DON BOSCO COLLEGE OF ENGINEERING
FATORDA, MARGAO, GOA – 403 602.

DEPARTMENT OF COMPUTER ENGINEERING

2024 – 2025

"Student Information System"

By

Mr. Malcolm Antao 2214025

Ms. Nathania Baptista 2214032

Mr. P. Jayesh Naidu 2214035

Term work report for the subject of

CE510 – Database Management and Query Processing

Semester V

October 2024

Under the guidance of

Dr. Norman Dias **Prof. Amey Tilve**

BACHELOR OF ENGINEERING: GOA UNIVERSITY

DON BOSCO COLLEGE OF ENGINEERING

FATORDA, MARGAO, GOA- 403602.

2024 – 2025

CERTIFICATE

This

Third Year Project Report

'Student Information System'

Submitted in partial fulfillment of the requirements for Bachelor's Degree in Computer Engineering of Goa University is the bonafide work of

Mr. Malcolm Antao 2214025

Ms. Nathania Baptista 2214032

Mr. P. Jayesh Naidu 2214035

Faculty Incharge	**Faculty Incharge**	**Head of Department**
Dr. Norman Dias Patkar	**Prof. Amey Tilve**	**Dr. Gaurang S**

ABSTRACT

The Student Information System (SIS) for a computer engineering college is a comprehensive solution designed to centralize and streamline the management of student-related data and academic processes. The primary objective of the system is to provide an efficient platform that handles various aspects of student administration, including the storage and retrieval of essential student information, course management, enrollment tracking, and grade reporting. The system also facilitates communication between students and faculty through features like course selection, mentorship, and announcement functionalities.

The SIS employs a structured database that integrates detailed information about students, such as their unique ID, contact details, and major field of study. It also records course information, including course descriptions, credits, and departmental affiliation. In addition, the system tracks student enrollments across different courses and semesters, providing a clear view of each student's academic progress. Instructor information, including contact details and teaching assignments, is also maintained to support faculty management.

Key findings of the project include the system's ability to significantly reduce manual administrative tasks, improve data accuracy, and enhance the overall efficiency of academic operations within the college. The SIS offers a user-friendly interface for students, instructors, and administrators, enabling smooth access to relevant data and facilitating better decision-making. In conclusion, the SIS plays a critical role in supporting both academic and administrative functions, promoting effective student management and improved educational outcomes.

ACKNOWLEDGEMENT

We would like to take this moment to extend our heartfelt gratitude to all those who supported and contributed to our project, whether directly or indirectly. We are deeply appreciative of the opportunity, expertise, and encouragement we received, as well as the invaluable mentorship that guided us forward. Without the support of each and every stakeholder, this project would have remained only an aspiration.

We are especially grateful to our Director, Rev. Fr. Kinley D'Cruz, and our Principal, Dr. Neena S. P. Panandikar, for their encouragement and for providing the facilities essential to this work. Your unwavering support was invaluable, and we are sincerely grateful.

Our sincere thanks go to our Head of Department, Dr. Gaurang Patkar, for his confidence in us and for his invaluable guidance, which enriched our journey with his extensive knowledge and insights.

Guides serve as beacons of direction in times of doubt and challenge, continually steering us toward our goals. In moments when the path seemed unclear, their supervision and wisdom helped us regain our focus. We are deeply grateful to our Internal Guides, Dr. Norman Dias and Prof. Amey Tilve from the Department of Computer Engineering, for their steadfast support and guidance throughout.

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

In a rapidly advancing educational landscape, the need for an efficient and centralized system to manage student information has become essential. The Student Information System (SIS) designed for the Computer Engineering College aims to streamline the management of student data, academic records, and administrative processes. This system integrates student details, course information, enrollment data, grades, instructor profiles, and departmental information, offering a comprehensive solution that enhances both academic and administrative functions. Traditional methods of handling such data—often relying on manual record-keeping or disconnected systems—prove to be inefficient and prone to errors. The SIS addresses these challenges by digitizing and centralizing all relevant student-related processes in one platform.

1.2 PROBLEM STATEMENT

The current approach to managing student and course-related information at the college level is fragmented, involving multiple platforms and manual input, which increases the risk of data loss, errors, and inefficiency. As student enrollment grows, the existing systems struggle to keep up with the demands for real-time data management, course selection, grade tracking, and communication between faculty and students. The absence of an integrated system hampers decision-making and administrative workflows. Therefore, the need for an advanced, centralized, and easily accessible system becomes evident to improve the management of student and academic data.

1.3 OBJECTIVES

The primary objectives of the SIS project include:

- **Centralizing Student Data**: To manage and store student profiles, including personal and academic information.
- **Course Management**: To store detailed course information and manage course offerings by departments.
- **Enrollment and Grade Tracking**: To track student enrollments in specific courses and maintain academic records.
- **Instructor and Department Management**: To store instructor details and facilitate course assignments by the Head of Department (HOD) to the teachers.

- **Administrative Efficiency**: To automate and improve the processes of course assignments, mentorship, and announcements.

1.4 SCOPE

This project is designed to cover the entire student lifecycle within the college, from the moment of enrollment until graduation. The system will handle student data, course information, enrollment, grades, instructor assignments, and department details. However, the scope does not extend to financial records, faculty performance evaluation, or external integrations with other university systems outside of the Computer Engineering College.

1.5 SIGNIFICANCE

The implementation of this MIS is crucial for enhancing both academic and administrative efficiency within the college. By streamlining the handling of vast amounts of data, the SIS reduces the likelihood of human error, improves access to accurate and up-to-date information, and facilitates better decision-making for both students and faculty. Furthermore, the system will support the academic growth of students by providing a clear, accessible platform for monitoring their progress. For administrators, the system's automation of tasks like course assignments and announcements will result in significant time savings, allowing them to focus on more strategic objectives. Ultimately, this SIS will contribute to improved operational efficiency and educational outcomes at the college.

CHAPTER 2

SYSTEM ANALYSIS

2.1 REQUIREMENTS GATHERING

2.1.1 FUNCTIONAL REQUIREMENTS

The Student Information System (SIS) is designed to provide a comprehensive set of features to manage various academic and administrative tasks. Key functionalities include:

- **Student Management:** Storage and retrieval of student details such as name, student ID, contact information, date of birth, and major field of study.
- **Course Management:** Recording and displaying course information, including course ID, name, description, credits, and the associated department.
- **Enrollment Management:** Tracking of student enrollments in courses, storing enrollment records with student and course IDs, semester, and year.
- **Grade Tracking:** Maintaining grade records linked to students and courses, enabling detailed academic performance tracking.
- **Instructor Management:** Storing instructor information, including their ID, name, contact details, and the courses they teach.
- **Department Operations:** Allowing the Head of Department (HOD) to assign courses to instructors and manage departmental tasks.
- **Mentorship and Announcements:** Enabling mentorship access to student details and managing the announcement system for both teachers and students.

2.1.2 NON - FUNCTIONAL REQUIREMENTS

To ensure the system operates efficiently and securely, several non-functional requirements are considered:

- **Performance:** The system should handle large volumes of student, course, and enrollment data without significant lag.
- **Security:** Student and instructor data must be securely stored, with role-based access control to restrict unauthorized access to sensitive information.
- **Scalability:** The system should be able to scale as the number of students, courses, and instructors increases over time.
- **Usability:** The user interface should be intuitive, allowing easy navigation for administrators, instructors, and students.

- **Reliability:** The system must be highly reliable, with minimal downtime to ensure smooth academic operations.

2.2 FEASIBILITY STUDY

2.2.1 TECHNICAL FEASIBILITY

The technical infrastructure required for the SIS is widely available and includes reliable database management systems, secure web technologies, and user-friendly interfaces. The required expertise in software development, database design, and web application development exists within the project team, making the project technically feasible.

2.2.2 ECONOMIC FEASIBILITY

The cost of implementing the SIS, including development, hardware, and software resources, is justified by the potential efficiency gains. By centralizing student information and streamlining administrative tasks, the system reduces the manual workload of staff and minimizes data redundancy, leading to significant long-term savings.

2.2.3 OPERATIONAL FEASIBILITY

The system aligns with the college’s current operations and procedures, ensuring smooth integration with minimal disruption. It enhances the existing processes by automating repetitive tasks and providing real-time access to information. The staff and faculty are expected to adapt to the new system with minimal training, further validating its operational feasibility.

CHAPTER 3

SYSTEM DESIGN

3.1 ARCHITECTURE DIAGRAM

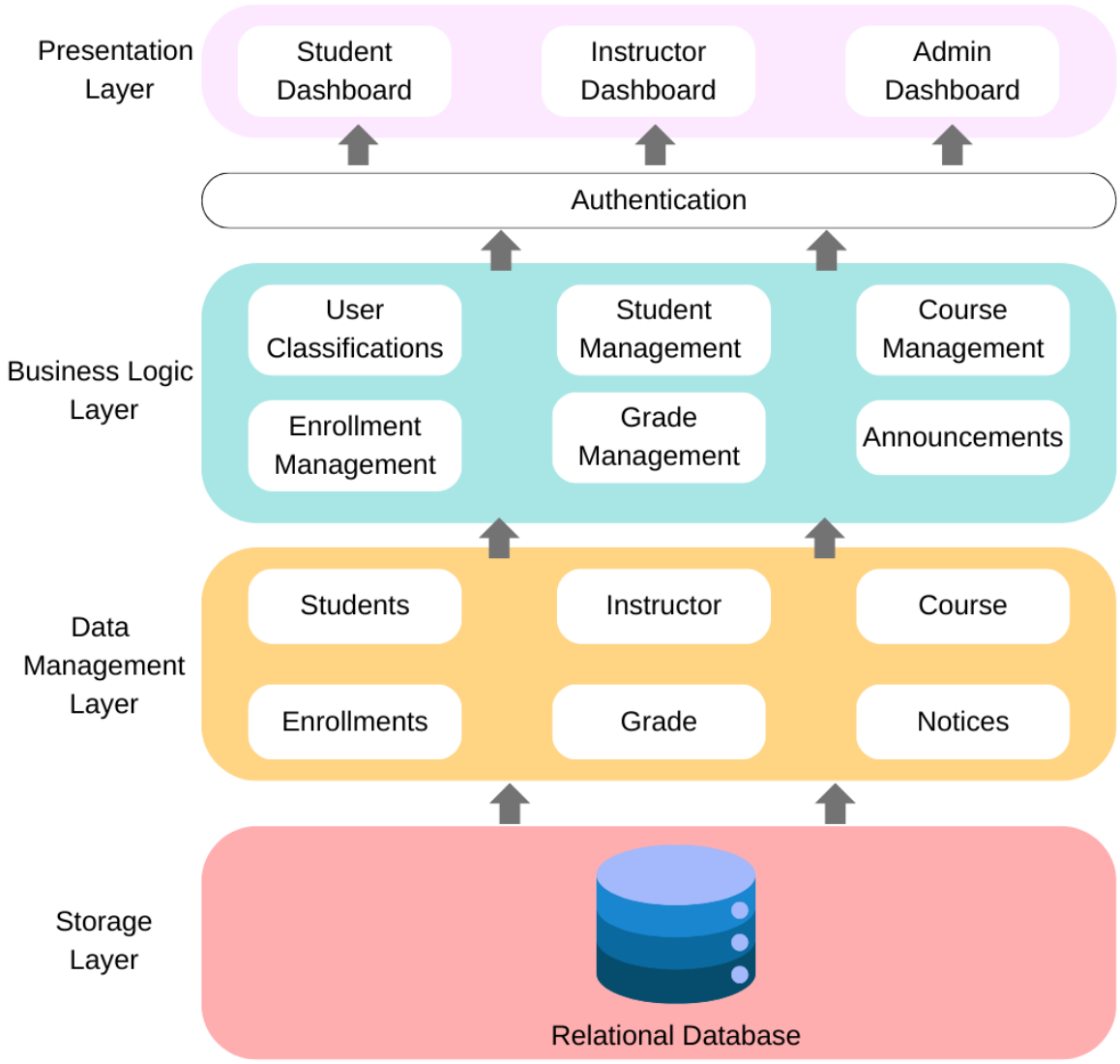

Fig: 3.1 System Architecture Diagram

The System Architecture diagram illustrates a multi-layered architecture for a Student Information System (SIS). At the base is the **Storage Layer** with a relational database that stores data on students, instructors, courses, enrollments, grades, and notices. Above it is the **Data Management Layer**, which organizes this data into categories like students, instructors, courses, enrollments, grades, and notices. The **Business Logic Layer** handles core system functions, including user classification, student and grade management, course management, enrollment processing, and announcements. The **Presentation Layer** offers user interfaces for students, instructors, and admins through dashboards specific to their roles. Between the business logic and presentation layers is an **Authentication** mechanism that manages secure access based on user roles and permissions.

3.2 DATABASE DESIGN

3.2.1 ENTITY RELATIONSHIP DIAGRAM

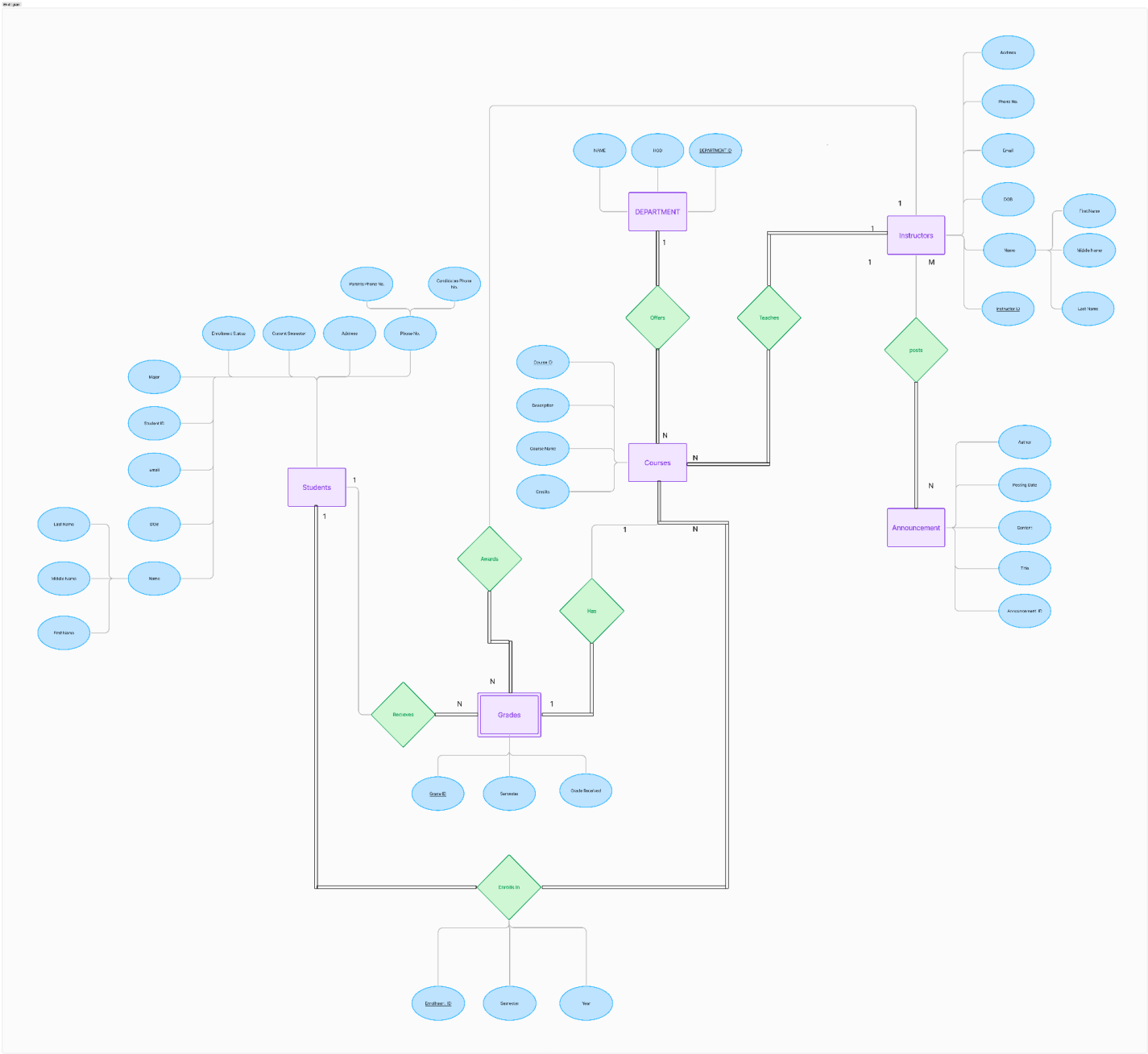

Fig: 3.2.1 Entity Relationship Diagram

3.2.2 SCHEMA DEFINITION

The provided ER diagram represents the entities and relationships between students, instructors, departments, courses, and grades.

- **Students:** Each student has attributes like Student_ID, First Name, Middle Name, Last Name, Current Semester, Date of Birth, Gender, and Address. A student can be enrolled in multiple courses, as shown by the relationship to the Enrolls_in entity.
- **Courses:** Courses are offered by departments, and each course has attributes like Course_ID, CourseName, Credits, and Description. A course can have multiple students enrolled and be taught by different instructors.
- **Instructors:** Instructors are associated with one or more courses, indicating that they can teach multiple courses. Each instructor has attributes like Instructor_ID, First Name, Last Name, Specialization, Rank, and Contact Information.
- **Grades:** The system tracks grades for students, with attributes such as IT1, IT2, IT3, Semester, and Grade Review. The grades are connected to both students and courses, signifying that each student receives a grade for a course.
- **Departments:** Each department can offer multiple courses and have several instructors. Departments have attributes like Dept_ID, Name, and Contact Information.
- **Announcements:** The diagram also includes an Announcement entity, which is related to the department. Announcements have attributes like Author, Title, Date, and Message, suggesting that departments can communicate updates or information to students and instructors.

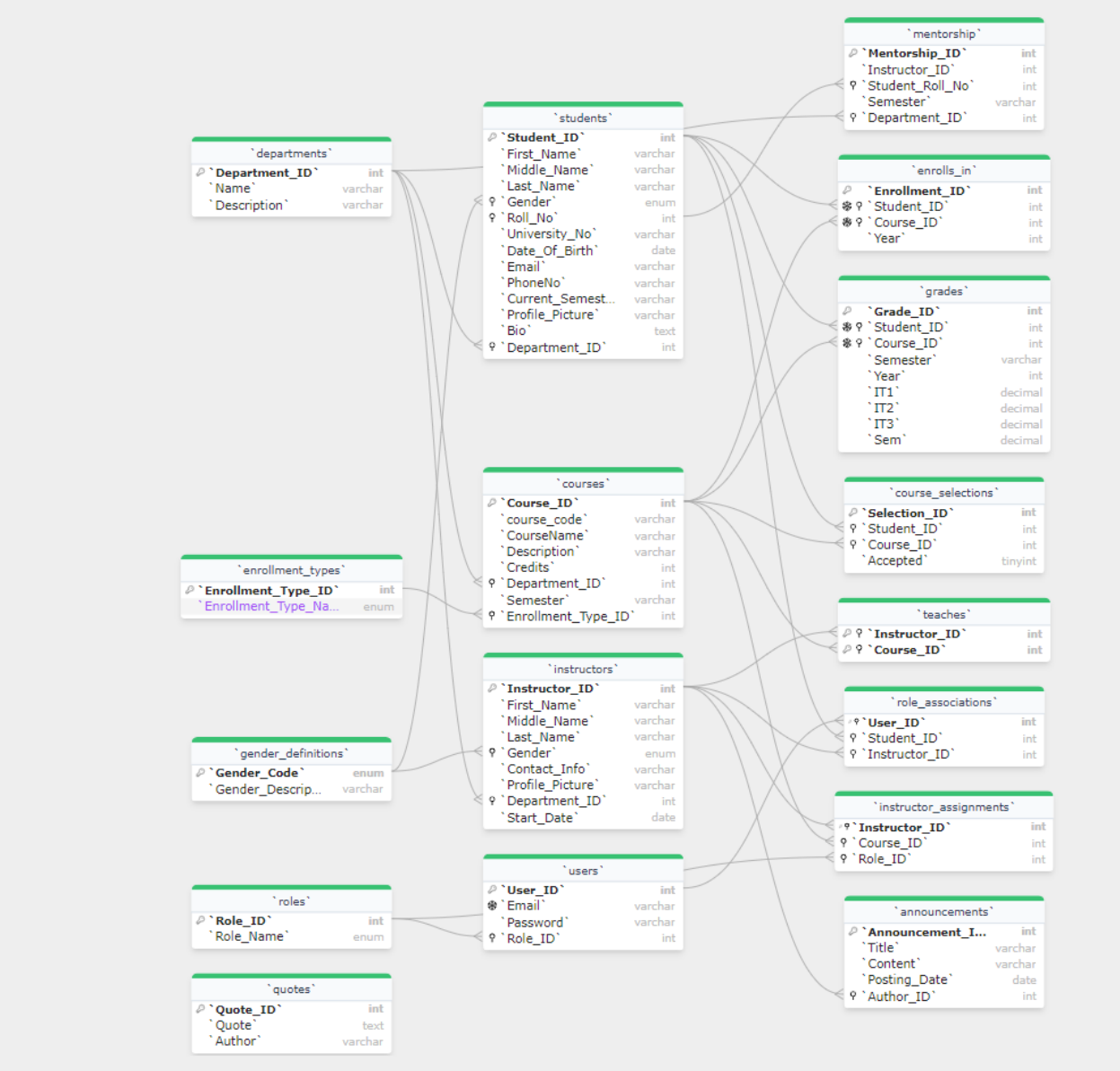

Fig: 3.2.2 Schema Diagram

3.2.3 NORMALIZATION

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	announcements	Browse Structure Search Insert Empty Drop	15	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	courses	Browse Structure Search Insert Empty Drop	40	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	course_selections	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	departments	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	enrollment_types	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	enrolls_in	Browse Structure Search Insert Empty Drop	44	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐	gender_definitions	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	grades	Browse Structure Search Insert Empty Drop	18	InnoDB	utf8mb4_general_ci	80.0 KiB	-
☐	instructors	Browse Structure Search Insert Empty Drop	16	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	instructor_assignments	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	mentorship	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	quotes	Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	roles	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	role_associations	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	students	Browse Structure Search Insert Empty Drop	15	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐	student_sgpa_cgpa	Browse Structure Search Insert Edit Drop	~0	View	---	-	-
☐	teaches	Browse Structure Search Insert Empty Drop	18	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	users	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	48.0 KiB	-
	18 tables	Sum	~222	InnoDB	utf8mb4_general_ci	688.0 KiB	0 B

Fig: 3.2.3.1 Student Database

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	Grade_ID	int(11)			No	None		AUTO_INCREMENT
☐	2	Student_ID	int(11)			No	None		
☐	3	Course_ID	int(11)			No	None		
☐	4	Semester	varchar(20)	utf8mb4_general_ci		No	None		
☐	5	Year	int(11)			No	None		
☐	6	IT1	decimal(5,2)			Yes	NULL		
☐	7	IT2	decimal(5,2)			Yes	NULL		
☐	8	IT3	decimal(5,2)			Yes	NULL		
☐	9	Sem	decimal(5,2)			Yes	NULL		

Fig: 3.2.3.2 Grades Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Instructor_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**First_Name**	varchar(50)	utf8mb4_general_ci		No	*None*		
3	**Middle_Name**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*		
4	**Last_Name**	varchar(50)	utf8mb4_general_ci		No	*None*		
5	**Gender**	enum('M', 'F', 'X')	utf8mb4_general_ci		No	*None*		
6	**Contact_Info**	varchar(150)	utf8mb4_general_ci		Yes	*NULL*		
7	**Profile_Picture**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*		
8	**Department_ID**	int(11)			Yes	*NULL*		
9	**Start_Date**	date			Yes	*NULL*		

Fig: 3.2.3.3 Instructors Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Instructor_ID**	int(11)			No	*None*		
2	**Course_ID**	int(11)			Yes	*NULL*		
3	**Role_ID**	int(11)			Yes	*NULL*		

Fig: 3.2.3.4 Instructor_Assignments Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Mentorship_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Instructor_ID**	int(11)			No	*None*		
3	**Student_Roll_No**	int(11)			No	*None*		
4	**Semester**	varchar(20)	utf8mb4_general_ci		No	*None*		
5	**Department_ID**	int(11)			No	*None*		

Fig: 3.2.3.5 Mentorship Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Quote_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Quote**	text	utf8mb4_general_ci		No	*None*		
3	**Author**	varchar(100)	utf8mb4_general_ci		No	*None*		

Fig: 3.2.3.6 Quotes Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Role_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Role_Name**	enum('Student', 'Teacher', 'Admin', 'HOD', 'ClassT...	utf8mb4_general_ci		No	*None*		

Fig: 3.2.3.7 Roles Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Instructor_ID	int(11)			No	None		
2	Course_ID	int(11)			No	None		

Fig: 3.2.3.8 Teaches Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	User_ID	int(11)			No	None		AUTO_INCREMENT
2	Email	varchar(150)	utf8mb4_general_ci		No	None		
3	Password	varchar(255)	utf8mb4_general_ci		No	None		
4	Role_ID	int(11)			No	None		

Fig: 3.2.3.9 Users Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Announcement_ID	int(11)			No	None		AUTO_INCREMENT
2	Title	varchar(100)	utf8mb4_general_ci		No	None		
3	Content	varchar(500)	utf8mb4_general_ci		No	None		
4	Posting_Date	date			No	None		
5	Author_ID	int(11)			No	None		

Fig: 3.2.3.10 Announcements Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Selection_ID	int(11)			No	None		AUTO_INCREMENT
2	Student_ID	int(11)			No	None		
3	Course_ID	int(11)			No	None		
4	Accepted	tinyint(1)			Yes	0		

Fig: 3.2.3.11 Course_Selections Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Course_ID	int(11)			No	None		AUTO_INCREMENT
2	course_code	varchar(50)	utf8mb4_general_ci		No	None		
3	CourseName	varchar(100)	utf8mb4_general_ci		No	None		
4	Description	varchar(255)	utf8mb4_general_ci		No	None		
5	Credits	int(11)			No	None		
6	Department_ID	int(11)			No	None		
7	Semester	varchar(10)	utf8mb4_general_ci		No	None		
8	Enrollment_Type_ID	int(11)			Yes	NULL		

Fig: 3.2.3.12 Courses Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Department_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Name**	varchar(100)	utf8mb4_general_ci		No	*None*		
3	**Description**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*		

Fig: 3.2.3.13 Departments Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Enrollment_Type_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Enrollment_Type_Name**	enum('Major', 'Minor', 'Professional Elective', 'O...	utf8mb4_general_ci		No	*None*		

Fig: 3.2.3.14 Enrollment_Types Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Enrollment_ID**	int(11)			No	*None*		AUTO_INCREMENT
2	**Student_ID**	int(11)			No	*None*		
3	**Course_ID**	int(11)			No	*None*		
4	**Year**	int(11)			No	*None*		

Fig: 3.2.3.15 Enrolls_In Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**Gender_Code**	enum('M', 'F', 'X')	utf8mb4_general_ci		No	*None*		
2	**Gender_Description**	varchar(20)	utf8mb4_general_ci		No	*None*		

Fig: 3.2.3.16 Gender_Definitions Table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**User_ID**	int(11)			No	*None*		
2	**Student_ID**	int(11)			Yes	*NULL*		
3	**Instructor_ID**	int(11)			Yes	*NULL*		

Fig: 3.2.3.17 Role_Associations Table

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**Student_ID**	int(11)			No	0		
☐	2	**First_Name**	varchar(50)	utf8mb4_general_ci		No	*None*		
☐	3	**Last_Name**	varchar(50)	utf8mb4_general_ci		No	*None*		
☐	4	**SGPA_Sem1**	decimal(5,4)			Yes	*NULL*		
☐	5	**SGPA_Sem2**	decimal(5,4)			Yes	*NULL*		
☐	6	**SGPA_Sem3**	decimal(5,4)			Yes	*NULL*		
☐	7	**SGPA_Sem4**	decimal(5,4)			Yes	*NULL*		
☐	8	**SGPA_Sem5**	decimal(5,4)			Yes	*NULL*		
☐	9	**SGPA_Sem6**	decimal(5,4)			Yes	*NULL*		
☐	10	**SGPA_Sem7**	decimal(5,4)			Yes	*NULL*		
☐	11	**SGPA_Sem8**	decimal(5,4)			Yes	*NULL*		
☐	12	**CGPA**	decimal(5,4)			Yes	*NULL*		

Fig: 3.2.3.18 CGPA SGPA View

	Name	Table	Time	Event
☐	**after_course_acceptance**	course_selections	AFTER	UPDATE
☐	**after_course_insert**	courses	AFTER	INSERT
☐	**auto_enroll_core_courses**	students	AFTER	INSERT
☐	**auto_enroll_core_courses_after_update**	students	AFTER	UPDATE
☐	**update_instructor_email**	instructors	AFTER	UPDATE
☐	**update_student_email**	students	AFTER	UPDATE

Fig: 3.2.3.19 Triggers

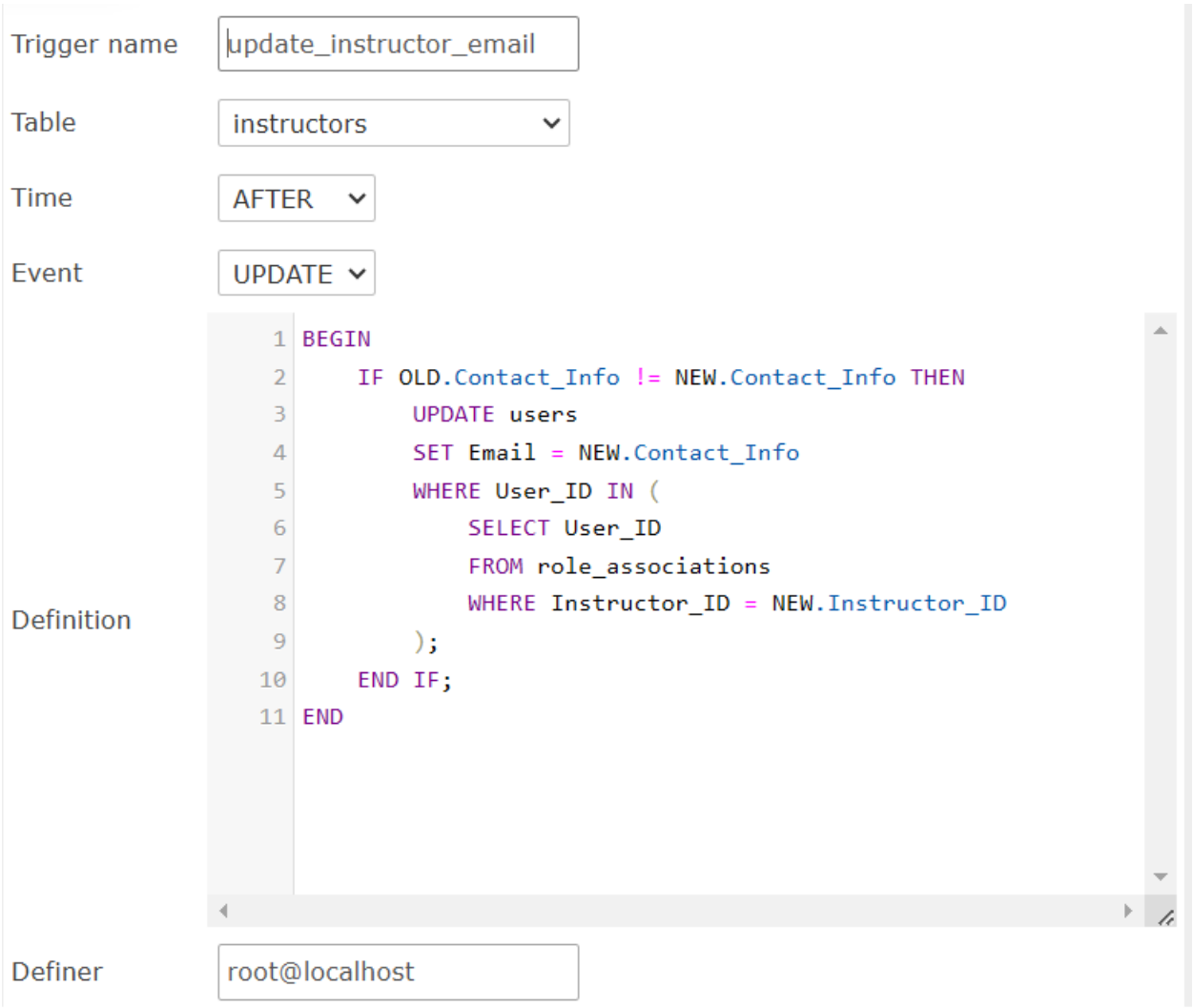

Fig: 3.2.3.20 update_instructor_email Trigger

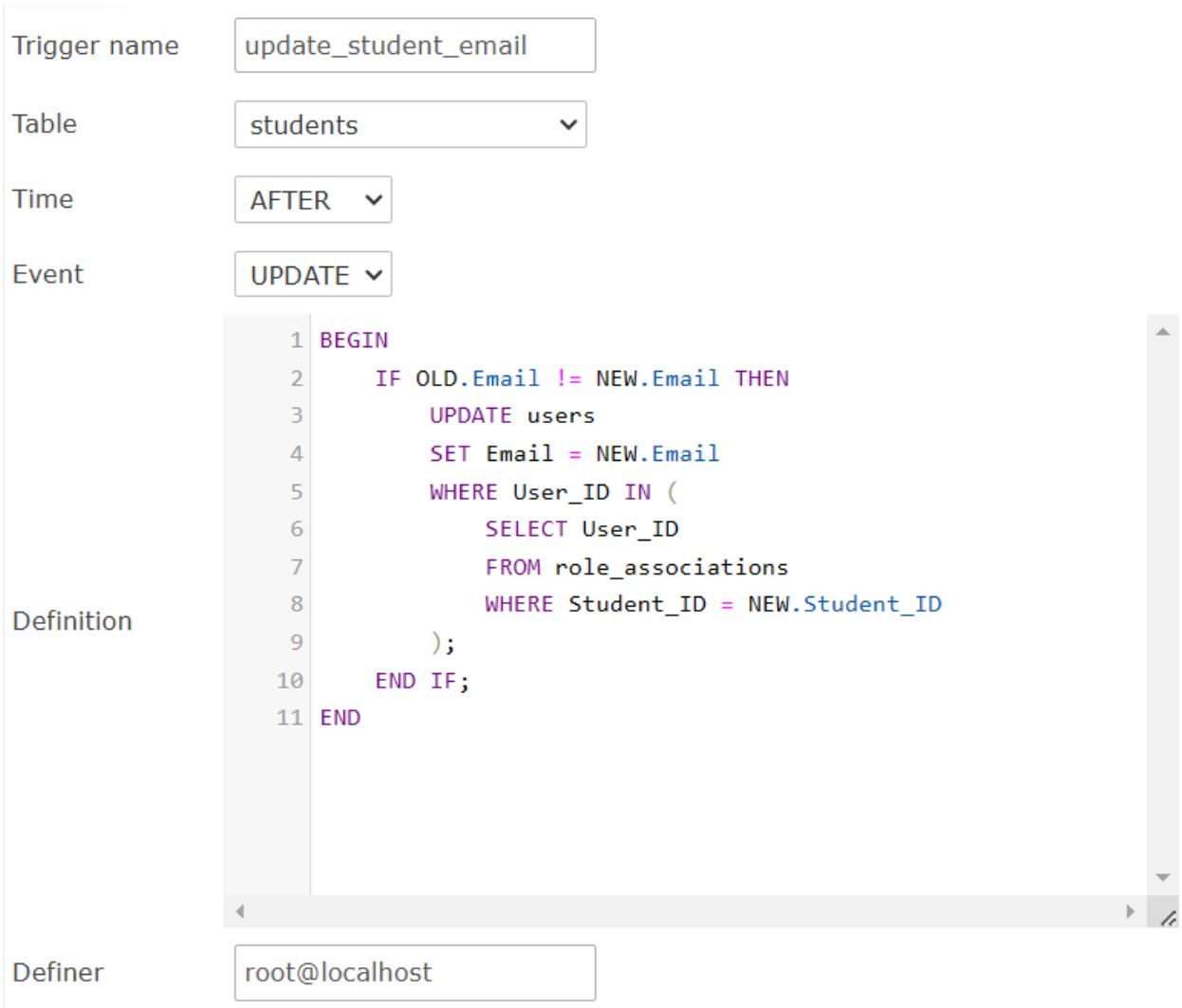

Fig: 3.2.3.21 update_student_email Trigger

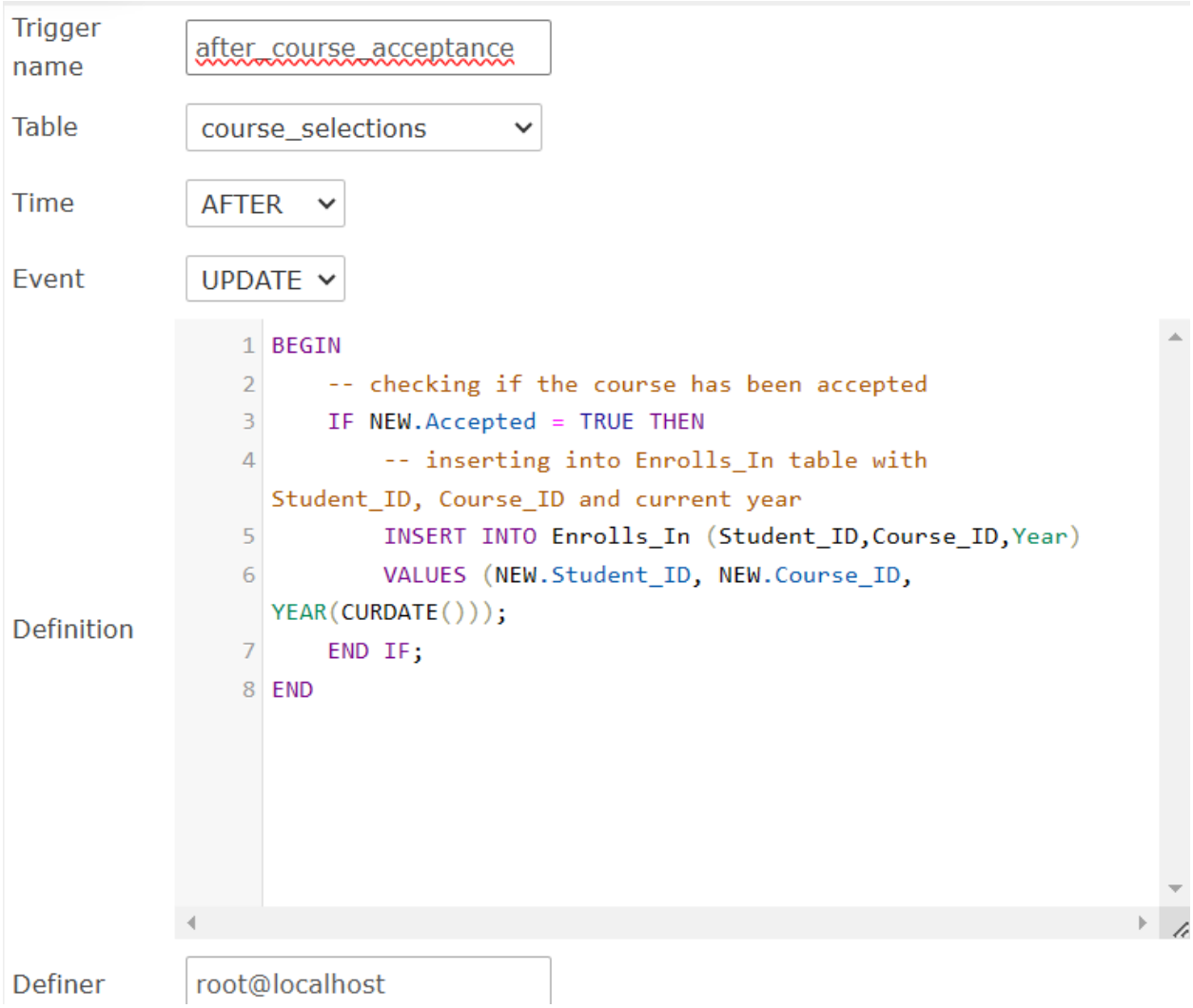

Fig: 3.2.3.22 after_course_acceptance Trigger

Trigger name: after_course_insert

Table: courses

Time: AFTER

Event: INSERT

Definition:

```
BEGIN
    -- Checking if the inserted course is of type 'Core' (i.e Enrollment_Type_ID = 5)
    IF NEW.Enrollment_Type_ID = 5 THEN
        -- Inserting enrollment records for all students of the same department and semester as that of the core course
        INSERT INTO enrolls_in (Student_ID, Course_ID, Year)
        SELECT s.Student_ID, NEW.Course_ID, YEAR(CURDATE())
        FROM students s
        WHERE s.Department_ID = NEW.Department_ID
          AND s.Current_Semester = NEW.Semester;
    END IF;
```

Definer: root@localhost

Fig: 3.2.3.23 after_course_insert Trigger

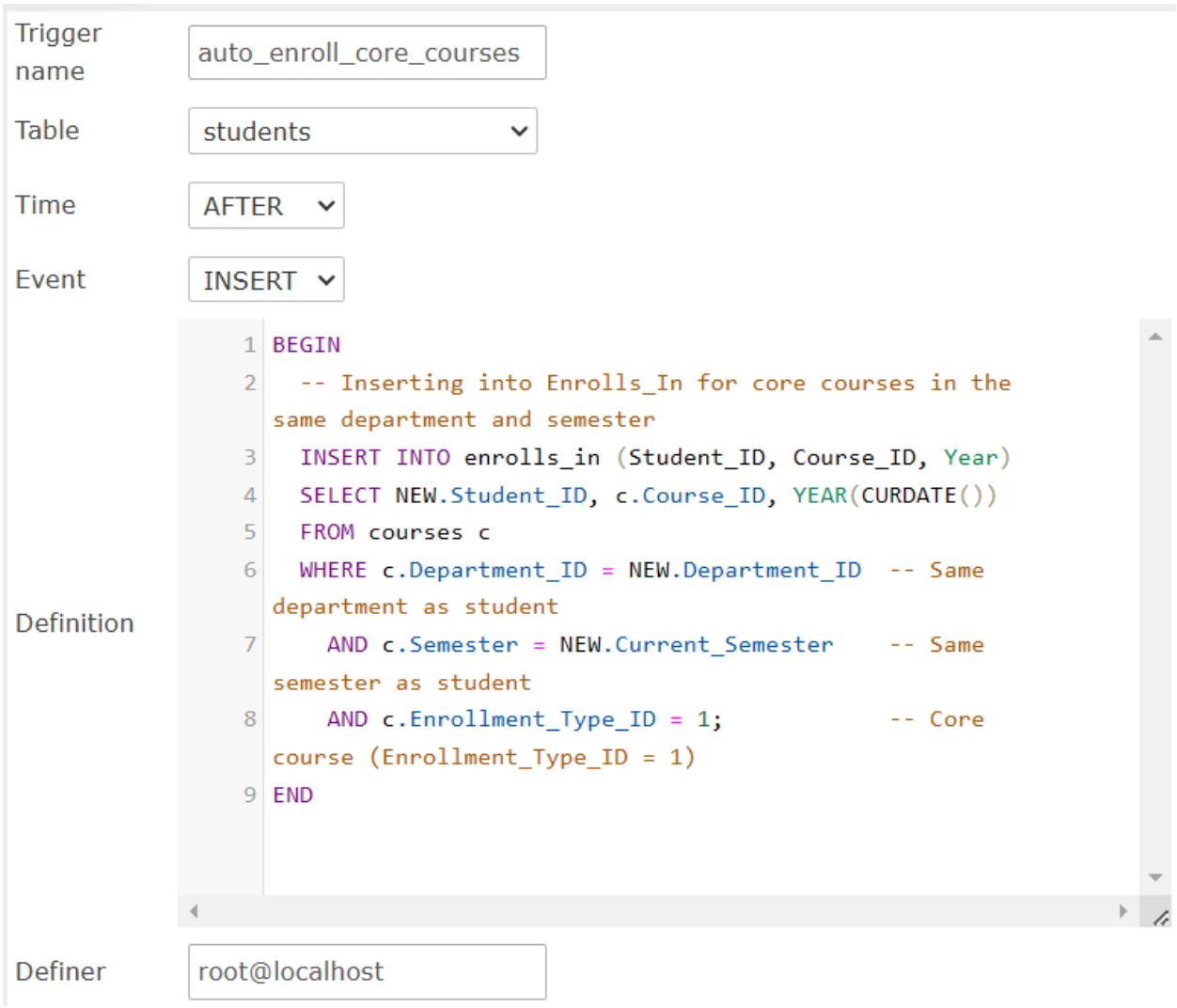

Fig: 3.2.3.24 auto_enroll_core_courses Trigger 1.1

Trigger name: auto_enroll_core_courses_

Table: students

Time: AFTER

Event: UPDATE

Definition:

```
BEGIN
  -- Check if the department or current semester has been changed
  IF NEW.Department_ID != OLD.Department_ID OR NEW.Current_Semester !=
OLD.Current_Semester THEN
    -- Insert into Enrolls_In for core courses in the updated department and semester
    INSERT INTO enrolls_in (Student_ID, Course_ID, Year)
    SELECT NEW.Student_ID, c.Course_ID, YEAR(CURDATE())
    FROM courses c
    WHERE c.Department_ID = NEW.Department_ID  -- Same updated department as student
      AND c.Semester = NEW.Current_Semester    -- Same updated semester as student
      AND c.Enrollment_Type_ID = 1;            -- Core course (Enrollment_Type_ID = 1)
  END IF;
END
```

Fig: 3.2.3.25 auto_enroll_core_courses Trigger 1.2

3.3 MODULE DESIGN

The Student Information System (SIS) for the Computer Engineering College is designed to optimize the management of student data, course details, enrollment records, instructor assignments, and other administrative functions. The system is modular, with each component focusing on a specific aspect of academic and student administration, enabling streamlined workflows and improving efficiency.

3.3.1 STUDENT MANAGEMENT MODULE:

This module is central to the SIS, as it handles all aspects of student information. It stores personal details such as names, contact information, date of birth, and major field of study. Each student is assigned a unique `Student_ID`, and their academic journey is tracked through this module. Students can also update their profiles and view their current semester, roll number, and university ID.

Key Functions:

- Profile management (view/update personal information)
- View academic progress (current semester, grades, and enrollment status)
- Access announcements and updates

3.3.2. COURSE MANAGEMENT MODULE:

The course management module contains detailed information about the courses offered at the college, including `Course_ID`, course name, description, credit hours, and the department offering the course. It allows departments to manage the courses each semester and helps students in selecting the right courses based on their major or requirements.

- **Key Functions:**
  - Store and manage course details (description, credits, department)
  - Manage course offerings for each semester
  - Support course enrollment and scheduling

3.3.3. ENROLLMENT MANAGEMENT MODULE:

The enrollment management module handles the tracking of student course registrations. It records which students are enrolled in which courses using foreign key relationships between `students` and `courses`. It also stores

information about the enrollment type (e.g., major, minor, elective) and keeps track of the semester and year of enrollment.

Key Functions:

- Track student enrollment in courses by semester and year
- Support course registration and approval by teachers
- Display enrolled courses and allow for viewing or updating selections

3.3.4. GRADE TRACKING MODULE:

The grade tracking module maintains academic performance records for each student. Grades are associated with the courses a student has completed, and the module keeps track of individual test scores (`IT1`, `IT2`, `IT3`, and final semester grades) for each course. This module is critical for students to view their academic progress and for teachers to enter and update grades.

Key Functions:

- Record and update grades for each student
- View detailed academic performance across semesters
- Provide access to cumulative GPA (SGPA/CGPA) tracking

3.3.5. INSTRUCTOR MANAGEMENT MODULE:

The instructor management module stores details of faculty members, including their `Instructor_ID`, contact information, and the courses they are assigned to teach. The module ensures that instructor assignments are linked to the appropriate courses and departments, facilitating smoother administrative processes.

Key Functions:

- Manage instructor profiles and contact details
- Track the courses assigned to each instructor
- Support course assignments by the Head of Department (HOD)

3.3.6. DEPARTMENT MANAGEMENT MODULE:

This module supports the operations of various departments in the college. It enables departments to manage courses, assign instructors, and handle other administrative tasks. The `departments` table stores department-specific details,

and this module links students, instructors, and courses to the correct departments.

Key Functions:

- Manage department-specific course offerings
- Allow HODs to assign courses to instructors
- Oversee department administration tasks

3.3.7. MENTORSHIP AND ANNOUNCEMENTS MODULE:

This module serves as a communication and mentorship platform within the system. Mentors can access student information to provide guidance, while the announcements functionality allows teachers and administrators to post updates and important messages. This ensures smooth communication between students, faculty, and administration.

Key Functions:

- Enable mentorship access to student academic records
- Allow teachers to post announcements for students
- Provide a platform for communication and notifications

3.3.8. ROLE AND ACCESS CONTROL MODULE:

This module is responsible for handling access permissions and user roles within the system. There are predefined roles such as `student`, `teacher`, and `admin`, each with distinct levels of access. Administrators use this module to grant and manage access to various parts of the system based on user roles.

Key Functions:

- Define and manage user roles (student, teacher, admin)
- Grant permissions to different sections of the SIS based on roles
- Control access to sensitive data like grades, course details, and student profiles

3.3.9. INTERFACES FOR DIFFERENT USERS:

1. Student Interface:

The student interface provides students with an intuitive platform to view and manage their academic life. They can view their grades, enroll in courses, update

their profiles, track their progress across semesters, and view announcements from their instructors.

Key Functions:

- View and update personal profiles
- Enroll in courses and track academic progress
- View grades, cumulative GPA, and announcements

2. Teacher Interface:

The teacher interface enables faculty members to manage the academic progress of their students. Teachers can view student data, grade them, approve their course selections, and access their mentees' progress. HODs have additional permissions to assign courses to teachers and oversee departmental tasks.

Key Functions:

- Grade students and approve course selections
- View student profiles and academic performance
- Mentor assigned students and manage mentorship program
- HOD functions for course assignments and department management

3. Admin Interface:

The admin interface is a powerful tool that provides administrators with full control over the system. Admins can assign roles, manage access permissions, oversee announcements, and ensure data security. This interface allows them to ensure that the right users have access to the right data and functionality.

Key Functions:

- Assign roles and manage user permissions
- Oversee system operations and access controls
- Manage and moderate announcements and updates across the system.

CHAPTER 4

IMPLEMENTATION

4.1 DEVELOPMENT ENVIRONMENT

Tools and Technologies Used in the Student Information System

1. **PHP (Version 8.3)**: Used for server-side scripting and building backend logic to process and manage the system's data.
2. **MySQL (Version 8.0)**: Acts as the relational database to store and manage student, instructor, course, and grade information efficiently.
3. **Apache (Version 2.4)**: Serves as the web server that hosts and delivers the web application to users.
4. **XAMPP**: Provides a local development environment for testing the PHP-MySQL-based web application.
5. **PDO (PHP Data Objects)**: Used to securely interact with the MySQL database using prepared statements.
6. **HTML5 & CSS3**: Provide the structure and design for the user interface of the web pages.
7. **JavaScript**: Adds interactivity to the system, handling tasks like form validation and asynchronous updates.
8. **JSON (JavaScript Object Notation)**: Facilitates data exchange between the server and client by sending and receiving structured data in a lightweight format.
9. **Figma**: A design tool used to create wireframes and prototypes for the system's user interface before implementation.
10. **phpMyAdmin**: A web-based tool used to manage the MySQL database, making it easier to handle data.
11. **Visual Studio Code**: Integrated Development Environments (IDEs) used for writing and managing code.
12. **Git & GitHub**: Used for version control, enabling collaboration and tracking changes to the codebase.

4.2 KEY FEATURES IMPLEMENTATION

- **User Authentication and Role-Based Access Control:**
  s

- **Student Management:**
  A module to manage student profiles, including personal details, contact information, enrollment status, and current semester, with the ability to view individual records.

- **Course and Enrollment Management:**
  Allows administrators to create, edit, and delete course information, while students can enroll in available courses. Enrollment details are tied to student records and are trackable by year and type.

- **Grade Allocation and Tracking:**
  Teachers can input and update student grades for specific assessments, and students can view their grades in real-time. The system supports multiple assessment types (internal tests, final exams) for a comprehensive view of academic performance.

- **Announcements:**
  Enables instructors and administrators to post announcements related to courses or general school notices. Students can view these updates in their dashboard to stay informed of important events.

- **Mentorship Access:**
  Implemented limited viewing access for mentors, allowing them to see mentee details such as academic performance and attendance, facilitating better student guidance and counseling.

- **Dashboard with Real-Time Information:**
  A dynamic homepage showing users their classes, courses, upcoming assessments, latest announcements, providing a comprehensive overview.

- **Profile Management:**
  Users can manage their profiles, update personal information, and customize their bio and contact details, while teachers can upload their profiles with their details.

- **Database Management:**

A robust database system (MySQL) that organizes and manages large amounts of academic data, ensuring fast and secure data retrieval and updates.

4.3 USER INTERFACE DESIGN

- **Students:**

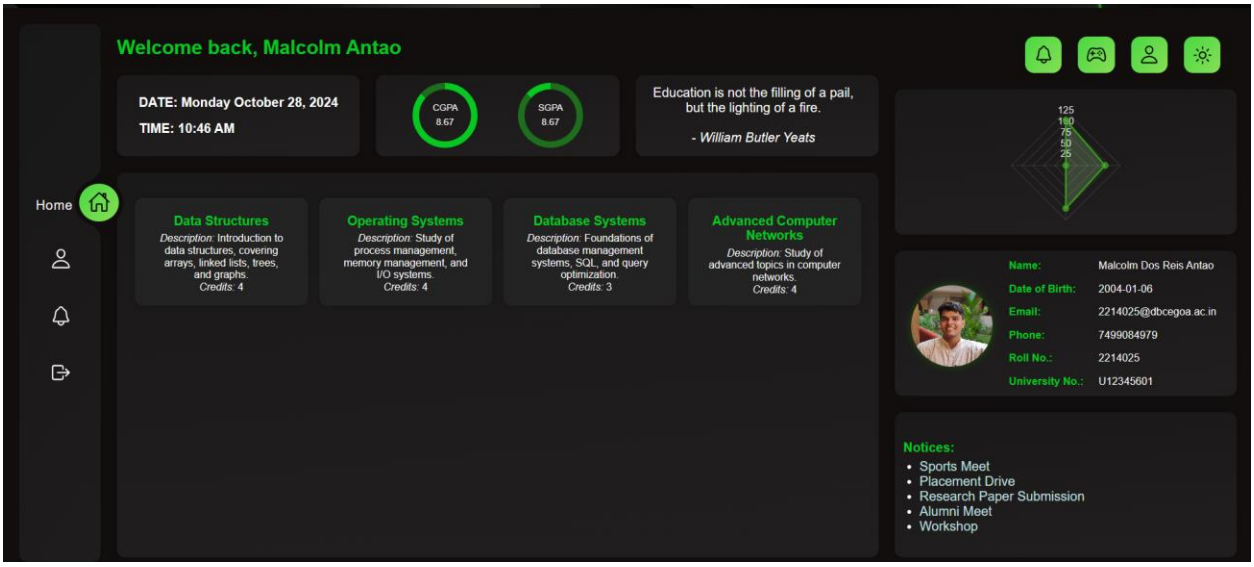

Fig: 4.3.1 Student Landing Page 1.1

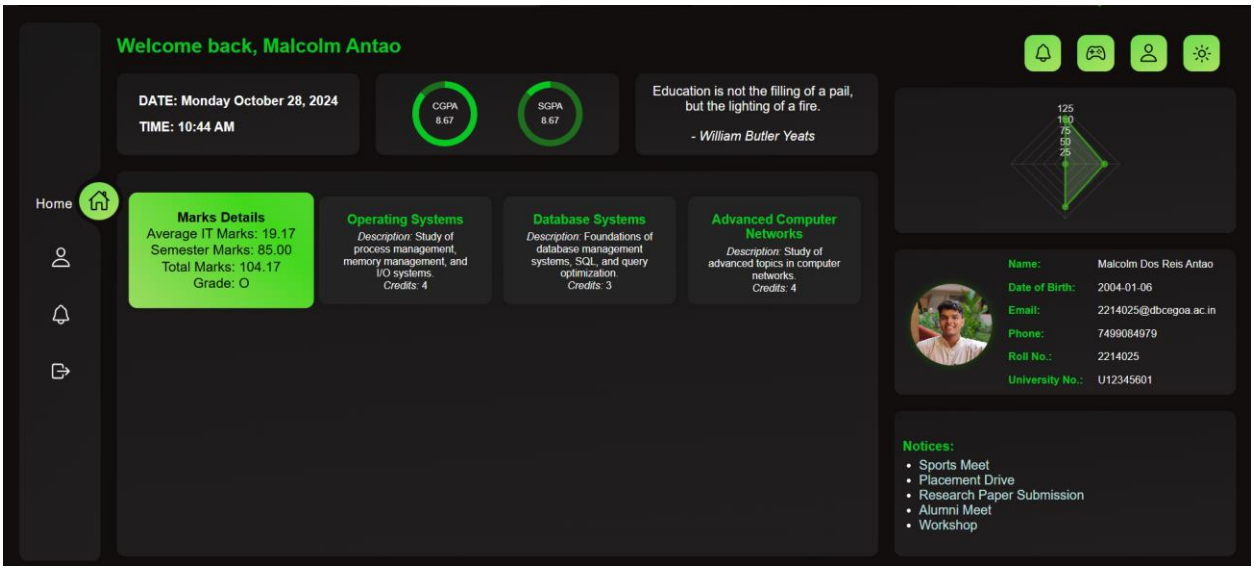

Fig: 4.3.2 Student Landing Page 1.2

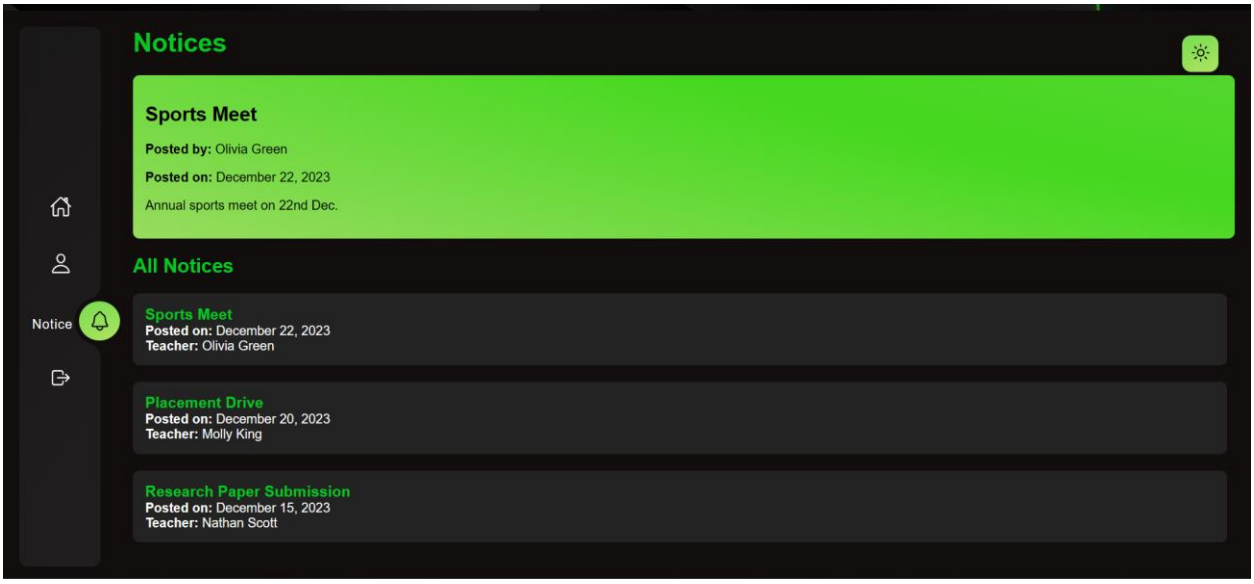

Fig: 4.3.3 Student Notice Page

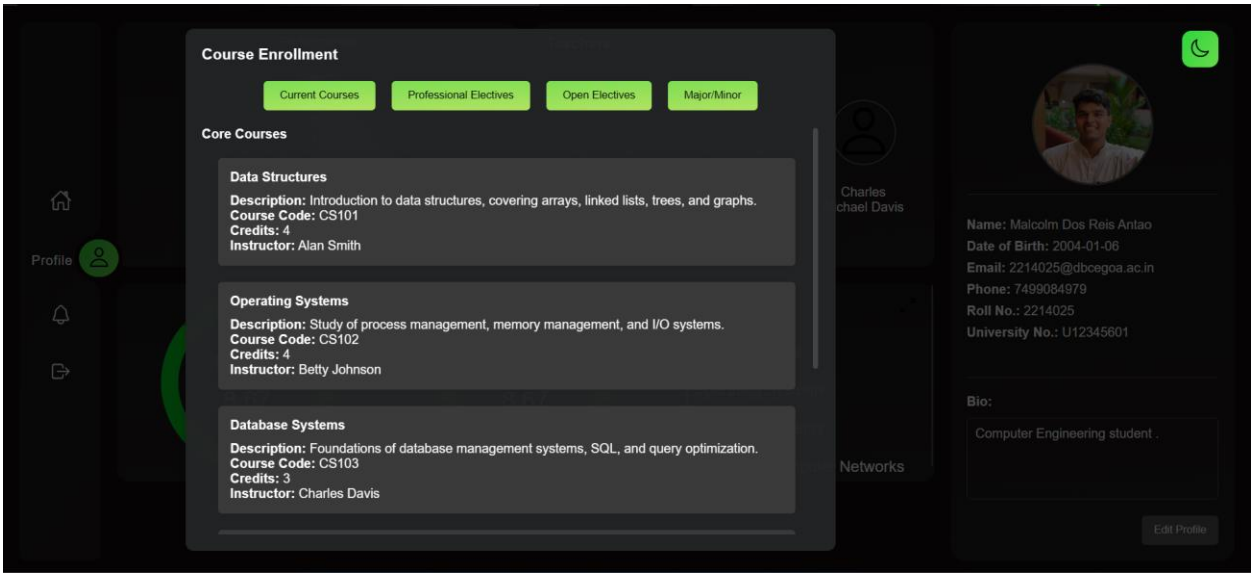

Fig: 4.3.4 Student Core Courses Page

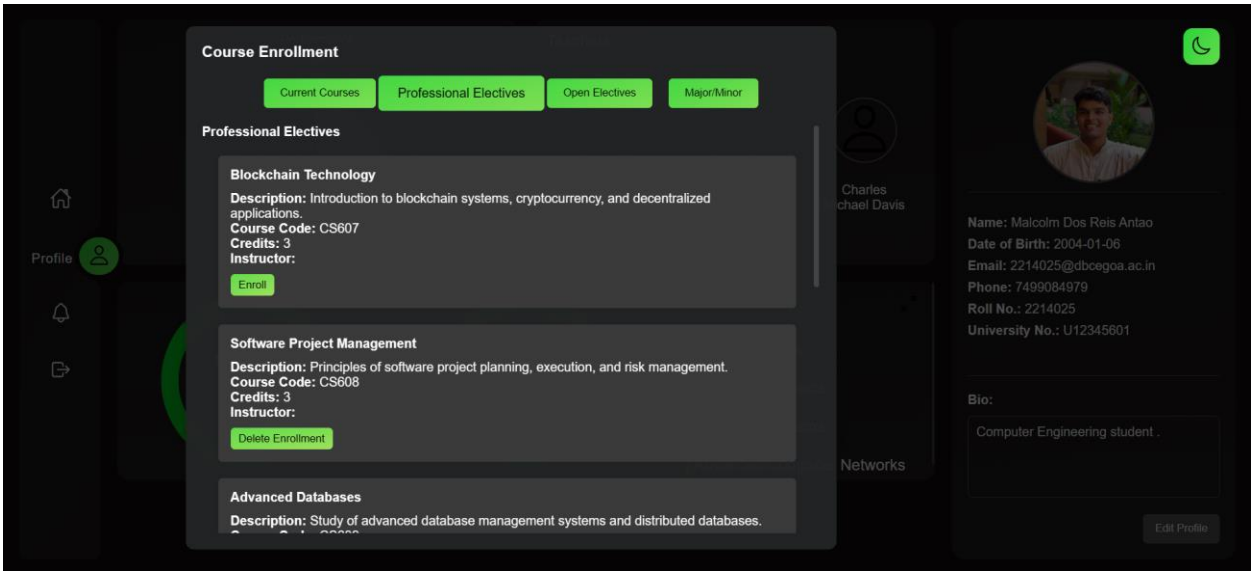

Fig: 4.3.5 Student Professional Electives Page

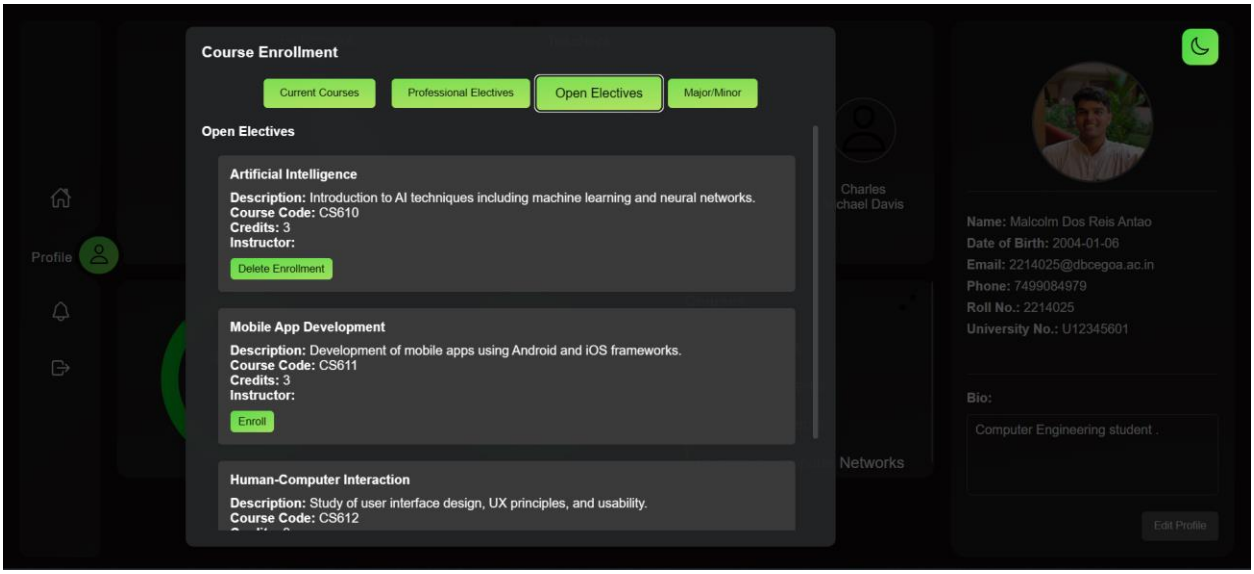

Fig: 4.3.6 Student Open Electives Page

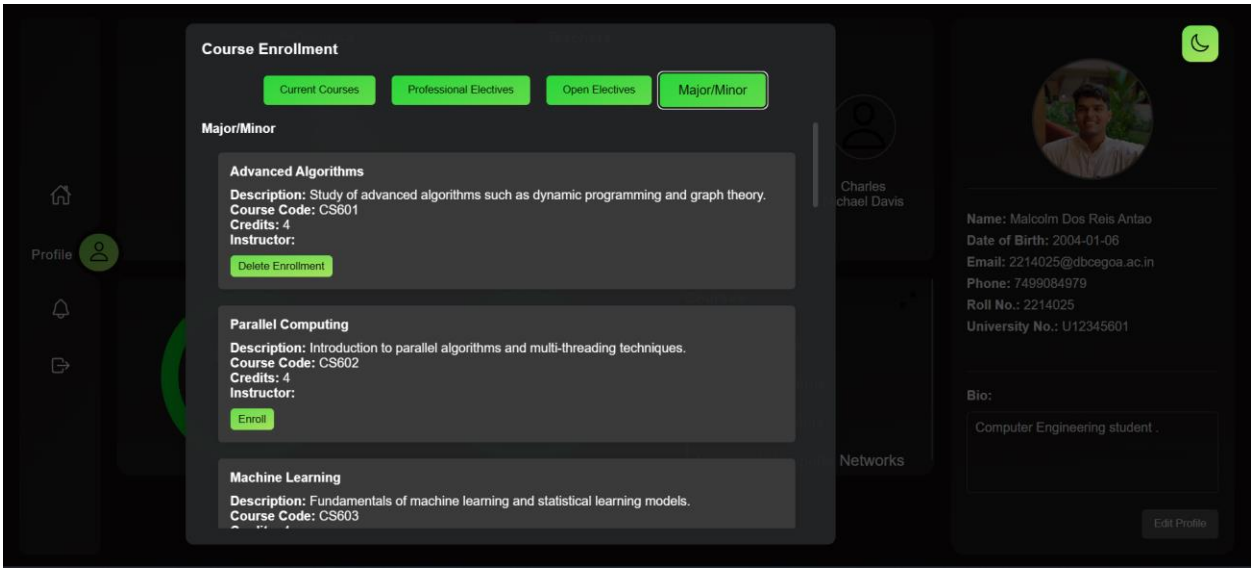

Fig: 4.3.7 Student Major/Minor Page

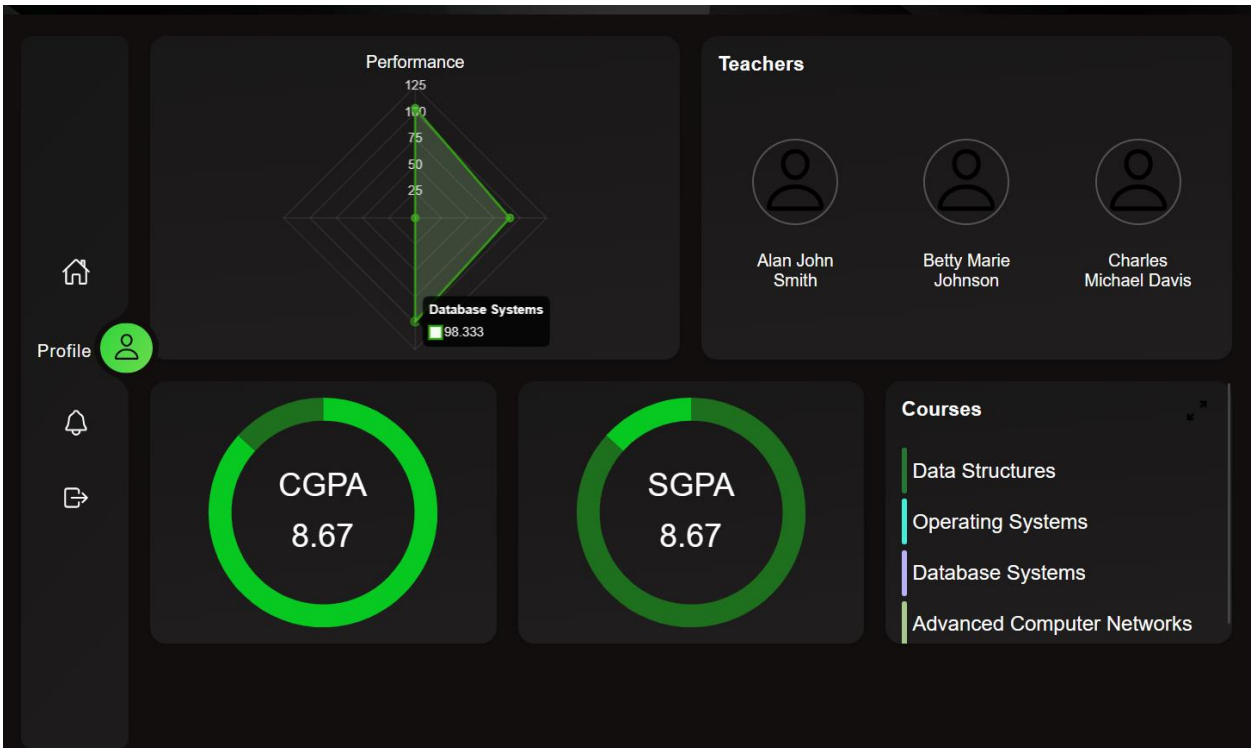

Fig: 4.3.8 Student Profile Page 1.1

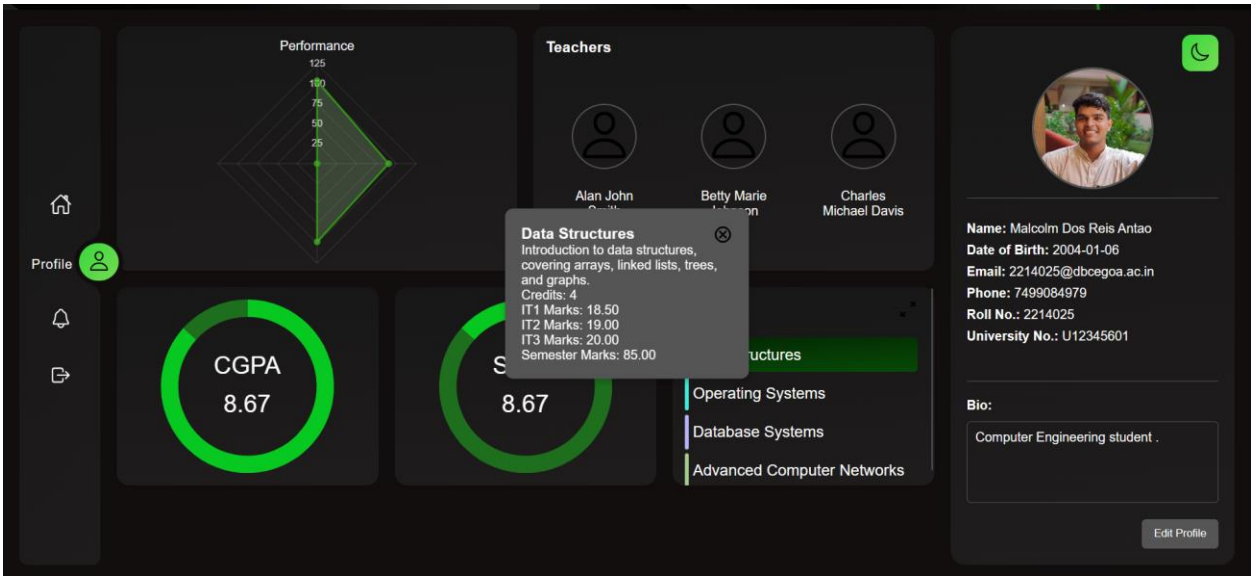

Fig: 4.3.9 Student Profile Page 1.2

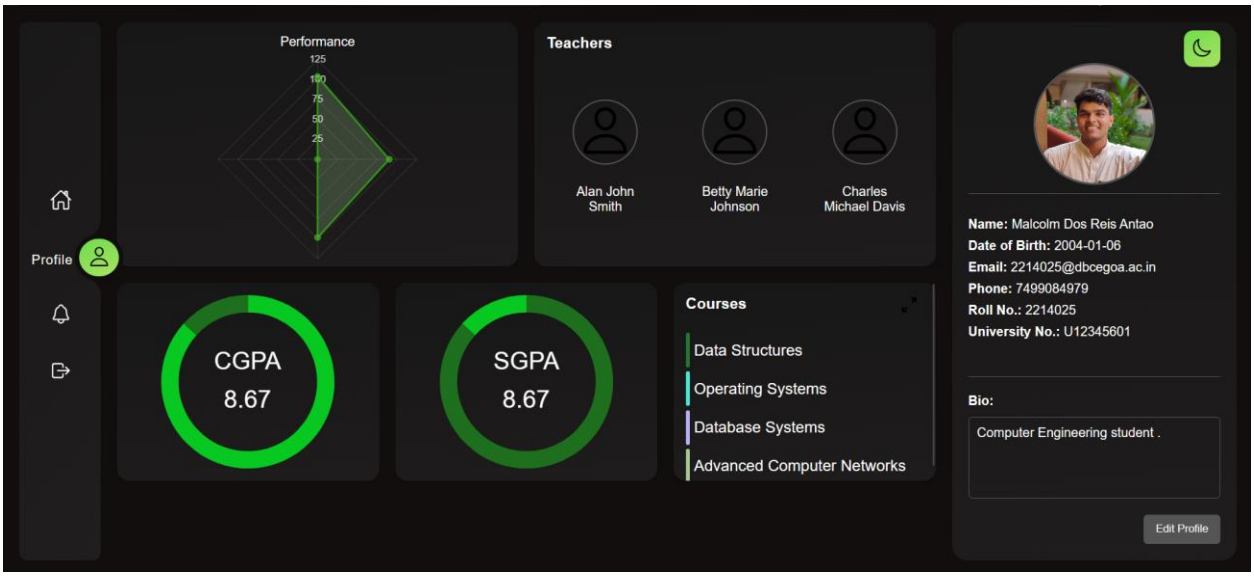

Fig: 4.3.10 Student Profile Page 1.3

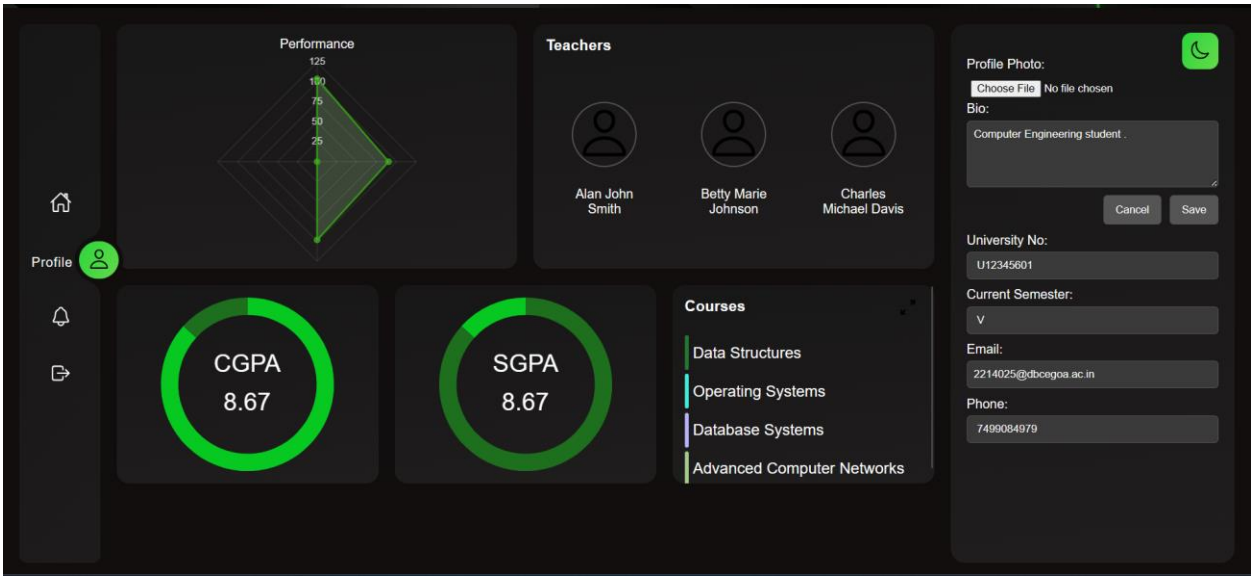

Fig: 4.3.11 Student Profile Page 1.4

- **Admin :**

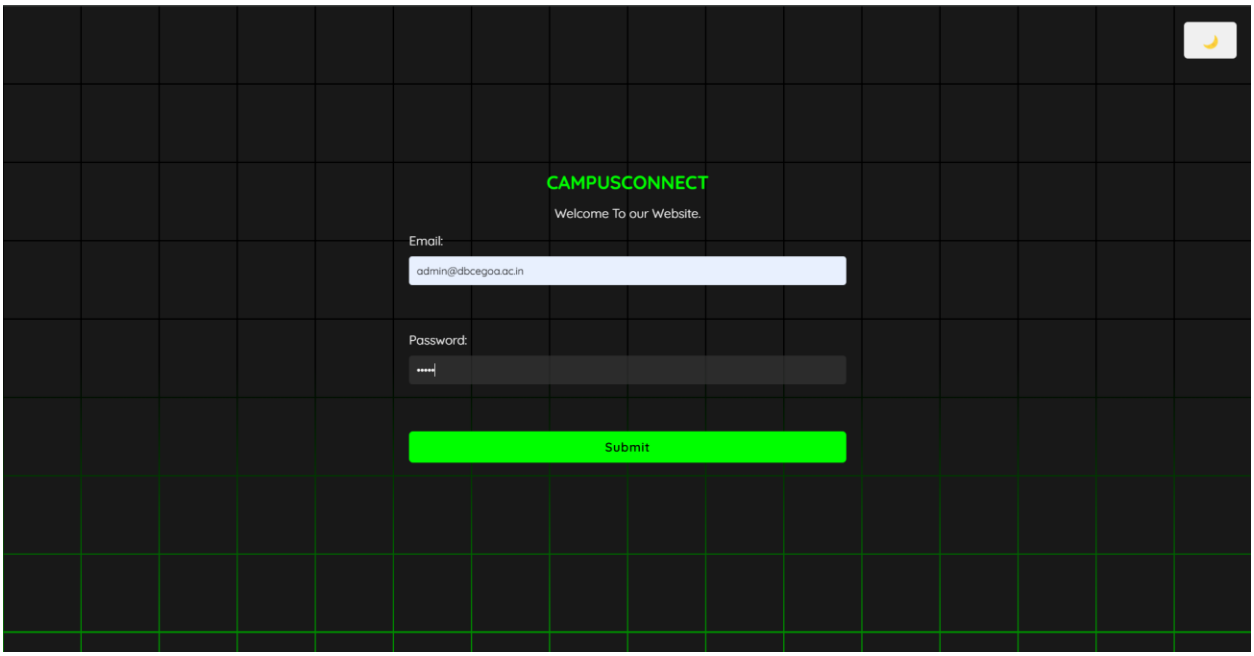

Fig: 4.3.12 Login Page

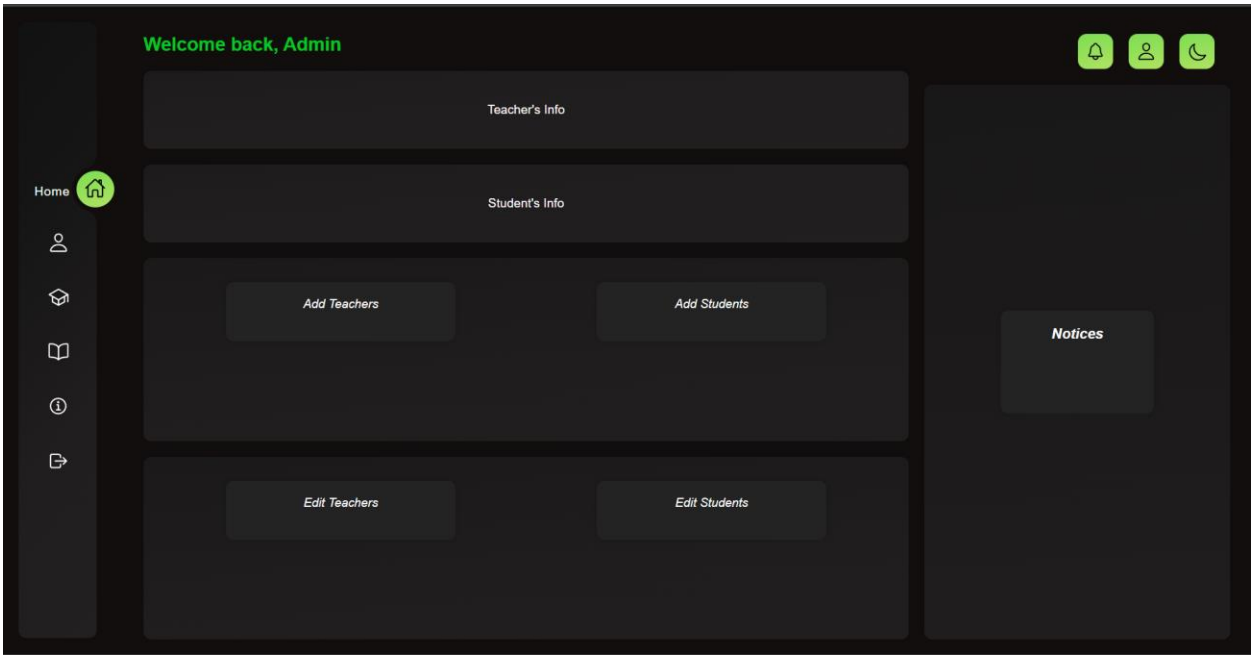

Fig: 4.3.13 Admin Landing Page

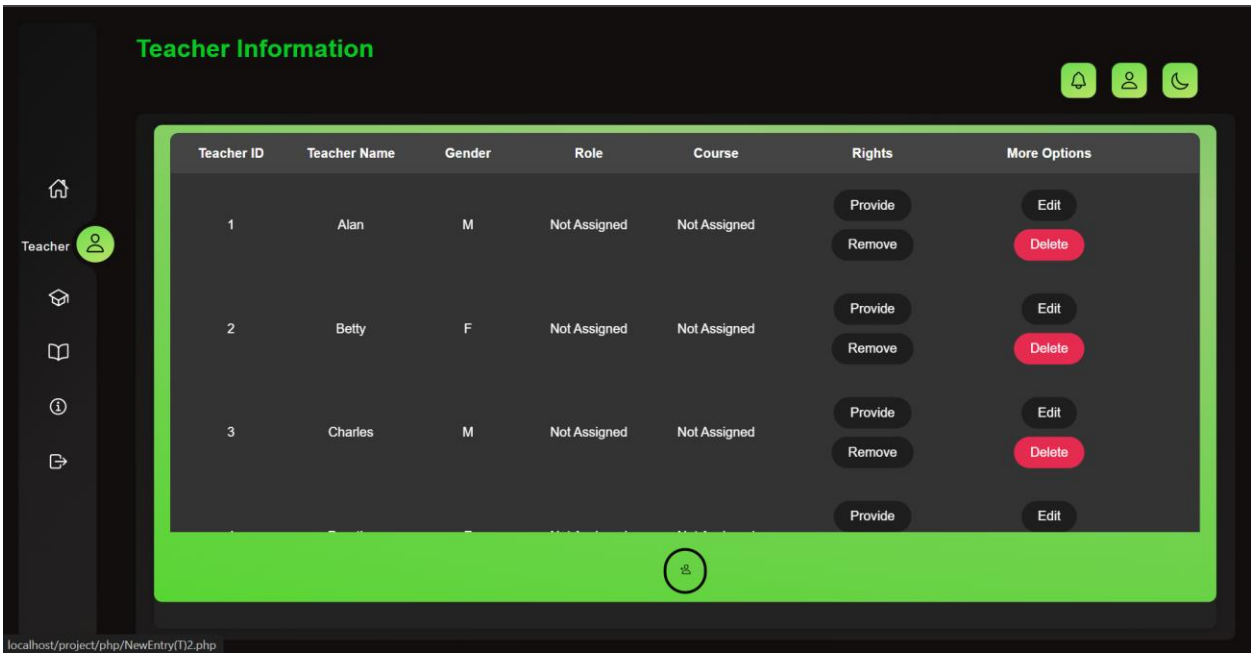

Fig: 4.3.14 Admin Teacher Information Page

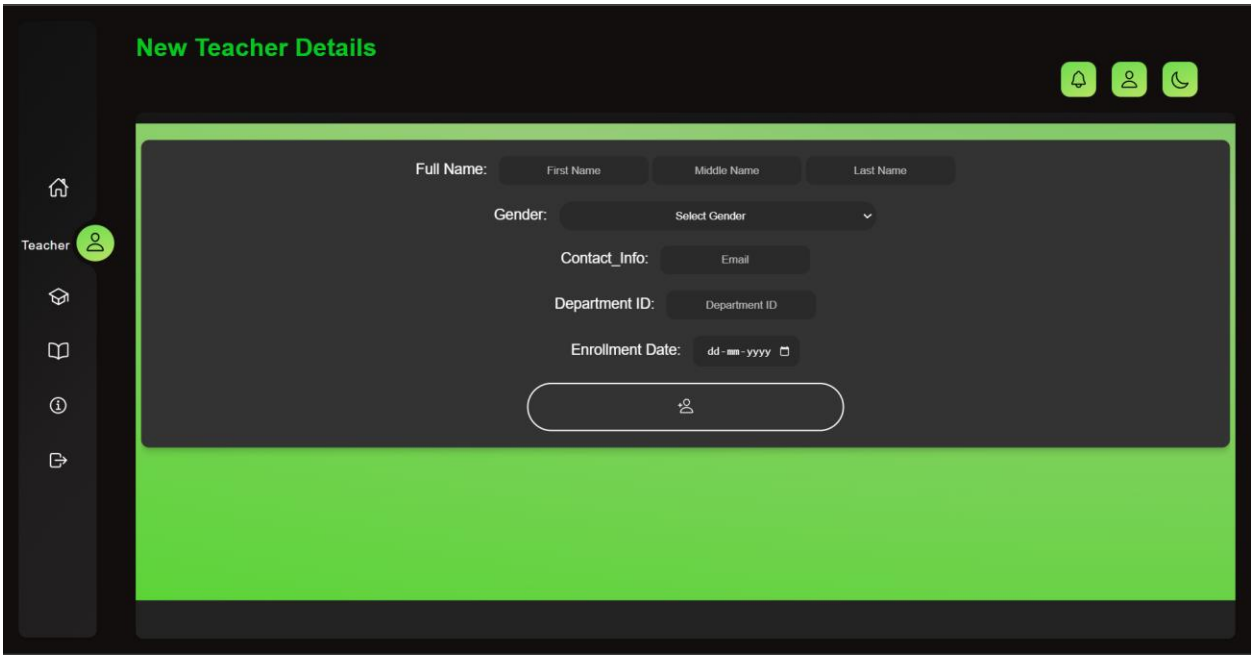

Fig: 4.3.15 Admin New Teacher Details Page

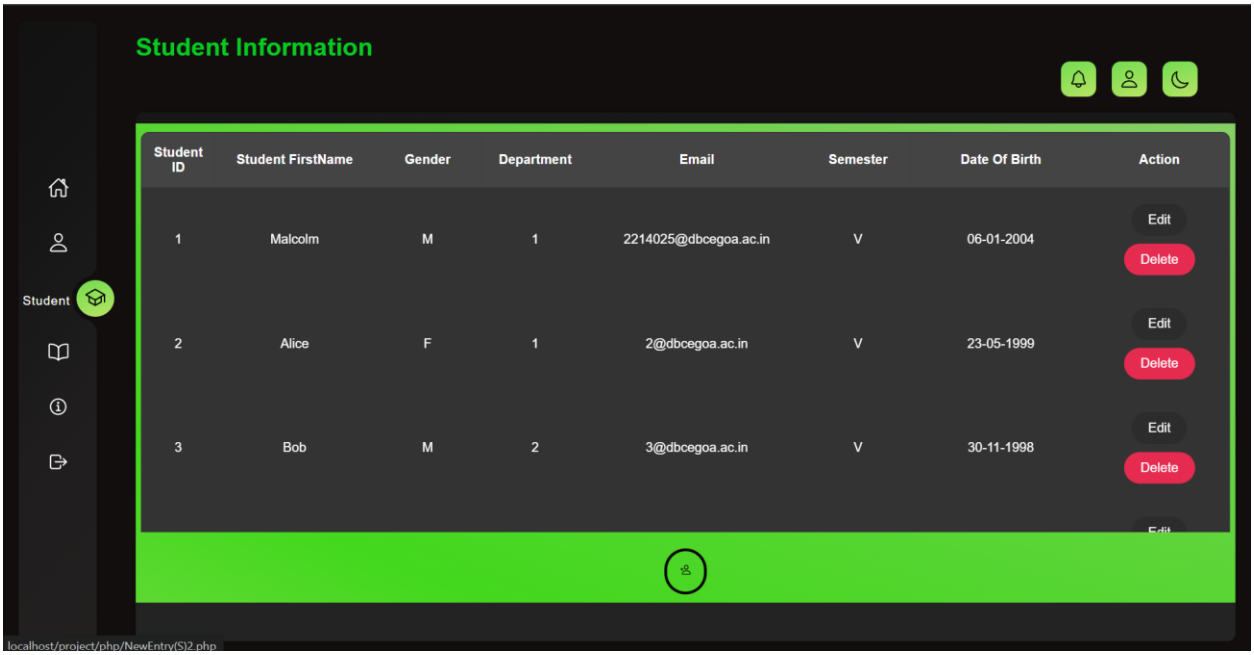

Fig: 4.3.16 Admin Student Information Page

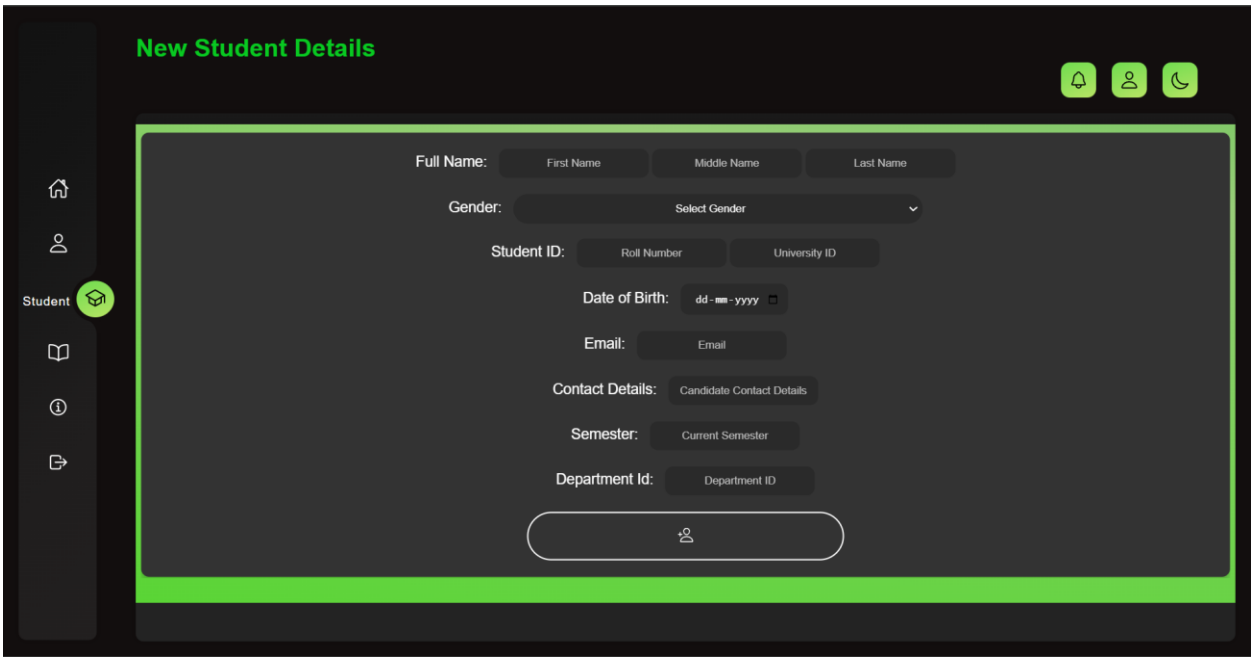

Fig: 4.3.17 Admin New Student Details Page

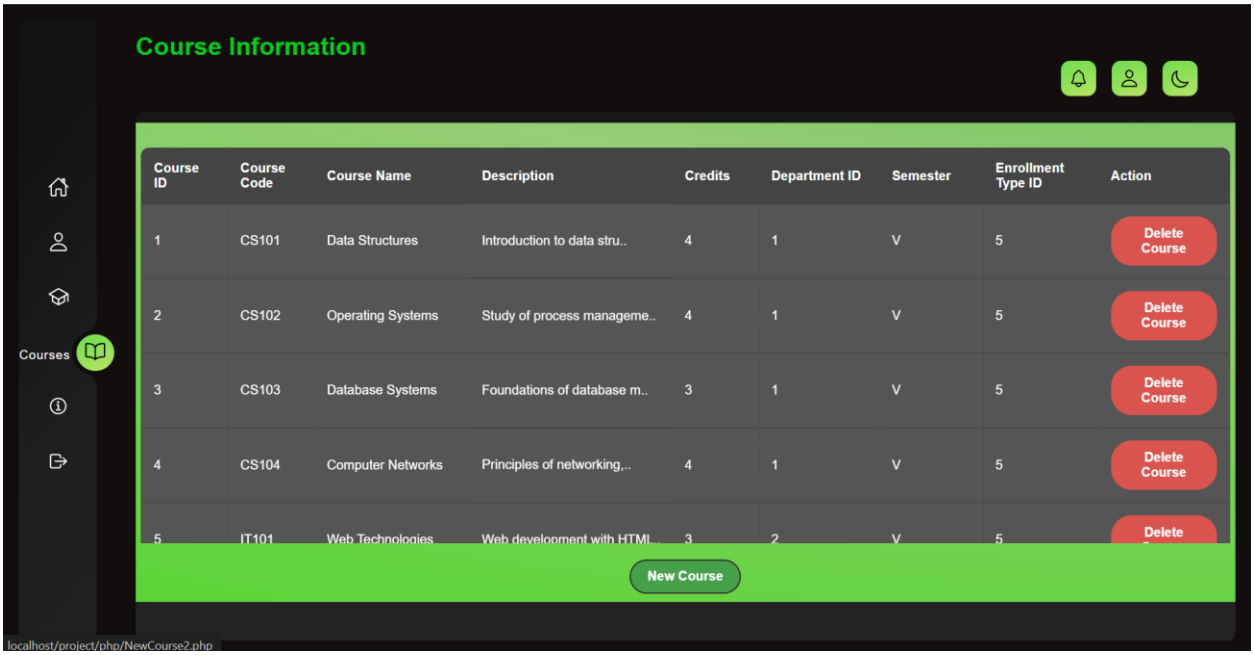

Fig: 4.3.18 Admin Course Information Page

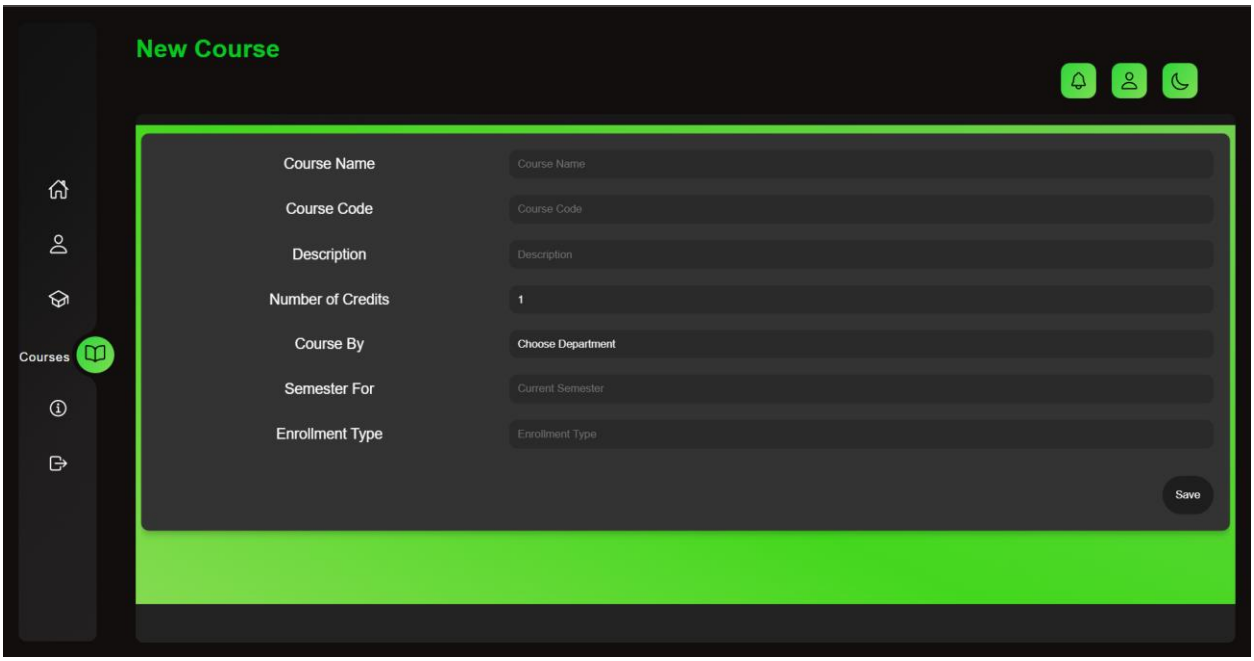

Fig: 4.3.19 Admin New Course Page

Fig: 4.3.20 Admin Notices Page

- **Teacher :**

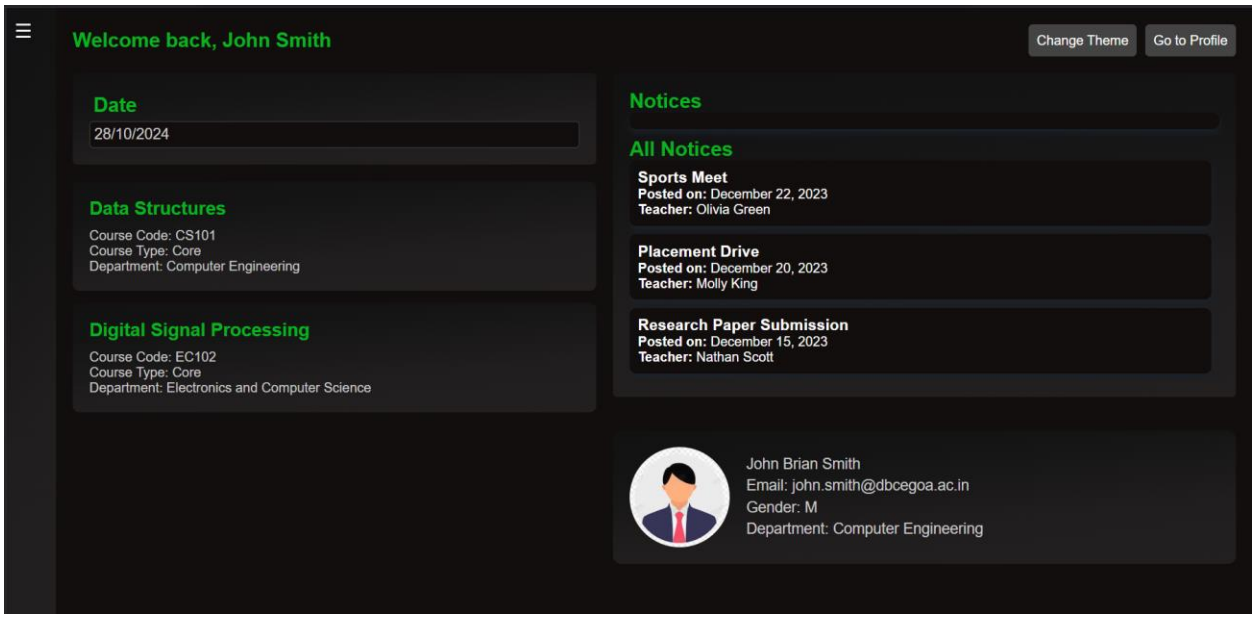

Fig: 4.3.21 Teacher Landing Page

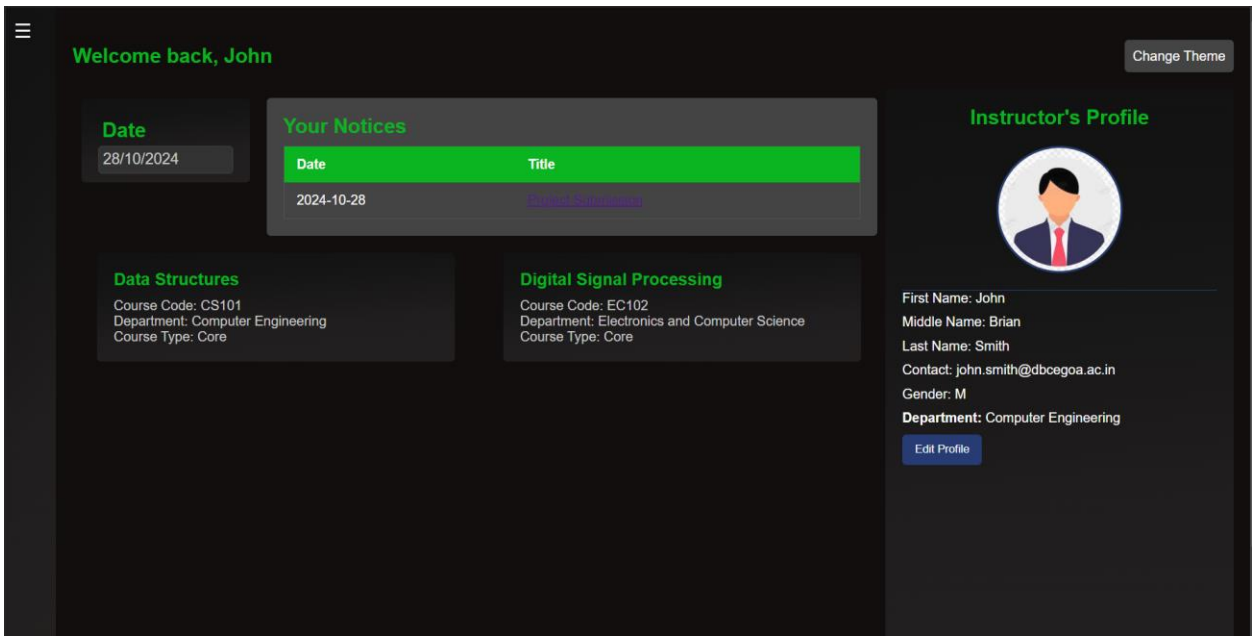

Fig: 4.3.22 Teacher Profile Page

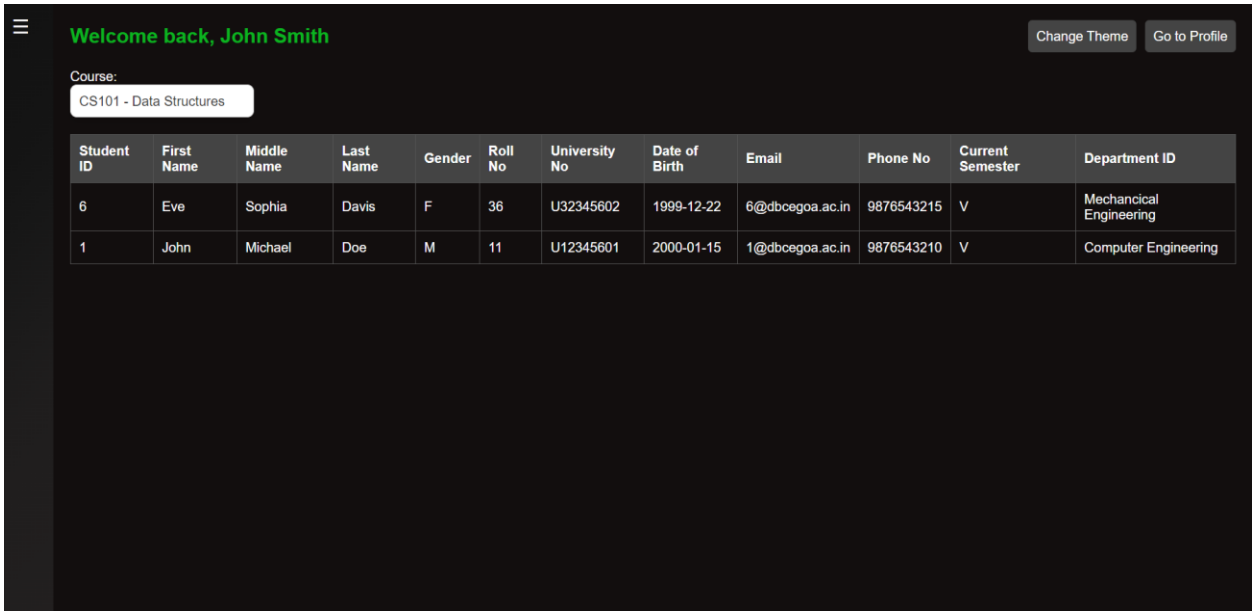

Fig: 4.3.23 Teacher Student Information Page

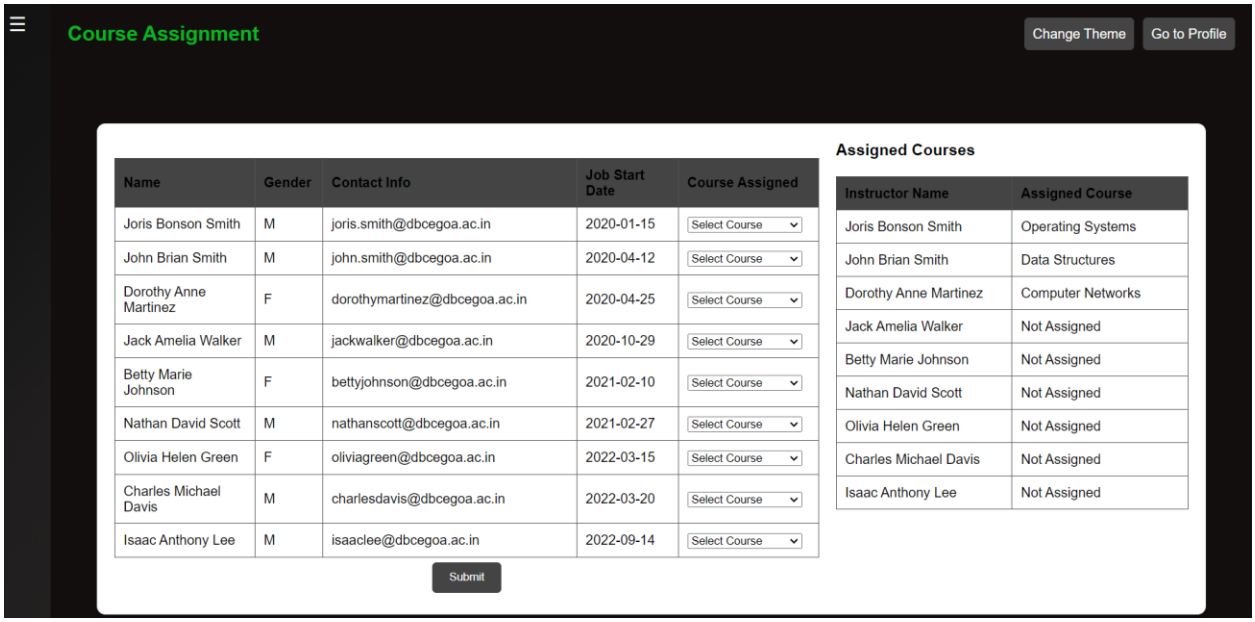

Fig: 4.3.24 HOD Course Assignments Page

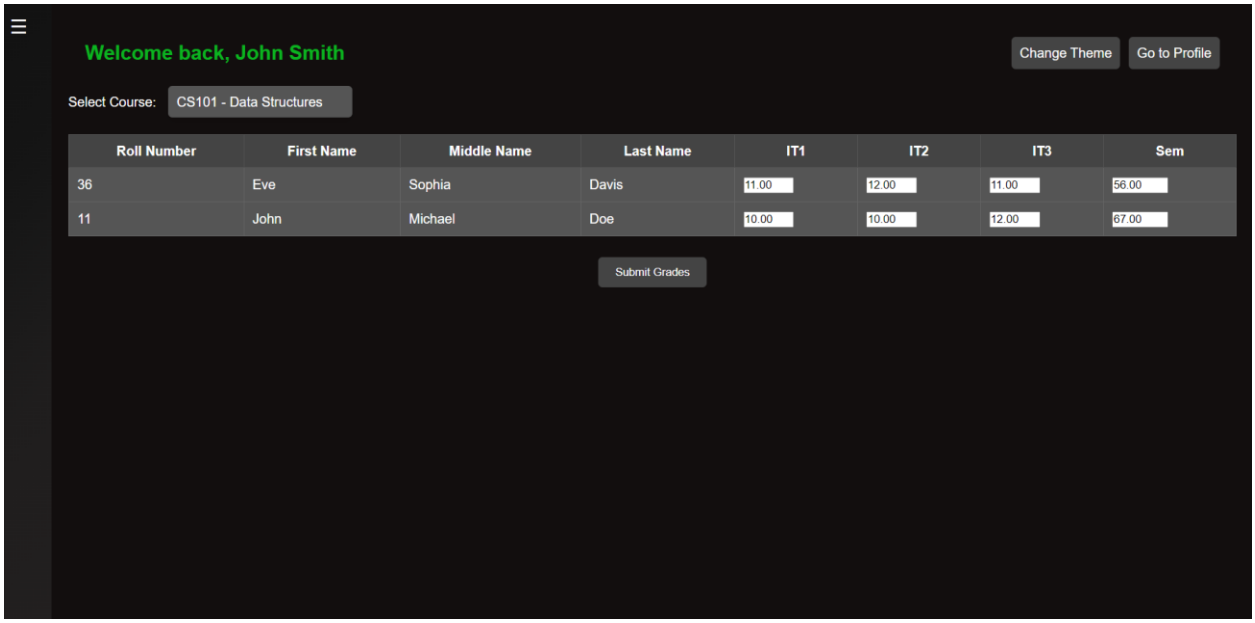

Fig: 4.3.25 Teacher Grade Allocation Page

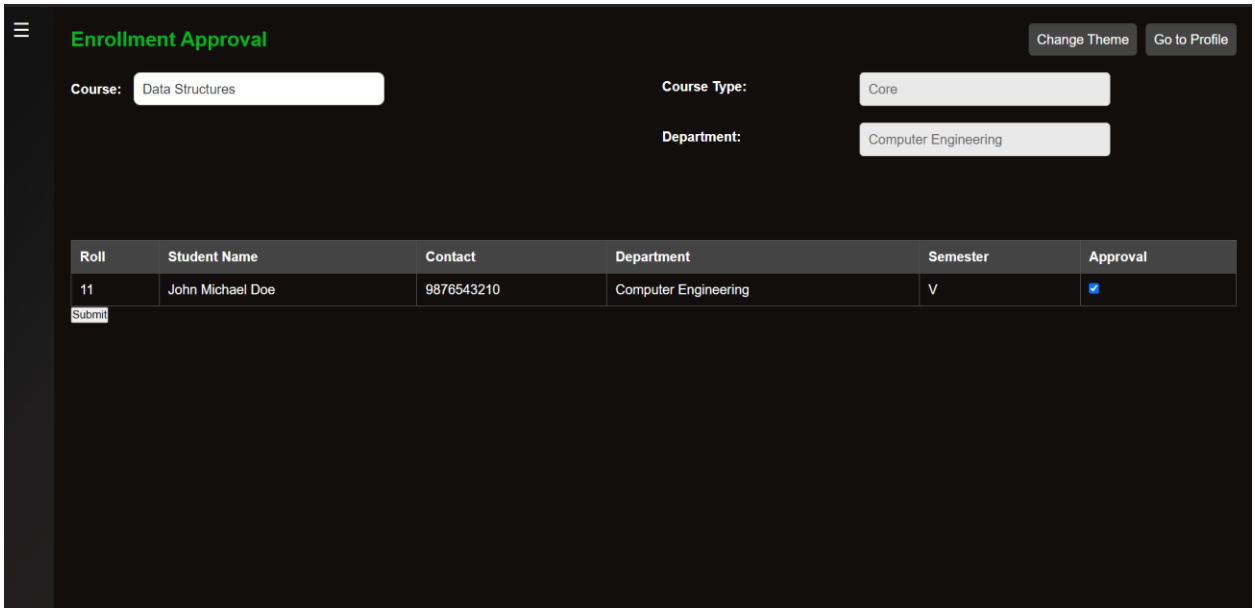

Fig: 4.3.26 Teacher Enrollment Approval Page

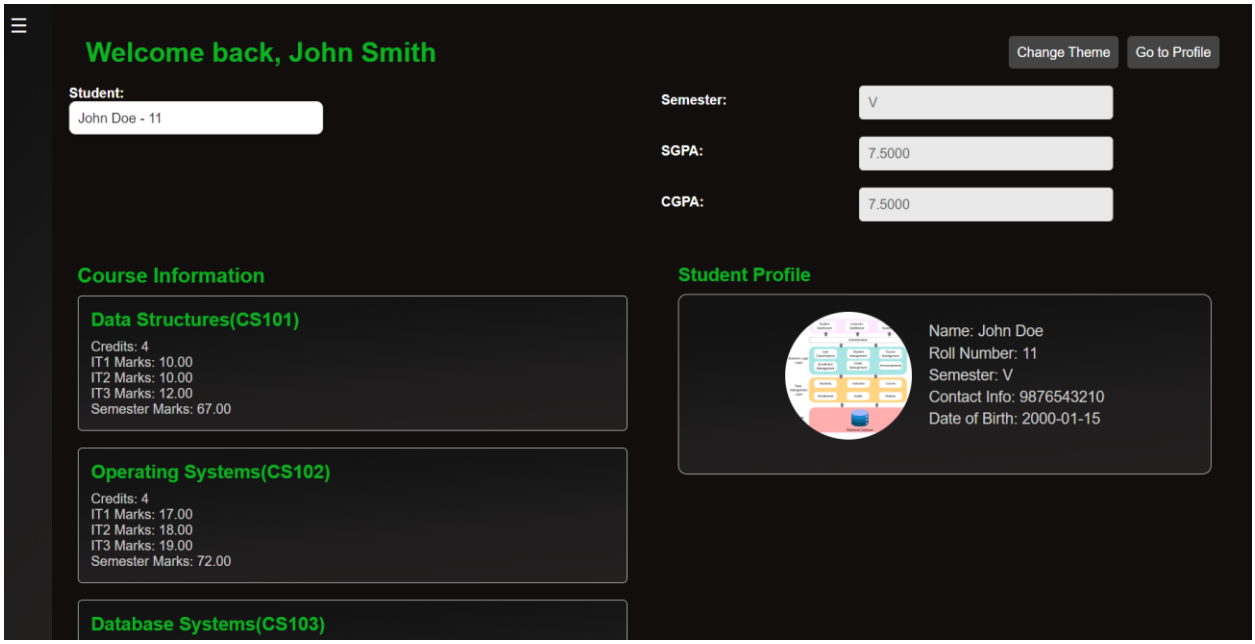

Fig: 4.3.27 Teacher Mentorship Page

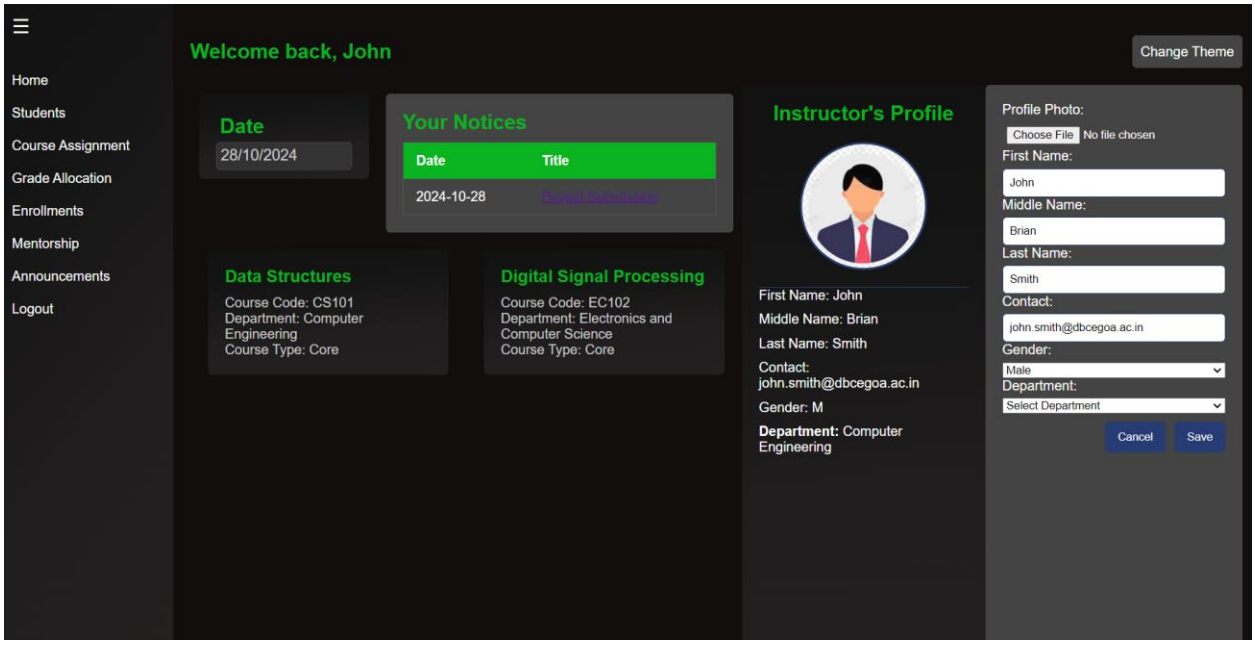

Fig: 4.3.28 Teacher Profile Page

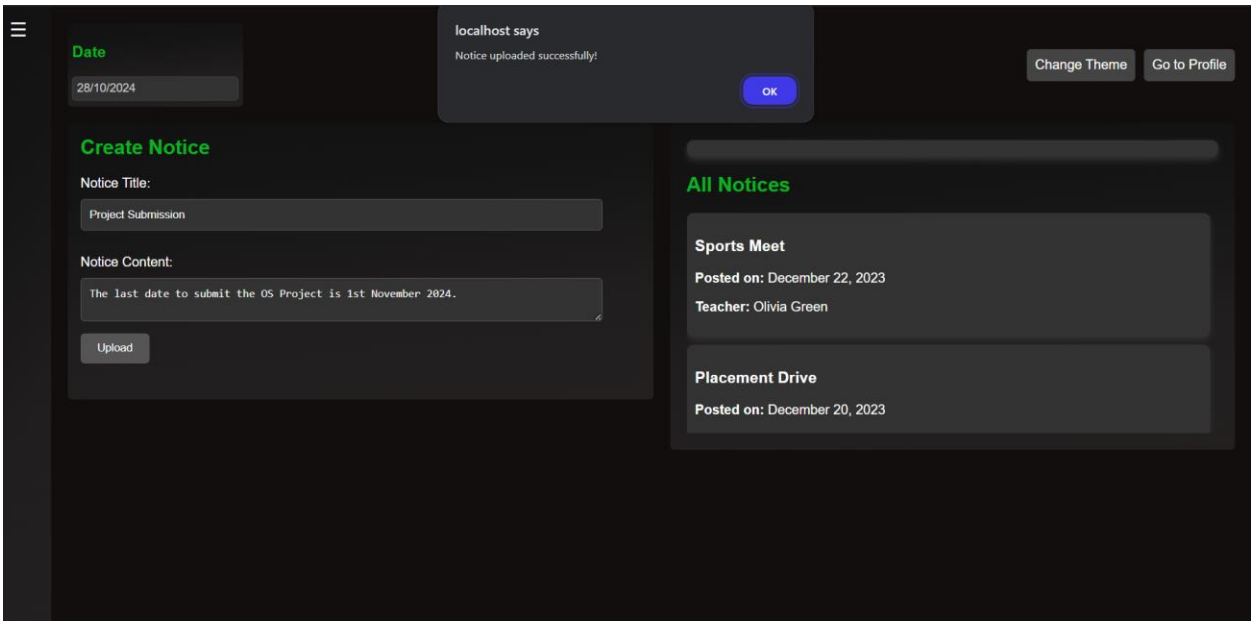

Fig: 4.3.29 Teacher Create Notices Page

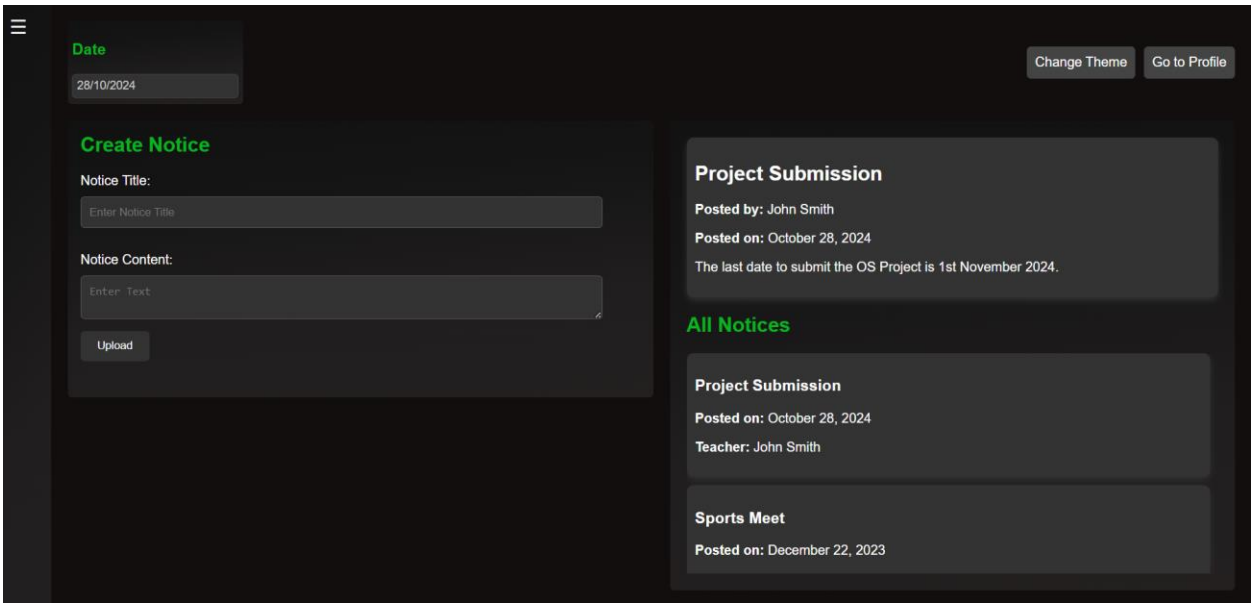

Fig: 4.3.30 Teacher Notices Page

CHAPTER 5

RESULTS AND DISCUSSION

5.1 SYSTEM PERFORMANCE:

The Student Information System (SIS) successfully meets the key objectives of streamlining academic operations, enhancing data management, and providing a user-friendly experience for students, teachers, and administrators. It ensures secure and efficient handling of student records, grades, and courses while maintaining accessibility for all user roles. With responsive design and role-based access, the system provides real-time updates, ensuring that users can manage their academic responsibilities with ease. The system's overall performance has been stable, handling large volumes of data effectively without noticeable delays, which meets our goal of creating a reliable and scalable solution for academic institutions.

5.2 USER FEEDBACK:

Feedback from early users, including teachers and administrative staff, has been positive. Users have particularly appreciated the streamlined grade allocation process, easy-to-navigate dashboard, and real-time access to information. Teachers found the system to be intuitive for inputting grades, while students appreciated the clear visibility of their academic performance and personalized dashboards. There have been suggestions for minor improvements, such as enhanced notification systems and attendance module, which are currently being considered for future updates.

5.3 CHALLENGES FACED:

Throughout the project, several challenges were encountered, particularly with integrating complex features like role-based access control and real-time updates across different user interfaces. Ensuring data security, especially when handling sensitive student information, required careful planning and the implementation of prepared statements to prevent SQL injection. Additionally, managing large data sets within the MySQL database while maintaining optimal performance demanded extensive optimization efforts. These challenges were successfully overcome through collaboration, iterative testing, and constant communication between the team, ensuring the system remained both secure and efficient.

5.4 COMPARATIVE ANALYSIS:

Compared to existing student information systems, our solution offers several key improvements, particularly in terms of user experience and performance. While traditional systems may struggle with providing real-time data updates, our system excels by giving users instant access to the most recent announcements, grades, and course information. The role-based access and mentorship features are also noteworthy innovations, allowing greater flexibility and control for institutions in managing user permissions and student support.

CHAPTER 6

REFERENCES

6.1 CITATIONS:

- francoisjacquet. (2024, October 5). *GitHub - francoisjacquet/rosariosis: RosarioSIS Student Information System for school management.* GitHub. https://github.com/francoisjacquet/rosariosis

- Thirumalai-Shaktivel. (2021). *GitHub- Thirumalai-Shaktivel/Student_Information_Management_System: Student Information Management System (SIMS) is a fully computerized system or a database where all the student-related data can be stored, retrieved, monitored & analyzed.* GitHub.https://github.com/Thirumalai-Shaktivel/Student_Information_Management_System?tab=readme-ov-file

- coderzaman. (2024). *GitHub - coderzaman/Student-Information-System: A comprehensive project for managing university operations, built with PHP, Bootstrap, HTML, CSS, JavaScript, and MySQL. Features include admin, teacher, and student panels for managing courses, fees, results, and profiles.* GitHub. https://github.com/coderzaman/Student-Information-System

6.2 RESOURCES:

- *Fundamental of Database systems*
  Ramez Elmasri, Shamkant B. Navathe, 7th Edition, Pearson, 2018.

- *DatabaseSystem Concepts*
  Abraham Silberschatz, Henry F. Korth, S. Sudarshan, 6th Edition, McGraw HillI, 2013

- *NOSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*
  Pramod J.Sadalage, Martin Fowler, 4th Edition, Pearson, 2014

- *Database Management Systems*
  Raghu Ramkrishnan, Johannes Gehrke, 3rd Edition McGraw-Hill, 2002