HALL TICKET GENERATOR

Mini Project submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Under the esteemed guidance of

G. Udaya Sree

Assistant Professor

By

T SAKETH REDDY (21R11A05R0)

Department of Computer Science and Engineering

Accredited by NBA

Geethanjali College of Engineering and Technology

(UGC Autonomous)

(Affiliated to J.N.T.U.H, Approved by AICTE, New Delhi)

Cheeryal (V), Keesara (M), Medchal.Dist.-501 301.

August-2024

Geethanjali College of Engineering & Technology

(UGC Autonomous)

(Affiliated to JNTUH, Approved by AICTE, New Delhi)

Cheeryal (V), Keesara(M), Medchal Dist.-501 301.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Accredited by NBA

CERTIFICATE

This is to certify that the B.Tech Mini Project report entitled **"HALL TICKET GENERATOR"** is a bonafide work done by **T Saketh Reddy(21R11A05R0)** in partial fulfillment of the requirement of the award for the degree of Bachelor of Technology in **"Computer Science and Engineering"** from Jawaharlal Nehru Technological University, Hyderabad during the year 2023-2024.

Internal Guide	**HOD - CSE**
G. Udaya Sree	**Dr A Sree Lakshmi**
Assistant Professor	Professor

External Examiner

Geethanjali College of Engineering & Technology

(UGC Autonomous)

(Affiliated to JNTUH Approved by AICTE, New Delhi)

Cheeryal (V), Keesara(M), Medchal Dist.-501 301.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Accredited by NBA

DECLARATION BY THE CANDIDATE

I, **T Saketh Reddy** bearing Roll No. **21R11A05R0,** hereby declare that the project report entitled **"HALL TICKET GENERATOR"** is done under the guidance of **Mrs. G. Udaya Sree, Assistant Professor**, Department of Computer Science and Engineering, Geethanjali College of Engineering and Technology, is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering.**

This is a record of bonafide work carried out by me/us in **Geethanjali College of Engineering and Technology** and the results embodied in this project have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other University or Institute for the award of any other degree or diploma.

T Saketh Reddy(21R11A05R0)

Department of CSE,

Geethanjali College of Engineering and Technology,

Cheeryal.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to our **Secretary Mr.G.R. Ravinder Reddy** for providing the necessary infrastructure to complete my project. I am also thankful to our **Principal Dr. S. Udaya Kumar** for providing an interdisciplinary & progressive environment.

I would like to express our sincere thanks **to Dr. A. Sree Lakshmi, Professor, Head of Department** of Computer Science, Geethanjali College of Engineering and Technology, Cheeryal, whose motivation in the field of software development has made me to overcome all hardships during the course of study and successful completion of project.

I would like to express my deep-felt gratitude and sincere thanks to my guide **G. Udaya Sree, Assistant Professor** of Computer Science, Geethanjali College of Engineering and Technology, Cheeryal, for her skillful guidance, timely suggestions and encouragement in completing this project sucessfully. I would like to express my profound sense of gratitude to all for having helped me in completing this dissertation.

Finally, I would like to express my heartfelt thanks to my parents who were very supportive both financially and mentally and for their encouragement to achieve my set goals.

T Saketh Reddy(21R11A05R0)

ABSTRACT

The Hall Ticket Generator project is designed to automate the process of issuing hall tickets and managing exam-related activities for students in educational institutions. Traditionally, students face long queues and manual verification steps for clearing dues and obtaining hall tickets for regular and Computer-Based Tests (CBT). This project addresses these inefficiencies by providing a web-based application where students can log in using their credentials to check for fee, fine, or library dues, make payments online, and download their hall tickets directly. The system is built using PHP for the backend, MySQL for the database, and Razorpay API for payment integration. The application features separate modules for students, staff, and administrators, each with specific functionalities like managing exam details, viewing and updating student records, and generating hall tickets in PDF format. This system significantly reduces the manual effort required, eliminates administrative bottlenecks, enhances data accuracy, and provides a seamless user experience. The Hall Ticket Generator aims to streamline the examination preparation process, saving time and effort for both students and staff while ensuring a more efficient and transparent system.

LIST OF FIGURES

S.No	Figure Name	Page No
1	System Architecture	13
2	Use Case Diagram	18
3	Sequence Diagram	19
4	Class Diagram	20
5	Activity Diagram	21
6	Modular Design	22
7	Home Page	51
8	Selecting Regular	51
9	Regular Page	52
10	Selecting Student	52
11	Validating Student Credentials	53
12	Fee Due	53
13	Payment Page	54
14	Payment Successful	54
15	Downloading Hall Ticket	55
16	Displaying Hall Ticket	55
17	Another Student Login	56
18	Library Due	56
19	Staff Page	57
20	Fee Management Login	57
21	Fee Management Page	58
22	Updating Fee	58
23	Librarian Login	59
24	Librarian Page	59
25	Updating Library Due	60
26	Selecting CBT	60
27	Verifying Student Credentials	61
28	CBT Page	61

29	CBT Payment	62
30	CBT Hall Ticket	62
31	Rechecking CBT Page	63
32	Selecting Admin	63
33	Verifying Admin Credentials	64
34	Select Sem	64
35	Year and Sem	65
36	Edit Regular Table	65
37	Regular Table	66
38	Edit CBT Table	66
39	Deleting a Record in CBT Table	67
40	Edit Student Data	67
41	Student Data	68
42	Editing Student Data	68
43	Verifying Change in Student Data	69
44	Manage Electives	69
45	Manage Electives Page	70
46	PE Mapping	70
47	PE Table	71
48	OE Mapping	71
49	OE Table	72

LIST OF ABBREVIATIONS

S.No	Acronym	Abbreviation
1	XAMPP	Cross-Platform (X), Apache (A), MariaDB (M), PHP (P), Perl (P)
2	HTML	Hypertext Markup Language
3	CSS	Cascading Style Sheets
4	PHP	Hypertext Preprocessor
5	JS	JavaScript
6	UML	Unified Modeling Language
7	SQL	Structured Query Language
8	API	Application Programming Interface
9	CRUD	Create, Read, Update, Delete
10	HTTP	Hypertext Transfer Protocol
11	JSON	JavaScript Object Notation
12	CBT	Computer Based Test
13	OE	Open Electives
14	PE	Professional Electives
15	PDF	Portable Document Format

TABLE OF CONTENTS

S.No	Contents	Page No
	Abstract	v
	List of Figures	vi
	List of Abbreviations	viii
1	**Introduction**	
	About the Project	1
	Objectives	2
2	**System Analysis**	
	Existing System	3
	Proposed System	5
	Feasibility Study	7
	Scope of Project	9
	System Configuration	10
3	**Literature Overview**	
	Literature Review	11
4	**System Design**	
	System Architecture	13
	Modules Description	15
	UML Diagrams	18
	System Design	
	Modular Design	22
	Database Design	25
5	**Implementation**	
	Working/Implementation	27
	Sample Code	29
6	**Testing**	
	Testing	48
	Test Cases	50
7	**Output Screens**	51

8	**Conclusion**	
	Conclusion	73
	Further Enhancements	73
9	**Bibliography**	
	Books References	75
	Websites References	76
	Technical Publication References	76
10	**Appendices**	77
11	**Plagiarism Report**	79

1. INTRODUCTION

1.1 ABOUT THE PROJECT

The Hall Ticket Generator project is designed to streamline and automate the process of issuing hall tickets and managing Computer-Based Tests (CBTs) within an academic setting. Traditionally, students had to manually visit multiple departments fee payment, library, and examination blocks to clear dues and receive their hall tickets. This involved cumbersome procedures, including verifying fees, clearing library dues, and obtaining signatures, leading to long queues and inefficiencies. The project addresses these challenges by providing a web-based platform where students can log in using their roll number and Aadhaar ID to check and settle any pending fees or fines. Once all dues are cleared, students can instantly download their hall tickets. Additionally, the system automates the CBT application process, enabling students to apply and pay for CBTs online, thus reducing the need for physical visits. The project also includes an admin panel to manage student data, fee structures, and CBT enrollments, enhancing administrative efficiency. By integrating all these functionalities into a single platform, the Hall Ticket Generator aims to significantly reduce administrative overhead and improve the overall student experience during examination periods.

At its core, the system is built using PHP for server-side scripting and HTML/CSS for the front-end interface, which interact seamlessly with a MySQL database. This combination provides a robust and scalable solution for managing diverse aspects of academic administration. The system allows for the efficient handling of student data, including the generation and management of hall tickets, the processing of library book records, and the monitoring of fee payments.

1.2 OBJECTIVES

The primary objective of the Hall Ticket Generator project is to modernize and automate the hall ticket issuance and CBT application processes to enhance efficiency and convenience. Specifically, the project aims to:

- Automate Hall Ticket Issuance: Enable students to access and download their hall tickets online after clearing any pending fees, fines, or library dues, eliminating the need for multiple physical visits and manual processing.
- Streamline Fee and Fine Management: Provide a unified platform for students to check and settle their fee and library dues, integrating payment processing directly into the system to facilitate seamless transactions.
- Simplify CBT Applications: Allow students to apply for CBTs and manage their applications online, reducing the need for physical slips and visits to different administrative blocks.
- Enhance Administrative Efficiency: Offer an admin interface for managing student records, fee structures, CBT applications, and elective course mappings, thus reducing administrative burden and errors.
- Improve User Experience: Provide a user-friendly interface that simplifies the processes of fee payment, library dues management, and hall ticket issuance, making these tasks more efficient and less time-consuming for both students and staff.

Through these objectives, the project seeks to address existing inefficiencies, improve accuracy in fee and fine tracking, and provide a more convenient and streamlined experience for students and administrative staff alike.

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

In the current manual system of hall ticket issuance and CBT management, students face a series of cumbersome and time-consuming steps. At the start of the examination period, students receive a physical slip with fields for roll number, regular fee due, and library dues. To obtain their hall tickets, students must follow a multi-step process:

- Fee Payment Section:
  Students first visit a separate fee payment section located in a different block of the campus. Staff members manually verify the student's roll number against the database to check for any outstanding fees or fines. If there are any dues, the student's slip is not marked as cleared, and the hall ticket is withheld. This requires students to physically present themselves at the fee counter, leading to delays and long queues, especially during peak periods.
- Library Dues Clearance:
  After clearing the fee, students must proceed to the library to address any overdue books or fines. This involves another round of manual verification and clearance, further extending the process.
- Hall Ticket Collection:
  Once both the fee and library dues are cleared, students must finally visit their class teacher to collect the hall ticket, resulting in additional waiting time and administrative overhead.
- CBT Enrollment:
  For CBTs, students must again go through a similar process: collect a slip, pay the CBT fee at the fee section, and then return to the examination block to receive a CBT-specific hall ticket.

Drawbacks of Existing System

- **Inefficiency and Redundancy**: The multiple steps and physical movement between departments create significant delays and inefficiencies.
- **Long Queues and Waiting Times**: Students often face long queues and extended waiting times during peak periods, impacting their overall examination preparation and experience.
- **Administrative Overhead**: Manual handling and verification of student data increase the workload and potential for errors among staff.
- **Lack of Integration**: The absence of a centralized system for managing fees, fines, and CBT applications complicates the process and reduces overall efficiency.

2.2 PROPOSED SYSTEM

The proposed Hall Ticket Generator system aims to address the inefficiencies and drawbacks of the existing manual process by providing an integrated, web-based solution that automates key functions:

- Centralized Platform: Students can access a single platform to check for fee dues, library fines, and CBT enrollment status. This eliminates the need to visit multiple departments, significantly reducing the time and effort required to obtain a hall ticket.
- Automated Fee and Fine Management: The system allows students to view and settle any pending fees or fines online. Integrated payment processing, using services like RazorpayX, ensures that transactions are completed seamlessly and updates are reflected immediately in the system.
- Instant Hall Ticket Download: Once all dues are cleared, students can instantly download their hall tickets from the platform, bypassing the need to collect physical slips from multiple locations.
- Efficient CBT Application: The system enables students to apply for CBTs and make payments online. This streamlines the CBT application process, allowing students to manage their enrollments and receive CBT hall tickets without additional physical paperwork.
- Admin Management Tools: The admin interface allows for efficient management of student records, fee structures, CBT enrollments, and elective courses. This reduces administrative overhead, minimizes errors, and improves data accuracy.
- Enhanced User Experience: By integrating various functionalities into a user-friendly web interface, the system simplifies the process for students and staff, making it more efficient and less stressful.

Benefits of the Proposed System

- Time and Effort Savings: Reduces the need for physical movement and multiple visits, saving time for both students and staff.
- Reduced Administrative Burden: Automates routine tasks and provides an efficient interface for managing records, reducing workload and errors.
- Improved Accuracy: Direct integration with the database ensures accurate and up-to-date information regarding fees, fines, and CBT applications.
- Enhanced Convenience: Offers a centralized, online solution for all relevant processes, making it easier for students to manage their examination-related tasks.

By transitioning to this automated system, the project aims to improve overall efficiency, reduce administrative overhead, and enhance the user experience for both students and staff.

2.3 FEASIBILITY STUDY

2.3.1 Details

The Hall Ticket Generator system is designed to streamline and automate the process of issuing hall tickets and managing CBT applications. It is a web-based application developed using XAMPP, PHP, and MySQL. The system offers functionalities for students to check and clear any pending fees or fines online, apply for CBTs, and download hall tickets directly from the platform. Administrators can manage student records, fee structures, and CBT enrollments through an intuitive web interface. This centralization of processes reduces manual effort, minimizes errors, and provides a more efficient way of handling examination-related tasks.

2.3.2 Impact on Environment

The implementation of the Hall Ticket Generator system has several environmental benefits. By transitioning from a paper-based system to a digital one, the project significantly reduces paper consumption, contributing to a decrease in deforestation and waste production. The digital nature of the system also lessens the need for physical infrastructure and transportation, thereby reducing the carbon footprint associated with administrative processes. Additionally, the time saved by reducing the need for physical movement between departments helps in optimizing resource use and minimizing the overall environmental impact.

2.3.3 Safety

The Hall Ticket Generator system ensures robust security measures to protect data and privacy. It employs secure login mechanisms and encrypted data storage to safeguard sensitive information such as student records and payment details. Access to the admin interface is restricted to authorized personnel, with credentials required for management tasks. The use of secure payment gateways like RazorpayX ensures that financial transactions are handled securely. The system also incorporates regular data backups to prevent loss of information and maintains network security to protect against unauthorized access.

2.3.4 Ethics

The development of the Hall Ticket Generator system adheres to ethical guidelines by prioritizing the security and privacy of users. The application ensures that no personal information is exposed or misused, implementing secure authentication and data protection measures. The system is designed to be user-friendly, avoiding any practices that could harm or inconvenience users. It respects user privacy by encrypting sensitive data and providing secure payment options. The ethical approach extends to ensuring that the system operates without bias and maintains the integrity of the examination process.

2.3.5 Cost

The cost of developing the Hall Ticket Generator system includes expenses related to software development, server hosting, and payment integration. Initial development costs involve setting up the web application, designing the user interface, and integrating payment gateways. Ongoing costs include server maintenance, software updates, and support. However, the system offers cost savings by reducing the need for physical infrastructure, minimizing administrative overhead, and streamlining processes. The efficiency gained through automation can lead to long-term savings in operational costs and reduced manpower requirements.

2.3.6 Type

The Hall Ticket Generator is a web-based application designed to simplify and automate the hall ticket issuance and CBT management processes. It integrates various functionalities into a single platform, providing a centralized solution for both students and administrators. The application is designed to be accessible through standard web browsers, making it compatible with a wide range of devices. Its web-based nature allows for easy updates and maintenance, and it can be accessed from anywhere with an internet connection.

2.3.7 Standards

The Hall Ticket Generator project follows the Agile SDLC model to ensure iterative development and continuous feedback. Agile methodologies enable rapid development cycles and regular updates based on user feedback. For quality management, the project adheres to CMMI (Capability Maturity Model Integration) standards to ensure that the development processes are well-defined and continuously improved. CMMI practices are employed to manage project risks, enhance product quality, and ensure that the application meets user requirements effectively.

2.4 SCOPE OF THE PROJECT

The scope of the Hall Ticket Generator project encompasses the automation of hall ticket issuance and the management of Computer-Based Test (CBT) applications within an academic institution. The primary functionalities include enabling students to:

- Check Fees and Fines: Students can log in to view any pending regular or library fees and fines.
- Make Payments: The system integrates a payment gateway to facilitate online fee payments and update records accordingly.
- Download Hall Tickets: Upon clearing dues, students can download their hall tickets directly from the application.
- Apply for CBTs: Students can view available CBT options, apply for them, and download CBT hall tickets.

For administrators, the scope includes:

- Student and Fee Management: Admins can access and edit student data, manage fee records, and update CBT applications.
- Database Management: Admins can manage various databases related to student records, fee structures, and CBT enrollments.
- Customizable Data: Admins can set parameters for issuing hall tickets and managing CBTs based on semester and year.

2.5 SYSTEM CONFIGURATION

The Hall Ticket Generator system operates in a web-based environment, configured as follows:

- Server: The application is hosted on a local XAMPP server, which includes Apache (for web server functionality), MySQL (for database management), and PHP (for server-side scripting).
- Database: MySQL is used to manage student records, fee data, CBT applications, and other relevant information. The database schema is designed to support efficient querying and updates.
- Web Technologies: The frontend of the application is developed using HTML, CSS, and JavaScript to provide a user-friendly interface. Backend functionalities are implemented using PHP.
- Payment Integration: RazorpayX is integrated for secure online payments. The payment gateway is configured in test mode during development to simulate transactions.
- PDF Generation: The application uses the mPDF library to generate and serve hall tickets and CBT documents in PDF format.
- Access: The application is accessible via standard web browsers, making it compatible with various devices. Users need to have valid credentials to access different functionalities of the system.

This configuration ensures that the application is robust, secure, and capable of handling the automation of hall ticket issuance and CBT management efficiently.

3. LITERATURE OVERVIEW

1."Automated Exam Management Systems: Innovations and Trends"

- Description: This paper reviews the latest advancements in automated exam management systems, focusing on innovations that streamline processes such as hall ticket issuance, fee management, and result processing. It highlights emerging trends and technologies that enhance system efficiency and user experience.
- Problem Identified: Traditional exam management systems face challenges such as inefficiency and administrative overhead, which automation aims to address.
- Publication Date: April 2023

2."Advancements in Online Payment Integration for Educational Institutions"

- Description: This study examines recent advancements in online payment systems integrated into educational institutions. It discusses new payment gateways, integration techniques, and their impact on improving the fee collection process and reducing administrative burden.
- Problem Identified: Outdated fee collection methods create inefficiencies and errors in processing payments and managing hall ticket issuance.
- Publication Date: September 2023

3. "Web-Based Solutions for Modern Student Services: A Comprehensive Review"

- Description: The paper provides a comprehensive review of recent web-based solutions designed to enhance student services, including fee management and hall ticket issuance. It analyzes how these systems address problems encountered in traditional methods and their impact on service delivery.
- Problem Identified: Traditional methods of managing student services are often inefficient and problematic, which web-based systems aim to resolve.
- Publication Date: June 2023

4."Ensuring Data Privacy and Security in Online Academic Systems: Recent Approaches"

- Description: This research focuses on contemporary approaches to securing online academic systems, with an emphasis on data privacy and transaction security. It outlines recent techniques and best practices for safeguarding sensitive student information.
- Problem Identified: Protecting sensitive data and ensuring secure transactions in online academic systems is a significant concern.
- Publication Date: February 2024

5."Optimizing Computer-Based Testing Systems: Current Strategies and Future Directions"

- Description: The paper explores current strategies for optimizing computer-based testing systems, highlighting improvements in automation and efficiency. It also discusses future directions and potential advancements in CBT technology.
- Problem Identified: Manual CBT processes can be inefficient and prone to errors, necessitating optimization through automation.
- Publication Date: July 2023

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

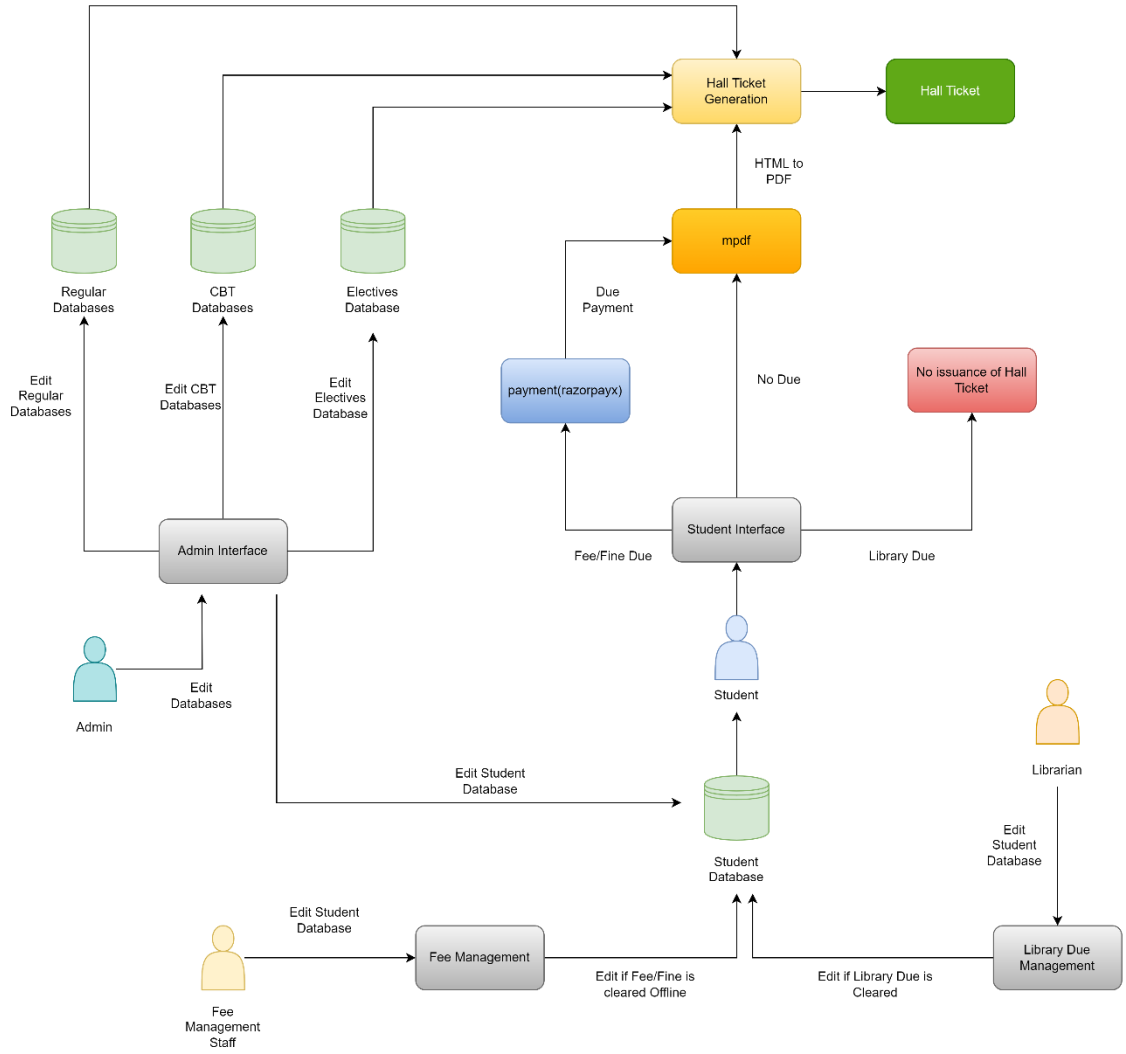

Fig 4.1 System Architecture

The system architecture for the Hall Ticket Generator project is designed to integrate multiple modules for handling hall ticket generation, fee management, library due management, and CBT registration processes. The architecture follows a client-server model where different users interact with the system through distinct interfaces, each serving specific functionalities. The diagram provides a visual representation of the overall system flow and its components.

Key Components of the System Architecture

Database Layer:

- The system architecture incorporates several databases (1-1, 1-2, 2-1, 2-2, 3-1, 3-2, 4-1, 4-2, cbt1-1, cbt1-2, etc.) that store information about the subjects, students, their dues, CBT registrations, and electives.
- Each semester has its dedicated database, which includes two tables (cse and ece) to store subject-related information such as subject_id, subject_name, date_of_exam, and time_of_exam.
- The electives database manages the open and professional electives chosen by students.
- The x database includes tables like slct for semester selection and std for student information, which is critical for verifying dues before hall ticket issuance.

User Interfaces:

- The system architecture supports three main user interfaces:
- Admin Interface: Allows administrators to manage all aspects of the application, including student records, subject details, and elective management. Admins can perform CRUD operations on all databases and update fee and library dues.
- Student Interface: Allows students to log in and perform actions such as viewing dues, making payments, selecting electives, and downloading hall tickets. Students interact with the system to apply for CBTs and view their exam schedules.
- Staff Interface: Provides staff members access to specific functionalities like fee and library management. They can update dues, verify payments, and ensure students' records are accurate.

Payment Gateway Integration:

- The system is integrated with Razorpay for secure online payment processing (in test mode). When a student has outstanding dues, the system prompts them to make the necessary payments through Razorpay. Upon successful payment, the respective dues are cleared in the database.

PDF Generation:

- The "mpdf" library is used for generating hall tickets in PDF format. Once a student clears all dues, the system fetches the relevant data from the database and creates a downloadable PDF file that serves as the hall ticket for the exams.

Processing Modules:

- Hall Ticket Generation: This module is responsible for generating hall tickets for students who have cleared all their dues. The hall tickets are produced in PDF format using the "mpdf" library.
- CBT Management: Manages student registration for CBTs. The system allows students to register for specific CBT subjects, tracks their registration status, and issues hall tickets specifically for CBT exams.
- Fee and Library Due Management: Modules to manage and update students' fee and library dues. These modules ensure that dues are recorded accurately, and updates are reflected promptly.

Workflow:

- When a student logs in, the system first checks for any pending dues via the std table. If dues exist, they are prompted to pay through the Razorpay payment gateway. Upon successful payment, the system updates the due status, and the hall ticket is generated using the "mpdf" library.
- If there are no dues, the hall ticket is generated directly.
- Admins and staff have additional privileges to manage student data and verify payments manually if needed.

4.1.1 Module Description

The Hall Ticket Generator project is structured into three main modules—Regular Module, CBT Module, and Admin Module—each of which is composed of various sub-modules that handle distinct functionalities. The modular architecture ensures a streamlined process for managing different academic processes, including fee payment, hall ticket generation, CBT management, and administrative tasks.

1. Index Module

Purpose: Acts as the entry point to the system.

Description: The Index Module serves as the main landing page that directs users to their respective modules based on their roles (Student, Staff, or Admin). It provides a seamless user experience by routing them to the appropriate functionalities they need to access.

2. Regular Module

Purpose: Manages all regular academic-related processes.

Sub-Modules:

- Student Module: This sub-module is specifically designed for student interactions with the system. It allows students to:
  View and pay any pending dues through the Payment Module, which integrates with payment gateways (like Razorpay) for seamless transactions.
  Generate hall tickets using the Hall Ticket Generation Module after confirming that all dues have been cleared.
- Staff Module: Tailored for faculty and staff members to manage dues and other administrative functions. It includes:
- Fee Module: Allows staff to update and manage fee-related information for students.
- Library Module: Enables staff to manage library dues, update book returns, and fines, ensuring all dues are cleared before students can generate hall tickets.

3. CBT Module

Purpose: Facilitates the management of Computer-Based Tests (CBT).

Sub-Modules:

- Student Module: Focuses on students' CBT-specific interactions. It allows them to:

View eligible subjects and register for CBT exams.

Make payments specific to CBT through the Payment Module.

Generate hall tickets for registered CBT subjects using the Hall Ticket Module, which becomes available only after successful registration and payment.

4. Admin Module

Purpose: Provides comprehensive control and management capabilities for system administrators.

Sub-Modules:

- Edit Regular Database: Allows admins to modify and update information related to regular academic processes, such as student details and subjects.
- Edit CBT Database: Enables administrators to manage and update CBT-specific records, ensuring accurate and up-to-date information.
- Manage Electives: Facilitates the management of elective subjects, allowing admins to update available options and track student selections for professional and open electives.
- Manage Student Database: Provides full access to the student database, allowing for the management of student records, including personal information, academic status, dues, and more.

4.2 UML DIAGRAMS

Use Case Diagram

A use case diagram for the Hall Ticket Generator outlines the main functionalities available to different users within the system, represented as actors such as students, staff, and admin. The diagram visually maps out the interactions between users and the system's functionalities, such as requesting hall tickets, making payments, applying for CBT, and managing electives or databases. It provides a clear view of the use cases that each actor can perform, representing how the system meets the different needs of its users. By showing how these actors engage with specific actions, the diagram simplifies the user interaction process, ensuring that all key functionalities are clearly defined.

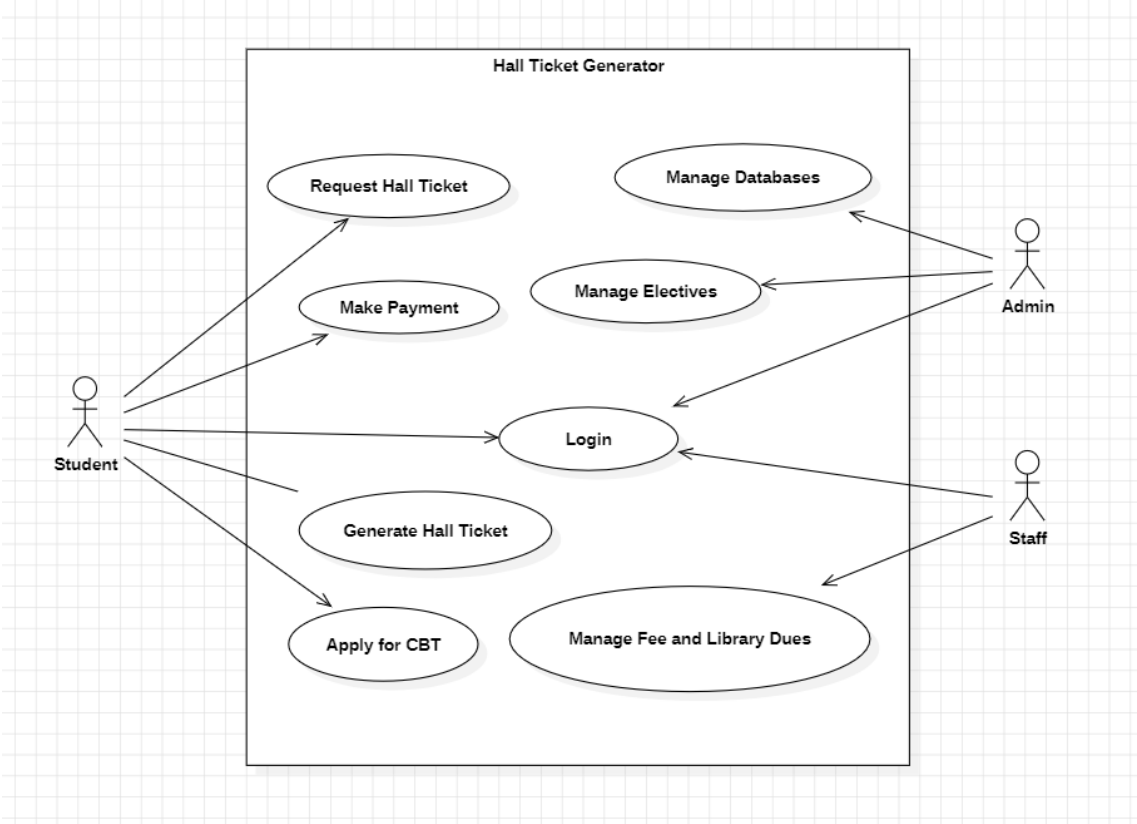

Fig 4.2.1 Use Case Diagram

Sequence Diagram

A sequence diagram for the Hall Ticket Generator depicts the time-ordered interactions between different system components during a student's request for a hall ticket. It begins with the student logging in, followed by validation of the login credentials by the system. The student then requests a hall ticket, prompting the system to check for any dues. If dues exist, the payment process is initiated, and the status is updated accordingly. Once the payment is cleared, the hall ticket is generated and displayed. This diagram emphasizes the interaction between the student, hall ticket generator, payment gateway, and database in a step-by-step manner, showing how the system ensures a smooth transition from login to hall ticket issuance.

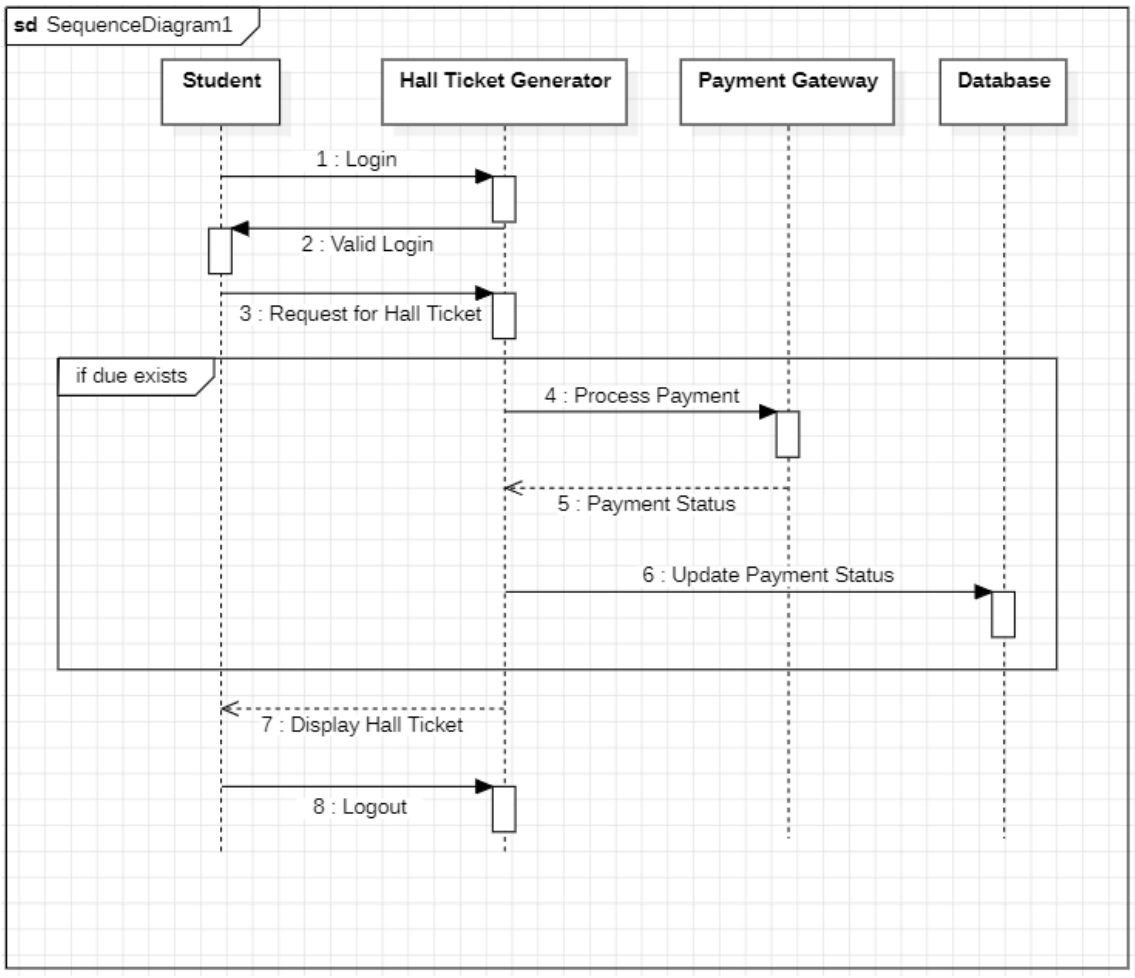

Fig 4.2.2 Sequence Diagram

Class Diagram

A class diagram for the Hall Ticket Generator system shows the structural components of the system, including key entities such as User, Student, Staff, Admin, and other relevant classes like Payment, Hall Ticket, Elective Management, and CBT. This diagram defines how each class is related to the others through associations and dependencies. It also highlights the main attributes and operations within each class, which represent the functionality of the system. For instance, students can request hall tickets or make payments, while staff can update dues and admins manage databases. Overall, the class diagram provides a blueprint of the system, defining its static structure and how different components interact to perform various tasks.

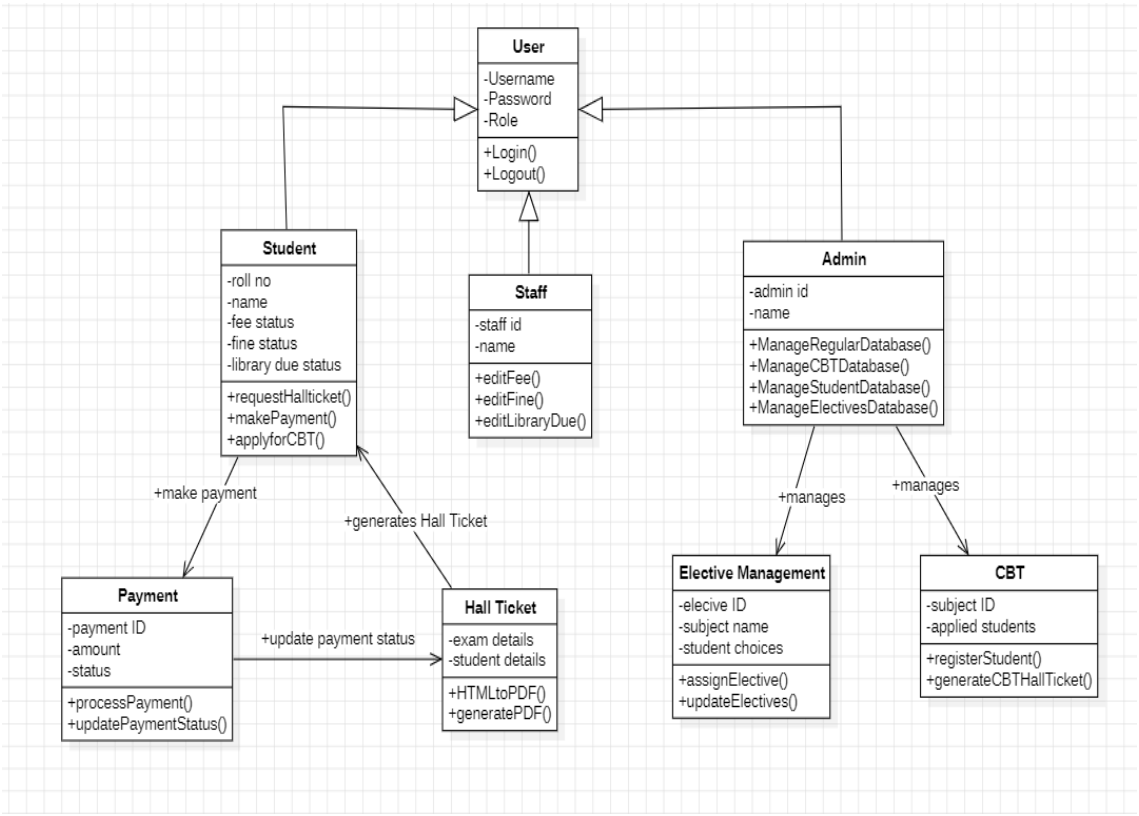

Fig 4.2.3 Class Diagram

Activity Diagram

An activity diagram for the Hall Ticket Generator shows the dynamic flow of the system, representing the sequence of actions taken by different user roles such as students, staff, and admin. It begins with login activities for each role and branches into specific actions depending on whether the login is valid or invalid. For students, the diagram details how due verification and payment processing occur before the hall ticket is generated. For staff, it focuses on updating dues, while admins manage databases. This diagram provides a clear view of the system's workflow, ensuring each user follows the necessary steps to complete their assigned tasks, thus reflecting the overall process of generating hall tickets.

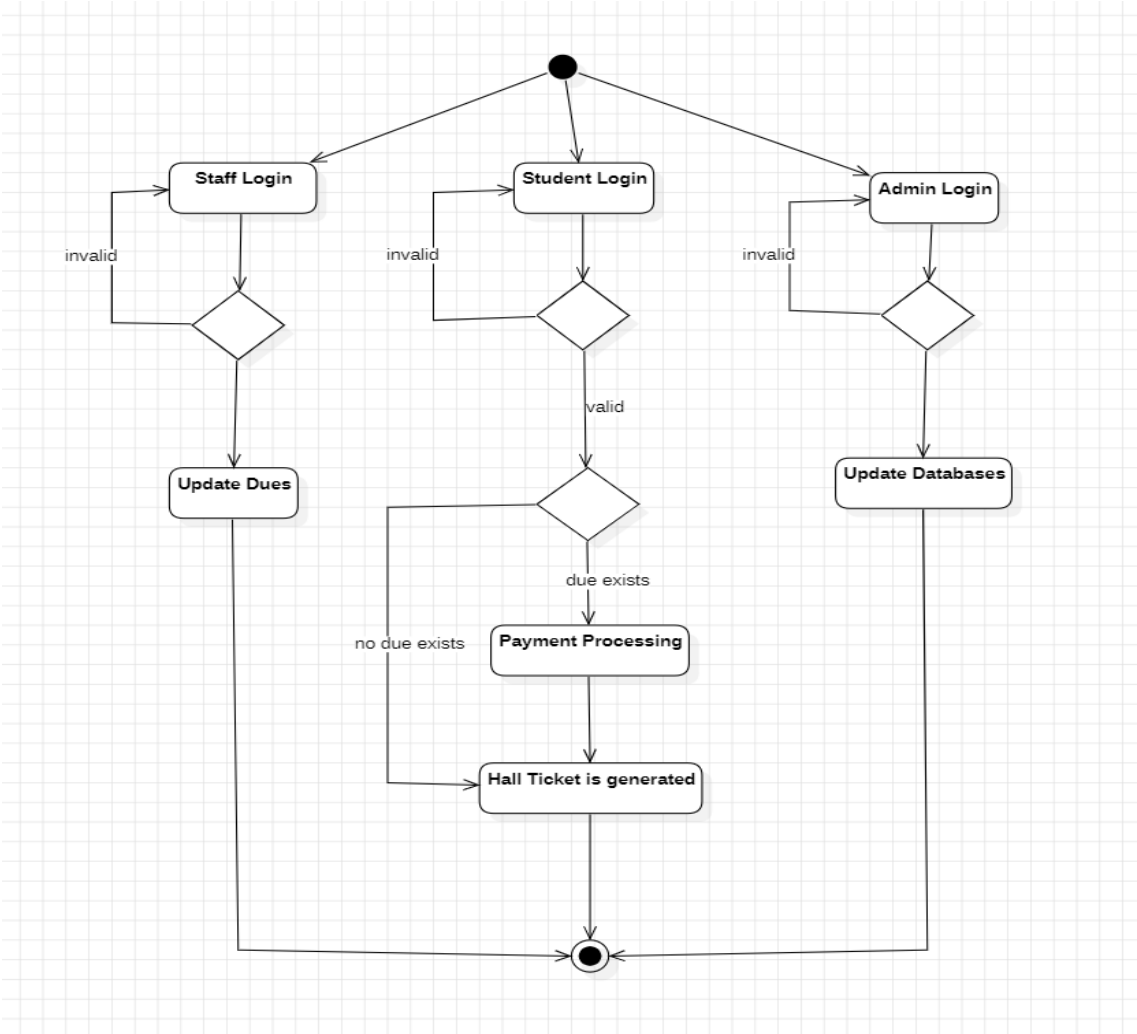

Fig 4.2.4 Activity Diagram

4.3 SYSTEM DESIGN

4.3.1 Modular Design

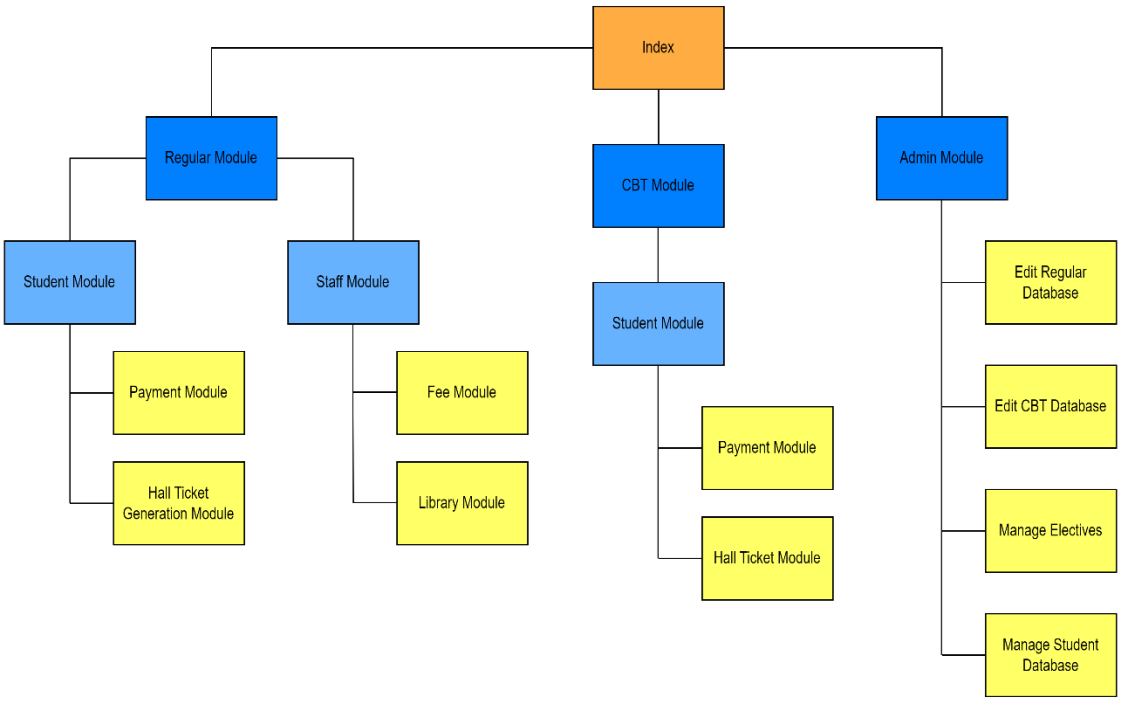

Fig: 4.3 Modular Design

The Hall Ticket Generator project is built using a modular design approach that divides the system into distinct modules, each responsible for specific functionalities. This modular architecture ensures scalability, maintainability, and easy integration of new features or updates. The system is organized into three main modules: Regular Module, CBT Module, and Admin Module, each comprising several sub-modules that handle different tasks within the application.

Key Modules and Their Descriptions:

Index Module:

The Index module serves as the entry point for the application. It provides the initial interface from which users (students, staff, and admins) can navigate to their respective modules. It acts as the central routing point, guiding users based on their roles and required functionalities.

Regular Module:

The Regular Module is designed to handle all regular academic activities and processes for both students and staff. It includes the following sub-modules:

1. Student Module:

Provides functionalities specific to students, such as viewing dues, making payments, and generating hall tickets.

Includes:

- Payment Module: Integrates with a payment gateway (e.g., Razorpay) for handling fee and other due payments. Once dues are cleared, the system updates the database accordingly.
- Hall Ticket Generation Module: Generates hall tickets for students after they clear all dues. The hall tickets are generated in PDF format using libraries like "mpdf".

2. Staff Module:

Allows staff members to manage student-related activities, such as updating fee and library dues.

Includes:

- Fee Module: Provides functionalities to update and manage fee dues for students.
- Library Module: Manages library dues, ensuring that books are returned on time and any fines are updated in the system.

CBT Module:

The CBT Module is specifically designed to handle Computer-Based Test (CBT) registrations and processes.

1. Student Module:

Similar to the regular module but focused on CBT-specific tasks. Students can view the CBT subjects they are eligible for and register for CBT exams.

Includes:

- Payment Module: Handles payments specifically for CBT registrations, ensuring students have cleared any dues related to CBT.
- Hall Ticket Module: Generates CBT-specific hall tickets for students who have completed the registration and payment process.

Admin Module:

The Admin Module provides comprehensive control over the entire system, allowing administrators to manage databases and configurations.

Includes:

- Edit Regular Database: Allows admins to edit information related to regular academic activities, such as subject details etc.
- Edit CBT Database: Allows modification and management of CBT-specific data.
- Manage Electives: Admins can manage professional and open electives, ensuring students have selected their choices correctly.
- Manage Student Database: Provides full access to manage and update all student-related information.

4.3.2 Database Design

The database design for the Hall Ticket Generator project is tailored to manage diverse aspects of student records, exam schedules, and elective courses. It is organized into several databases and tables to facilitate the efficient storage and retrieval of information:

Semester Databases:

- Databases: 1-1, 1-2, 2-1, 2-2, 3-1, 3-2, 4-1, 4-2
- Tables: cse, ece
- Columns: subject_id, subject_name, date_of_exam, time_of_exam
- Purpose: These databases store the subject information for Computer Science (CSE) and Electronics and Communication (ECE) streams for each semester. They include details about the subjects, exam dates, and times.

CBT Databases:

- Databases: cbt1-1, cbt1-2, cbt2-1, cbt3-1, cbt3-2, cbt4-1, cbt4-2
- Tables: cse, ece
- Columns: subject names from corresponding tables from semester database's tables
- Purpose: These databases manage CBT (Computer-Based Test) registrations for various subjects and semesters. Each table indicates whether a student (identified by roll number) has applied for a CBT in specific subjects.

Elective Databases:

- Databases: electives
- Tables:
  oes (Open Electives): id, subject_name
  oe (Open Electives Student Choices): roll_number, subject_id (all available subjects in oes)
  pes (Professional Electives): id, subject_name
  pe (Professional Electives Student Choices): roll_number, subject_id (all available subjects in pes)

- Purpose: These tables store information about open and professional electives, including available subjects and student choices for each elective category.

General Student and Exam Management Database:

- Database: x
- Tables:

  slct (Semester Selection): year, sem (tracks the current semester and year)

  std (Student Data): roll_number, password, fee_due, library_due, fine_due (stores student details including fees, fines, and library dues)
- Purpose: This database manages current semester information and student records, ensuring accurate tracking of dues and access to hall tickets.

5. IMPLEMENTATION

5.1 IMPLEMENTATION

The Hall Ticket Generator project is designed to automate the cumbersome process of hall ticket issuance and CBT (Computer-Based Test) management in educational institutions. The system is developed as a web-based application using PHP, MySQL, HTML, CSS, and JavaScript, which allows students, staff, and administrators to manage exam-related tasks seamlessly.

The project is structured into three main modules:

- Regular Hall Ticket Generation Module:
  This module allows students to log in using their roll numbers and a password generated using their Aadhaar ID. Upon login, the system checks for any outstanding fees, fines, or library dues stored in the std table of the x database. If there are no dues, the student is presented with an option to download the hall ticket as a PDF generated using the "mpdf" library. If there are dues, the student can pay them online using the Razorpay payment integration (in test mode). Upon successful payment, the dues are updated in the database, and the student is then allowed to download the hall ticket.
- CBT Management Module:
  This module manages the registration and hall ticket generation process for Computer-Based Tests (CBTs). Students can log in and view available CBT subjects and those they have already registered for. They can select the subjects they wish to apply for CBTs and make the necessary payments through the same Razorpay integration. Once payment is confirmed, a CBT hall ticket is generated and made available for download.
- Admin and Staff Management Module:
  The admin has full control over the application, including managing the databases and tables related to student details, subjects, and electives. The admin can add, edit, or delete data from the databases through a user-friendly interface. The staff members, on the other hand, have access to verify and update student dues in both fee and library sections. This is crucial for offline payments or adjustments.

Datasets Used

The project uses several MySQL databases and tables to manage different aspects of student and exam data:

- Semester Databases (1-1, 1-2, 2-1, 2-2, 3-1, 3-2, 4-1, 4-2):
  Each database contains two tables (cse and ece), corresponding to the two main streams.
  Tables store subject information, including subject_id, subject_name, date_of_exam, and time_of_exam.
- CBT Databases (cbt1-1, cbt1-2, cbt2-1, cbt3-1, cbt3-2, cbt4-1, cbt4-2):
  These databases track which students have applied for CBTs in different subjects.
  They contain tables (cse and ece) that map roll_number to subject_id and an applied status indicating whether the student has registered for CBT in a particular subject.
- Electives Database (electives):
  Contains tables like oes, oe, pes, and pe to manage elective courses.
  Stores available elective subjects and the respective choices of students.
- General Database (x):
  Includes tables like slct (current semester tracking) and std (student details, including fee_due, library_due, and fine_due).

5.2 SAMPLE CODE

1. Home Page

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Hall Ticket Generation System</title>

  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <style>

/* CSS styling */

</style>

</head>

<body>

<header class="header">

  <h1>Hall Ticket Generator</h1>

</header>

<div class="container text-center">

  <h2>Welcome to Hall Ticket Generation System</h2>

  <div class="row">

    <div class="col-sm-4">

      <a href="home.html" class="btn btn-option">
```

```
        <img src="https://cdn3.iconfinder.com/data/icons/education-
756/140/clipboard__paper__exams__education__sheet-128.png" alt="Regular">
        Regular
      </a>
    </div>
    <div class="col-sm-4">
      <a href="cbtstd.php" class="btn btn-option">
        <img src="https://cdn2.iconfinder.com/data/icons/education-and-learning-
47/60/question__quiz__exams__education__online-128.png" alt="CBT">
        CBT
      </a>
    </div>
    <div class="col-sm-4">
      <a href="adm.php" class="btn btn-option">
        <img src="https://cdn0.iconfinder.com/data/icons/network-technology-
19/64/systems-admin-owner-personal-programmer-128.png" alt="Admin">
        Admin
      </a>
    </div>
  </div>
</div>
<footer class="footer">
  <p>GCET</p>
</footer>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></scrip
t>

<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>
```

2. Regular Page

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Hall Ticket Generation System</title>

  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

  <style>

/* CSS styling */

</style>

</head>

<body>

  <header class="header">

    <h1>Welcome to Regular Hall Ticket Generation System</h1>

  </header>
```

```
 <div class="container text-center">
  <div class="row justify-content-center">
   <div class="col-md-6">
    <a class="btn btn-option" id="x" href="std.php">
     <img src="https://cdn0.iconfinder.com/data/icons/streamline-emoji-1/48/130-man-student-2-128.png">Student
    </a>
   </div>
   <div class="col-md-6">
    <a href="staff.html" class="btn btn-option" id="x">
     <img src="https://cdn3.iconfinder.com/data/icons/business-654/64/Personnel-staff-organization-company-user-networking-people-business-128.png">Staff
    </a>
   </div>
  </div>
  <a href="h1.html" class="btn small-button">Back</a>
 </div>
 <footer class="footer">
  <p>GCET</p>
 </footer>
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
```

```
 <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
```

3. Student Page

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Home</title>
   <meta charset="UTF-8" />
   <meta name="viewport" content="width=device-width" />
   <link rel="stylesheet" href="styles.css" />
</head>
<body>
   <div class="container">
      <div class="center">
         <h1>Login</h1>
         <form method="post" action="log.php">
            <div class="txt_field">
               <input type="text" name="roll" required>
               <span></span>
               <label>Roll Number</label>
            </div>
            <div class="txt_field">
```

```
                <input type="password" name="pass" required>
                <span></span>
                <label>Password</label>
            </div>
            <input name="submit" type="Submit" value="submit">
            <br>
             
        </form>
        <?php
        if (isset($_GET['error']) && $_GET['error'] == 1) {
            echo "<p style='color: red;text-align:center'>Wrong credentials. Please try
again.</p>";
        }
        ?>
      </div>
  </div>
</body>
</html>
```

4. Payment Page

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Due Payment</title>
    <style>
/* CSS styling */
</style>
</head>
<body>
    <div class="container">
        <?php
        session_start();
        $a = $_SESSION['fee'];
        $b = $_SESSION['lib'];
        $c = $_SESSION['roll'];
        $d = $_SESSION['fine'];
        echo "<h1>Hello, " . $c . "</h1>";
        if ($a > 0 || $d > 0) {
            echo "<div class='alert'><p>You have a due of " . ($a + $d) . " Rs (including both fee and library fine).</p>";
            echo "<a href='pay.php'>Click here to pay the due</a></div>";
        }
        if ($b > 0) {
            echo "<div class='info'><p>Please return the " . $b . " books to the library to download your hall ticket.</p></div>";
        }
        ?>
```

```
    </div>
</body>
</html>
```

5. Hall Ticket Generation Page

```
<?php
require_once __DIR__ .'/vendor/autoload.php';
$servername = "localhost";
$username = "root";
$password = "";
session_start();
$r=$_SESSION['roll'];
$s=substr($r,0,2);
$cy=date('Y');
$cys=substr((string)$cy,2,2);
$yr=(int)$cys-(int)$s;
$s=substr($r,4,1);
if($s=='5'){
    $yr+=1;
}
$conn = new mysqli($servername, $username, $password, 'x');
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```
$sql = "SELECT sem FROM slct where year=$yr";
$result = $conn->query($sql);
$row = $result->fetch_assoc();
$dbname = $row['sem'];
$s=substr($r,7,1);
$y = "cse";
if($s=='4'){
    $y="ece";
}
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT * FROM $y";
$result = $conn->query($sql);
$ticket_html = '
<!DOCTYPE html>
<html>
<head>
    <title>Hall Ticket</title>
    <style>
        .ticket {
            border: 2px solid black;
```

```
    padding: 20px;
    margin: 20px;
}
.ticket-header {
    text-align: center;
    margin-bottom: 20px;
}
.ticket-body {
    margin-bottom: 20px;
}
.ticket-table {
    width: 100%;
    border-collapse: collapse;
}
.ticket-table th, .ticket-table td {
    border: 1px solid black;
    padding: 8px;
    text-align: center;
}
.ticket-table th {
    background-color: #f2f2f2;
}
.ticket-footer {
```

```
            text-align: center;
            margin-top: 20px;
        }
    </style>
</head>
<body>
<div class="ticket">
    <div class="ticket-header">
        <h2>Hall Ticket</h2>
    </div>
    <div class="ticket-body">
        <p><strong>Roll Number:</strong> ' . $r . '</p>
        <table class="ticket-table">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Subject</th>
                    <th>Date</th>
                    <th>Time</th>
                </tr>
            </thead>
            <tbody>';
if ($result->num_rows > 0) {
```

```
while ($row = $result->fetch_assoc()) {
    $start_time = date("H:i", strtotime($row['time']));
    $end_time = date("H:i", strtotime($row['time'] . "+3 hours"));
    $subj=$row['subject'];
    $pid=$row['id'];
    $xs=substr($subj,0,1);
    if($xs=='p' || $xs=='o'){
        $tab="pe";
        if($xs=='o'){
            $tab="oe";
        }
        $con = new mysqli($servername, $username, $password,"electives");
        if ($con->connect_error) {
         die("Connection failed: " . $con->connect_error);
        }
        $sqlq="SELECT * from $tab where roll='$r' ";
        $result1 = $con->query($sqlq);
        $row1 = $result1->fetch_assoc();
        $subid=$row1[$subj];
        $tab="pes";
        if($xs=='o'){
            $tab="oes";
        }
```

```
            $sqlq="SELECT * from $tab where id='$subid' ";
            $result1 = $con->query($sqlq);
            $row1 = $result1->fetch_assoc();
            $subj=$row1['subject'];
            $con->close();
        }
        $ticket_html .= '
        <tr>
            <td>' . $pid . '</td>
            <td>' . $subj . '</td>
            <td>' . $row['date']. '</td>
            <td>' . $start_time . ' - ' . $end_time . '</td>
        </tr>';
    }
} else {
    $ticket_html .= '
    <tr>
        <td colspan="4">No records found</td>
    </tr>';
}
$ticket_html .= '
            </tbody>
        </table>
```

```
    </div>
    <div class="ticket-footer">
        <p>&copy; ' . date("Y") . ' GCET. All rights reserved.</p>
    </div>
</div>
</body>
</html>';
$mpdf = new \Mpdf\Mpdf();
$mpdf->WriteHTML($ticket_html);
$mpdf->Output();
?>
```

6. Staff Page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Staff Section</title>
    <style>
/* CSS styling */
 </style>
</head>
<body>
    <header>
```

```
        <h1>Staff Section</h1>
    </header>
    <main>
        <div class="button-container">
            <a href="fee.php" class="big-button">
                <img src="https://cdn1.iconfinder.com/data/icons/business-bw-vol-2/100/business-10-31-128.png" alt="Fee Icon" width="40">
                Fee
            </a>
            <a href="lib.php" class="big-button">
                <img src="https://cdn2.iconfinder.com/data/icons/school-supplies-7/64/164_school-open-study-education-book-library-128.png" alt="Library Icon" width="40">
                Library
            </a>
        </div>
        <a href="home.html" class="small-button">Back</a>
    </main>
    <footer>
        <p>GCET</p>
    </footer>
</body>
</html>
```

7. Fee Management Page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Fee and Fine Management</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <style>
/* CSS Styling */
</style>
</head>
<body>
    <header>
        <h1>Student Fee and Fine Management</h1>
    </header>
    <main>
        <div class="search-container">
            <form id="search-form">
                <label for="roll">Enter Roll Number:</label>
                <input type="text" id="roll" name="roll" required>
                <button type="submit">Search</button>
            </form>
```

```
        </div>
        <div class="result-container" id="result-container">
            <!-- Result will be displayed here -->
        </div>
        <a href="staff.html" class="btn btn-secondary">Back</a>
    </main>
    <footer>
        <p>GCET</p>
    </footer>
    <script>
        document.getElementById('search-form').addEventListener('submit',
function(event) {
            event.preventDefault();
            let roll = document.getElementById('roll').value;
            fetch(`feesearch.php?roll=${roll}`)
                .then(response => response.json())
                .then(data => {
                    if (data.success) {
                        document.getElementById('result-container').innerHTML = `
                            <form id="update-form">
                                <label>Roll: <input type="text" name="roll" value="${data.roll}"
readonly></label>
                                <label>Fee: <input type="text" name="fee"
value="${data.fee}"></label>
```

```
                    <label>Fine: <input type="text" name="fine"
value="${data.fine}"></label>
                    <button type="submit">Update</button>
                </form>
            `;
            document.getElementById('result-container').style.display = 'flex';
            document.getElementById('update-form').addEventListener('submit',
function(event) {
                event.preventDefault();
                let formData = new FormData(this);
                fetch('feeupdate.php', {
                    method: 'POST',
                    body: formData
                })
                .then(response => response.json())
                .then(data => {
                    if (data.success) {
                        alert('Record updated successfully');
                    } else {
                        alert('Failed to update record');
                    }
                });
            });
        } else {
```

```
                alert('No record found');
            }
        });
    });
    </script>
</body>
</html>
```

6. TESTING

6.1 TESTING

Testing is a critical phase in the development of the Hall Ticket Generator project to ensure the system functions as intended and meets the requirements specified. The testing process includes various levels and types of testing to validate the functionalities, performance, security, and usability of the system.

Unit Testing:

- Each module (e.g., Student Module, Staff Module, Admin Module, Payment Module) was tested individually to ensure that every function within the modules worked as expected.
- Example: Testing the login functionality for students, staff, and admin separately to verify correct input validation and authentication.

Integration Testing:

- After unit testing, modules were integrated, and tests were performed to check the interaction between different modules.
- Example: Integrating the Student Module with the Payment Module and verifying that payment status correctly updates the hall ticket generation process.

Functional Testing:

- Conducted to ensure that the system performs according to the specified functional requirements.
- Example: Testing the hall ticket generation process under various scenarios like “no dues” or “after dues are cleared.”

System Testing:

- The complete system was tested to validate that it meets the overall system requirements.

- Example: Testing the entire process from student login, due clearance, payment, to hall ticket download.

User Acceptance Testing (UAT):

- The system was tested with a group of end users (students, staff, and admins) to validate that it meets their expectations and is easy to use.
- Example: Testing by real users to ensure that the process is streamlined and the interface is intuitive.

Performance Testing:

- Tested to ensure the system's responsiveness, stability, and scalability under expected load conditions.
- Example: Testing how the system handles concurrent logins and multiple payment requests during peak times.

Security Testing:

- Conducted to identify any potential vulnerabilities and to ensure that user data is secure.
- Example: Testing for SQL injection, cross-site scripting (XSS), and ensuring secure password management.

Regression Testing:

- After any changes or enhancements, regression testing was conducted to ensure that the new changes did not adversely affect the existing functionalities.
- Example: After integrating RazorpayX for payment, testing the entire flow to ensure previous functionalities remain intact.

6.2 TEST CASES

S.No	Test Case Description	Expected Result	Status
1	Verify login with valid student credentials	Student is successfully logged in.	Success
2	Verify login for Admin	Admin is successfully logged in	Success
3	Verify login for Staff	Staff is successfully logged in.	Success
4	Verify payment processing.	Payment is processed successfully.	Success
5	Verify updating payment status after successful transaction	Hall ticket is generated and available for download in PDF format	Success
6	Verify hall ticket generation for students with no dues	Hall ticket is generated and available for download in PDF format	Success
7	Verify no hall ticket generation for students with pending dues	Message displayed: "Clear dues to generate hall ticket"	Success
8	Verify hall ticket generation after dues are cleared	Hall ticket is generated and available for download after dues are cleared	Success
9	Verify admin can add a new student	Student is added successfully	Success
10	Verify admin can edit student details	Student details are updated successfully	Success
11	Verify admin can delete a student record	Student record is deleted successfully	Success
12	Verify applying for a CBT (Computer-Based Test)	Subjects are successfully applied for CBT	Success
13	Verify CBT status update after application	CBT status is updated in the database	Success

7. OUTPUT SCREENS

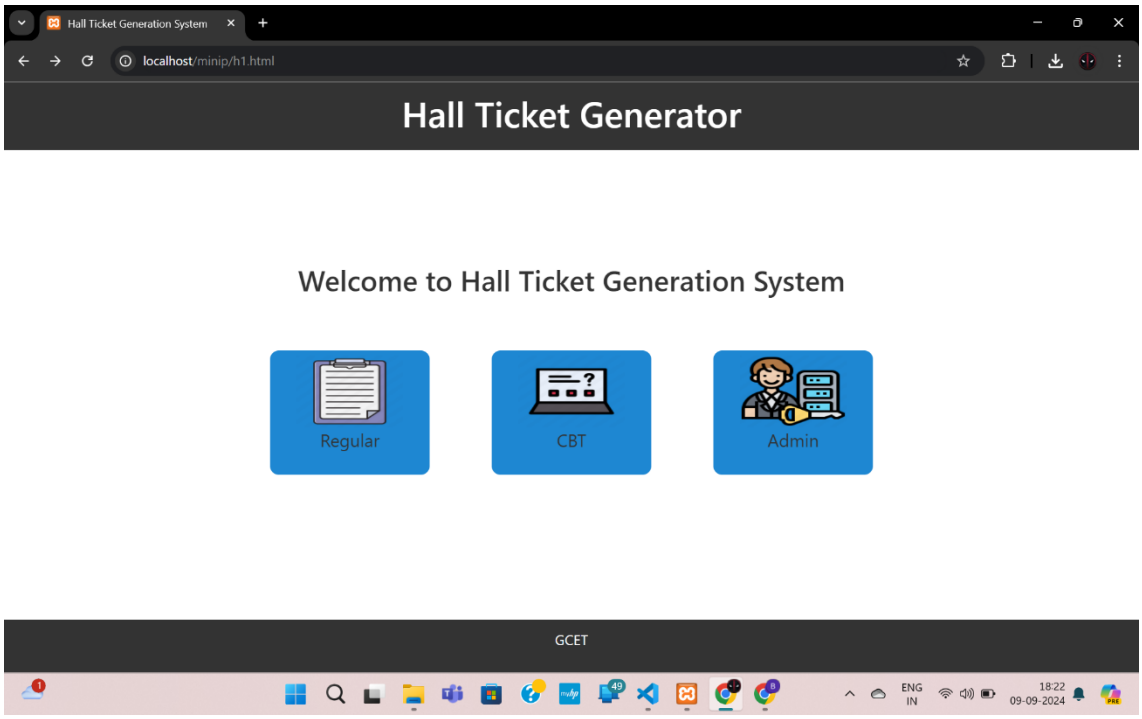

Fig 7.1 Home Page

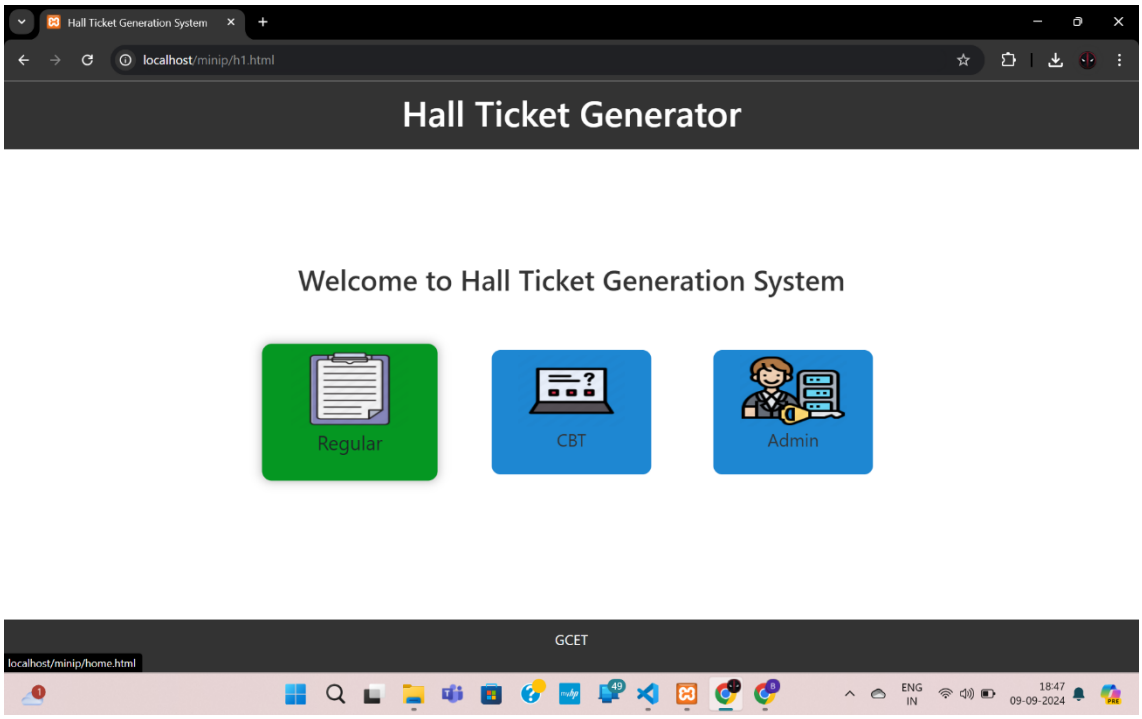

Fig 7.2 Selecting Regular

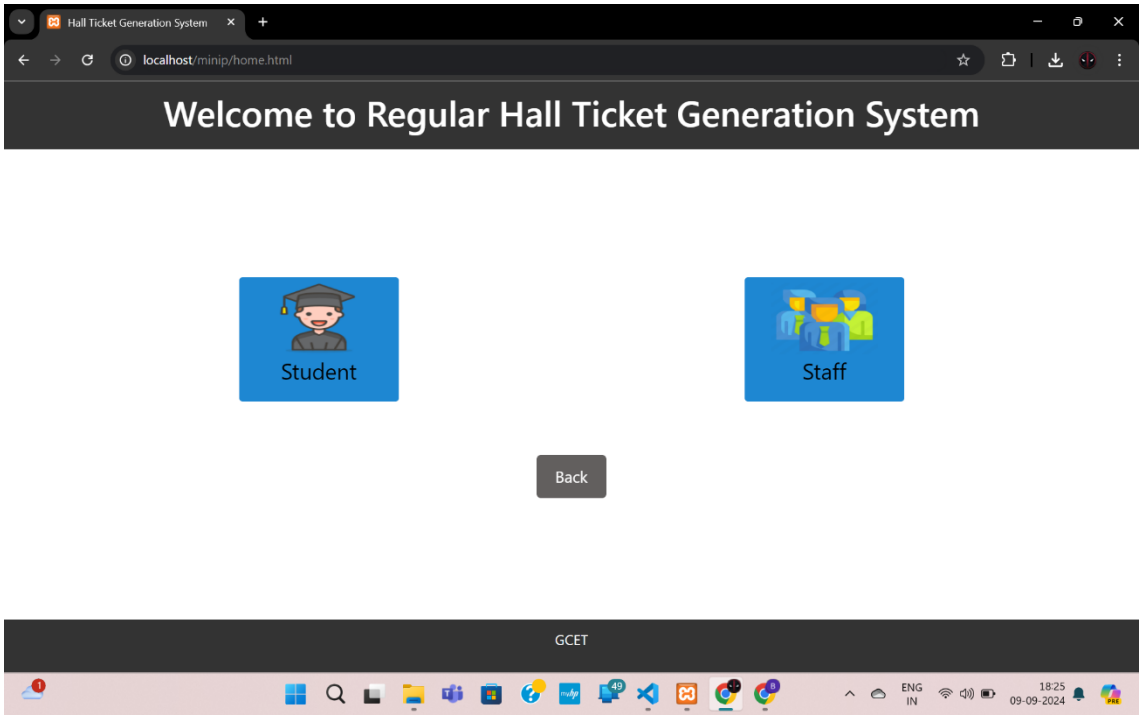

Fig 7.3 Regular Page

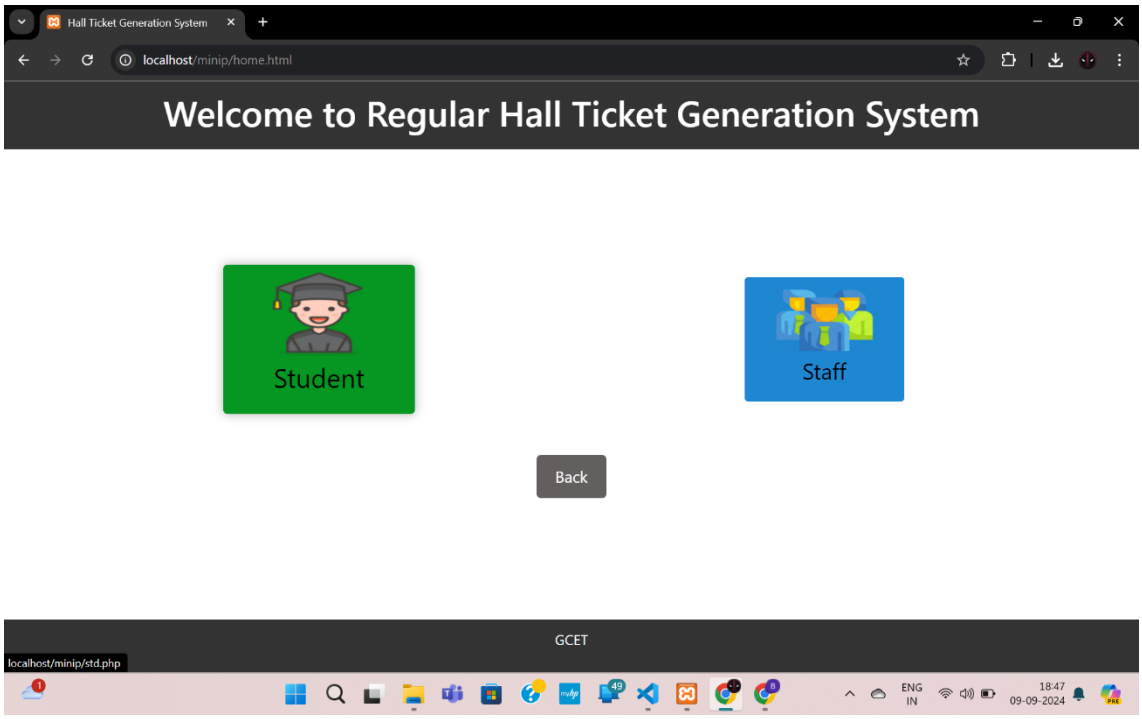

Fig 7.4 Selecting Student

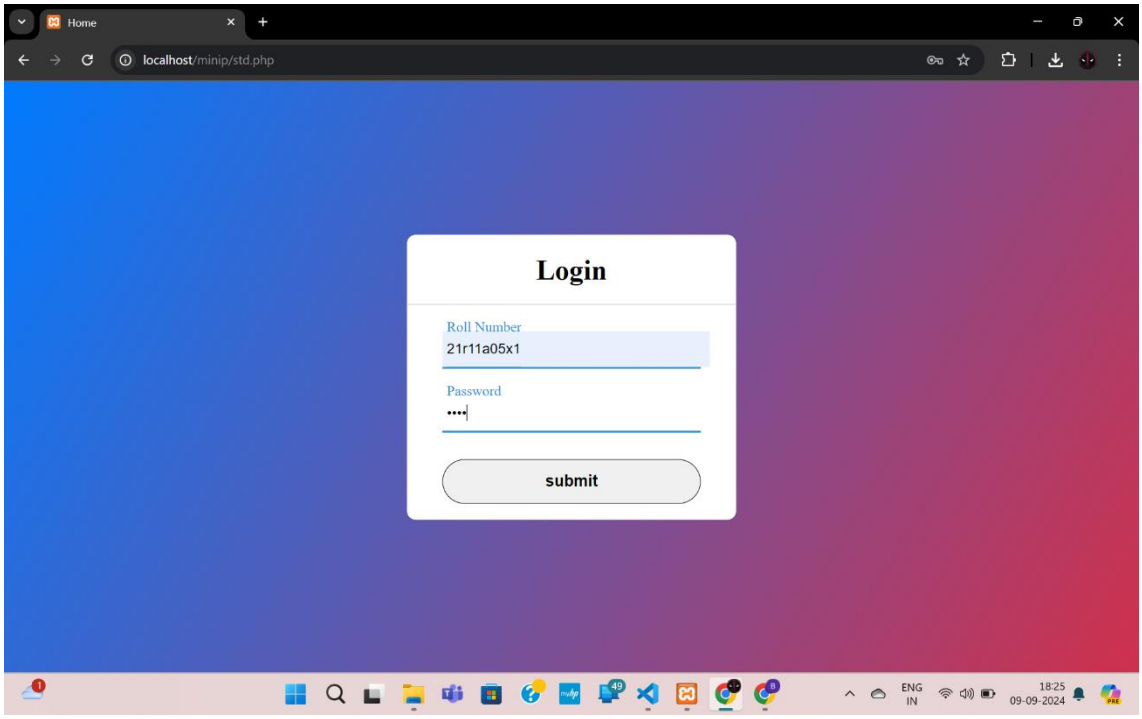

Fig 7.5 Validating Student Credentials

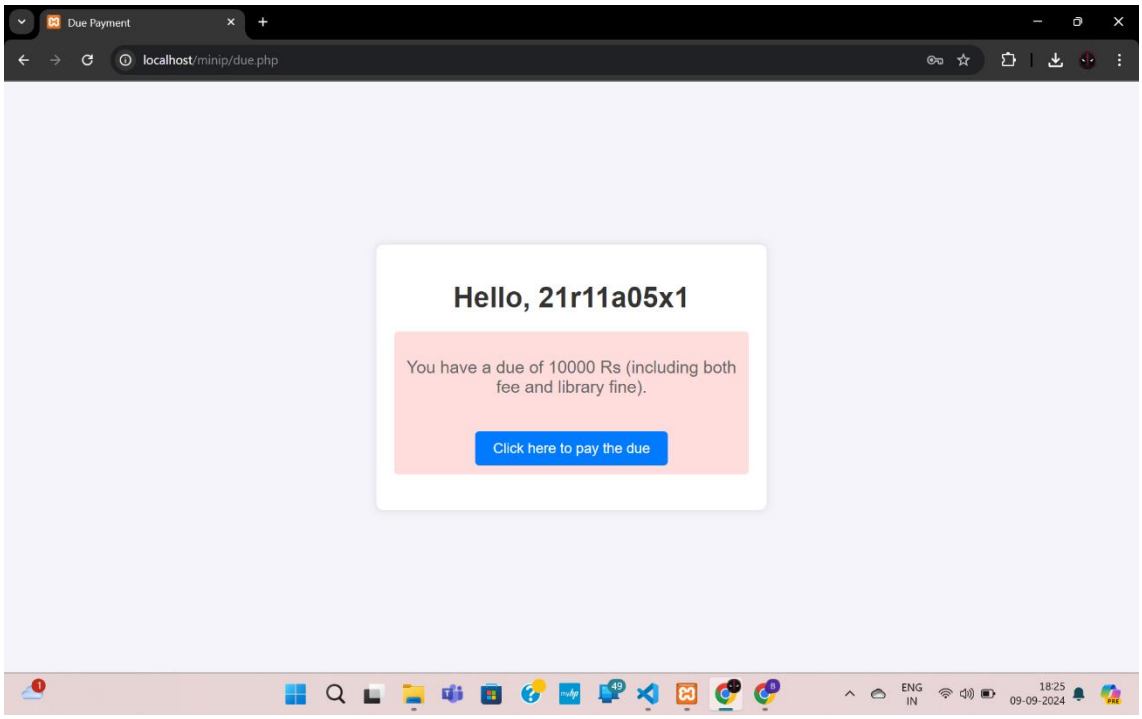

Fig 7.6 Fee Due

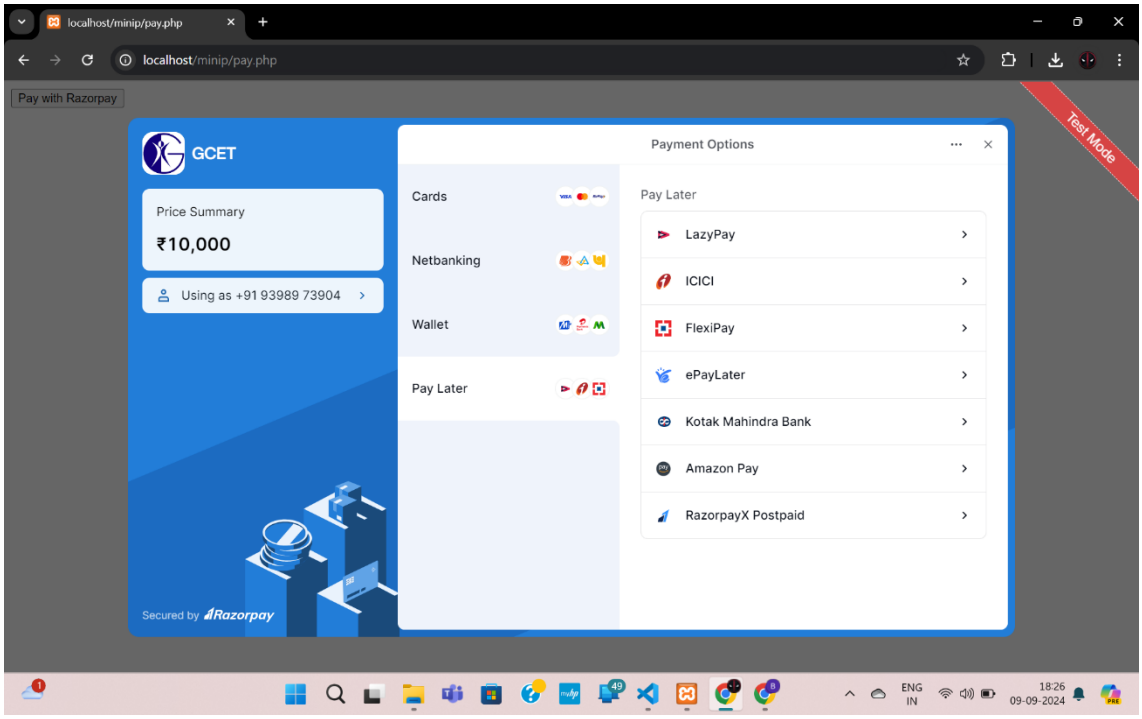

Fig 7.7 Payment Page

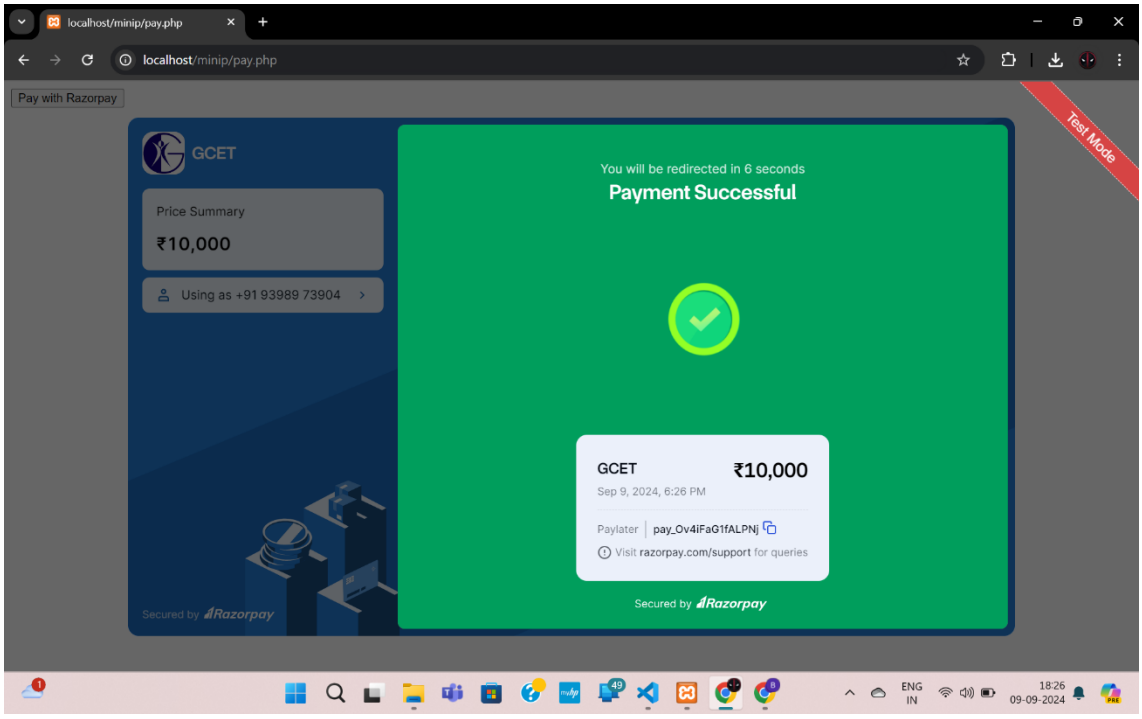

Fig 7.8 Payment Successful

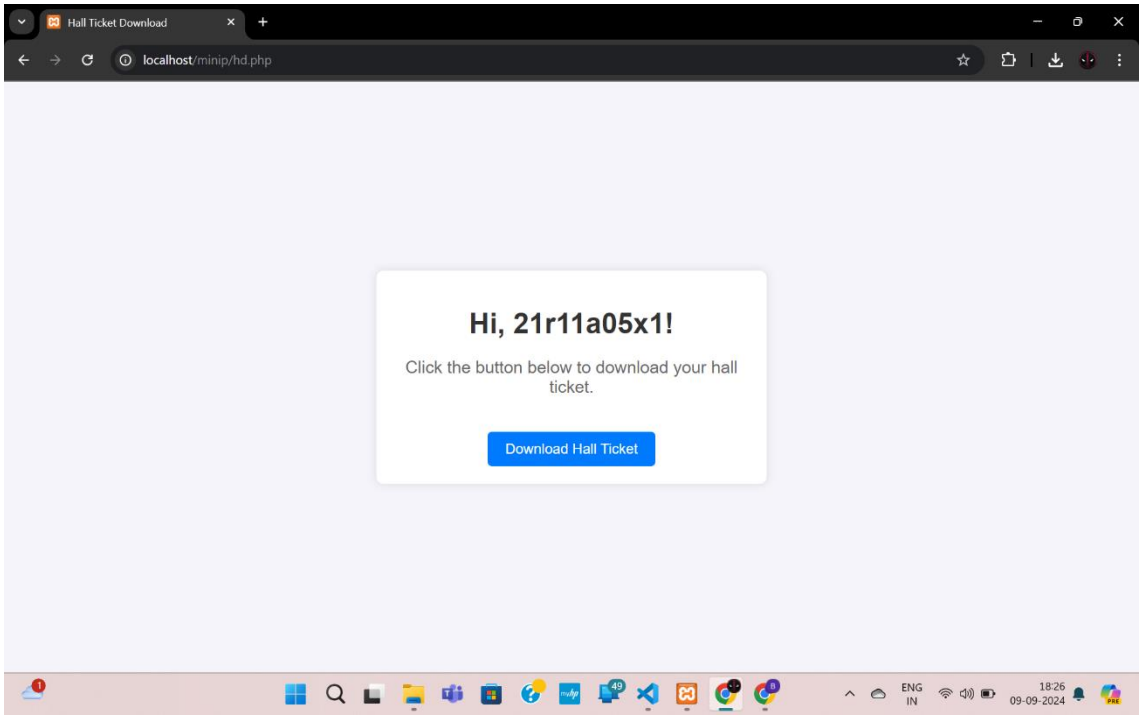

Fig 7.9 Downloading Hall Ticket

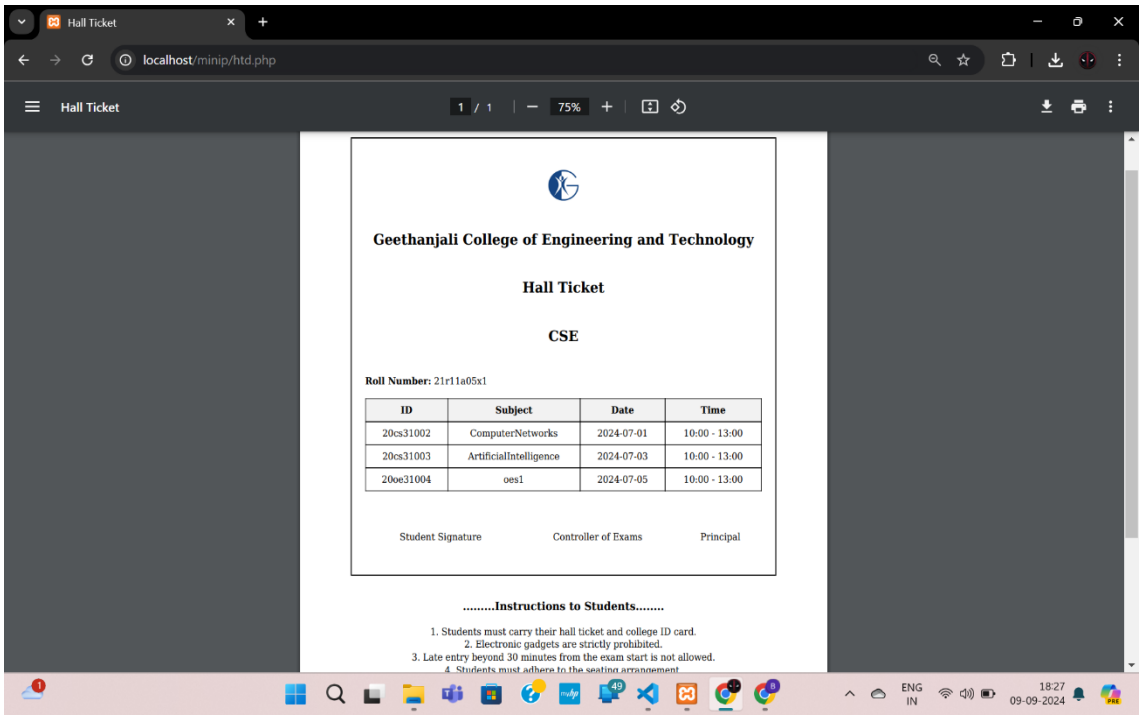

Fig 7.10 Displaying Hall Ticket

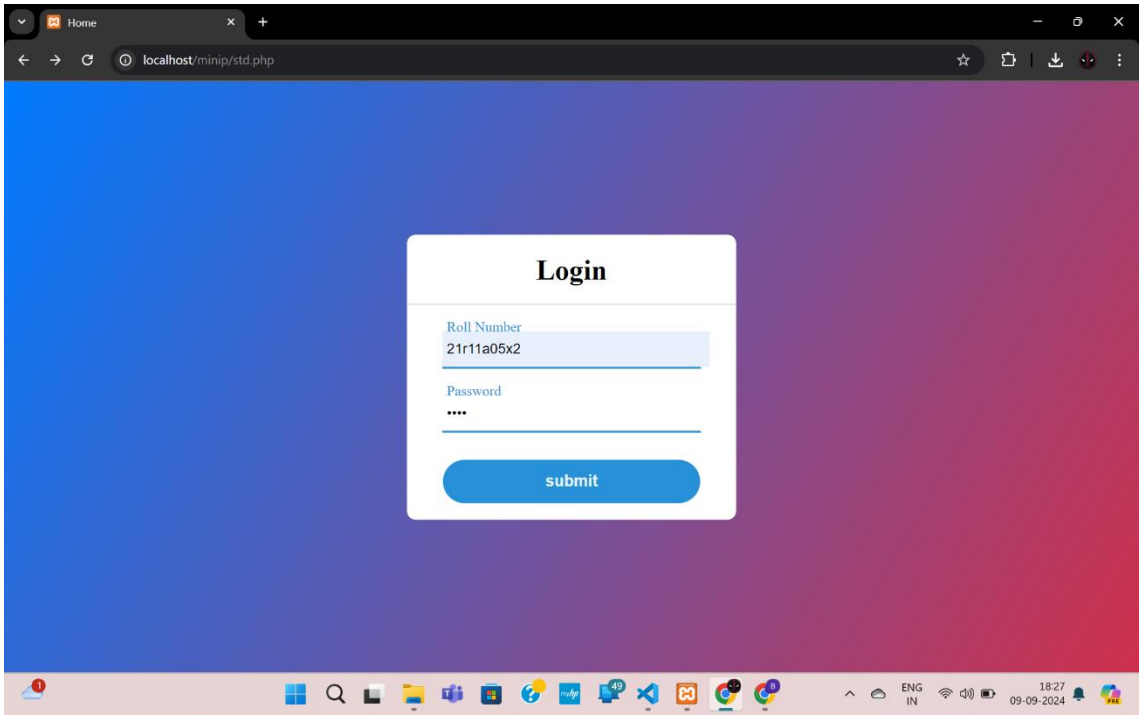

Fig 7.11 Another Student Login

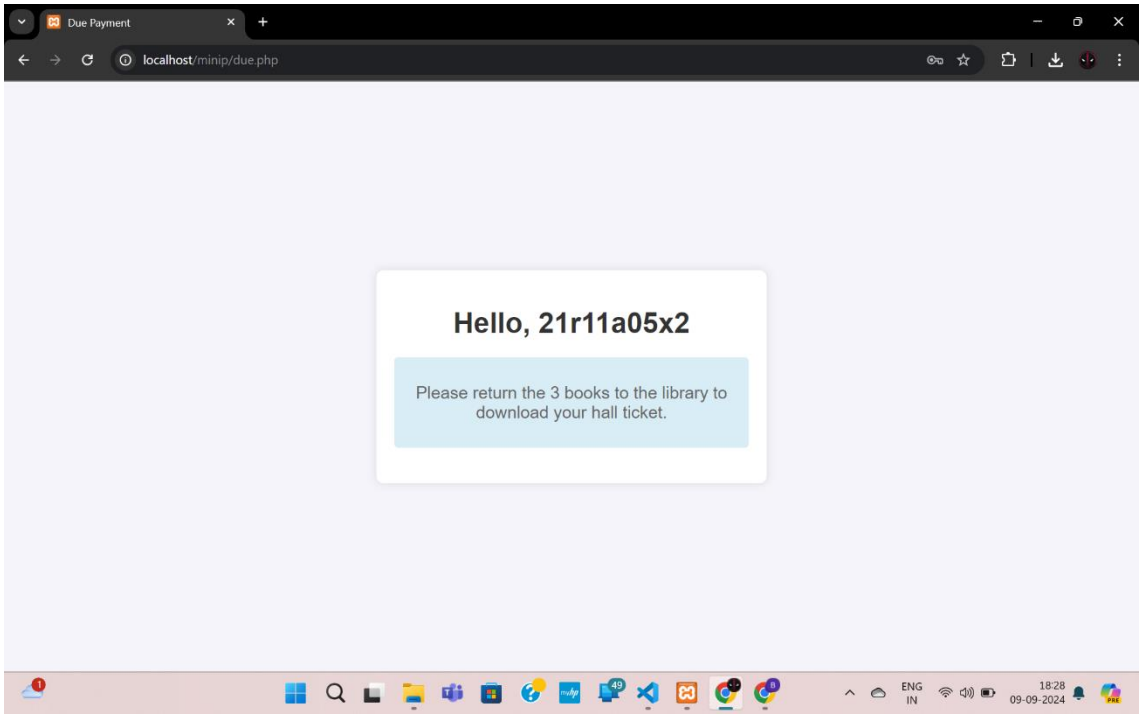

Fig 7.12 Library Due

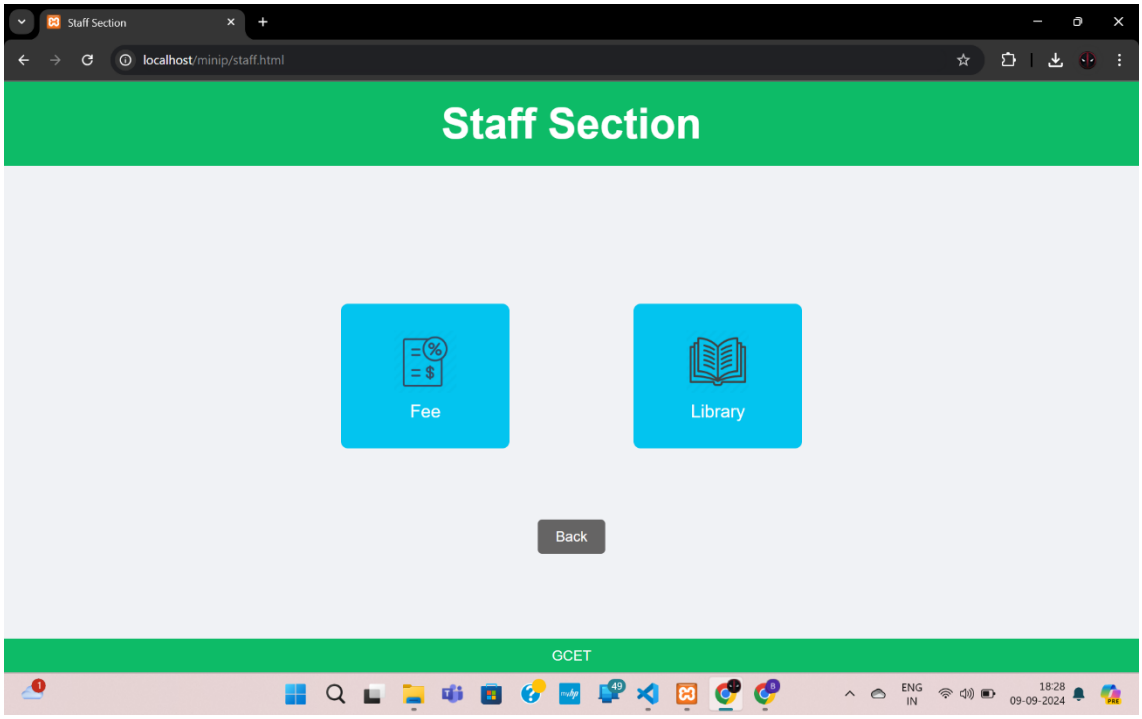

Fig 7.13 Staff Page

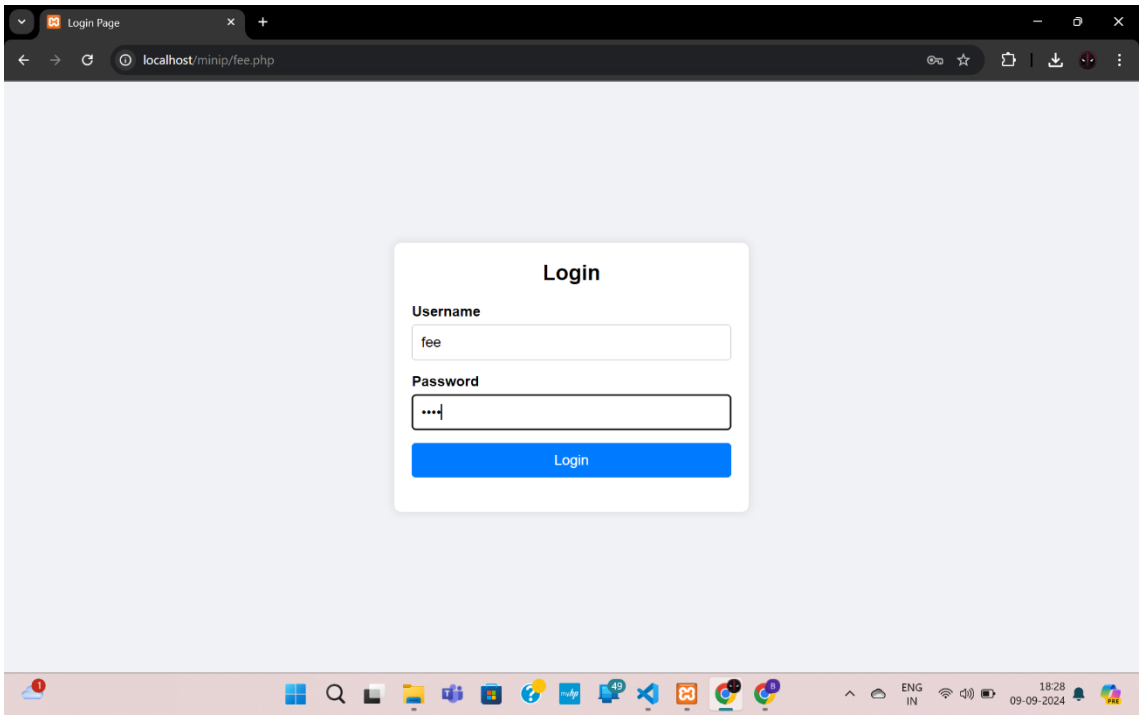

Fig 7.14 Fee Management Login

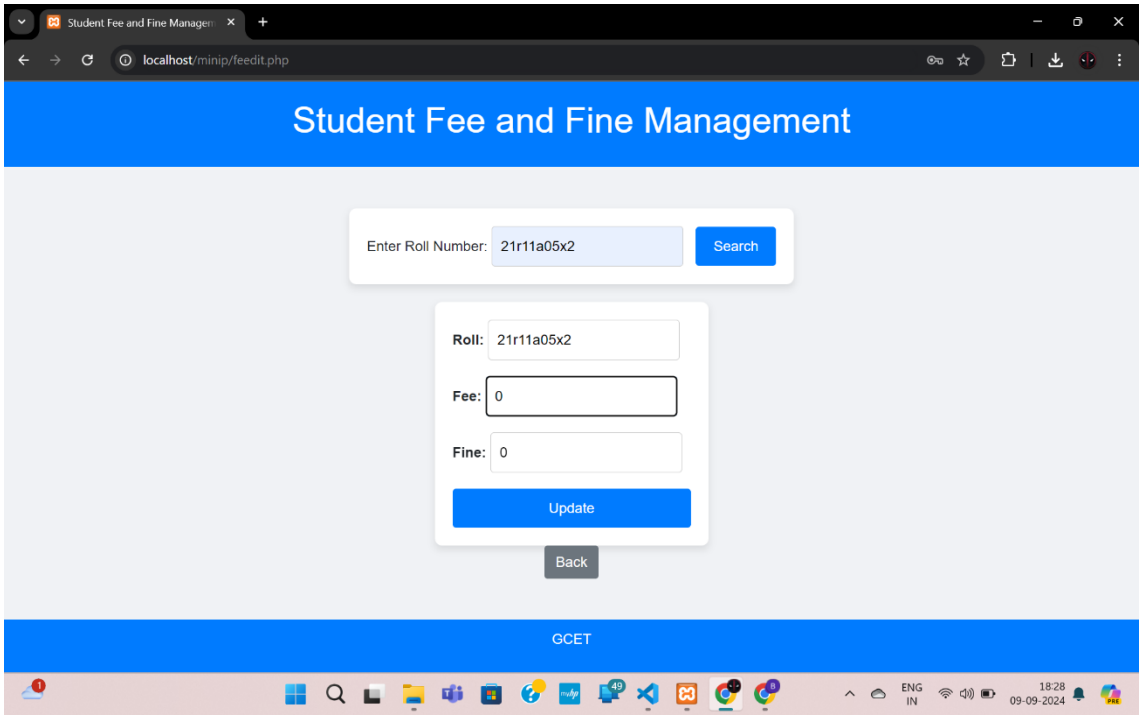

Fig 7.15 Fee Management Page

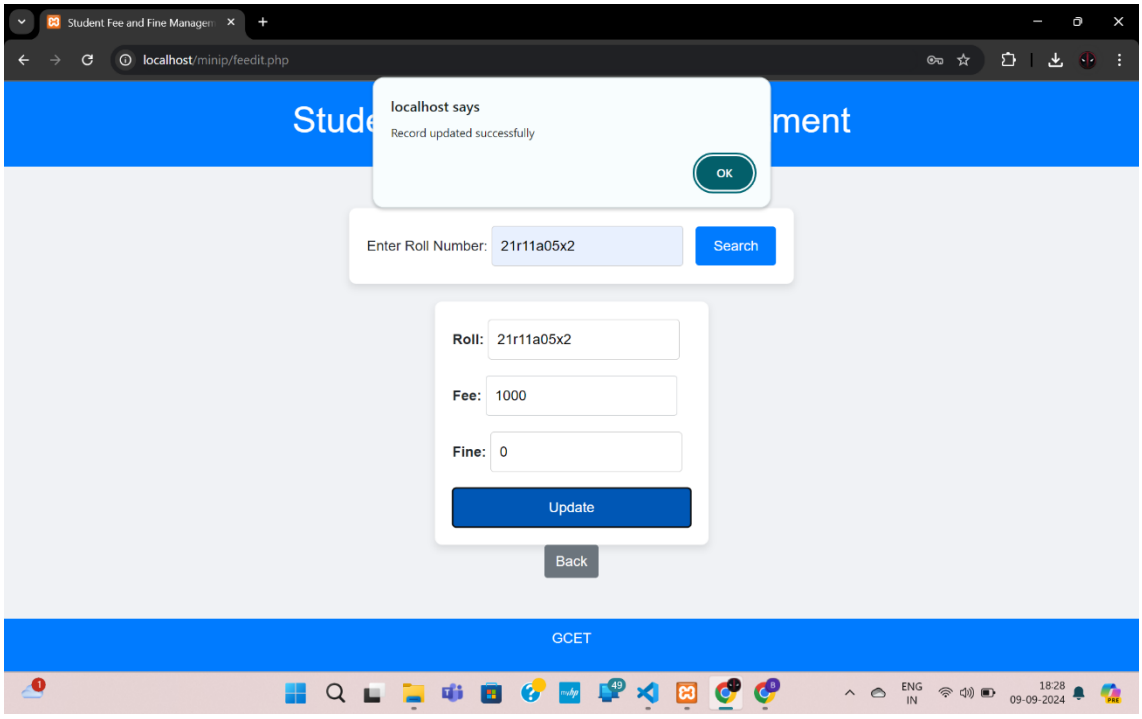

Fig 7.16 Updating Fee

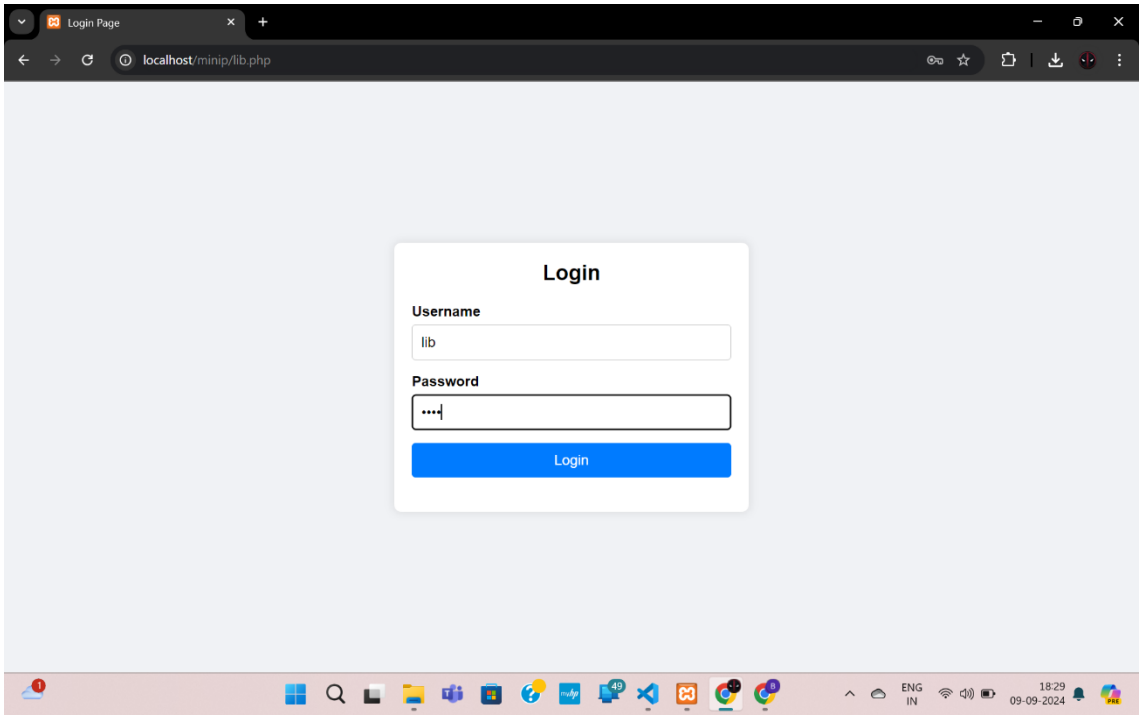

Fig 7.17 Librarian Login

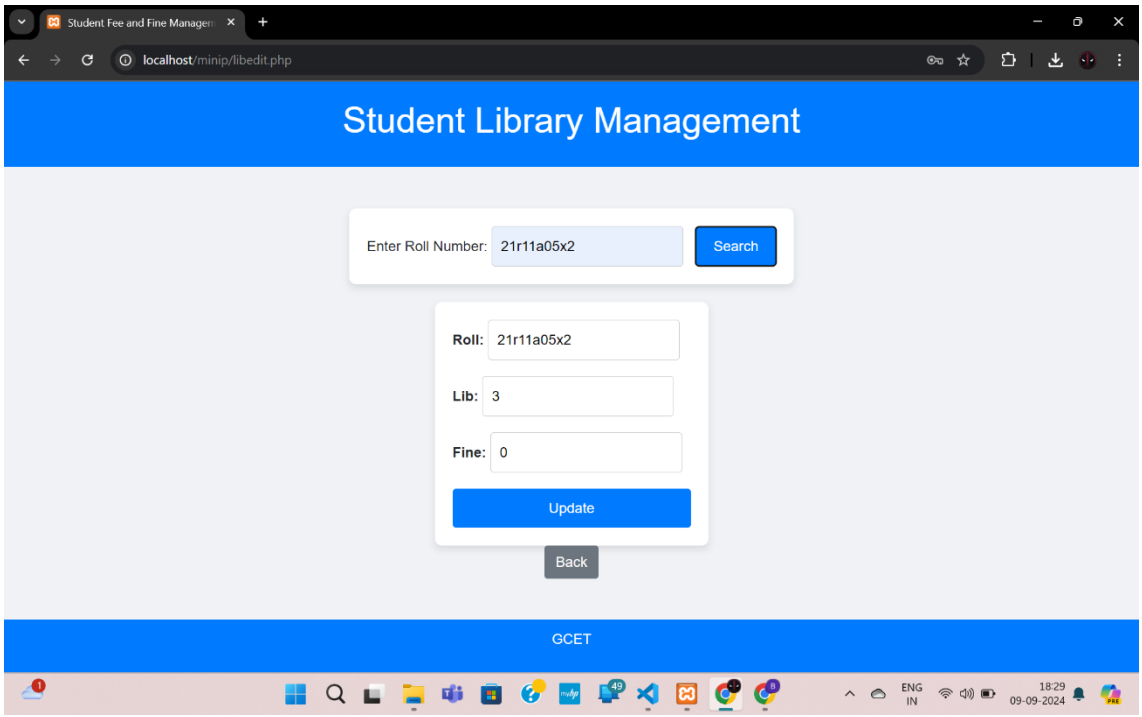

Fig 7.18 Librarian Page

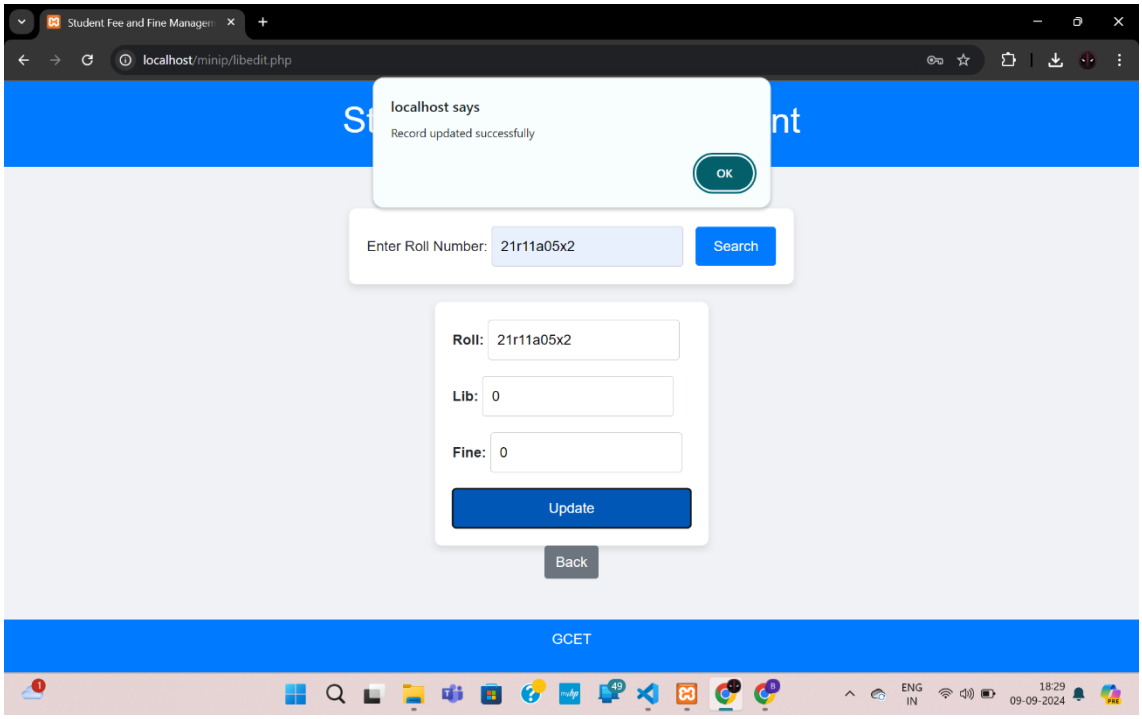

Fig 7.19 Updating Library Due

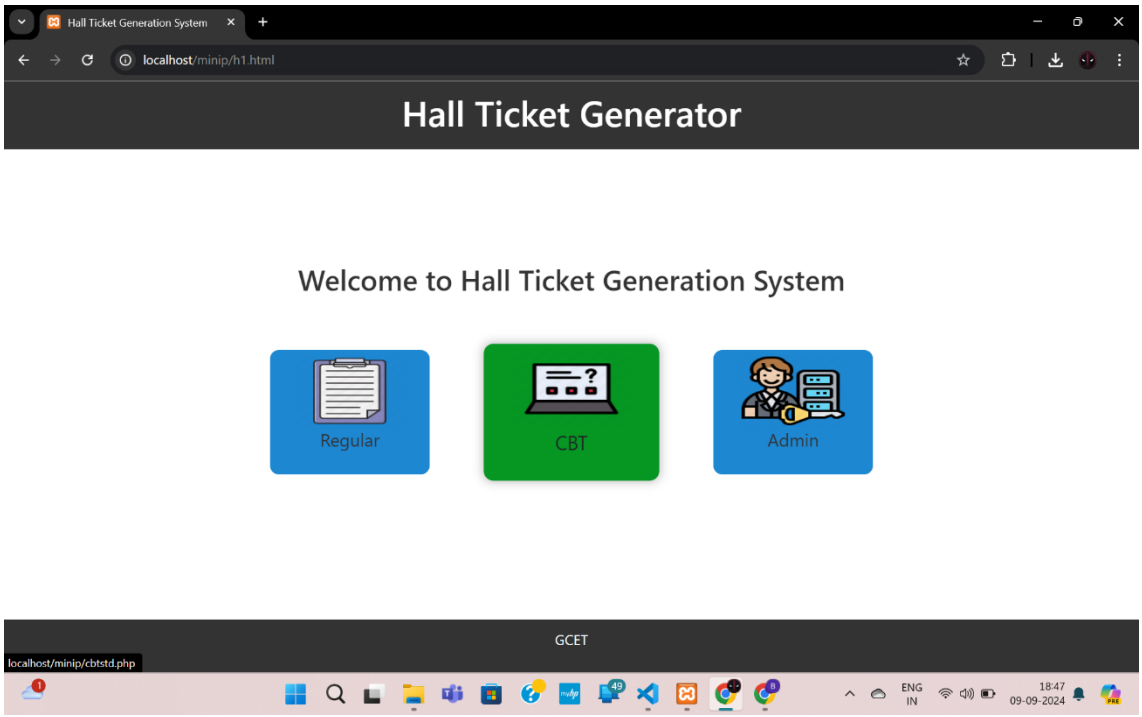

Fig 7.20 Selecting CBT

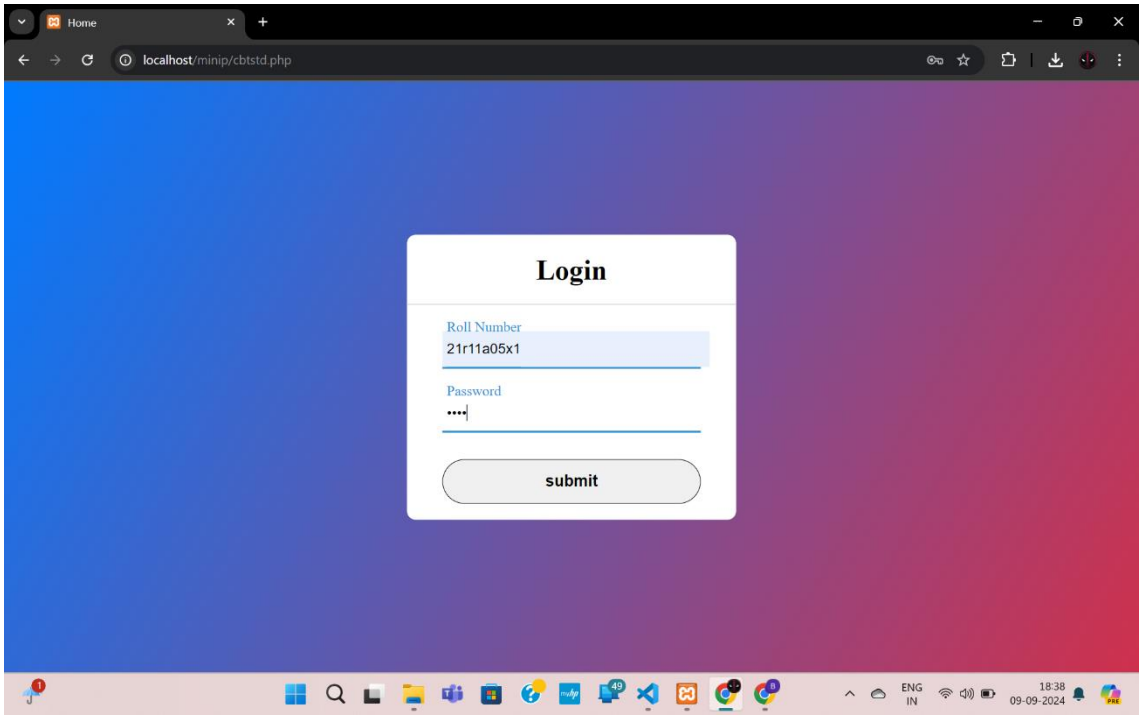

Fig 7.21 Verifying Student Credentials

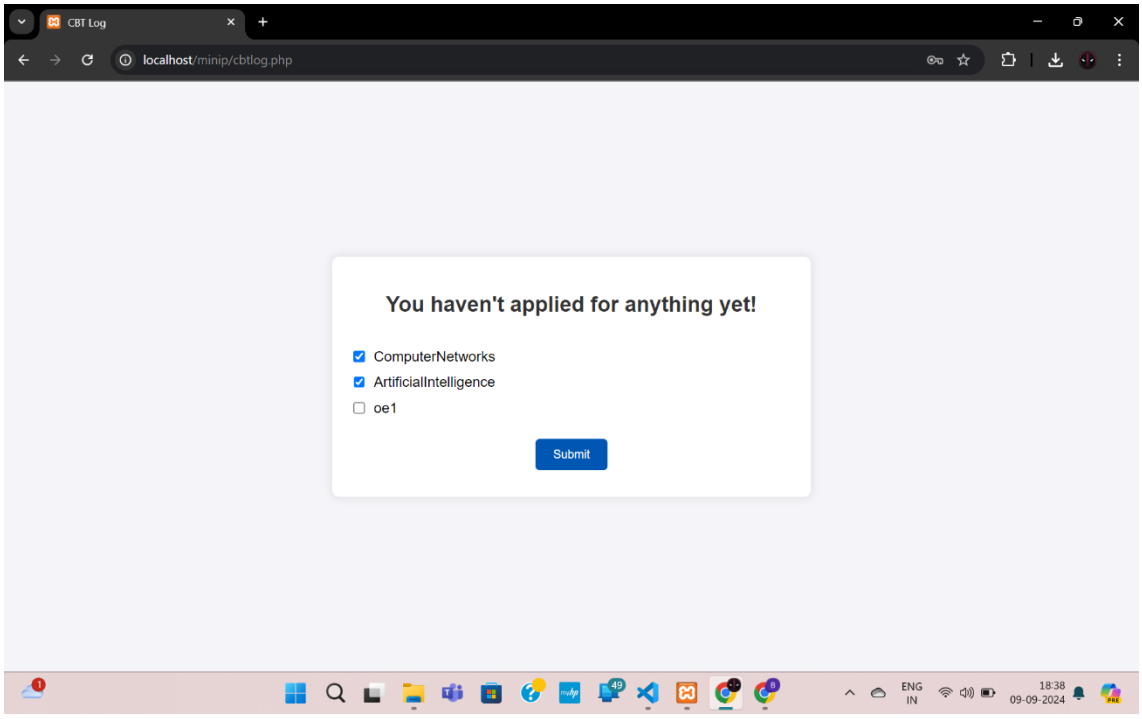

Fig 7.22 CBT Page

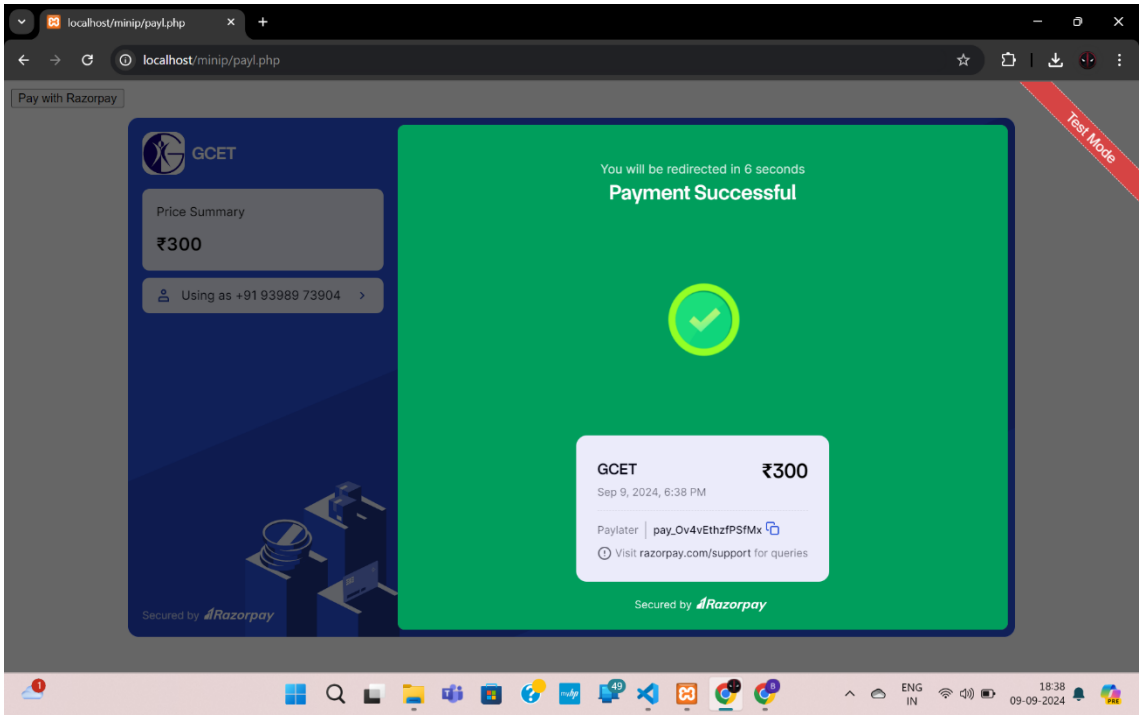

Fig 7.23 CBT Payment

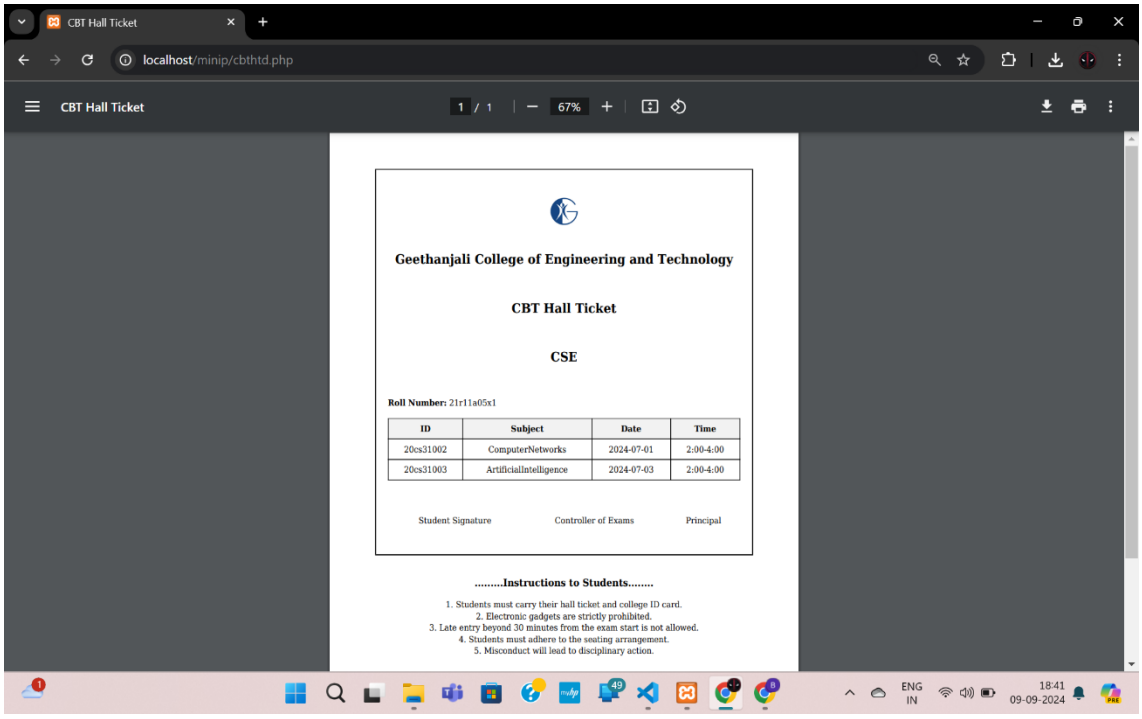

Fig 7.24 CBT Hall Ticket

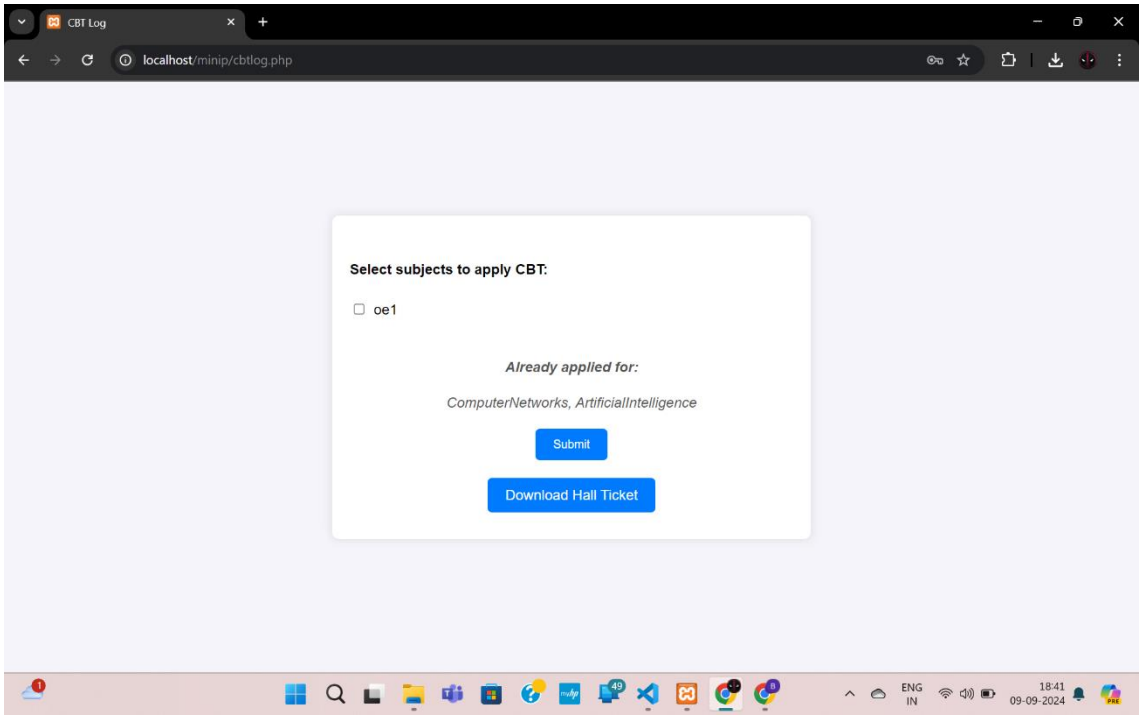

Fig 7.25 Rechecking CBT Page

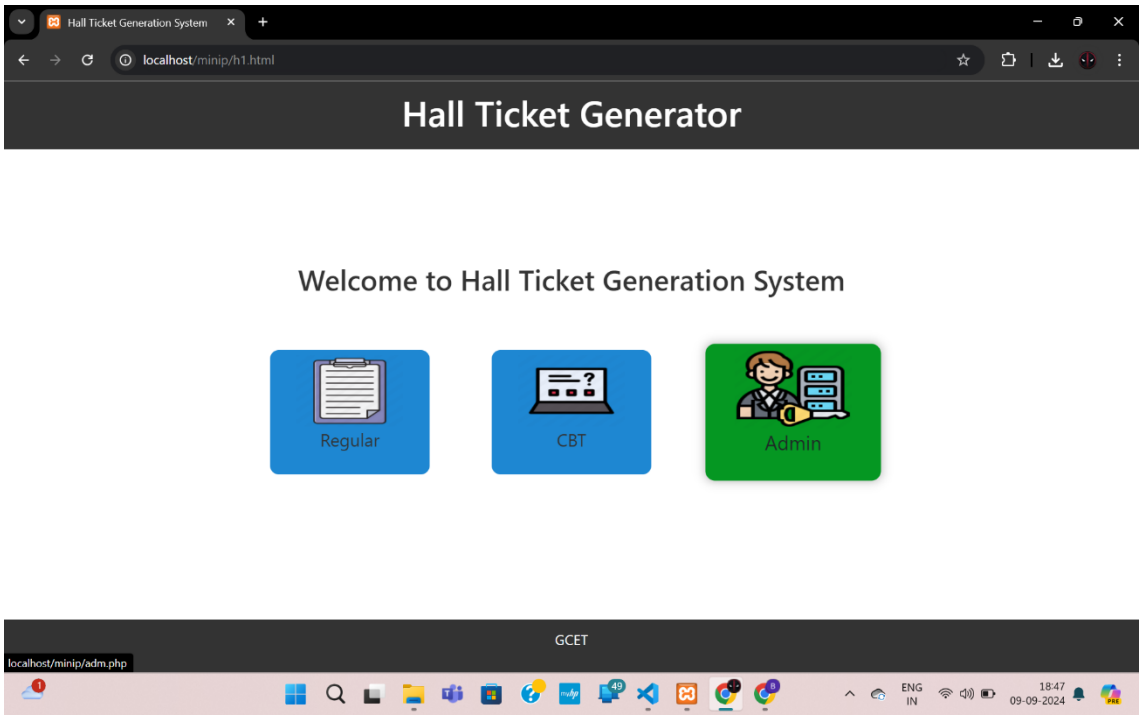

Fig 7.26 Selecting Admin

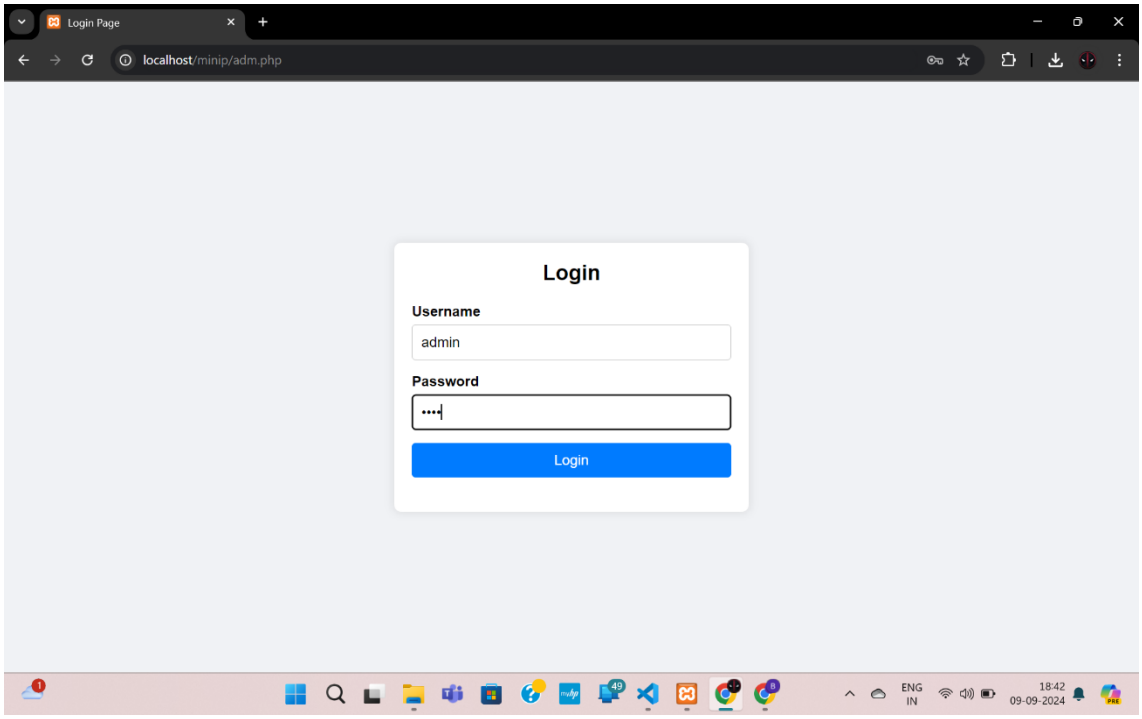

Fig 7.27 Verifying Admin Credentials

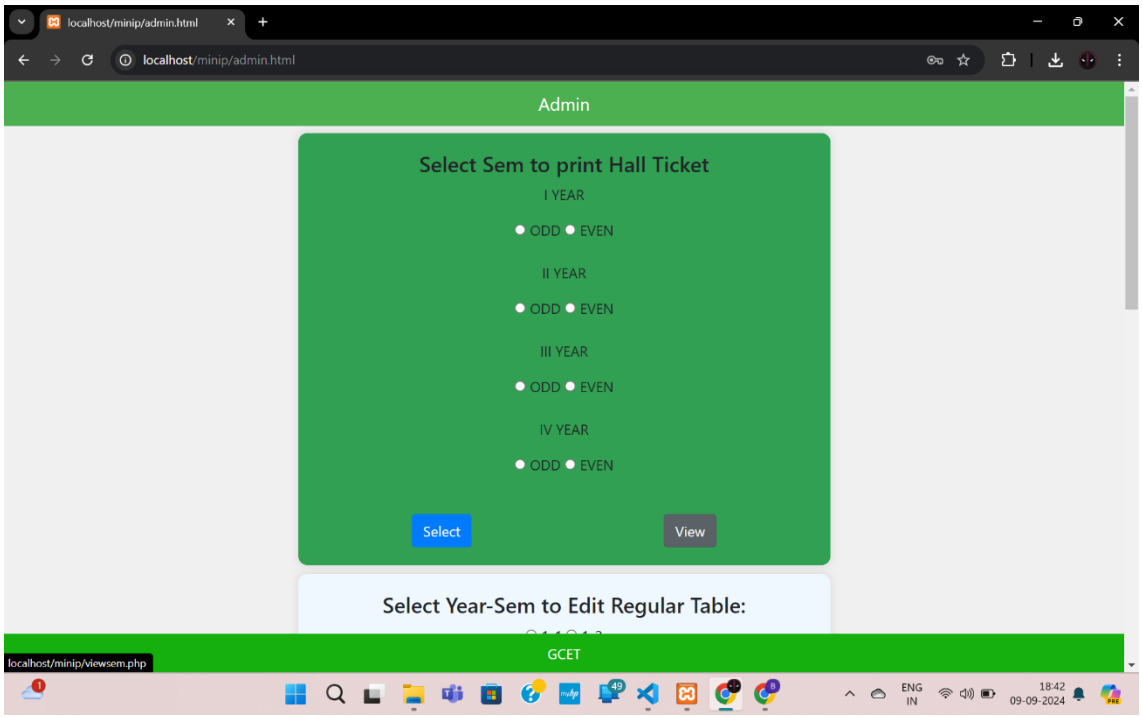

Fig 7.29 Select Sem

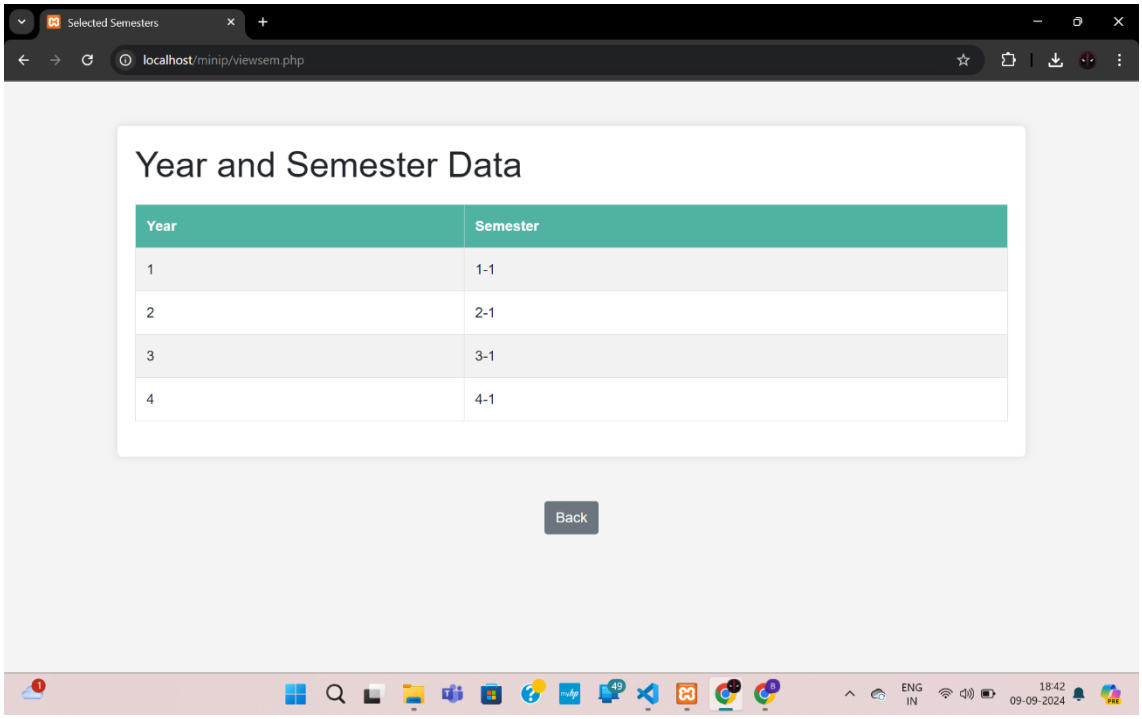

Fig 7.28 Year and Sem

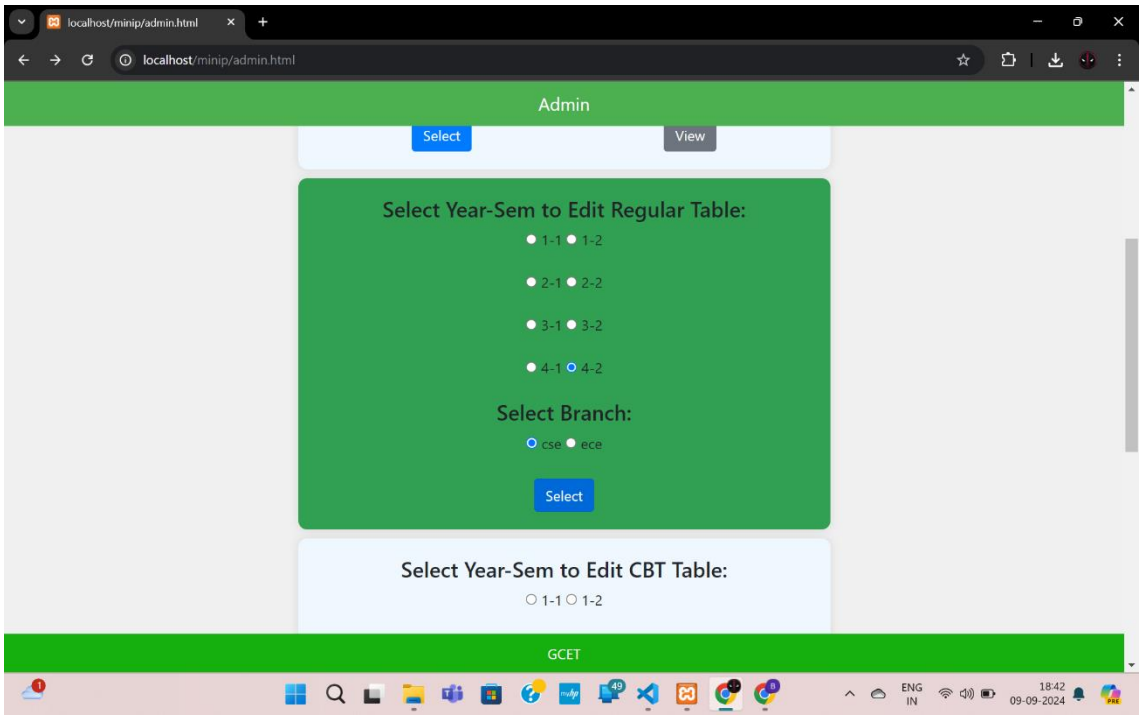

Fig 7.30 Edit Regular Table

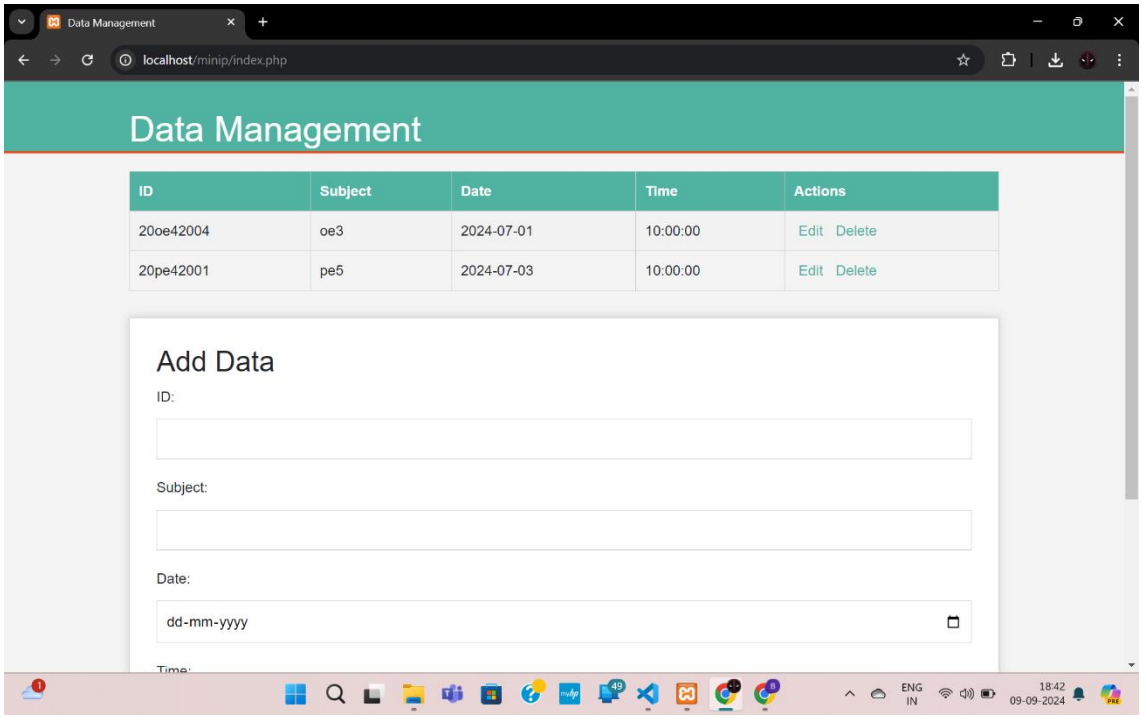

Fig 7.31 Regular Table

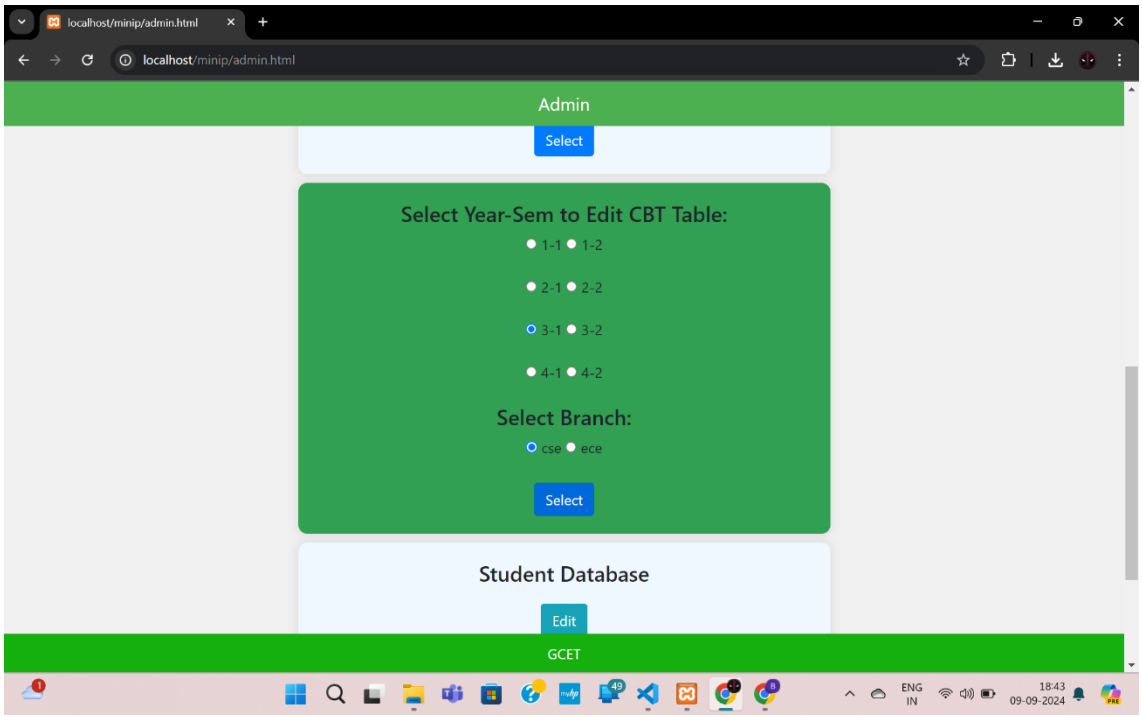

Fig 7.32 Edit CBT Table

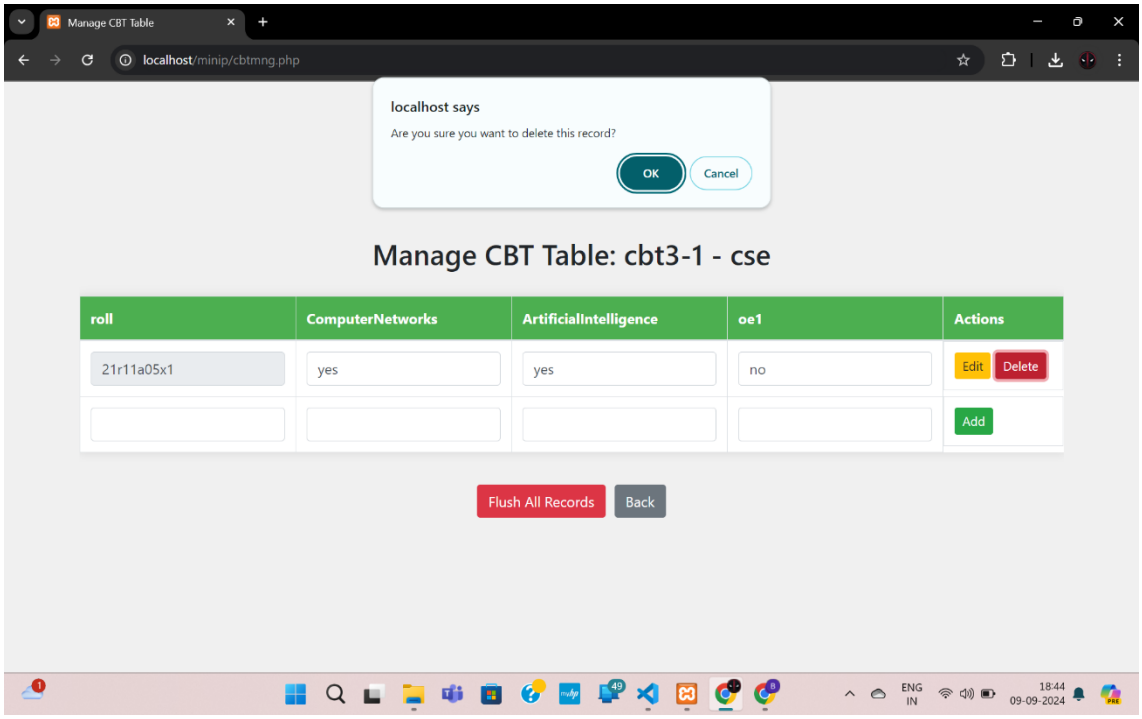

Fig 7.33 Deleting a Record in CBT Table

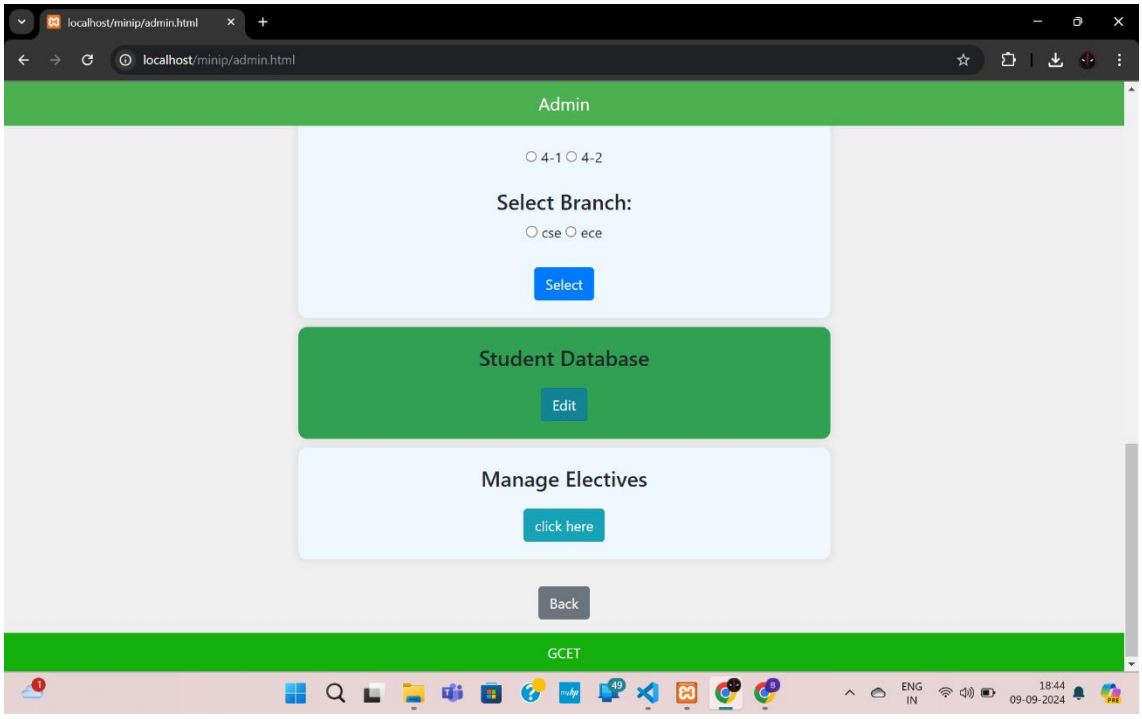

Fig 7.34 Edit Student Data

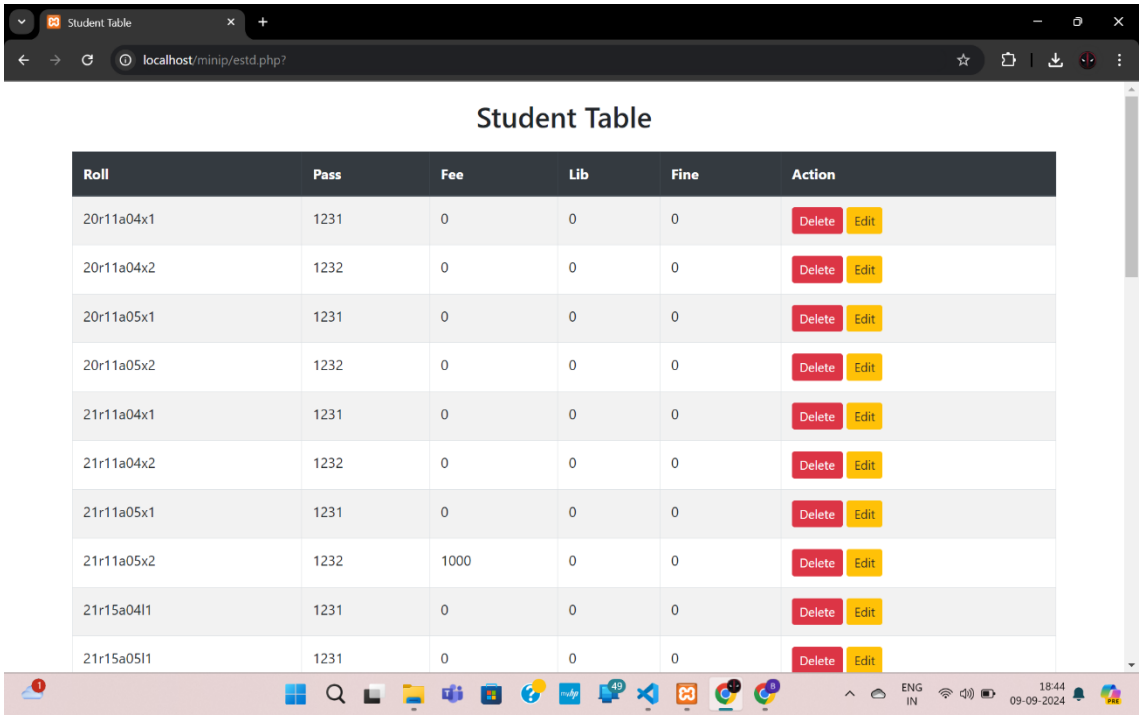

Fig 7.35 Student Data

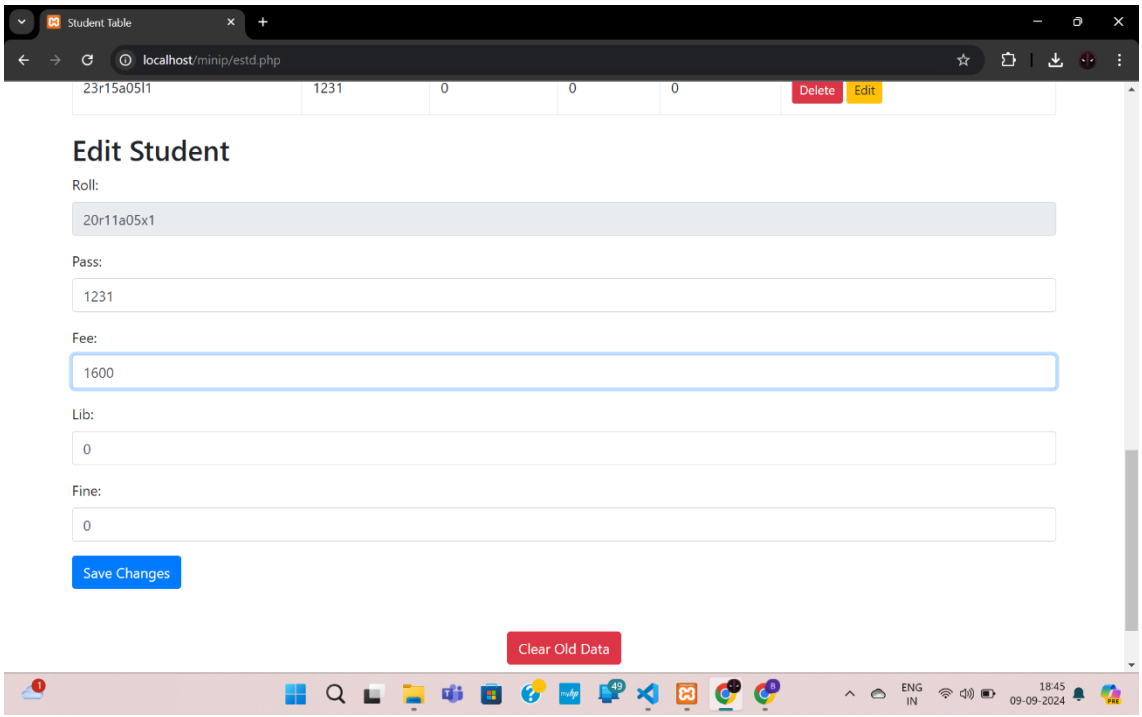

Fig 7.36 Editing Student Data

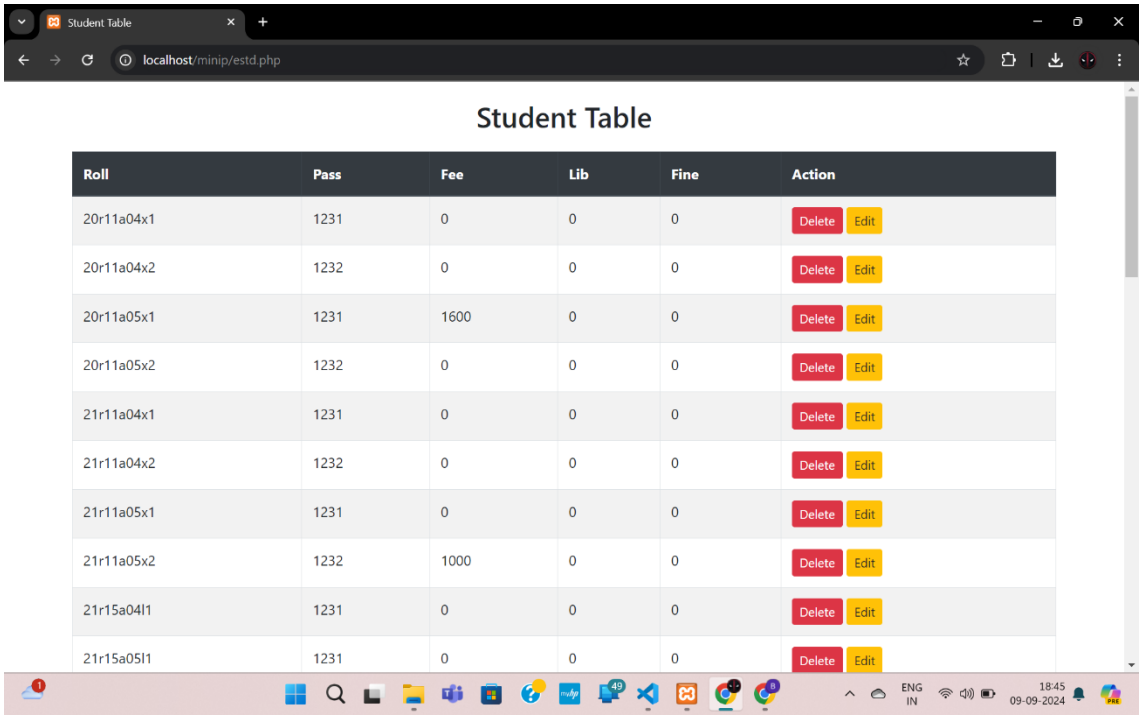

Fig 7.37 Verifying Change in Student Data

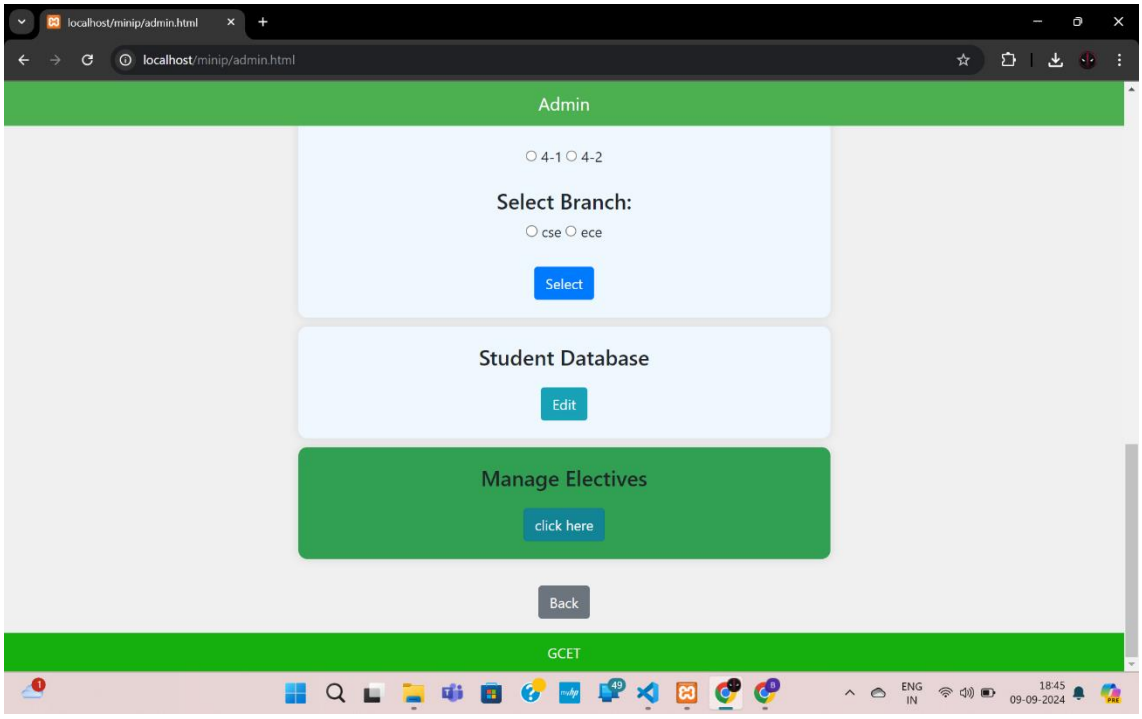

Fig 7.38 Manage Electives

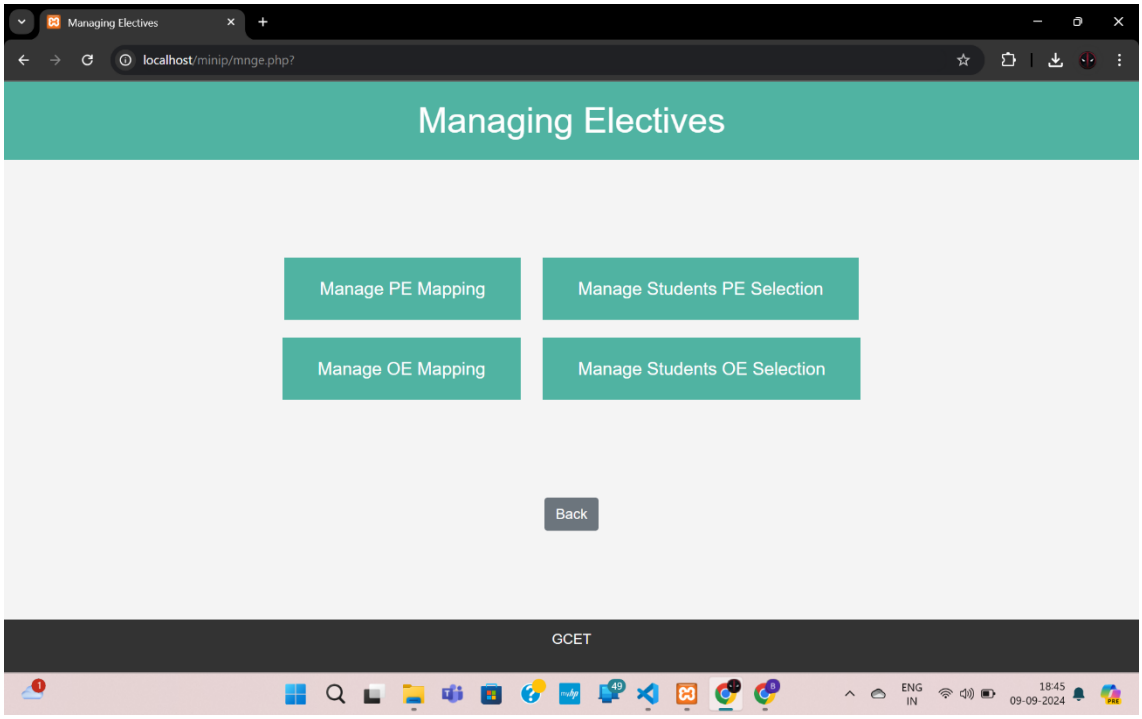

Fig 7.39 Manage Electives Page

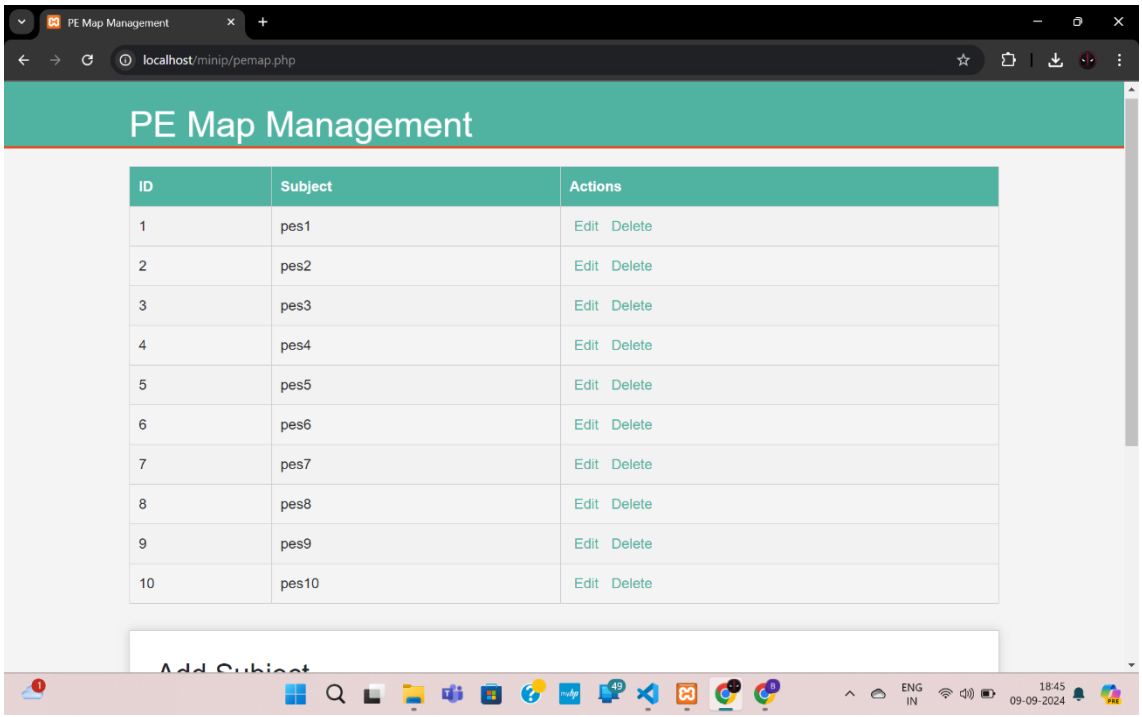

Fig 7.40 PE Mapping

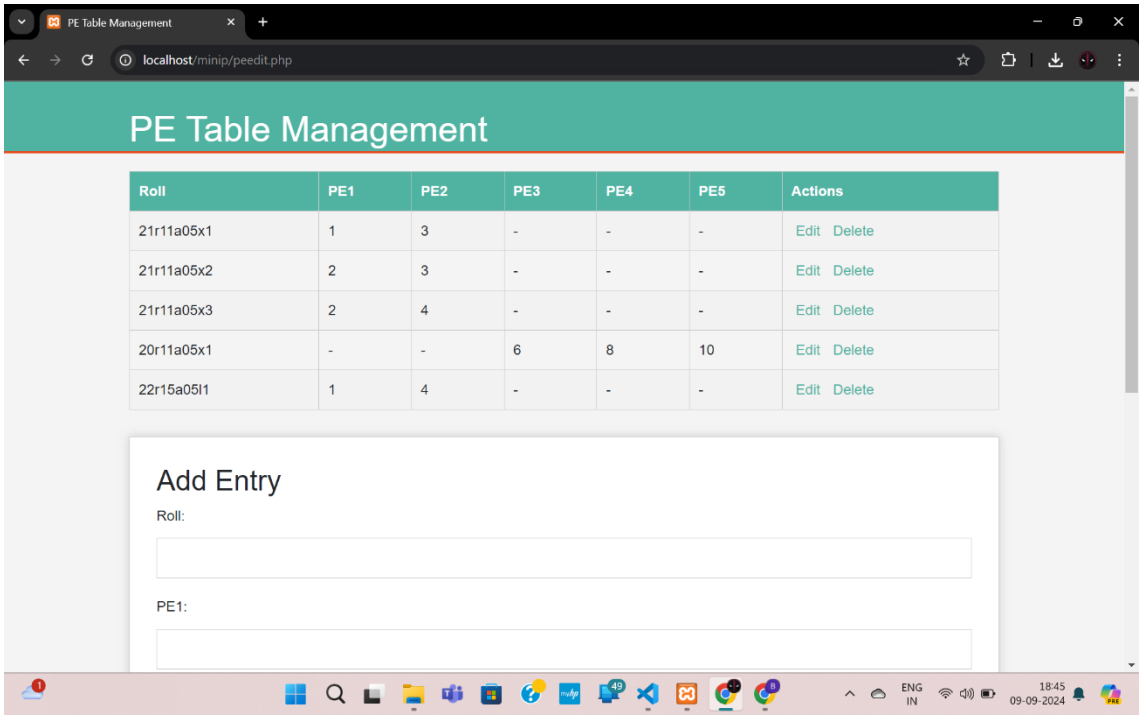

Fig 7.41 PE Table

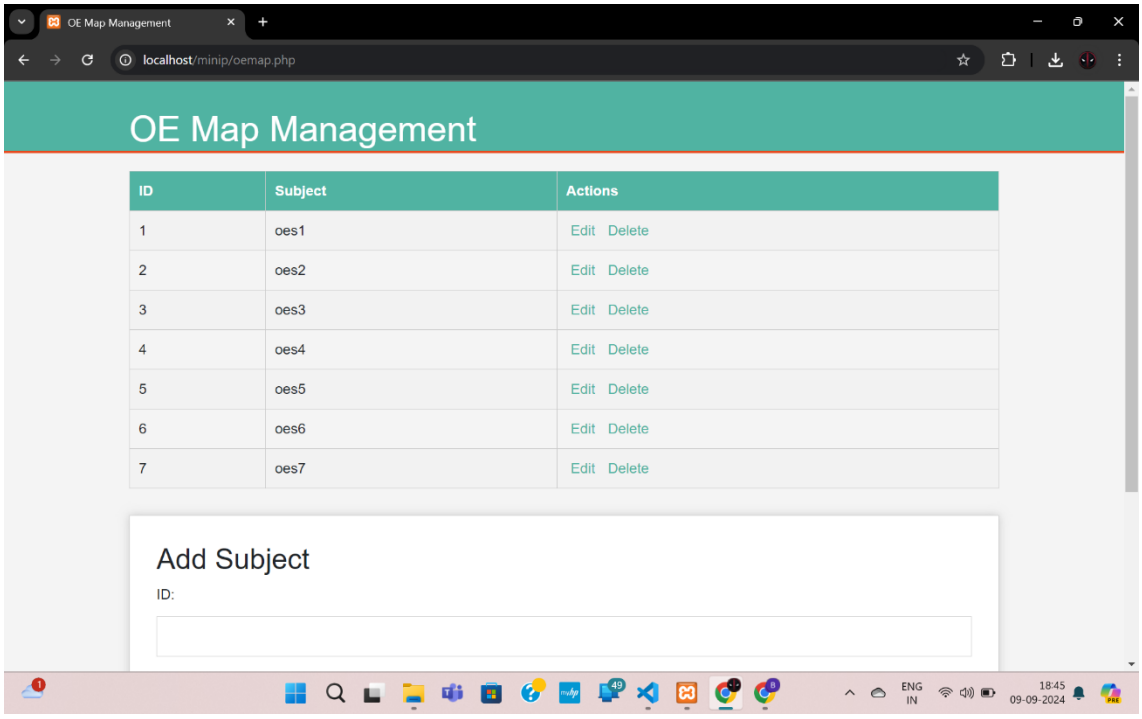

Fig 7.42 OE Mapping

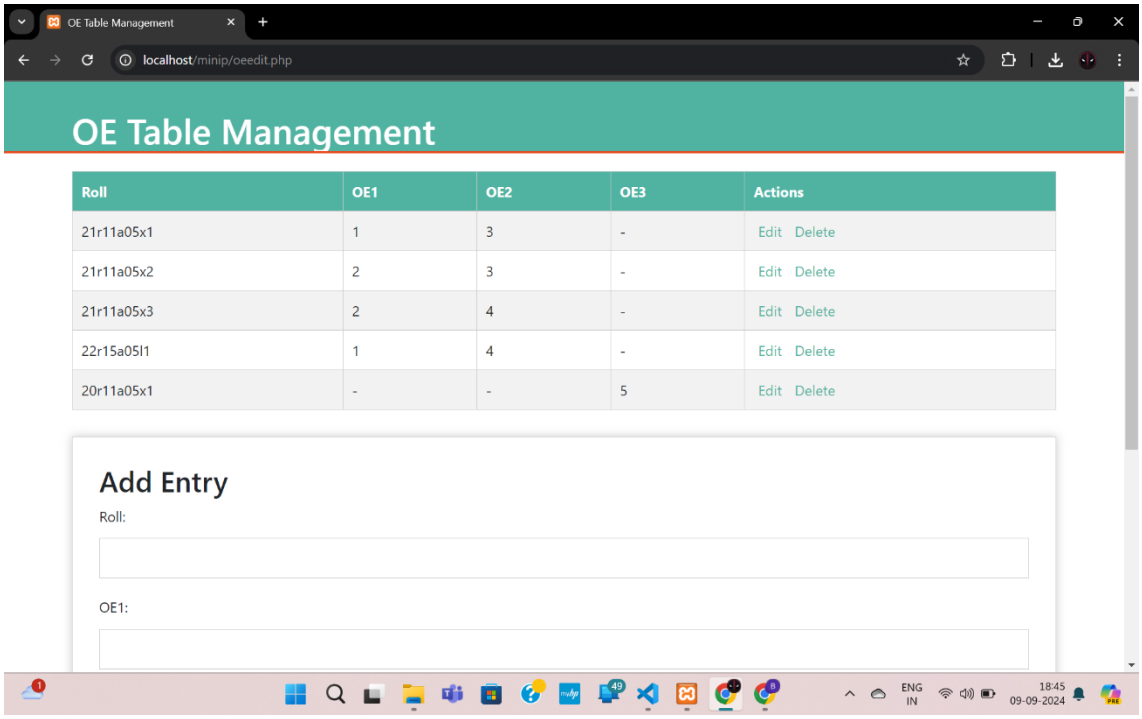

Fig 7.43 OE Table

8. CONCLUSION

8.1 CONCLUSION

The Hall Ticket Generator project provides an automated solution for managing hall ticket issuance, fee payments, library dues, and elective selection for students in academic institutions. By leveraging a combination of technologies such as PHP, JavaScript, HTML/CSS, Razorpay API, mpdf, and robust database management, the system integrates multiple administrative functions into a unified platform. The project significantly streamlines the process for both students and administrative staff, reducing manual intervention, errors, and time consumption.

The system ensures that students can conveniently manage their academic dues and generate hall tickets online after fulfilling the required conditions, such as fee payments and clearing library dues. The administrative interface allows staff to manage and monitor student records, including fee status, library dues, and elective registrations, thereby enhancing transparency and accuracy in data management.

Moreover, the system's modular design and layered architecture make it scalable and adaptable for future requirements. With secure login mechanisms and database management, the project ensures data integrity, privacy, and security, aligning with ethical standards for software development. Overall, the Hall Ticket Generator project offers an efficient, user-friendly, and reliable solution for academic management processes.

8.2 FURTHER ENHANCEMENTS

While the Hall Ticket Generator project successfully automates and optimizes several key administrative processes, there are numerous potential enhancements that can further improve the system:

- Mobile Application Development: Extending the project by developing a mobile application version can increase accessibility and convenience for students and administrators, allowing them to perform tasks on the go.
- Integration with Other Campus Management Systems: The project can be expanded by integrating with other existing systems, such as attendance

management, course registration, and exam grading systems, to provide a comprehensive solution for academic management.

- Automated Email and SMS Notifications: Enhancing the system to include automated email and SMS notifications for students regarding hall ticket generation, payment confirmations, and deadlines can improve communication and reduce the likelihood of missed deadlines.
- AI-Powered Insights and Analytics: Introducing an AI module that analyzes data related to fee payments, library dues, elective selections, and exam performances can provide valuable insights for institutional planning and decision-making.
- Enhanced Security Features: While the project already includes secure login mechanisms, incorporating advanced security features such as two-factor authentication (2FA) and biometric authentication can further safeguard sensitive data and prevent unauthorized access.
- Multi-Language Support: Adding multi-language support to the platform would cater to a more diverse user base, especially in regions with multiple languages.
- Feedback and Support Module: Developing a module where students and staff can provide feedback or request support directly within the system can help in continuously improving the platform based on user needs.
- Payment Gateway Options: While Razorpay is used for payment processing, adding multiple payment gateway options like PayPal, Google Pay, or UPI can offer flexibility and convenience to users.

By incorporating these enhancements, the Hall Ticket Generator project can evolve into a comprehensive, highly efficient, and user-centric academic management system that caters to the dynamic needs of modern educational institutions.

9. BIBLIOGRAPHY

9.1 BOOKS REFERENCES

- **"Database System Concepts" by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan.**

  This book provides a comprehensive introduction to database management systems, covering fundamental concepts such as data models, SQL, normalization, transactions, and database design. It was used as a reference for designing and managing the various databases (Regular, CBT, Electives, etc.) used in the project.

- **"Web Development with JavaScript and AJAX Illuminated" by Richard Allen and Mark Lassoff.**

  This book offers a deep dive into web development using JavaScript, HTML, CSS, and AJAX. It provided insights into creating dynamic and interactive web pages for the student and admin interfaces of the system.

- **"PHP and MySQL Web Development" by Luke Welling and Laura Thomson.**

  A comprehensive guide that covers building dynamic websites using PHP and MySQL, focusing on security, optimization, and best practices. This book provided insights into structuring the backend of the Hall Ticket Generator project.

- **"Modern PHP: New Features and Good Practices" by Josh Lockhart.**

  This book explores modern PHP development practices, including advanced features, security, and performance improvements that were useful for optimizing the codebase.

- **"Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5" by Robin Nixon.**

  A good reference for integrating front-end and back-end technologies to create dynamic and interactive web applications like the one used in this project.

9.2 WEBSITES REFERENCES

- **PHP Manual** (https://www.php.net/manual/): The official PHP documentation is a valuable resource for understanding core PHP functionalities, error handling, and best practices used in the project.
- **Razorpay Documentation** (https://razorpay.com/docs/): Official documentation for Razorpay's payment gateway integration used in the project to handle online payments for dues.
- **MySQL Documentation** (https://dev.mysql.com/doc/): The official guide for MySQL, covering the essential aspects of relational database design, SQL queries, and integration with PHP.

9.3 TECHNICAL PUBLICATION REFERENCES

- **"Building Secure Web Applications with PHP" by OWASP Foundation.**
  An online publication that provides security guidelines and practices specific to PHP to prevent vulnerabilities like SQL injection and cross-site scripting (XSS).
- **"Efficient Web-Based Student Information System" in International Journal of Computer Applications (2020).**
  Discusses methodologies for creating web-based systems to manage academic processes efficiently, aligning with the objectives of the Hall Ticket Generator project.
- **"Payment Gateway Integration for Web Applications" in International Journal of Advanced Computer Science and Applications (2021).**
  This paper outlines best practices for integrating payment gateways securely, relevant to the Razorpay integration in the project.
- **"Modern Database Management Techniques for Educational Institutions" - Journal of Information Technology Education (2023).**
  This technical paper provided a detailed overview of efficient database management practices, which were useful for managing the student and exam databases in the project.

10. APPENDICES

A. Software Used:

- **XAMPP:** Used to set up a local server environment for PHP and MySQL development.
- **PHP:** Server-side scripting language used to build the backend logic and connect with the database.
- **MySQL:** Relational database management system used to store and manage data related to students, subjects, dues, payments, and more.
- **HTML, CSS, JavaScript:** Frontend technologies used to build the user interface for students, staff, and admin.
- **RazorpayX API:** Used for integrating payment gateway functionality to handle fee payments.
- **MPDF:** A PHP library used to generate PDF documents for hall tickets.

B. Methodologies Used:

- **Modular Design:** The project is divided into distinct modules (Regular, CBT, Admin), each handling specific functionalities such as fee management, hall ticket generation, and database management.
- **Iterative Development:** The project follows an iterative development approach, where each module is developed, tested, and refined in phases.
- **Database Normalization:** Applied normalization techniques to ensure efficient data storage and reduce redundancy in databases like 1-1, 1-2, 2-1, etc. and cbt1-1, cbt1-2, electives.

C. Testing Methods Used:

- **Unit Testing:** Testing individual modules such as student login, fee payment, and hall ticket generation to ensure each part works correctly.
- **Integration Testing:** Testing the interaction between different modules, such as the integration between the student module and the payment gateway.
- **User Acceptance Testing (UAT):** Ensuring that the system meets the end-user requirements and works as expected in real-life scenarios.
- **Functional Testing:** Verifying that each function of the software application operates in conformance with the requirement specification.

11. PLAGIARISM REPORT

Dupli Checker

PLAGIARISM SCAN REPORT

0%
Plagiarised

100%
Unique

Date	2024-09-10
Words	178
Characters	1297

Content Checked For Plagiarism

The Hall Ticket Generator project is designed to automate the process of issuing hall tickets and managing exam-related activities for students in educational institutions. Traditionally, students face long queues and manual verification steps for clearing dues and obtaining hall tickets for regular and Computer-Based Tests (CBT). This project addresses these inefficiencies by providing a web-based application where students can log in using their credentials to check for fee, fine, or library dues, make payments online, and download their hall tickets directly. The system is built using PHP for the backend, MySQL for the database, and Razorpay API for payment integration. The application features separate modules for students, staff, and administrators, each with specific functionalities like managing exam details, viewing and updating student records, and generating hall tickets in PDF format. This system significantly reduces the manual effort required, eliminates administrative bottlenecks, enhances data accuracy, and provides a seamless user experience. The Hall Ticket Generator aims to streamline the examination preparation process, saving time and effort for both students and staff while ensuring a more efficient and transparent system.

Matched Source

No plagiarism found

Check By: Dupli Checker