile-friendly version or dedicated app for easy access by staff and administrators.
* Automated Notifications: Sending automated SMS or email notifications to students, parents, or administrators for attendance alerts.
* Cloud-Based Storage: Moving the system to a cloud-based infrastructure to ensure scalability and accessibility from anywhere.
* AI-Powered Attendance Insights: Using machine learning algorithms to analyze attendance patterns and provide predictive insights.
* Multilingual Support: Adding language options to make the system accessible to a diverse group of users.

By implementing these future enhancements, the system can further increase efficiency, accuracy, and security, making it a more advanced and effective attendance management solution.

# BIBILIOGRAPHY

## A. SAMPLE CODING

Student Controller:

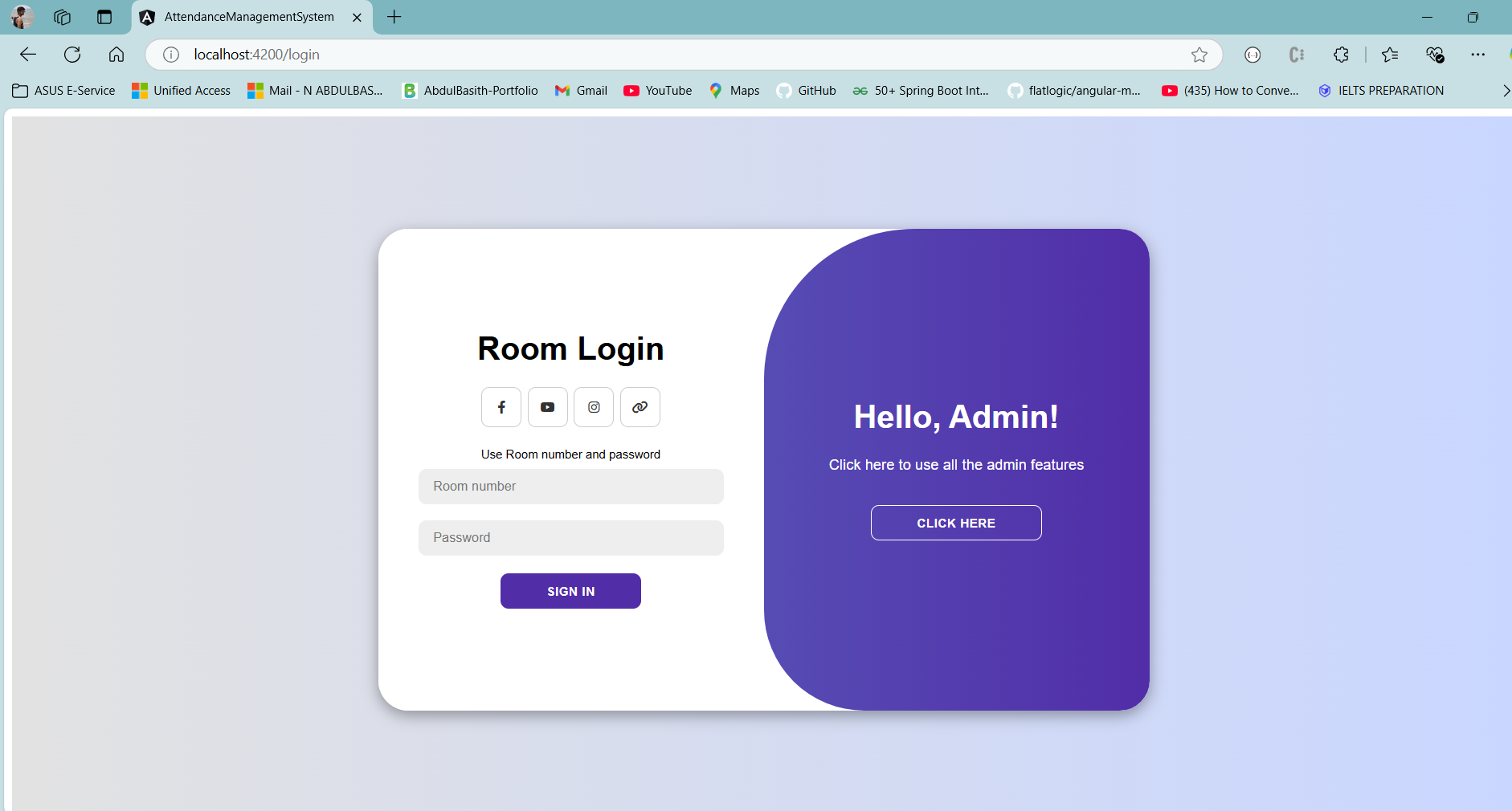
|  |
| --- |
| package com.example.attendance.Controller;  import java.util.List;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.web.bind.annotation.CrossOrigin;  import org.springframework.web.bind.annotation.DeleteMapping;  import org.springframework.web.bind.annotation.GetMapping;  import org.springframework.web.bind.annotation.PathVariable;  import org.springframework.web.bind.annotation.PostMapping;  import org.springframework.web.bind.annotation.PutMapping;  import org.springframework.web.bind.annotation.RequestBody;  import org.springframework.web.bind.annotation.RestController;  import com.example.attendance.Details.StudentDetails;  import com.example.attendance.Details.StudentDetailsEdit;  import com.example.attendance.Models.Students;  import com.example.attendance.Response.ClassAssignResponse;  import com.example.attendance.Response.ClassEditResponse;  import com.example.attendance.Response.StudentDeleteResponse;  import com.example.attendance.Response.StudentEditResponse;  import com.example.attendance.Response.StudentsResponse;  import com.example.attendance.Servic.StudentServices;  @RestController  @CrossOrigin  public class StudentController {  @Autowired  private StudentServices Service;  // Add new Student  @PostMapping("/add/new/student")  public StudentsResponse AddNewStudent(@RequestBody StudentDetails details) {  return Service.AddStudent(details);  }  // Assign students to class  @GetMapping("/assign/student\_to\_class/{rollno}/{classname}")  public ClassAssignResponse AssignClass(@PathVariable("rollno") String rollno,  @PathVariable("classname") String classname) {  return Service.AssignClass(rollno, classname);  }  // Delete the student  @DeleteMapping("/delete/student/{rollno}")  public StudentDeleteResponse DeleteStudent(@PathVariable("rollno") String roolno) {  return Service.deleteStudent(roolno);  }  // Update the student details  @PutMapping("/edit/student/{id}")  public StudentEditResponse EditStudent(@RequestBody StudentDetailsEdit details, @PathVariable("id") Long id) {  return Service.EditStudent(details, id);  }  // Find all the Students based on gender  @GetMapping("/findall/students/{gender}")  public List<Students> FindAllStudentsByGender(@PathVariable("gender") boolean gender) {  return Service.FindAllStudentsByGender(gender);  }  // Find all the Students  @GetMapping("/findall/students")  public List<Students> FindAllStudents() {  return Service.FindAllStudents();  }  // Find student based on year and batch  @GetMapping("/findbyyear/{year}/{startyear}/{endyear}/{gender}")  public List<Students> FindStudentByYearAndBatch(@PathVariable("year") int year,  @PathVariable("startyear") String startyear, @PathVariable("endyear") String endyear,  @PathVariable("gender") boolean gender) {  return Service.FindStudentByYearAndBatch(year, startyear, endyear, gender);  }  // Find student based on roomno  @GetMapping("/findbyroom/{roomno}")  public List<Students> FindStudentByRoomNO(@PathVariable("roomno") String roomno) {  return Service.FindyByRoomno(roomno);  }  // Clear student based on roomno////  @GetMapping("/clear/students/{roomno}")  public ClassEditResponse ClearStudentsByRoom(@PathVariable("roomno") String roomno) {  return Service.ClearStudentsByRoomNo(roomno);  }  } |

Attendance Controller:

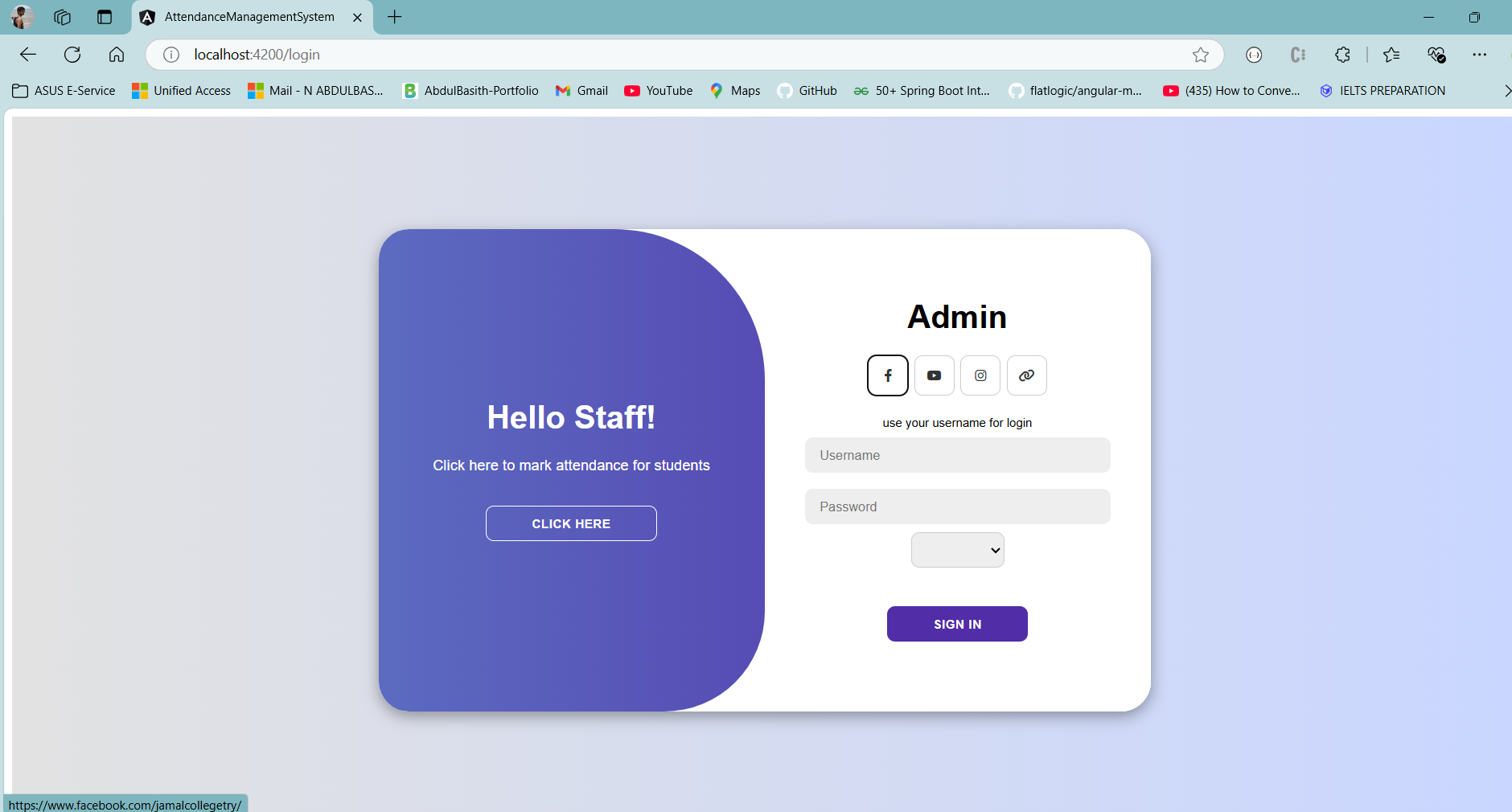
|  |
| --- |
| package com.example.attendance.Controller;  import java.io.ByteArrayOutputStream;  import java.io.IOException;  import java.time.LocalDate;  import java.util.List;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.http.HttpHeaders;  import org.springframework.http.HttpStatus;  import org.springframework.http.ResponseEntity;  import org.springframework.web.bind.annotation.CrossOrigin;  import org.springframework.web.bind.annotation.GetMapping;  import org.springframework.web.bind.annotation.PathVariable;  import org.springframework.web.bind.annotation.RestController;  import com.example.attendance.Models.StudentsAttendance;  import com.example.attendance.Response.AttedanceResponse;  import com.example.attendance.Response.DownloadResponse;  import com.example.attendance.Service.Imple.AttendanceServiceImple;  @RestController  @CrossOrigin  public class AttendanceController {  @Autowired  private AttendanceServiceImple Service;  // Mark attendance  @GetMapping("/mark/attendance/{rollno}/{status}")  public AttedanceResponse MarkAttendance(@PathVariable("rollno") String rollno, @PathVariable boolean status) {  return Service.MarkaAttendance(rollno, status);  }  //Edit the attendance  @GetMapping("edit/attendance/{date}/{rollno}/{status}")  public AttedanceResponse EditAttendance(@PathVariable("date") String strdate, @PathVariable("rollno") String rollno, @PathVariable("status") boolean status ) {  LocalDate date = LocalDate.parse(strdate);  return Service.EditAttendance(date, rollno, status);  }    //View attendance based on year for particular date  @GetMapping("/view/attendance/{year}/{startyear}/{endyear}/{date}/{gender}")  public List<StudentsAttendance> viewAttendanceByYear(@PathVariable("year") int year,  @PathVariable("startyear") String startyear, @PathVariable("endyear") String endyear,  @PathVariable("date") String strdate,@PathVariable("gender") boolean gender) {  LocalDate date = LocalDate.parse(strdate);  return Service.viewAttendance(year, startyear, endyear, date, gender);  }  //View attendance for particular students  @GetMapping("/view/attendance/{rollno}")  public List<StudentsAttendance> viewAttendanceByRollno(@PathVariable("rollno") String rollno) {  return Service.viewAttendanceByRollno(rollno);  }  // Endpoint to download the filtered attendance in Excel format based year and  // batch  @GetMapping("/download/{year}/{startyear}/{endyear}/{date}/{gender}")  public ResponseEntity<?> downloadAttendance(@PathVariable("year") int year,  @PathVariable("startyear") String startyear, @PathVariable("endyear") String endyear,  @PathVariable("date") String strdate,@PathVariable("gender") boolean gender) throws IOException {  LocalDate date = LocalDate.parse(strdate);  List<StudentsAttendance> attendanceList = Service.viewAttendance(year, startyear, endyear, date, gender);  if (attendanceList == null || attendanceList.isEmpty()) {  // Return a message if no attendance data found  return ResponseEntity.status(HttpStatus.NOT\_FOUND).body(  new DownloadResponse("Attendance not found for " + year + " Yr " + startyear + "-" + endyear + " On this date " + date, HttpStatus.NOT\_FOUND, false));  }  ByteArrayOutputStream excelFile = Service.exportAttendanceToExcel(attendanceList);  String Gender = (gender) ? "Male" : "Female";  String fileName = Service.generateExcelFilename(date, startyear, endyear, Gender);  // Set headers and return the file as a response  HttpHeaders headers = new HttpHeaders();  headers.add("Content-Disposition", "attachment; filename=" + fileName);  return ResponseEntity.ok().headers(headers).body(excelFile.toByteArray());  }  // Endpoint to download the filtered attendance in Excel format based on rollno  @GetMapping("/download/{rollno}")  public ResponseEntity<?> downloadAttendanceByRoolno(@PathVariable("rollno") String rollno) throws IOException {  List<StudentsAttendance> attendanceList = Service.viewAttendanceByRollno(rollno);  if (attendanceList == null || attendanceList.isEmpty()) {  // Return a message if no attendance data found  return ResponseEntity.status(HttpStatus.NOT\_FOUND).body(  new DownloadResponse("Attendance not found for Student: " + rollno, HttpStatus.NOT\_FOUND, false));  }  ByteArrayOutputStream excelFile = Service.exportAttendanceToExcel(attendanceList);  String fileName = Service.generateExcelFilenameByRollno(rollno);  // Set headers and return the file as a response  HttpHeaders headers = new HttpHeaders();  headers.add("Content-Disposition", "attachment; filename=" + fileName);  return ResponseEntity.ok().headers(headers).body(excelFile.toByteArray());  }  } |

## B. SCREEN SHOTS

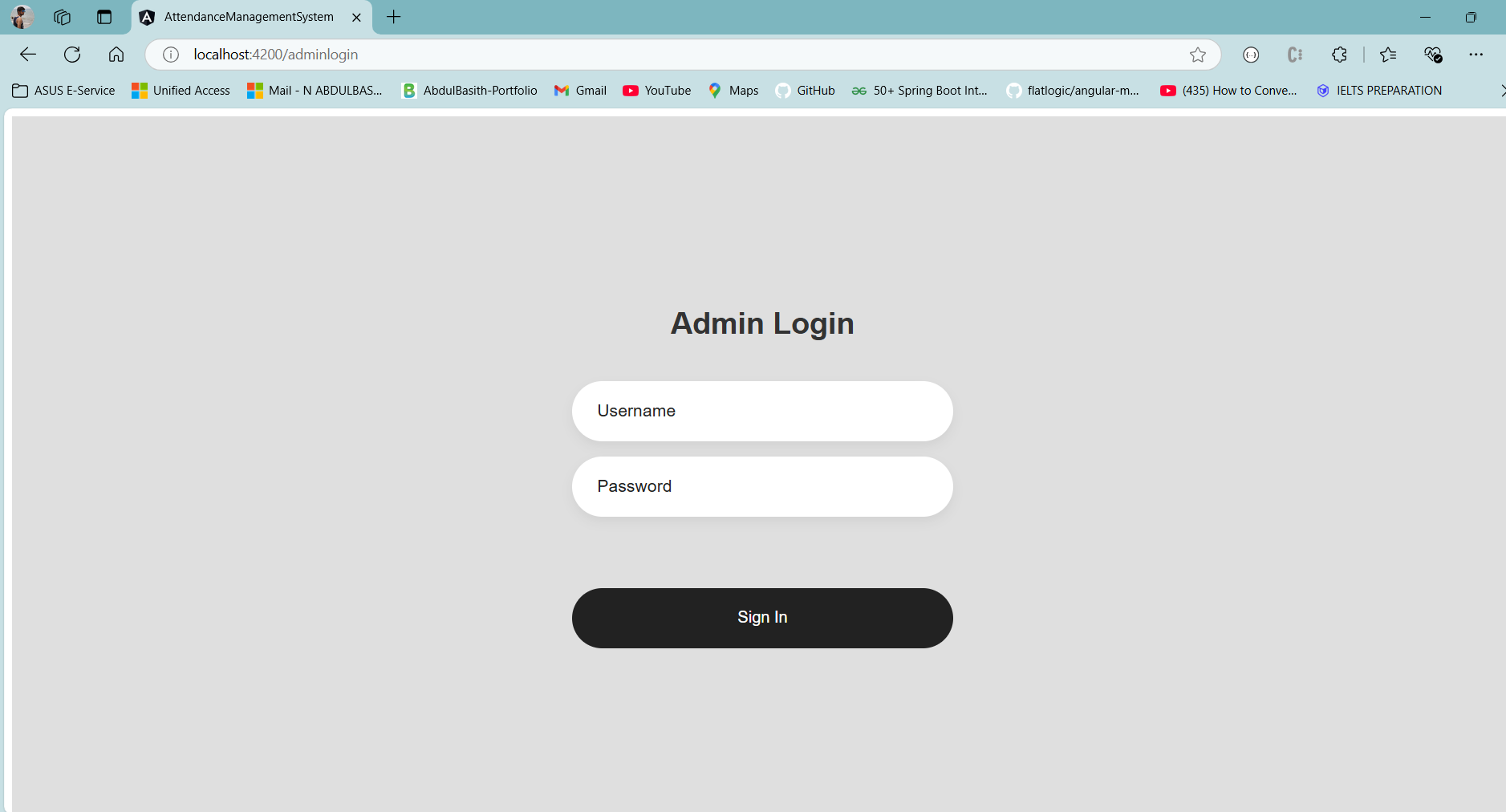
Staff Login Page:



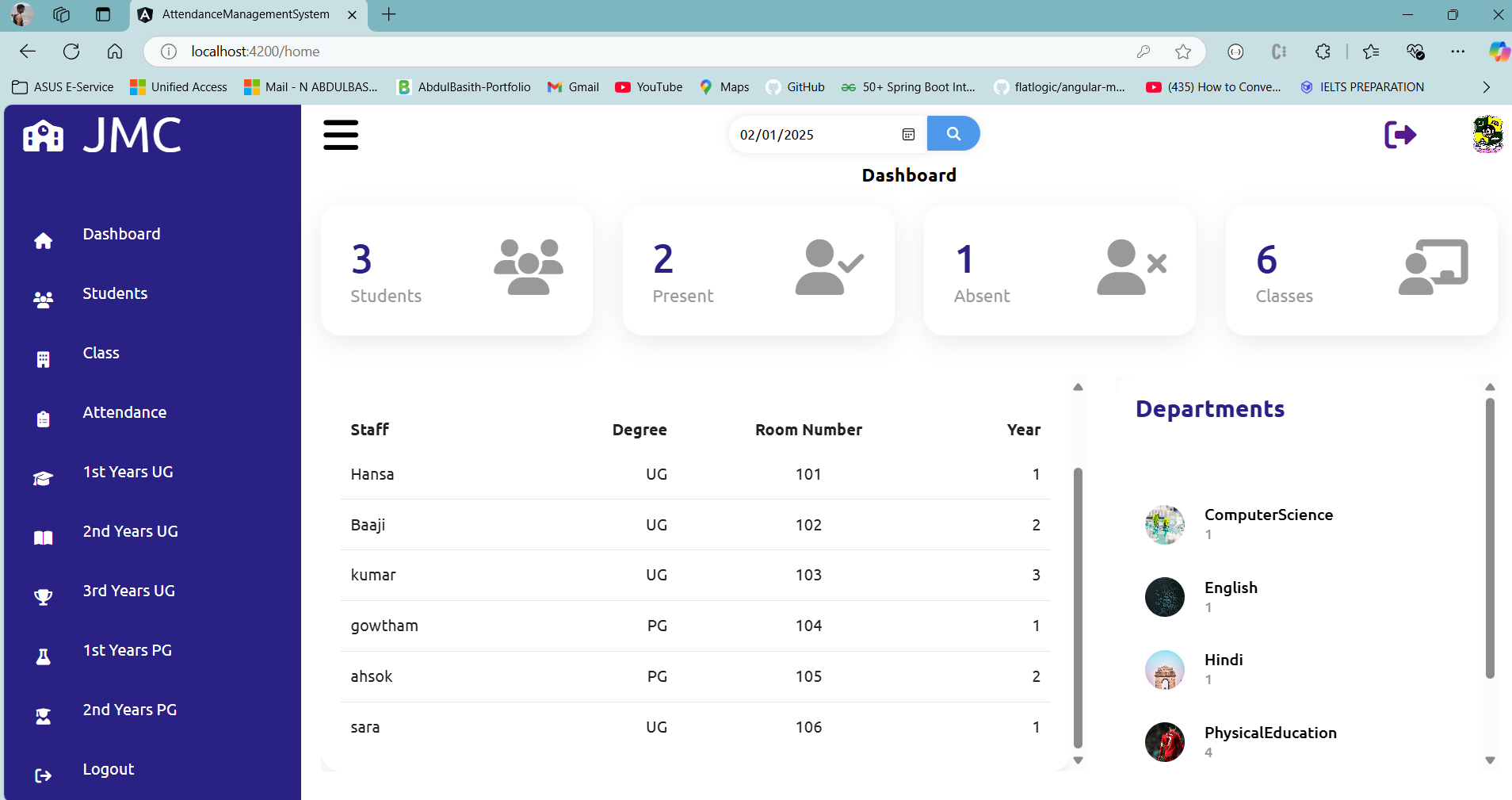
Co-Ordinator Login Page:



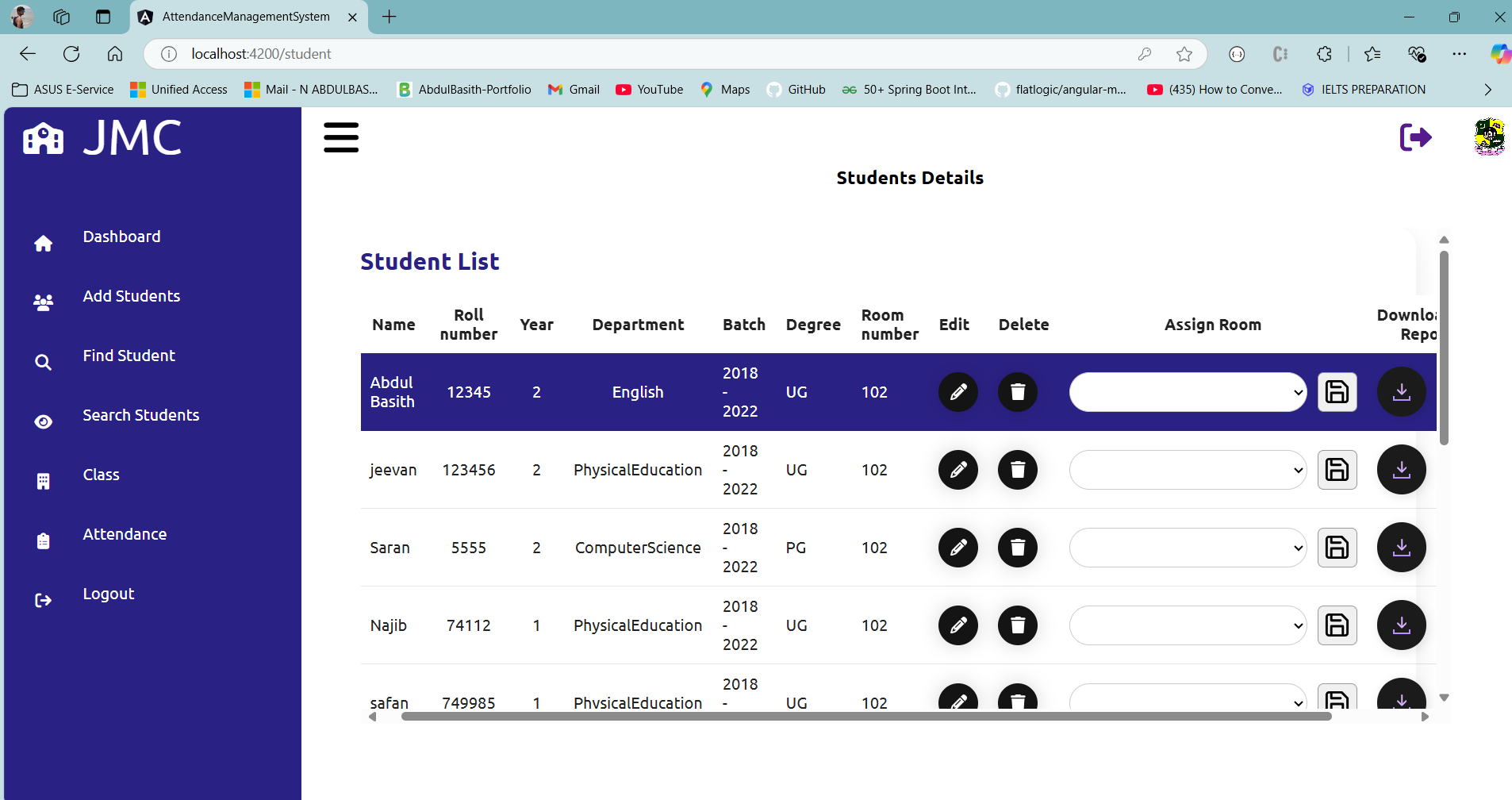
Admin Login Page:



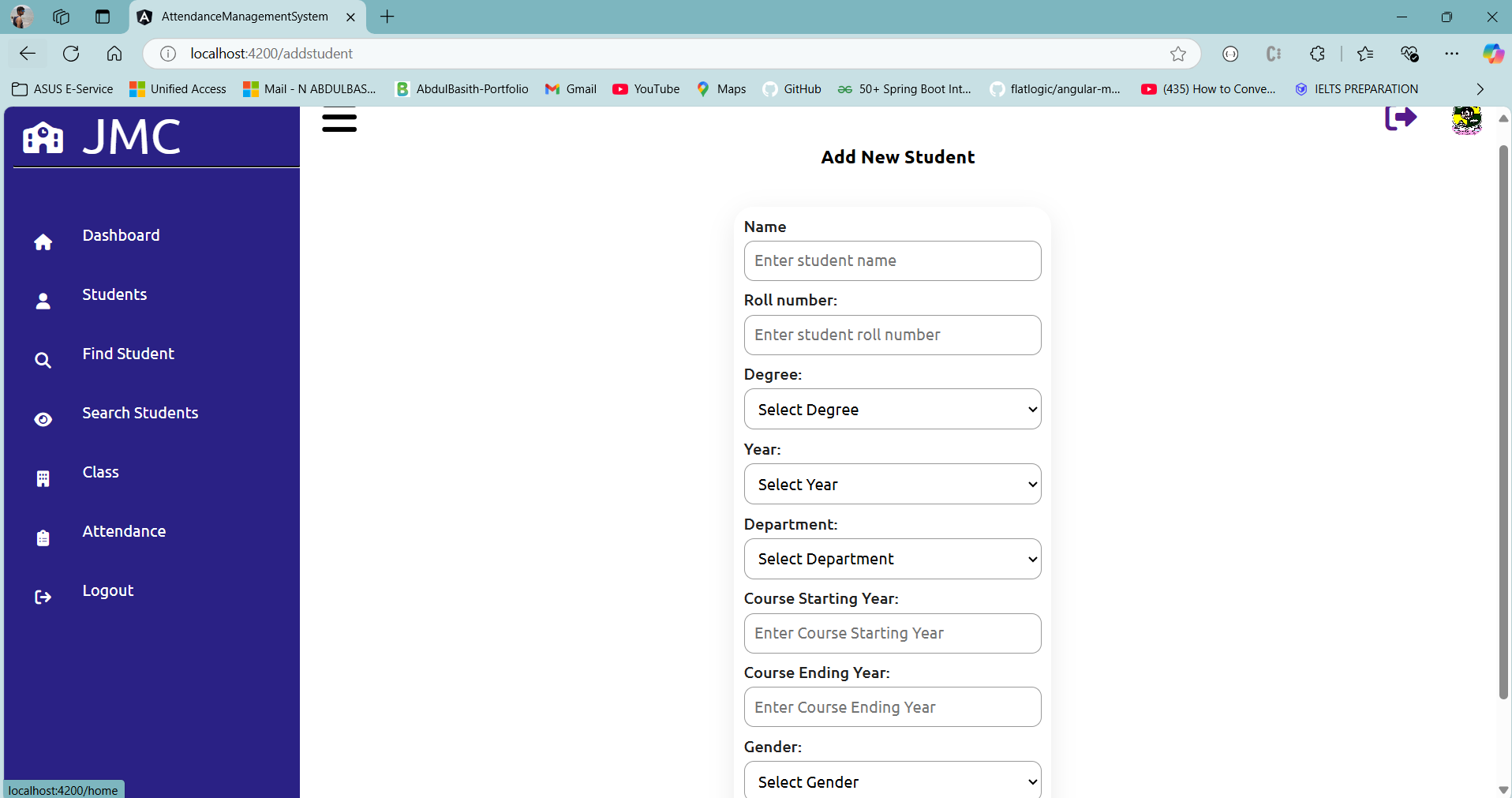
Co-Ordiantor Dashboard:



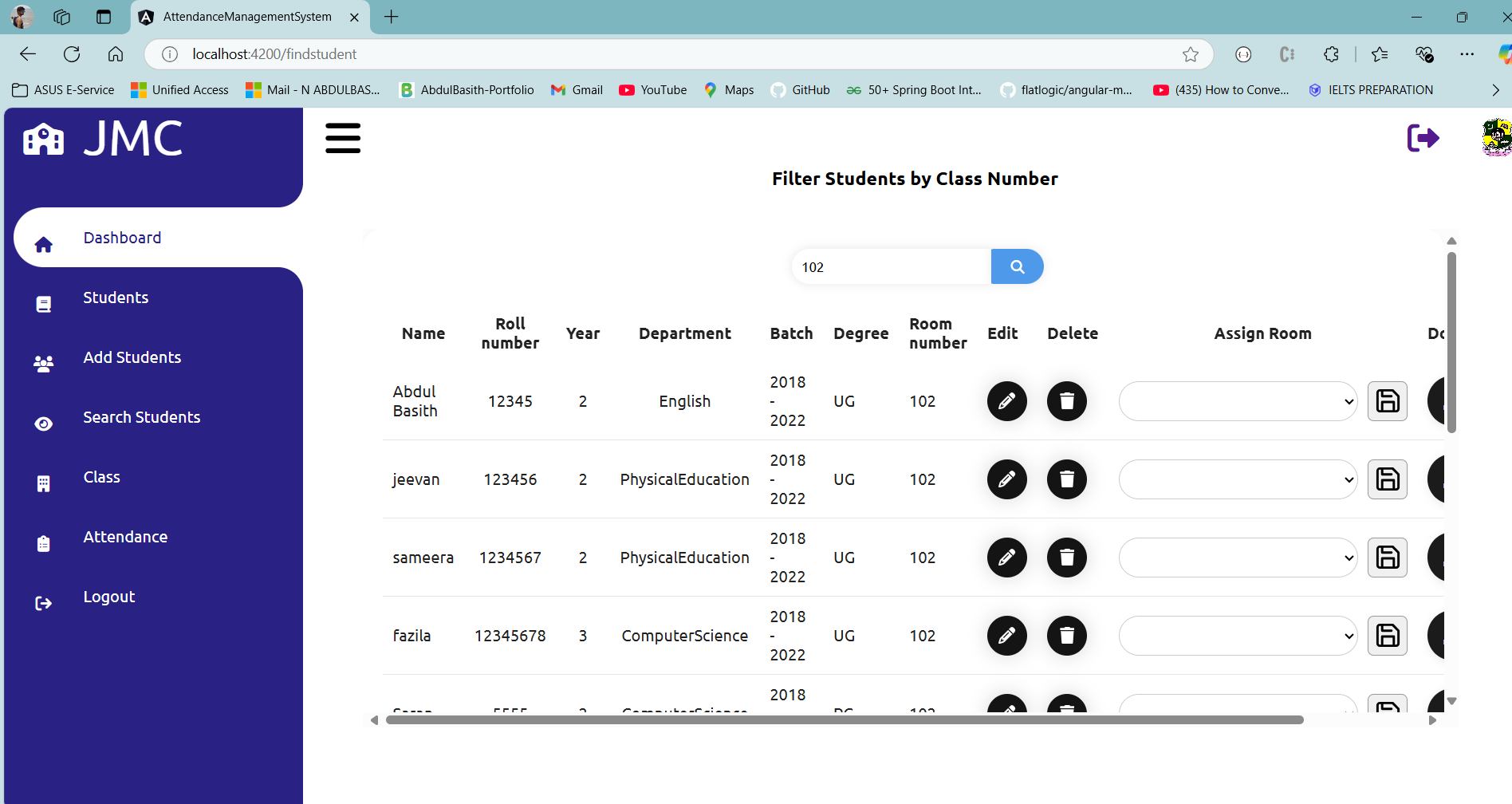
Delete/Update/Assign room Student Page:



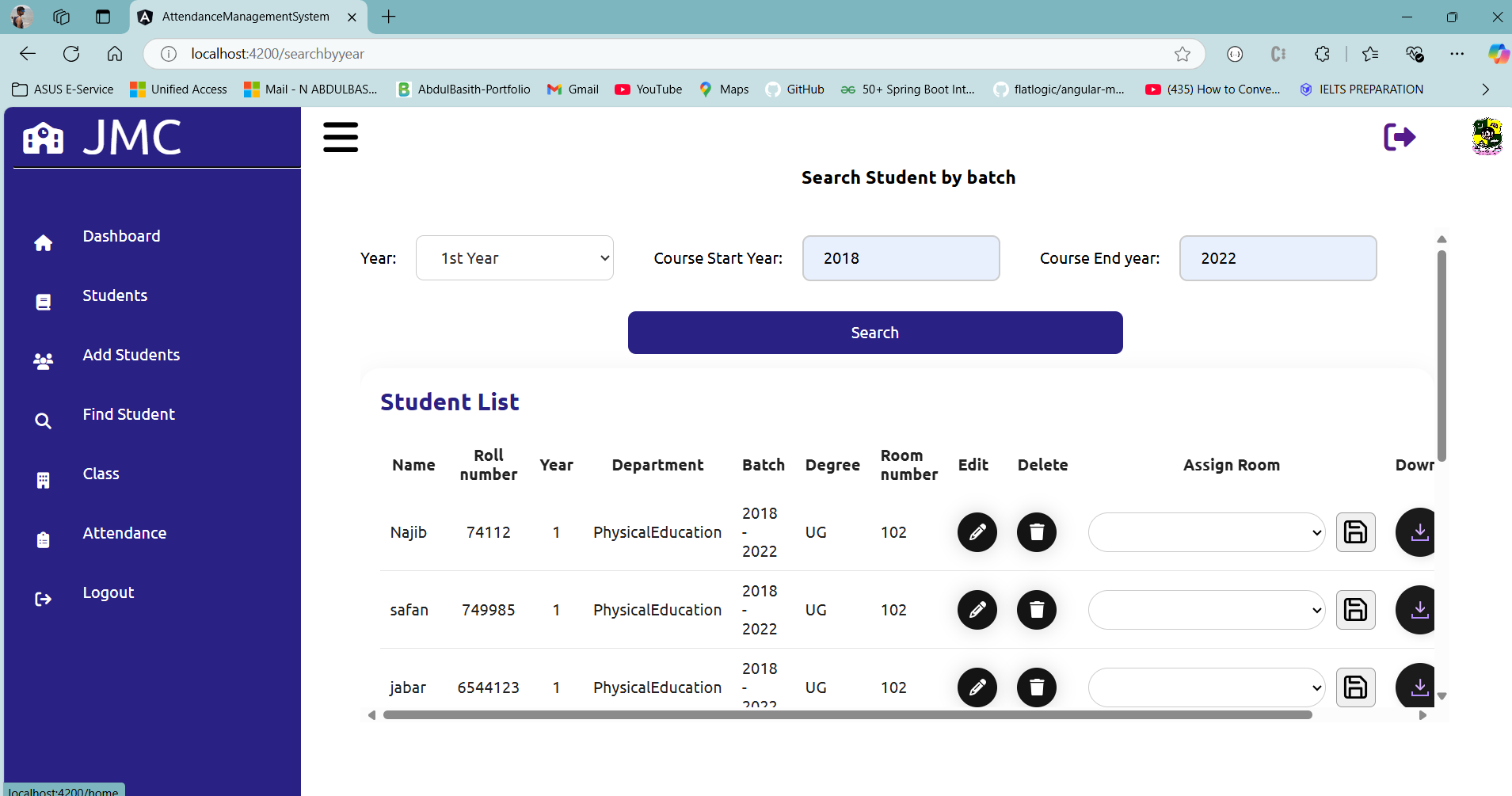
Add student page:



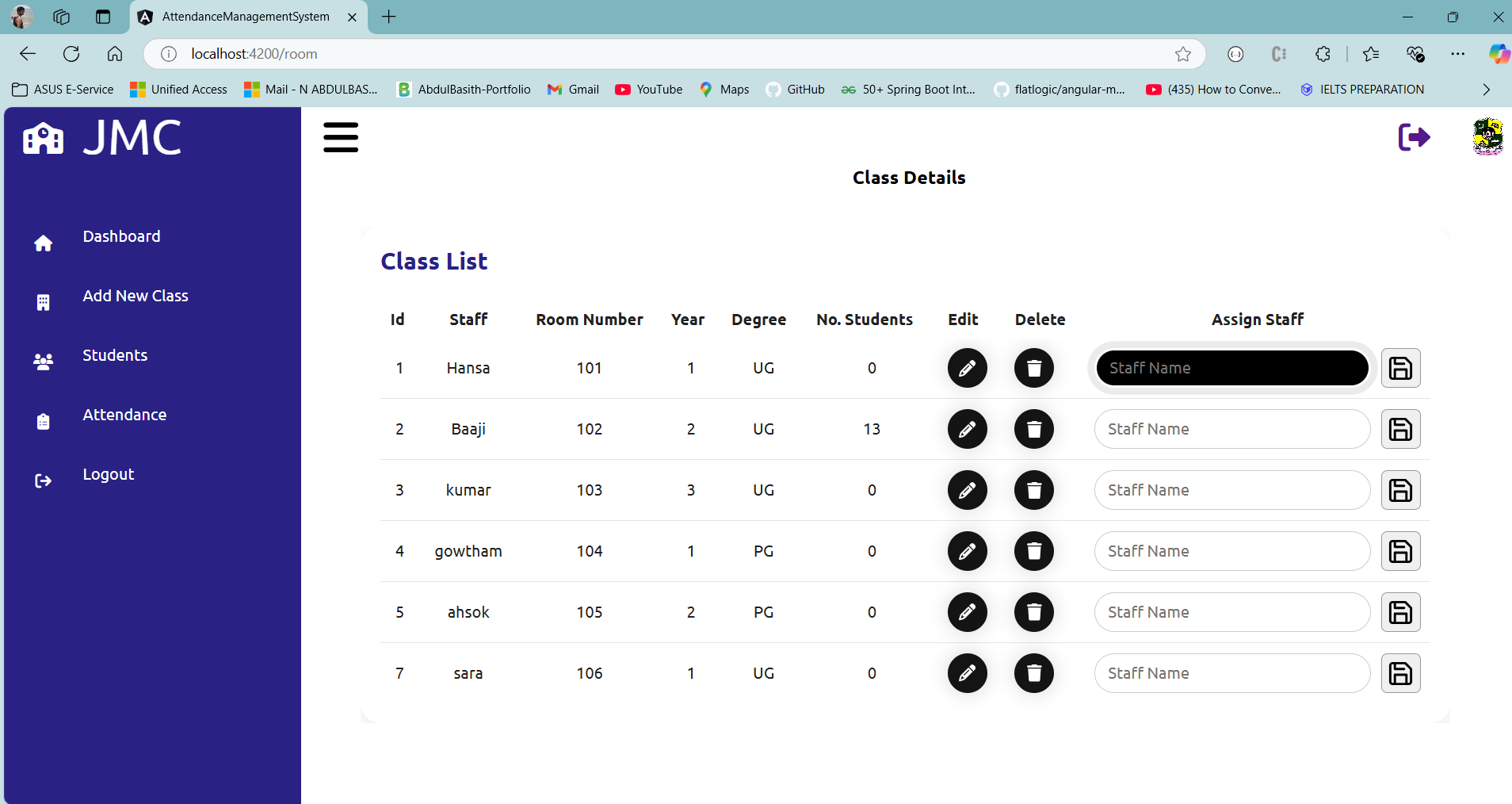
Search Student Room number:



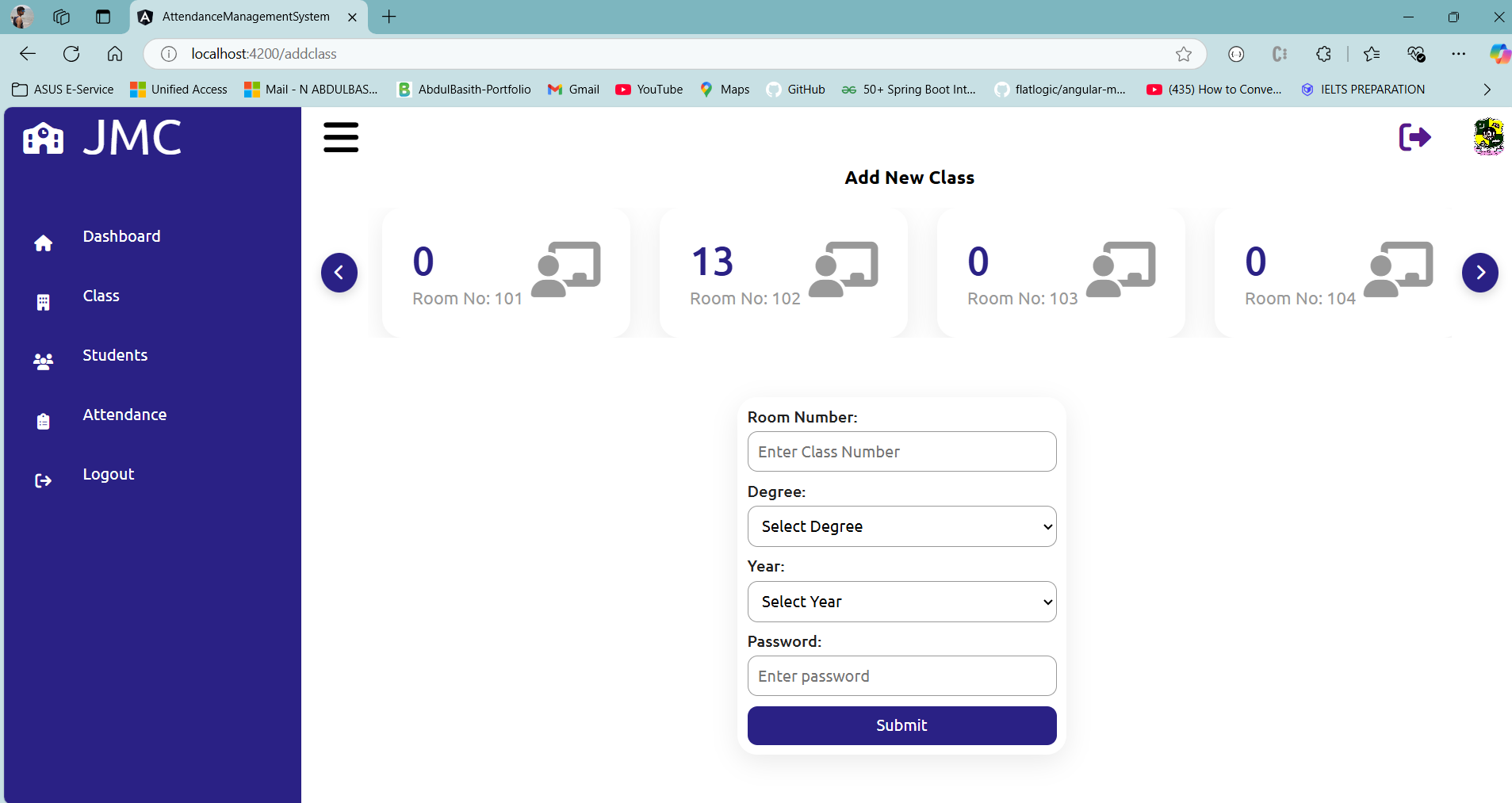
Search student by batch wise:



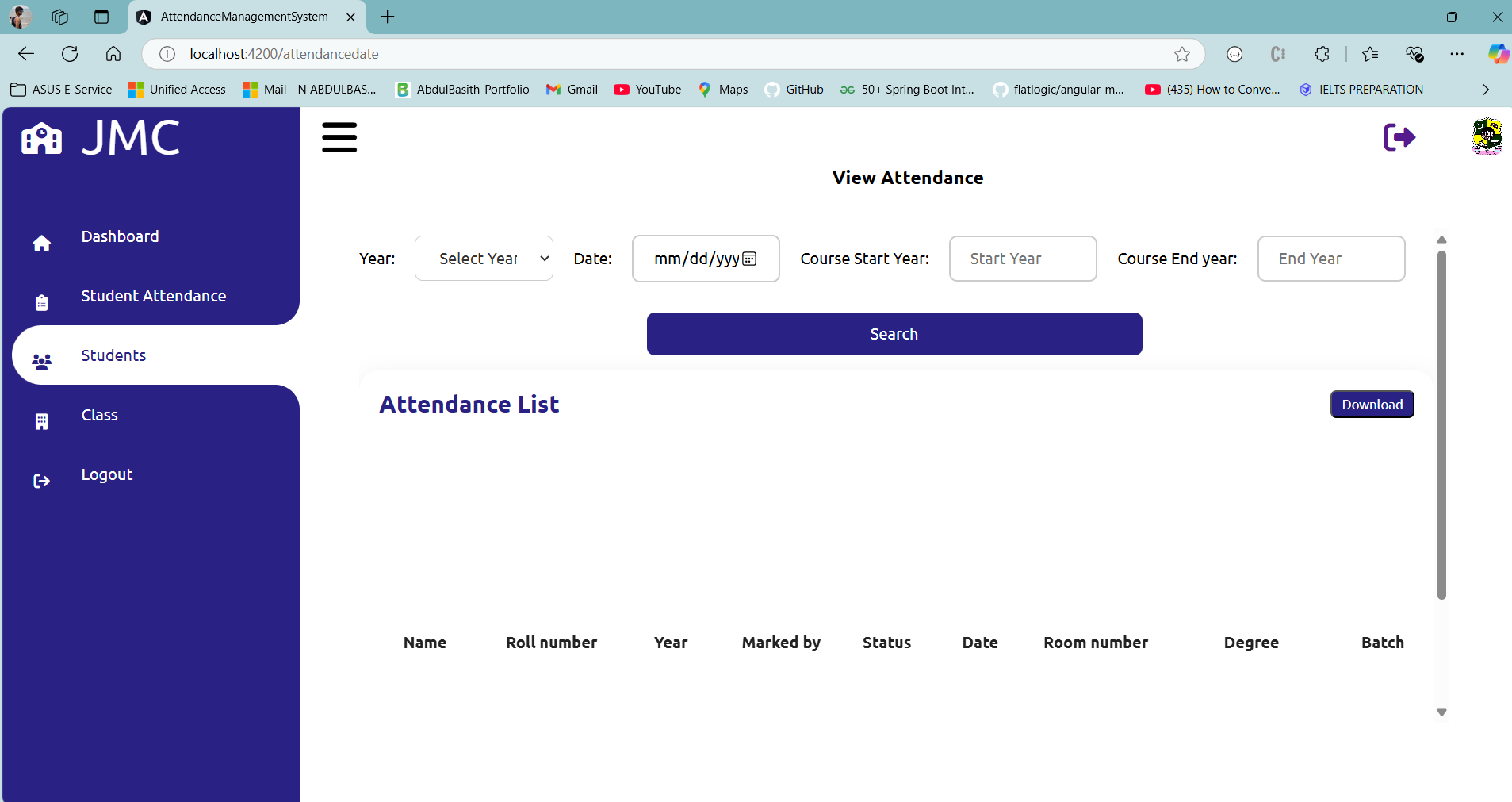
Assign staff /Delete/Edit Class page:



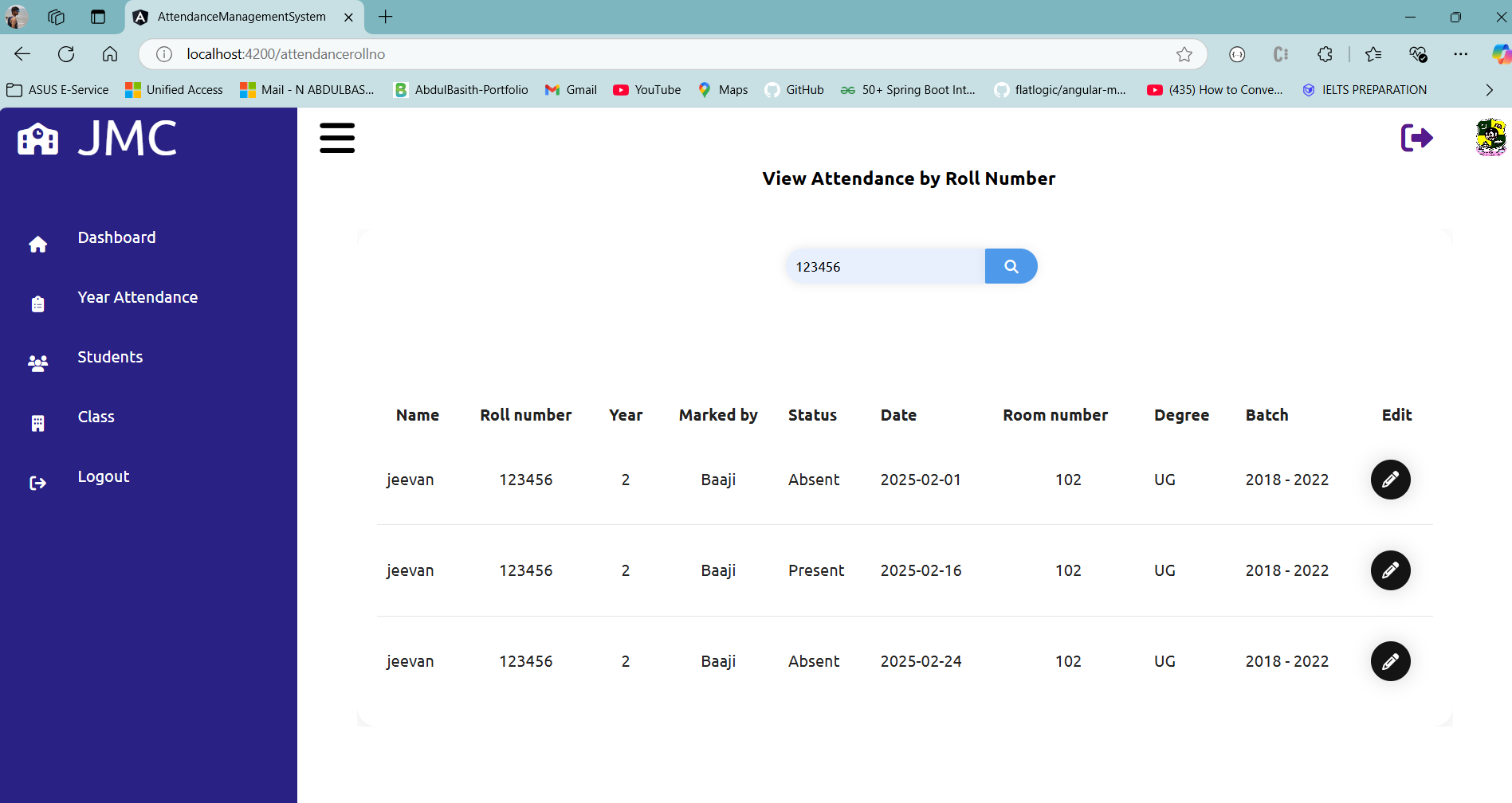
Add Class Page:



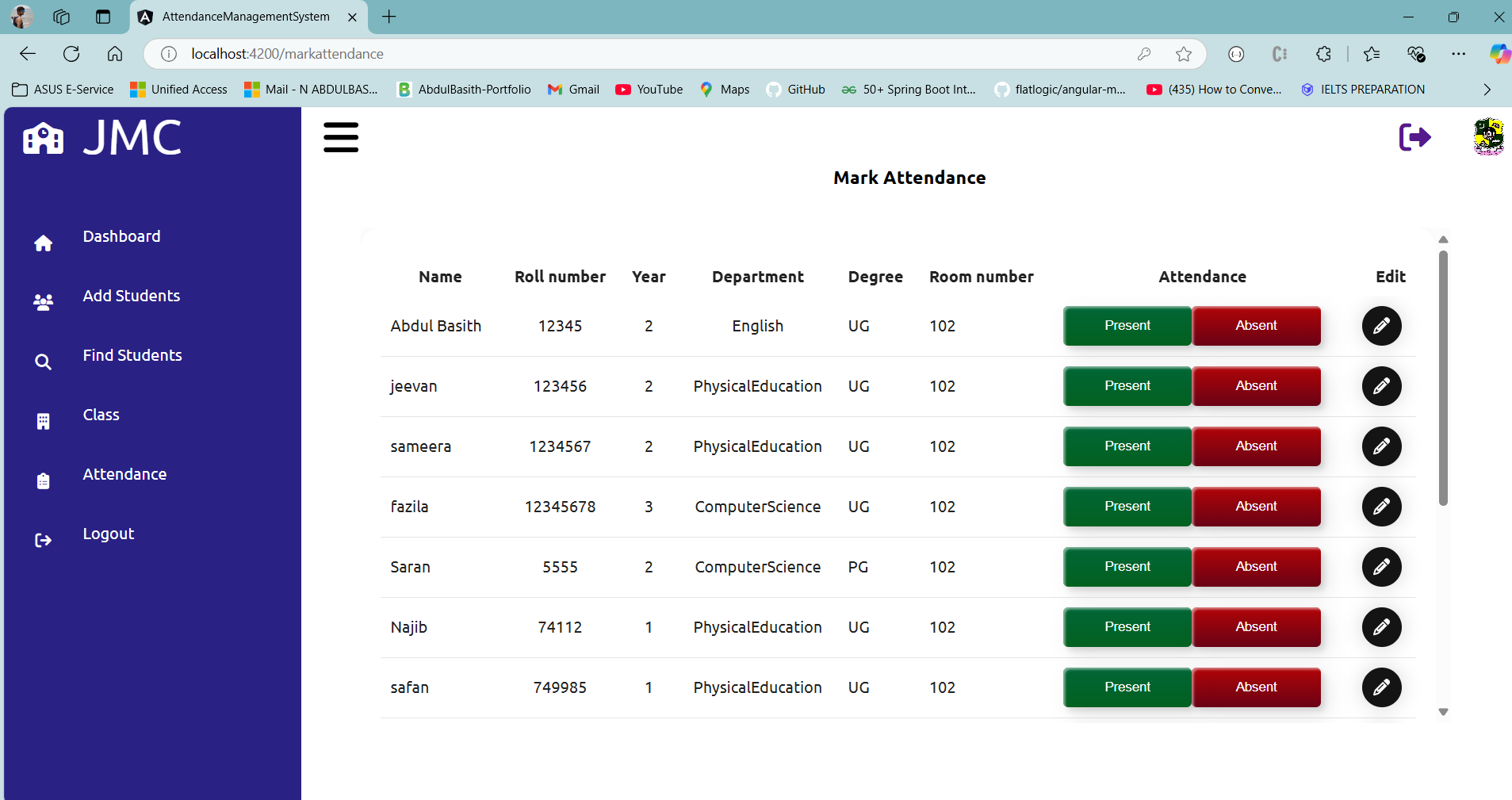
View/Download attendance year wise:



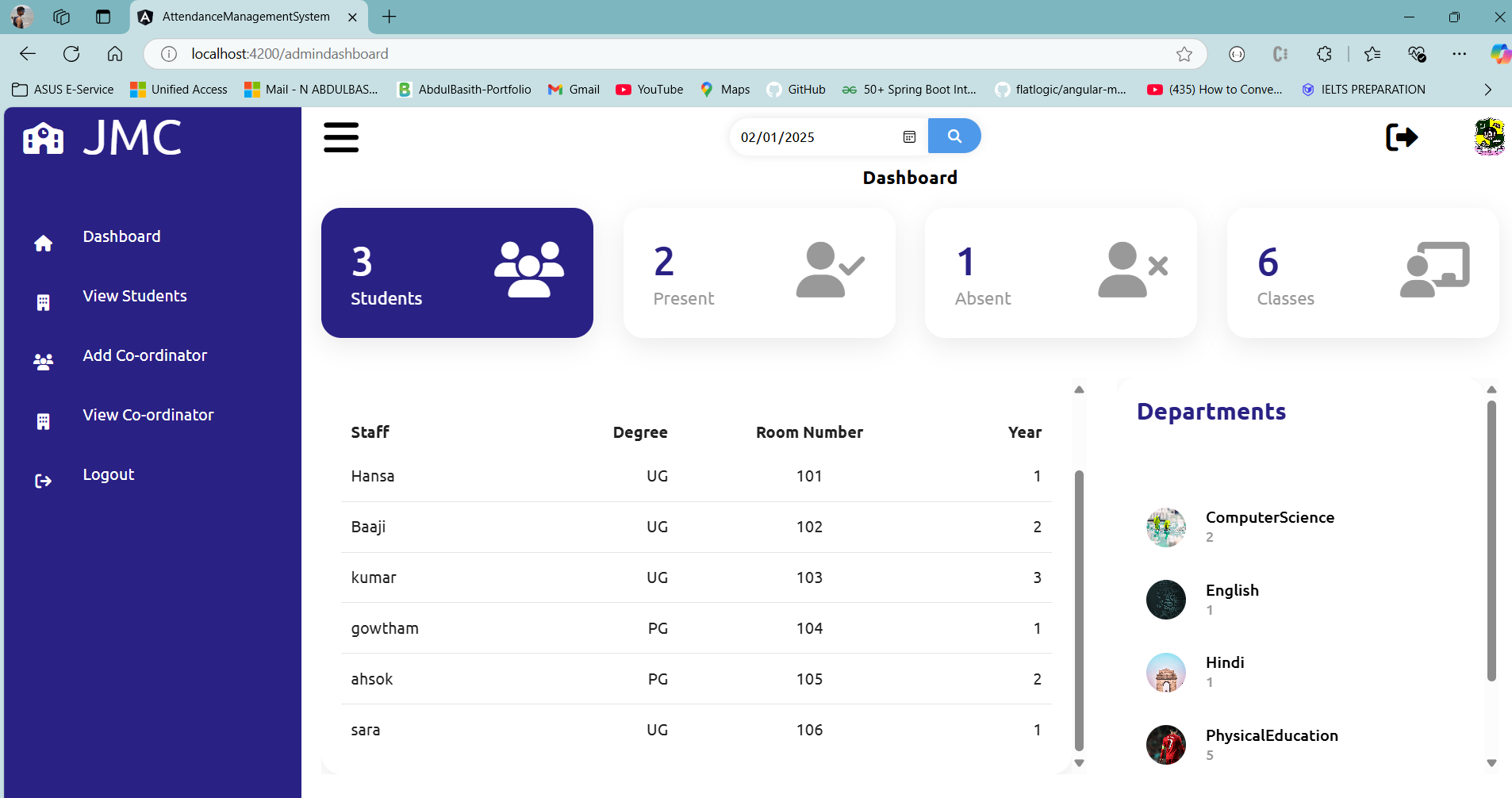
View/ Download attendance by roll number:



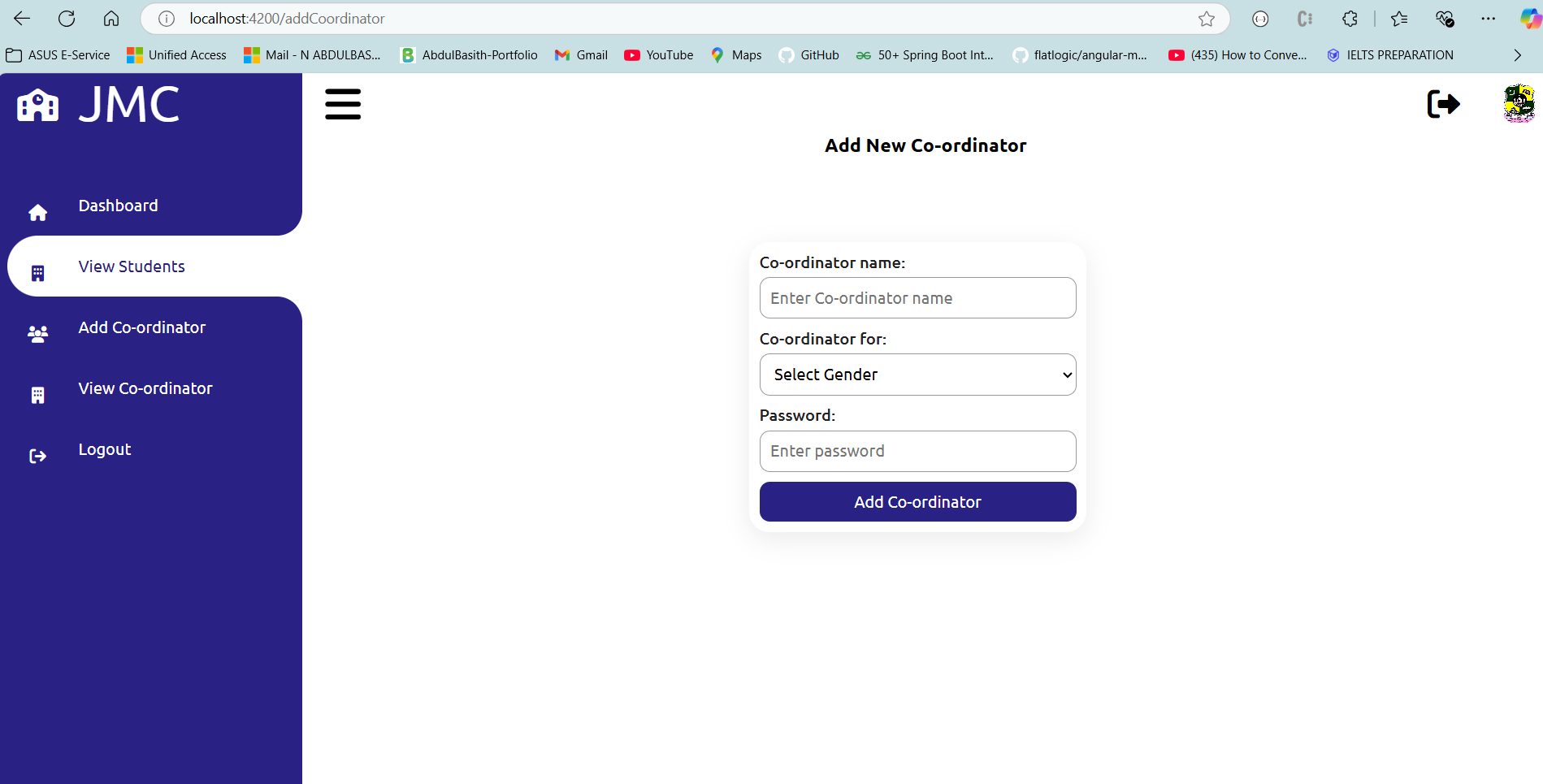
Mark attendance page:



Admin Dashboard:



Add Co-Ordinator:



Co-Ordinator details:

