EXPERIMENT 13

AIM: MINI PROJECT

Submit the Project Report which will consist of documentation related to different phases of Software Development Life Cycle:

- Title of the Project, Abstract, Introduction
- Software Requirement Specification
- Conceptual Design using ER features, Relational Model in appropriate Normalize form
- Graphical User Interface, Source Code
- Testing document
- Conclusion

Develop application considering:

- **Front End :** Java/Perl/PHP/Python/Ruby/.net/any other language
- **Backend :** MongoDB/MySQL/Oracle

A Mini Project Report

on

"COURSE ENROLLMENT SYSTEM"

by

AKASH AMRUTE (TEACOMP03)

ROHIT BADGUJAR (TEACOMP11)

TUSHAR CHAVHAN (TEACOMP23)

Under the guidance of

Dr. Nikita Kulkarni

Department of Computer Engineering

K J's Educational Institute

K J College of Engineering & Management Research, Pune

SAVITRIBAI PHULE PUNE UNIVERSITY

2025-2026

K J's Educational Institute

K J College of Engineering & Management Research

(Accredited by NAAC)

Affiliated to Savitribai Phule Pune University (SPPU), Approved by Govt. of Maharashtra, Recognized by AICTE, New Delhi.

CERTIFICATE

This is to certify that,

AKASH AMRUTE (TEACOMP03)

ROHIT BADGUJAR (TEACOMP11)

TUSHAR CHAVHAN (TEACOMP23)

of class **T.E Computer**; have successfully completed their mini project work on **"COURSE ENROLLMENT SYSTEM"** at **K J College of Engineering & Management Research** in the partial fulfilment of the Graduate Degree course in T.E at the department of **Computer Engineering** in the academic Year 2025-2026 Semester – I as prescribed by the Savitribai Phule Pune University.

Dr. Nikita Kulkarni

Project Guide

Dr. Nikita Kulkarni

Head of Department

Acknowledgements

With deep sense of gratitude, we would like to thanks all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work.

It is our proud privilege to express deep sense of gratitude to **Dr. Suhas S. Khot**, Principal of **K J College of Engineering & Management Research**, Pune for his comments and kind permission to complete this project. We remain indebted to Dr. Nikita Kulkarni, H.O.D. of Computer Engineering Department for her timely suggestion and valuable guidance.

The special gratitude goes to Dr. Nikita Kulkarni for her excellent and precious guidance in completion of this work .We thanks to all the colleagues for their appreciable help for our working project. With various industry owners or lab technicians to help, it has been our endeavor to throughout our work to cover the entire project work.

We also thankful to our parents who providing their wishful support for our project completion successfully. And lastly we thanks to our all friends and the people who are directly or indirectly related to our project work.

AKASH AMRUTE (TEACOMP03)

ROHIT BADGUJAR (TEACOMP11)

TUSHAR CHAVHAN (TEACOMP23)

Contents

Abstract

The **Course Enrollment System** is a dynamic, web-based application designed to **streamline the process of student course registration and management**. It provides secure user and admin login, a clear dashboard for viewing **Available Courses**, and allows students to **Enroll** or **Delete** (un-enroll) from courses in real-time. The system incorporates robust database design features, including a many-to-many relationship between users and courses, ensuring efficient and accurate record-keeping. Built using **PHP and MySQL**, the system offers a user-friendly interface for students and a dedicated panel for administrators to manage course catalogs.

Chapter 1: Introduction

1.1 Introduction:

The motivation behind this project stems from the need to modernize and digitize the course enrollment process. Traditional methods often rely on manual forms and outdated systems, leading to registration errors, limited course visibility, and poor record tracking. Our system digitizes the entire workflow—from user registration and login to course enrollment, profile management, and un-enrollment. It aims to provide a secure, user-friendly platform that enhances the student experience while simplifying course administration for educational institutions.

1.2 Problem Statement:

Educational institutions and students face several challenges due to outdated, manual enrollment systems. These include enrollment errors, lack of real-time visibility into course availability, duplicate entries, and difficult record management. Additionally, students often struggle with limited access to manage their enrolled courses. The absence of a centralized digital platform restricts users from viewing their status or receiving instant confirmation. There is a pressing need for a robust, web-based **Course Enrollment System** that offers secure login, a complete course catalog, real-time enrollment updates, and profile management.

1.3 Framework of the Proposed Work in Project:

The development of the **Course Enrollment System** follows a structured framework to ensure the system is reliable, user-friendly, and scalable. The key phases are:

1. Needs Assessment and Requirement Gathering

- Identify the specific needs of students, course administrators, and faculty.
- Analyze existing enrollment methods and gather feedback to understand pain points such as manual errors, lack of real-time course availability, and limited profile management.
- Define core requirements like secure student and admin login, course enrollment and drop functionality, user profile updates, and course catalog management.

2. System Design and Architecture

- Design the overall architecture, including the database schema with **Users**, **Courses**, and **Enrollments** tables, the user interface layout, and backend logic.
- Ensure the system supports modular components like user authentication, the course dashboard, and the student profile page.

- Focus on intuitive UI/UX using **Bootstrap** and responsive design principles for accessibility on various devices.

3. Development and Coding

- Implement core functionalities such as **user registration**, **login**, **viewing available courses**, **enrolling in a course**, and **updating the profile**.
- Use **PHP** and **MySQL** for backend logic and data handling.
- Integrate **session protection** and **input validation** to secure user data and enrollment records.

4. Testing and Quality Assurance

- Perform **unit testing** and **integration testing** to ensure all modules work correctly.
- Validate the **enrollment logic** (preventing duplicate enrollments), **profile update functionality**, and **session handling**.
- Fix bugs and optimize performance for a smooth user experience.

5. User Training and Documentation

- Prepare documentation for users and admins explaining how to use the system.
- Include screenshots and step-by-step instructions for login, course enrollment, and profile management.
- Provide error-handling guidance and FAQs for common issues.

6. Launch and Deployment

- Deploy the system locally using **XAMPP** for testing and demonstration.
- Conduct pilot testing with sample users to gather feedback.
- Refine UI and logic based on user suggestions before final submission.

7. Scalability and Enhancement

- Design the system to support future upgrades like **payment gateway integration** for course fees, **email confirmation** for enrollments, and an **admin panel** to add/edit courses.
- Ensure the database and codebase are scalable to handle an increasing number of students and courses.

Chapter 2: Literature Review

2.1 Introduction:

The **Course Enrollment System** represents a vital and transformative component within the educational management domain, leveraging advanced technology to address the complexities faced by students and academic administrators. In this era of digital transformation, understanding the existing landscape of literature is paramount to effectively inform the design, development, and deployment of a modern enrollment solution.

Educational institutions have long been responsible for managing the critical process of course registration, student record keeping, and academic planning. However, the traditional methods employed, often relying on manual forms, physical queues, and siloed paper-based systems, have faced significant limitations in the face of rapidly increasing student populations and the demand for real-time data. Our system aims to bridge these gaps, and the literature review serves as a comprehensive examination of current practices, technologies, and challenges within the academic administration sector.

This literature review will delve into various facets of academic management, including existing **Student Information Systems (SIS)**, **online course catalog platforms**, **user authentication protocols**, and **database design methodologies**. By synthesizing knowledge from academic research, industry reports on educational technology, and case studies of successful university systems, this review will provide the necessary context for the development and implementation of a sophisticated solution that addresses the ever-changing needs of modern student enrollment.

2.2 Existing Methodologies:

The development of the Course Enrollment System draws upon established methodologies commonly used in the creation of robust web applications and database-driven systems:

1. Agile Development for Web Applications:

- The **Agile methodology**, including frameworks like Scrum and Kanban, is favored for its flexibility and collaborative nature in software development.
- It has been successfully employed in the iterative development of educational platforms, allowing developers to prioritize features like user enrollment and course management, respond quickly to changing academic requirements, and foster close cooperation between developers and system administrators.

2. User-Centered Design (UCD):

- **User-Centered Design (UCD)** methodologies are crucial in the design of high-traffic academic websites, emphasizing the importance of understanding student and administrator needs.
- User research, creating personas (e.g., student vs. admin), and usability testing have been leveraged to create intuitive, user-friendly interfaces, ensuring the enrollment and course selection processes are straightforward and error-free.

3. Security-Centric Approaches:

- Literature emphasizes the critical need for robust security measures in systems handling sensitive student data, such as login credentials and academic records.
- Security frameworks, including the principles of the **OWASP Top Ten methodology**, are explored to identify and mitigate vulnerabilities like SQL Injection and session hijacking, ensuring the safeguarding of valuable user data and enrollment records.

4. Responsive Web Design (RWD):

- Given that students and administrators access the system using a diversity of devices (laptops, tablets, and smartphones), **Responsive Web Design (RWD)** principles are widely adopted.
- RWD ensures the website is accessible and visually consistent across various screen sizes and platforms, from the registration page to the course selection dashboard.

5. Database Management System (DBMS) Principles:

- Unlike CMS frameworks, enrollment systems often rely on strong DBMS principles, particularly **Relational Model design**, to manage complex relationships between students, courses, and enrollments.
- This includes using **Normalization Forms** to reduce data redundancy and improve data integrity, which is vital for maintaining accurate student transcripts and course availability.

6. System Performance Optimization:

- Literature highlights the importance of optimizing the system for performance, particularly during peak registration periods.
- Strategies like **database indexing**, efficient query design, and client-side optimization are instrumental in ensuring the system handles concurrent user traffic without delays or crashes.

2.3 Proposed Methodology:

The development of the Course Enrollment System will follow a structured, iterative methodology to ensure a reliable and user-focused final product.

1. Requirements Analysis:

- Conduct extensive stakeholder consultations to understand the specific needs and objectives of students and academic administrators.
- Document detailed system requirements, including core functionalities such as secure authentication, the ability to **Enroll** and **Delete** courses, and manage user profiles.

2. System Design and Prototyping:

- Create a comprehensive system design, specifically defining the **Relational Database Structure** (Users, Courses, Enrollments, etc.), user interfaces (Dashboard, Profile Page), and backend logic.
- Develop interactive **prototypes** to visualize the system's layout and functionality for feedback on the enrollment workflow and user experience.

3. Agile Development and Testing:

- Employ an **Agile development approach** with iterative sprints to build the system incrementally, prioritizing the most critical features first (e.g., Enrollment and Login).
- Rigorously test each module, including **unit testing** (individual functions like enrollment logic), **integration testing** (database connections), and **user acceptance testing (UAT)** to identify and rectify issues before release.

4. User Training and Documentation:

- Prepare user documentation and training materials to facilitate student and administrator onboarding.
- Conduct training sessions to ensure proficient system use, particularly for new student registration and administrative course management tasks.

5. System Deployment and Migration:

- If applicable, assist the institution in migrating any existing student/course data into the new MySQL database structure.
- Deploy the system locally using **XAMPP** for final testing and demonstration, followed by potential deployment to a pilot group for live feedback.

6. Launch, Support, and Scalability:

- Officially launch the Course Enrollment System to the intended student user base.
- Plan for **scalability** and future enhancements (like payment integration or mobile access) based on real-world usage data and ongoing user feedback.

Chapter 3: Software Requirement Specification

3.1 Hardware Requirements

To develop and test the **Course Enrollment System** efficiently, the following hardware configuration was used:

- **Laptop Model:** Asus TUF F15
- **Processor:** Intel Core i5 (12th Gen or higher)
- **RAM:** Minimum 8 GB
- **Storage:** 256 GB SSD or higher
- **Display:** 15.6” Full HD
- **Network:** Stable internet connection for testing and deployment

3.2 Software Requirements

The system was built using widely adopted development tools and technologies to ensure compatibility, scalability, and ease of use:

- **Operating System:** Windows 11
- **Backend:** MySQL Workbench 8.0 (Database Management)
- **Server Environment:** XAMPP (Apache + MySQL + PHP)
- **Frontend Technologies:**
  - ✧ **HTML5** (Structure and content)
  - ✧ **CSS3** (Styling and layout)
  - ✧ **Bootstrap 5.3** (Responsive design and UI components)
- **Programming Language:** PHP 8.2 (Server-side scripting)
- **Browser Compatibility:** Google Chrome, Microsoft Edge

Chapter 4: Assumptions

The development and deployment of the **Course Enrollment System** is based on the following key assumptions. These assumptions help define the scope, guide the implementation, and ensure the system functions effectively within its intended environment:

1. **User Access and Connectivity**

   - It's assumed that users (students and admins) will have access to **internet-enabled devices** (laptops, smartphones) and a stable internet connection to interact with the system.

2. **Course and Catalog Data Availability**

   - It's assumed that the **admin has access to accurate and up-to-date information** about **courses**, course codes, descriptions, and prerequisites (if implemented) to populate the system.

3. **User Registration and Authentication**

   - It's assumed that **students will register and log in** before viewing their enrollment status or managing their courses.
   - **Session management** is assumed to restrict unauthorized access to enrollment and profile features.

4. **Enrollment Logic and Availability**

   - It's assumed that **course availability** (i.e., whether a course is open for enrollment) is predefined and consistent, allowing the system to accurately manage the **enroll** and **delete** (un-enroll) processes.

5. **Admin Control and Permissions**

   - It's assumed that **only authorized admins** can add or manage the course catalog.
   - Admin login is protected by secure credentials.

6. **Local Hosting Environment**

   - It's assumed that the system will be hosted locally using **XAMPP** for development and demonstration purposes, without requiring external hosting or domain setup.

7. **Security Measures**

   - It's assumed that **basic security features** like input validation, session protection, and restricted access are implemented to safeguard student data and enrollment records.

8. **Testing and Debugging**

   - It's assumed that the system will be **tested thoroughly** for bugs, session errors, and enrollment logic before final deployment or submission.

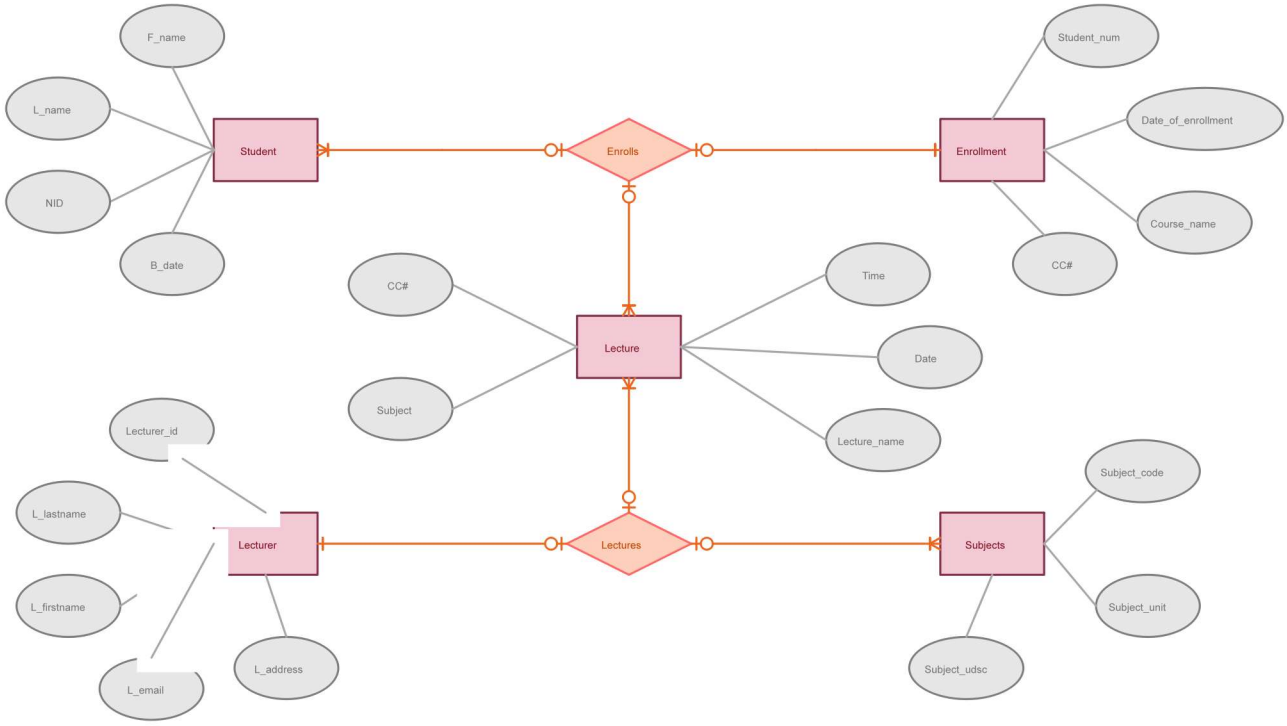
9. **Scalability and Future Enhancements**

   - It's assumed that the system is designed to support future upgrades such as **payment gateway integration** for fees, **email confirmation** for enrollments, and **mobile app compatibility**.

10. **Documentation and User Guidance**

    - It's assumed that user manuals and admin instructions will be provided to ensure smooth onboarding and usage of the system.

Chapter 5 : Entity Relationship Diagram



Chapter 6 : Tables/Collections

The **Course Enrollment System** uses a relational database designed in **MySQL**. Below are the key tables derived from the ER Diagram, along with their attributes and relationships.

- 6.1 Student Table

```
CREATE TABLE Student (
   NID VARCHAR(20) PRIMARY KEY,          -- National ID / Student ID as Primary Key
   F_name VARCHAR(100),                  -- First Name
   L_name VARCHAR(100),                  -- Last Name
   B_date DATE                           -- Birth Date
);
```

- 6.2 Admin Table

```
CREATE TABLE Admin (
   id INT AUTO_INCREMENT PRIMARY KEY,
   username VARCHAR(50) UNIQUE,
   password VARCHAR(255)
);
```

- 6.3 Subjects Table

```
CREATE TABLE Subjects (
   Subject_code VARCHAR(10) PRIMARY KEY,
   Subject_name VARCHAR(100),
   Subject_unit INT,
   Subject_desc TEXT
);
```

- 6.4 Lecturer Table

```
CREATE TABLE Lecturer (
   Lecturer_id VARCHAR(15) PRIMARY KEY,
   L_firstname VARCHAR(100),
   L_lastname VARCHAR(100),
   L_email VARCHAR(100) UNIQUE,
```

```
    L_address VARCHAR(255)
);
```

- 6.5 Enrollment Table

```
CREATE TABLE Enrollment (
    NID VARCHAR(20),                    -- Foreign Key to Student (Student_num in ERD is mapped to NID)
    CC_hash VARCHAR(10),                -- Course Code Hash (CC# in ERD, could be Subject_code)
    Date_of_enrollment DATE,
    Course_name VARCHAR(100),           -- Redundant, but kept for simplicity based on ERD attribute
    PRIMARY KEY (NID, CC_hash),
    FOREIGN KEY (NID) REFERENCES Student(NID),
    FOREIGN KEY (CC_hash) REFERENCES Subjects(Subject_code)
);
```

- 6.6 Lecture Table

```
CREATE TABLE Lecture (
    Lecture_ID INT AUTO_INCREMENT PRIMARY KEY,
    CC_hash VARCHAR(10),                -- Foreign Key to Subjects
    Lecturer_id VARCHAR(15),            -- Foreign Key to Lecturer
    Lecture_name VARCHAR(100),
    Date DATE,
    Time TIME,
    FOREIGN KEY (CC_hash) REFERENCES Subjects(Subject_code),
    FOREIGN KEY (Lecturer_id) REFERENCES Lecturer(Lecturer_id)
);
```

Chapter 7 : Features

Features of the Course Enrollment System

1. Enrollment Management

- **Description:** The system allows students to manage their academic record by enrolling in available courses and dropping currently enrolled ones with real-time updates.
- **Capabilities:**

  - ✧ Efficiently manage student enrollment in courses using a many-to-many relationship in the database.
  - ✧ Prevent double-enrollment in the same course and ensure accurate course availability.
  - ✧ Store enrollment history for future reference and transcript generation.

2. Course Catalog Management

- **Description:** Admins can add, manage, and update the list of available courses, including codes and descriptions.
- **Capabilities:**

  - ✧ Maintain course details including name, unique course code (e.g., WEB301), and course credits.
  - ✧ Allow admins to update or delete course entries as needed to maintain an up-to-date catalog.
  - ✧ Link courses to relevant departments or types for organized data handling.

3. User Profile and Authentication

- **Description:** The system provides secure user registration, login, and profile management capabilities.

- **Capabilities:**

  - ✧ Secure login and session management to restrict unauthorized access to student enrollment features.

  - ✧ Allow registered users to view and update their personal details and securely change their password.
  - ✧ Differentiate between user roles (Student and Admin) for role-based access control.

4. Ease of Use and Error Prevention

- **Description:** The system is designed to be intuitive for both students and administrators and reduce errors during data entry.
- **Capabilities:**

  - ✧ Clean dashboard interface with clear navigation for viewing available courses and current enrollments.
  - ✧ Input validation to prevent incorrect or incomplete data during registration and profile updates.
  - ✧ Real-time feedback for invalid entries or session issues.

  ○

5. Web Application in PHP/MySQL

- **Description:** The system is built as a dynamic web application using PHP and MySQL.
- **Capabilities:**

  - ✧ Server-side scripting (PHP) for secure login logic, enrollment processing, and session handling.
  - ✧ Database integration (MySQL) for storing users, the course catalog, and the enrollment records.
  - ✧ Real-time data synchronization and backend validation across all transactions.

6. User-Friendly Interface

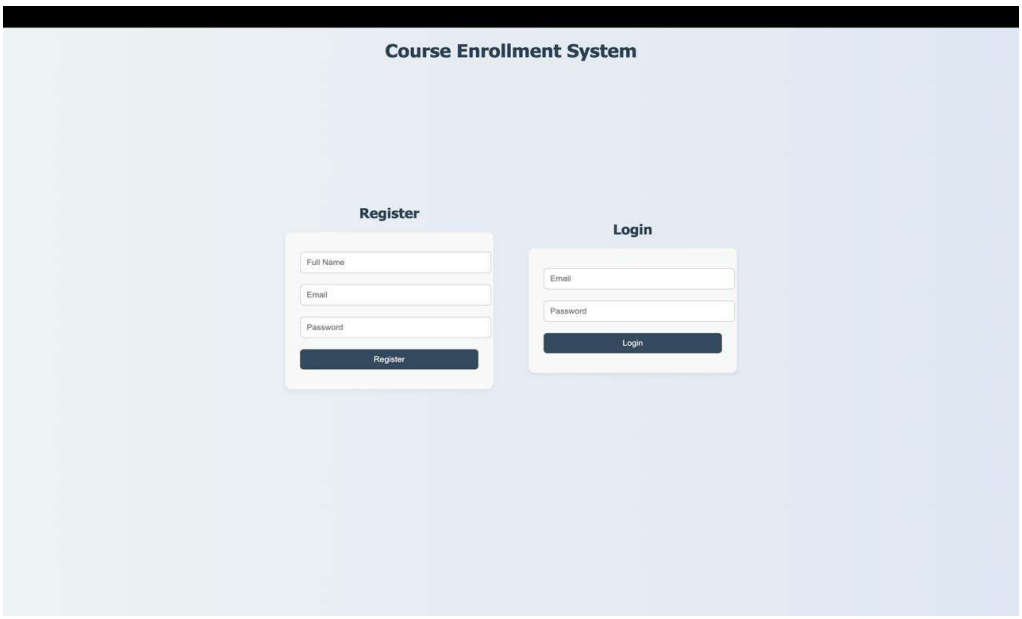
- **Description:** The interface is designed for clarity, responsiveness, and ease of navigation across different devices.

- **Capabilities:**

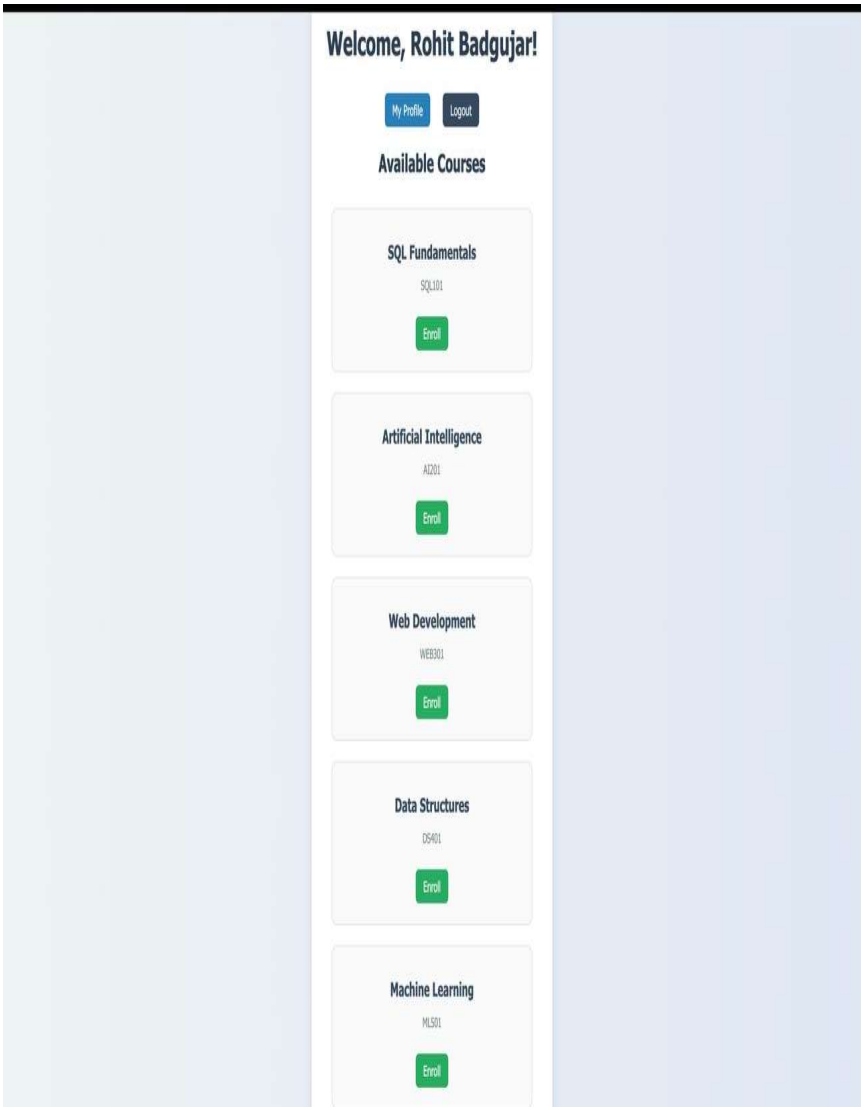
  - ✧ **Bootstrap-powered layout** for mobile and desktop compatibility.
  - ✧ Personalized welcome messages (e.g., "Welcome, Rohit Badgujar!") upon successful login.
  - ✧ Clear visual distinction between **"Enroll"** (green) and **"Delete"** (red) actions for better user experience (UX).

Chapter 8 : Screenshots

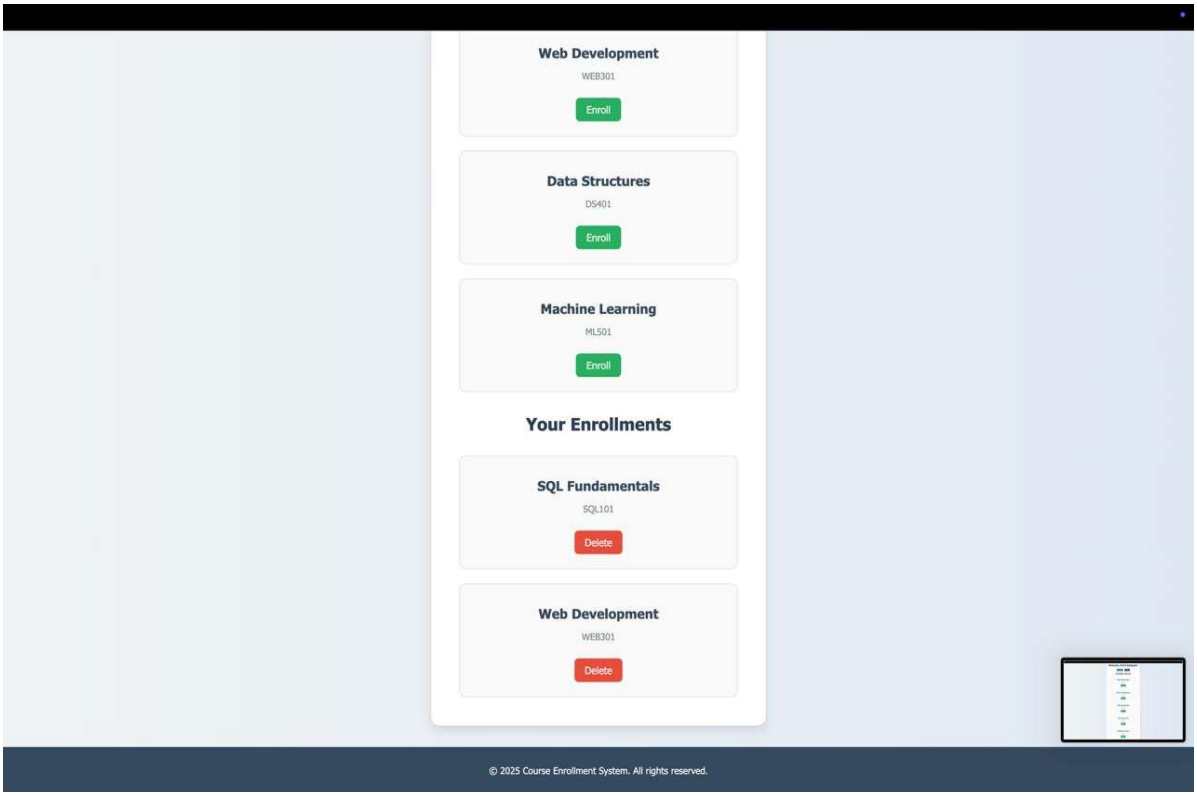
- 8.1 Register and Login Page



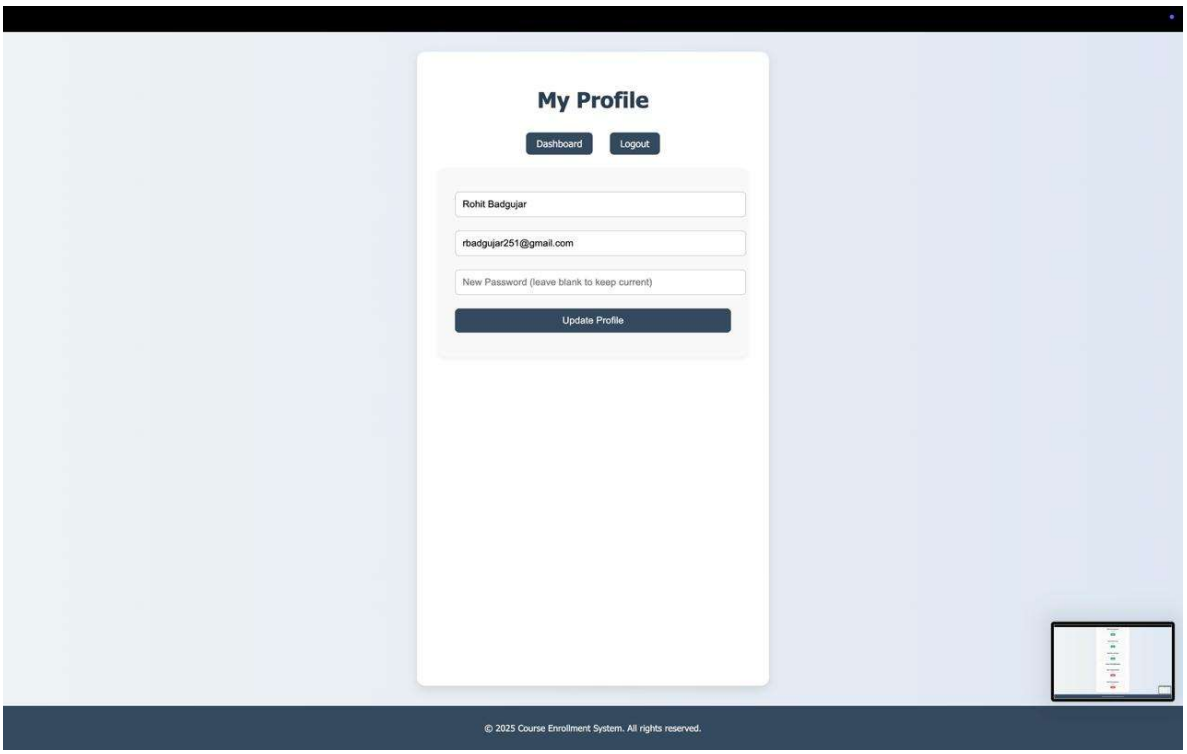
8.2 Available Courses



8.3 Course Enrollments



8.4 Profile



Chapter 9: Conclusion

DBMS Lab Mini-Project Report – Course Enrollment System

To conclude, the **Course Enrollment System** is a dynamic web application built using **PHP** and **MySQL** that efficiently manages student enrollment, user authentication, and the course catalog. It successfully overcomes the limitations of manual student record systems and provides a secure, scalable, and user-friendly platform for both students and academic administrators.

Key Achievements:

- **Effective Data Management:** Implemented a robust relational model to manage the many-to-many relationship between Students and Subjects via the **Enrollment** table.
- **Easy Implementation:** Achieved functional deployment in a local server environment (XAMPP).
- **Core Logic:** Implemented secure login, profile updates, and real-time enrollment/un-enrollment logic.
- **User Experience:** Features a **Responsive Design** for multi-device access and a clean interface for clear navigation.

The project has vast scope for future enhancement. It is modular and can be upgraded as new requirements arise, making it highly adaptable for expansion in a real-world academic setting. The system is now fully functional, allowing students to accurately manage their course records and enabling admins to oversee the course catalog efficiently.

Future Scope:

- **Integration of Payment Gateway:** For online payment of course fees during the enrollment process.
- **Email Confirmation:** Automated SMS/email confirmation for successful course enrollment or drop actions.
- **Mobile App Version:** Development of a dedicated mobile application for Android/iOS users.
- **Analytics Dashboard:** An advanced dashboard for admin insights into popular courses, enrollment trends, and student demographics.
- **Pre-requisite Checks:** Implementing logic to prevent enrollment in a course until the required prerequisite courses are completed.