SRI SIDDHARTHA ACADEMY OF HIGHER EDUCATION

(Declared as Deemed to be University, under section 3 of UGC Act 1956)

Accredited 'A+' Grade by NAAC

AGALAKOTE, TUMAKURU-572107

Project Report
On
"SCHOLARSUITE
TRANSFORMING CAMPUS ADMINISTRATION AND ELEVATING ACADEMIC EXCELLENCE"

Submitted in partial fulfillment of requirements for the award of degree

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

Submitted

By

BALAJI	**21IS010**
BRUNDA K	**21IS014**
KARAN R JOSHI	**21IS036**
NITHYANANDA R S	**21IS056**

Under the Guidance of

Mr. Gangadhar M L, M.Tech,(Ph.D)

Assistant Professor, Dept. of ISE,

SSIT, TUMAKURU

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

Accredited by NBA (2022-2028)

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(A Constituent College of Sri Siddhartha Academy of Higher Education)

MARALURU, TUMAKURU-572105

2024-25

SRI SIDDHARTHA ACADEMY OF HIGHER EDUCATION

(Declared as Deemed to be University, under section 3 of UGC Act 1956)
Accredited 'A+' Grade by NAAC
AGALAKOTE, TUMAKURU-572107

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(A Constituent College of Sri Siddhartha Academy of Higher Education)
MARALURU, TUMAKURU-572105

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"SCHOLARSUITE TRANSFORMING CAMPUS ADMINISTRATION AND ELEVATING ACADEMIC EXCELLENCE",** is a bonafide work carried out by **Mr. BALAJI (21IS010), Ms. BRUNDA K (21IS014), Mr. KARAN R JOSHI (21IS036), and Mr. NITHYANANDA R S (21IS056)** in partial fulfillment of the requirement for the award of Bachelor of Engineering degree in **Information Science and Engineering** of Sri Siddhartha Academy of Higher Education, Tumakuru during the year 2024-25.

It is certified that all corrections/suggestions indicated have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Signature of the Guide	Signature of the HoD	Signature of the Principal
Mr. Gangadhar M L Asst. Professor, Dept. of ISE, SSIT, TUMAKURU	**Dr. H.S.Annapurna** Professor and Head, Dept. of ISE, SSIT, TUMAKURU.	**Dr. M. S. Raviprakasha** Principal, SSIT, TUMAKURU.

External Viva

Name of the Examiners:	**Signature with date**
1. _____	_____
2. _____	_____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without mentioning the people who made it possible and their support had been a constant source of encouragement, which crowned our efforts with success.

We are deeply indebted and would like to express our sincere thanks to our beloved **Principal, Dr. M. S. Raviprakasha,** for providing us the opportunity to do this project.

Our special gratitude to **Dr. H. S. Annapurna, HOD**, department of ISE, S.S.I.T for her guidance, constant encouragement and wholehearted support.

Our special thanks to our guide **Mr. Gangadhar M L**, **Assistant Professor**, department of ISE, S.S.I.T, for her guidance, constant encouragement and wholehearted support.

Finally, we would like to express our sincere thanks to all the staff members of ISE department, S.S.I.T. for their valuable guidance & support.

Place: Tumakuru

Date:

Project Associates:

BALAJI	**(21IS010)**
BRUNDA K	**(21IS014)**
KARAN R JOSHI	**(21IS036)**
NITHYANANDA R S	**(21IS056)**

DECLARATION

We, **BALAJI (21IS010), BRUNDA K (21IS014), KARAN R JOSHI (21IS036)** and **NITHYANANDA R S (21IS056)** of VIII semester, Department of Information Science and Engineering of Sri Siddhartha Institute of Technology, Tumakuru, hereby declare that this project titled, **"SCHOLARSUITE TRANSFORMING CAMPUS ADMINISTRATION AND ELEVATING ACADEMIC EXCELLENCE"**, has been carried out by us under the supervision of **Mr. Gangadhar M L**, Assistant Professor, Department of Information Science and Engineering**,** in partial fulfillment of requirements for the award of degree in Information Science and Engineering, **Sri Siddhartha Institute of Technology, Tumakuru**.

Date:

Place: Tumakuru

BALAJI (21IS010)

BRUNDA K (21IS014)

KARAN R JOSHI (21IS036)

NITHYANANDA R S (21IS056)

CONTENTS

Ch.No	Chapters	Page.No
1.	**INTRODUCTION**	**1**
	1.1 OVERVIEW	1
	1.2 OBJECTIVE	2
	1.3 FUNCTIONAL HIGHLIGHTS OF SCHOLARSUITE	2
	1.4 BENEFITS OF SCHOLARSUITE	3
	1.5 PROPOSED SYSTEM	4
	1.6 SYSTEM ARCHITECTURE	5
2.	**LITERATURE SURVEY**	**7**
3.	**METHODOLOGY**	**10**
	3.1 DEVELOPMENT STACK OVERVIEW	10
	3.2 DATABASE AND DATA MODELS	10
	3.3 BACKEND API'S	11
	3.4 FRONTEND COMPONENTS AND ROUTING	11
	3.5 SOFTWARE REQUIREMENTS	12
	3.6 HARDWARE REQUIREMENTS	12
	3.7 SYSTEM ARCHITECTURE	13
	3.8 SYSTEM WORKFLOW	13
4.	**SYSTEM DESIGN**	**15**
	4.1 USE CASE DIAGRAM	15
	4.2 CLASS DIAGRAM	17
	4.3 SEQUENCE DIAGRAM - LOGIN FLOW	18
	4.4 SYSTEM ARCHITECTURE	19
	4.5 SUMMARY	20
5.	**RESULTS AND DISCUSSION**	**21**
	5.1 INTRODUCTION TO TESTING	21
	5.2 TEST CASES DESIGN & EXECUTION	24
	5.3 RESULTS	32
	5.4 RESULT SUMMARY	49
	5.5 DISCUSSION ON OUTCOMES	49
6.	**CONCLUSION**	**50**

APPENDIX

A. Technodea Certificate

LIST OF FIGURES

Figures	Page No
1.1: System Architecture	5
4.1: Use Case Diagram	15
4.2: Class Diagram	17
4.3: Sequence Diagram	18
5.1: Landing Page	32
5.2: About Page	32
5.3: Sign-in page	33
5.4: Incorrect Sign-in Page	33
5.5: Admin Dashboard	34
5.6: List of Teachers	34
5.7: Form to create a new Teacher	35
5.8: Teacher profile details	35
5.9: List of Students	36
5.10: Form to create a new Student	36
5.11: Student profile details	37
5.12: Lessons and its Details	37
5.13: Subjects and its Details	38
5.14: Exams and its Details	38
5.15: Assignments and its Details	39
5.16: Results with details	39
5.17: Parents and their details	40
5.18: Form to add new subject	40
5.19: Form to add new class	41
5.20: Form to create new Lesson	41
5.21: Form to create a new Exam	42
5.22: Form to create new result	42
5.23: Calculator to calculate the percentage from CGPA	43
5.24: Calculator to calculate the CGPA from SGPA's	43
5.25: Digital Library	44
5.26: List of Alumni	44
5.27: Profile details of Alumni	45
5.28: Account details and password update	46

5.29: Credit deficiency details of a student 47
5.30: Login details at clerk security service provider 47
5.31: Prisma OMR 48

LIST OF TABLES

Tables	Page No.
5.1: Testcase 1	24
5.2: Testcase 2	25
5.3: Testcase 3	26
5.4: Testcase 4	27
5.5: Testcase 5	28
5.6: Testcase 6	29
5.7: Testcase 7	30
5.8: Testcase 8	31

ABSTRACT

An organized and systematic office solution is essential for all universities and organizations. There are many departments of administration for the maintenance of college information and student databases in any institution. All these departments provide various records regarding students. Most of these track records are needed to maintain information about the students. This information could be the general details like student name, address, performance, attendance or specific information related to departments like collection of data. All the modules in college administration are interdependent. They are maintained manually. So, they need to be automated and centralized as, Information from one module will be needed by other modules. For example, when a student needs his course completion certificate it needs to check many details about the student like his name, reg number, year of study, exams he attended and many other details. So, it needs to contact all the modules that are office, department and examination and result of students. With that in mind, we completely changed the existing Student Database Management System and made necessary improvements to make the processes simpler. Administrators using the system will find that the process of recording and retrieving student's information and managing their classes, including marking of attendance, is now a breeze. In general, this project aims to enhance efficiency and at the same time maintain information accuracy. Later in this report, features and improvements that allow achievement to this goal will be demonstrated and highlighted. Our work is useful for easy user interface. We are planning to utilize the powerful database management, data retrieval and data manipulation. We will provide more ease for managing the data than manually maintaining in the documents. Our work is useful for saving valuable time and reduces the huge amount of paperwork.

Chapter 1

INTRODUCTION

1.1 Overview

In today's fast-paced environment, **modern universities** require **comprehensive digital systems** to efficiently manage a wide array of and administrative tasks. From student enrollment to faculty assignments and from attendance tracking to grade reporting, higher education institutions are increasingly turning toward automation for better productivity, accuracy, and transparency.

A **Academic Management System (AMS)** serves as a **digital solution** that helps automate core institutional functions. As AMS facilitates the management of student registration, fee payments, attendance, and performance. These systems often function as **educational ERP platforms**, unifying the roles and responsibilities of teachers, students, parents, and administrators onto a single digital interface.

The benefits of such systems are numerous:

- **Automation of workflows** such as exam scheduling, result generation, and announcement broadcasting.
- **Reduction of manual paperwork**, resulting in improved **data consistency**, **security**, and **accessibility**.
- **Transparency across departments**, minimizing errors and administrative burden.
- Enhanced **decision-making through real-time insights and reports**.

As institutions scale, **manual methods become insufficient and error-prone**, making **integrated AMS solutions essential** for operational efficiency.

According to the project, integrated management systems not only streamline campus operations but also empower decision-makers through real-time analytics and reporting.

1.2 Objective

This project presents **ScholarSuite**, a tailored **Academic Management System** developed using React and TypeScript, aimed at bridging the gap between needs and technological solutions. The **main objectives** of ScholarSuite are:

- To **automate core workflows** like attendance tracking, exam management, and grading.
- To provide **planning tools**, such as **CGPA calculators** for students.
- To enable **role-specific access** and control for administrators, teachers, parents, Alumni and students.
- To improve **efficiency, accuracy, and transparency** through automation.
- To **enhance user experience** via an intuitive web-based dashboard.

ScholarSuite is designed to be a **modular, scalable, and user-friendly** system that can be adopted by institutions of varying sizes. It reflects real-world administrative challenges and provides robust solutions using modern web development tools and secure authentication practices.

1.3 Functional Highlights of Scholarsuite

Upon registration, users can select their role as **Admin**, **Student**, or **Teacher**. The system then grants role-specific access to functionalities:

- **Students**:
  - Compute **semester GPA and cumulative CGPA**
  - View **personal attendance reports**
  - Check **exam schedules and results**
- **Teachers**:
  - Record **marks and attendance**
  - Schedule **examinations**
  - Monitor **class performance**

- **Administrators**:
    - Manage **user accounts and roles**
    - Oversee **activity**
    - Generate **institutional reports**

By consolidating data into a centralized platform, ScholarSuite ensures **data-driven decision-making**, **minimal redundancy**, and **streamlined communications** between different institutional stakeholders.

1.4 Benefits of ScholarSuite

The implementation of **ScholarSuite** offers numerous benefits to modern educational institutions seeking to streamline and administrative functions. Key benefits include:

- **Automation of Routine Tasks**: Activities such as attendance tracking, exam scheduling, and grade recording are handled digitally, saving time and reducing human error.
- **Data Consistency and Centralization**: All and administrative data is stored in a centralized system, ensuring data integrity and ease of access.
- **Role-Based Dashboards**: Customized dashboards for Students, Teachers, and Admins improve user experience and make navigation intuitive.
- **Improved Decision-Making**: Real-time reports and analytics empower faculty and administrators to make informed decisions.
- **Reduced Manual Workload**: Automating repetitive tasks minimizes paperwork and administrative overhead.
- **Increased Transparency**: Stakeholders including students and faculty gain visibility into performance, schedules, and institutional updates.
- **Scalability**: The system is designed to accommodate the growing needs of institutions, from small colleges to large universities.
- **Security and Privacy**: Features such as password hashing (bcrypt) and session management ensure user data is protected.

1.5 Proposed System

The **Proposed System**, ScholarSuite, is a web-based Academic Management System designed to cater to the management needs of students, teachers, and administrators. The system is built with a focus on:

- **User Role Segmentation**: Upon registration, users choose a role (Admin, Teacher, or Student), which determines their access and functionalities.
- **Modular Features**:
    - **Students** can view attendance, compute CGPA/SGPA, and check exam schedules.
    - **Teachers** can record marks and attendance, schedule assessments, and analyze class performance.
    - **Admins** have full control over the system including user management, report generation, and system monitoring.
    - **Parents** can view and access their particular pupil details and monitor them.
    - **Alumni** can update their profiles and view other alumni only.
- **User-Friendly Interface**: Developed using React, Next.js, Tailwind CSS, TypeScript, Node.js, Next,js API Routes, the UI is clean, responsive, and easy to use.
- **Secure Authentication**: User login is protected using clerk password hashing and session handling techniques are handled automatically by clerk to prevent unauthorized access.
- **Database Integration**: The system uses PostgreSQL for lightweight and effective data management.
- **Scalable and Extendable**: ScholarSuite can be easily extended with additional modules like library management, placement tracking, and more.

The goal of the proposed system is to **minimize administrative burden**, **maximize transparency**, and **promote efficient processes** within a centralized digital ecosystem

1.6 System Architecture

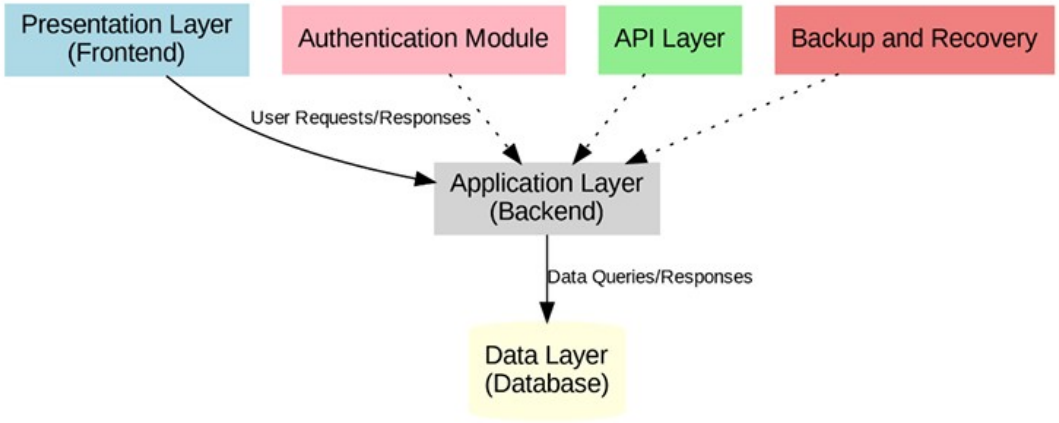


Figure 1.1: System Architecture

ScholarSuite follows a **multi-tiered architecture** that separates concerns and ensures flexibility and maintainability. The key layers of the architecture are:

- **Presentation Layer (Frontend)**
  - Built using React, Next.js, Tailwind CSS, TypeScript for responsive design.
  - Renders pages dynamically based on user roles (Student, Teacher, Admin, Parents, Alumni).
  - Provides forms for login, data entry (e.g.,marks), and viewing records.
- **Application Layer (Backend)**
  - Developed using Node.js, Next.js API Routes .
  - Handles all **business logic**, including authentication, data validation, session handling, and role-based access control.
  - API routes manage the interaction between frontend forms and backend processing.
- **Data Layer (Database)**
  - Uses Prisma, PostgreSQL as the relational database management system.
  - Stores user credentials, attendance data, marks, subject information, and schedules.

- **Security Layer**

  - Implements **hashing** for password encryption.
  - Manages **secure user sessions** using clerk JSON web tokens (JWT's).
  - Includes basic input validation and protections against advanced web vulnerabilities.

- **System Workflow Overview**

  - **User Registration/Login**: Admin create accounts and select their role.
  - **Role-Based Redirection**: Users are directed to dashboards based on their role.
  - **Data Operations**: Users perform operations like uploading marks, computing CGPA, etc.
  - **Result Display**: Outputs are rendered dynamically and stored in the database.
  - **Admin Monitoring**: Admin has the ability to view and manage system-wide data.

This **modular and layered approach** ensures that ScholarSuite remains **robust**, **maintainable**, and **scalable**, with clear separation of concerns and efficient codebase management.

Chapter 2

LITERATURE SURVEY

[1] Pandey, V., Anand, A., & Racherla, S. (2022). Student Profile Management System. International Research Journal of Engineering and Technology (IRJET).

The student profile management system aims to centralize and streamline the handling of student data within educational institutions. This system focuses on efficiently managing student information, including personal details, records, and attendance, with secure access controls to ensure data privacy. It employs web technologies such as React, Next.js, Tailwind CSS, TypeScript, Node.js, Next.js API Routes and Postgres for database.

The primary objectives include enhancing data management efficiency, providing easy accessibility for authorized users, and ensuring data security. Despite its potential benefits, the system lacks real-world implementation examples and case studies to validate its effectiveness. Additionally, integration challenges with existing college management systems are not explored, presenting a potential area for future research and development.

GAPS

The project focuses on student profiling and online enrollment but lacks realworld implementation examples or case studies to validate the claims. Additionally, it doesn't explore integration challenges with existing college management systems. IJRET[1]

[2] Singh Jaryal, V., & Goel, S. (2022). Student Management System. Jaypee Academic of Information Technology.

The "Student Management System" project aims to develop a comprehensive system to manage student-related activities efficiently. The system includes functionalities for handling student profiles, course registrations, attendance, and records. It is designed to replace manual processes with an automated, user-friendly interface. The project was carried out under the supervision of Dr. Shubham Goel at Jaypee Academic of Information Technology, Waknaghat, Himachal Pradesh. The system ensures data

security and provides easy access to authorized users, such as students, faculty, and administrators.

GAPS

The project provides a theoretical framework for managing student records but does not address potential scalability issues or how the system handles complex legal disputes in real-world scenarios.

[3] Selwyn, N., Hillman, T., Rensfeldt, A. B., & Perrotta, C. (2021). Digital Technologies and the Automation of Education — Key Questions and Concerns. Postdigital Science and Education.

This article delves into the growing impact of digital technologies and automation on education. It examines how automation is becoming increasingly embedded in various educational processes, from administrative tasks to teaching and learning activities. The authors discuss key concerns related to the delegation of decision-making to automated systems, the potential loss of professional judgment, and the implications for teacher-student relationships. The article also highlights the need for critical examination of the assumptions and promises of educational automation, emphasizing the importance of maintaining human oversight and accountability in automated educational environments.

GAPS

Discusses the infusion of technology in education but lacks a focus on practical implementation and challenges faced by educational institutions[3].

[4] P, S. (2022). College ERP System.

The College ERP System is a comprehensive college management system built using the Django framework. It facilitates interactions between students and teachers by providing features such as attendance tracking, marks management, and timetable scheduling. The system is designed to streamline administrative processes and improve communication within the educational institution. It includes functionalities for managing student profiles, course registrations, and records, ensuring data security and easy access for authorized users.

GAPS

A comprehensive college management system built using Django framework. It includes features like attendance, marks,and timetable management but lacks a discussion on integration with existing systems[4].

[5] P, S. (2022). Student Management System.

This project is a simple web-based student management software written in PHP and JavaScript, specifically designed for schools or educational organizations. It includes features such as adding student information, managing multiple programs, handling student payments, attendance systems, generating student ID cards, and sending results and notices by SMS. The system ensures that all activities are auto-saved, and both admins and users can view changes made. It also supports multiple themes and provides an easy installation process.

GAPS

A project designed to manage student records, but it lacks a discussion on data security and privacy concerns[5].

Chapter 3

METHODOLOGY

3.1 Development Stack Overview

The ScholarSuite platform is built using a **modern full-stack React and TypeScript architecture** that ensures scalability, maintainability, and performance.

Frontend:

- **React** with **Next.js**: Enables **server-side rendering** and **client-side routing**.
- **Tailwind CSS**: Utility-first CSS framework ensuring responsive and accessible UI.
- **TypeScript**: Enforces **static type checking**, reducing bugs and improving developer experience.

Backend:

- **Node.js**: Executes Next.js API routes and handles core **business logic**.
- **Prisma ORM**: Facilitates seamless interaction with SQL databases like **PostgreSQL**.
- **Authentication**: Uses clerk for password hashing and **session cookies** for session management.

This modern stack balances **rapid development**, **strong type safety**, and **future scalability**.

3.2 Database and Data Models

ScholarSuite employs a **relational database** managed through Prisma ORM. The database schema includes models for:

- **Users** (with roles: Admin, Teacher, Student,Parents,Alumni)
- **Students**, **Teachers**
- **Courses** (Subjects), **Classes**
- **Attendance records**
- **Exams** and **Marks**

Key Relationships:

- A **User** has a defined **role** determining access rights.
- **Courses** are linked to **teachers** and **classes**.
- **Attendance** links **students**, **courses**, and **dates**.
- **Exams** associate students with subjects and store **marks**.

Prisma supports **schema migrations** that evolve the database structure safely and incrementally, ensuring **referential integrity** and clean querying (e.g., prisma.attendance.findMany()).

3.3 Backend API's

The backend uses **Next.js API routes** for handling all server-side operations.

Core Operations:

- **User Authentication**: Registration, login, role-based redirection.
- **CRUD Operations**: For students, courses, attendance, marks, and exams.
- **Access Control**:
    - **Admins** can manage all entities.
    - **Teachers** can record attendance and marks for their subjects.
    - **Students** can only **view** their records.
    - **Parents** can only track their pupil details.

All server responses are in **JSON format**, and the **frontend fetches data via API calls** for dynamic updates.

3.4 Frontend Components and Routing

ScholarSuite uses **Next.js pages and components** to construct the user interface. Routing is implemented using **Next.js Link** or useRouter() to enable fast transitions without full reloads.

Key Components:

- **Authentication Pages**: Sign-up and login forms collect credentials and assign roles.
- **Dashboard**: Role-based interface showing relevant modules (e.g., class list for teachers, CGPA tool for students).
- **CGPA Calculator**: Allows students to enter subject-wise grades and credits, performs weighted GPA calculation.
- **Attendance Tracker**: Enables teachers to mark attendance via a calendar interface and view student percentages.
- **Exam Scheduler**: Admins create exams, students view upcoming events.
- **Marks Entry**: Teachers enter scores, the UI calculates and displays grades.
- **Profile Pages**: Users can edit and view their personal information.

3.5 Software Requirements

Softwares and tools required for development

Operating System	Any OS capable of running a browser (Mac, Windows, Linux)
Front-end	React, Next.js, Tailwind CSS, TypeScript
Backend	Node.js, Next,js API Routes
Database	Prisma, PostgreSQL
Tools Required	ESLint, Prettier, GitHub
Authentication	Clerk
IDEs	VS-Code
Container	Docker

3.6 Hardware Requirements

Processor	Intel core i3 and above
Hard Disk	2 GB or above
RAM	2 GB or above
Internet	4 Mbps or above *(Wired or Wireless)*.

3.7 System Architecture

ScholarSuite follows a **three-tier architecture**:

1. Presentation Layer (Frontend):

- Handles all user interactions using **React + Next.js**
- Renders dynamic pages like /dashboard, /cgpa-calculator, /attendance

2. Application Layer (APIs & Logic):

- **Next.js API routes** manage data transactions and implement business logic (e.g., CGPA calculation, attendance tracking)
- Performs **authentication**, **role checks**, and **data validation**

3. Data Layer (Database + ORM):

- Managed by **Prisma ORM** interacting with **PostgreSQL databases**
- Stores and retrieves structured data

3.8 System Workflow

The internal system flow is structured as follows:

- **Login & Role Fetching**:
  - The user logs in, and their **role** is fetched from the database.
  - Based on the role, the **user is redirected** to the appropriate dashboard.
- **CGPA Calculator Flow**:
  - A student enters subject-wise marks and credits into a dynamic form.
  - The system validates input and calculates GPA **client-side** without server interaction.
  - Results are displayed instantly for user feedback.

- **Attendance Submission Flow**:

  - A teacher marks attendance via a UI calendar.
  - The UI sends a POST request to /api/attendance.
  - Prisma writes the data to the database; the server confirms the operation.
  - The frontend then refreshes attendance statistics.

Chapter 4

SYSTEM DESIGN

This chapter outlines the **system design of ScholarSuite**, presenting the architectural blueprint through **UML diagrams** and describing the relationships between the system's components. The design aims to ensure clarity, maintainability, modularity, and scalability, using standard design principles and modern software architecture.

4.1 Use Case Diagram

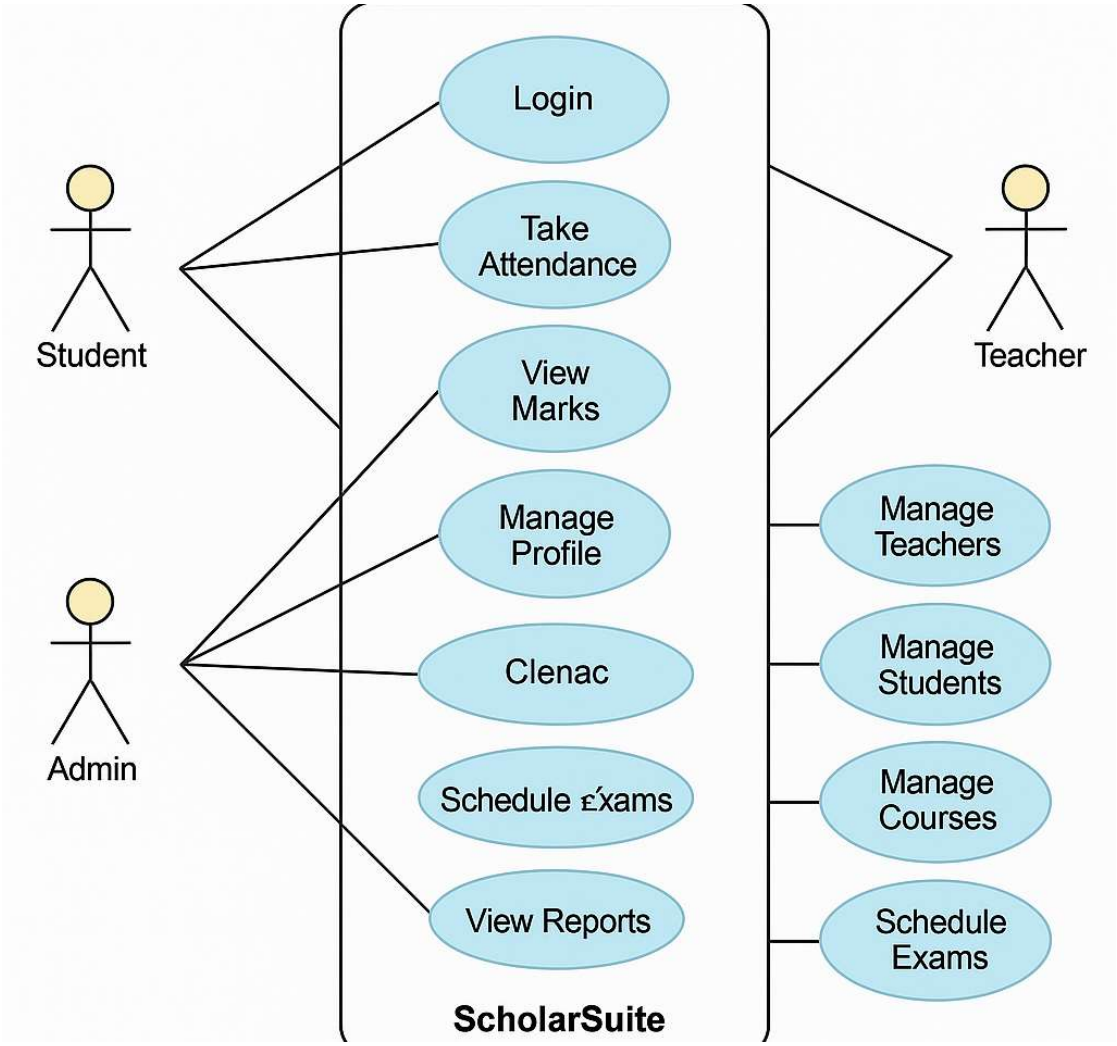


Figure 4.1: Use Case Diagram

The **Use Case Diagram** depicts the core interactions between the system and its three main user roles: **Student**, **Teacher**, **Parents, Alumni** and **Admin**.

Key Use Cases per Role:

- **All Users (Student, Teacher, Admin, Parents, Alumni)**:
  - Login
- **Student**:
  - Calculate CGPA
  - View Attendance
  - View Marks
  - Manage Profile
  - View Class
  - View Results and Credit deficiency
  - View Notes and Alumni
- **Teacher**:
  - Take Attendance
  - Manage Exams
  - Enter Marks
- **Admin**:
  - Manage Teachers
  - Manage Students
  - Manage Courses
  - Schedule Exams
  - View Reports
  - Manage Assignments
  - Manage Alumni
  - Manage Library
  - Manage events and Announcements

4.2 Class Diagram

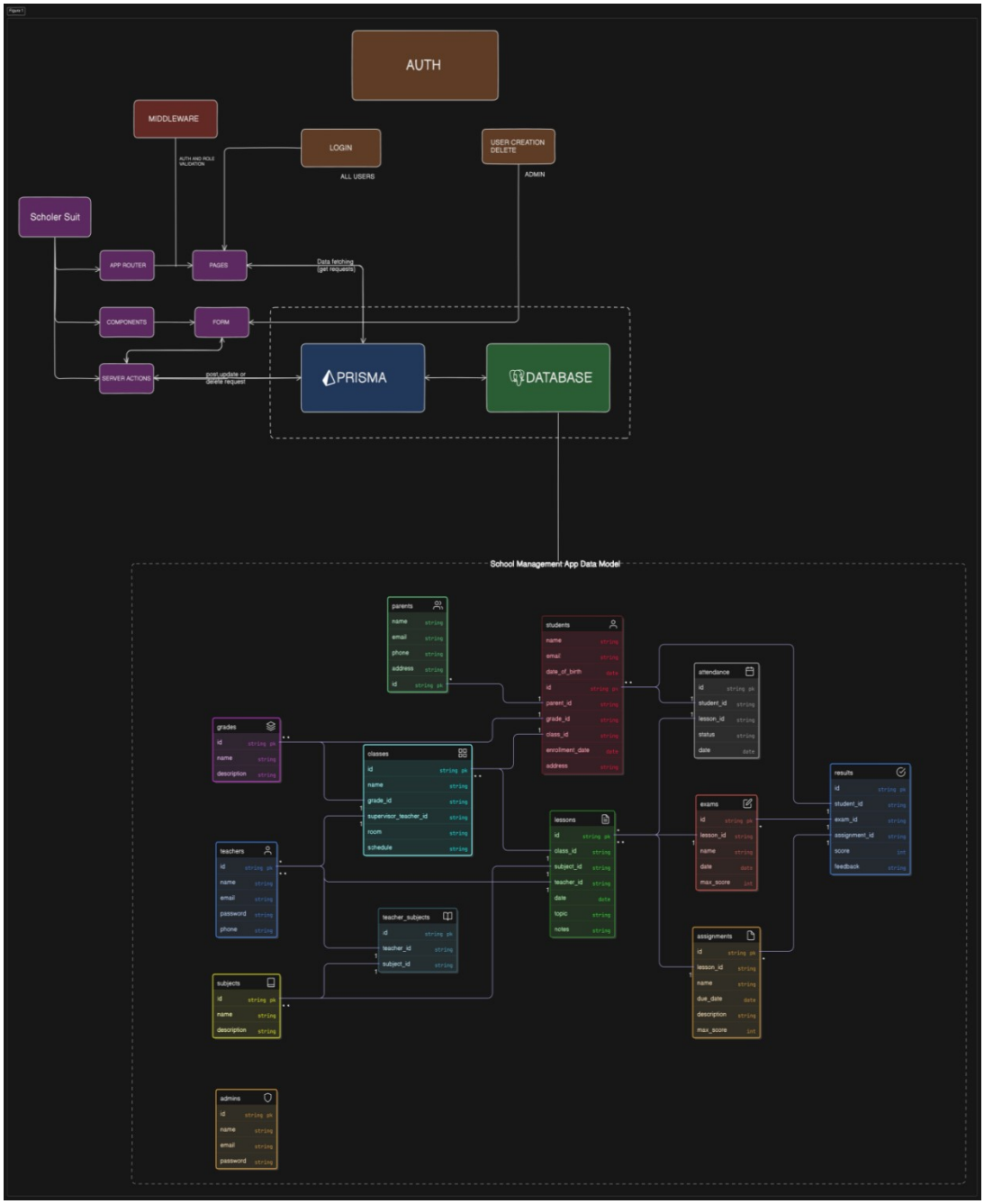


Figure 4.2: Class Diagram

ScholarSuite's data and logic layers are built on **object-oriented principles**, and its structure mirrors the relational database schema implemented via **Prisma ORM**.

Main Classes:

- User: Base class with attributes like userId, name, email, passwordHash, and role.
- Student, Teacher, Admin: Inherit from User and possess role-specific attributes and methods.
- Course: Represents a subject, linked to a Teacher and multiple Students.
- Attendance: Contains date, status, studentId, and courseId.
- Exam: Includes examId, courseId, examDate, description.
- Marks: Links a studentId with an examId, and stores marksObtained, grade.

Key Methods:

- Student.calculateCGPA()
- Teacher.enterMarks()
- Admin.manageCourses()

These classes and their associations reflect real relationships. For example, one teacher teaches many courses, one student attends many courses, and each attendance record ties one student to one course on one date.

4.3 Sequence Diagram - Login Flow

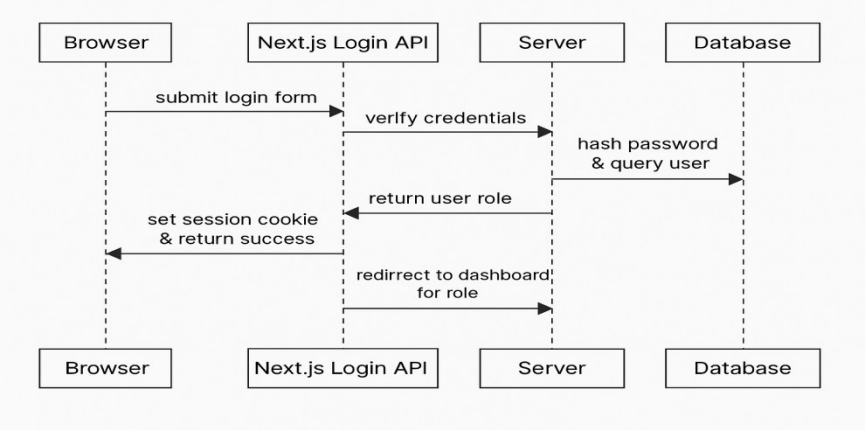


Figure 4.3: Sequence Diagram

The **Login Process** is central to ScholarSuite's secure access. Below is a step-by-step breakdown of the **login sequence**:

1. **User submits login form** via the UI.
2. The browser sends a POST request to clerk.
3. The server hashes the entered password.
4. It queries the database via **Prisma** for a matching user.
5. If valid, the server sets a **session cookie** and returns the **user role**.
6. The frontend **redirects** to the corresponding dashboard (/student, /teacher, or /admin).
7. The server may also send user-specific data (name, department) for dashboard customization.

This sequence ensures **authentication**, **role-based redirection**, and **secure session handling**.

4.4 System Architecture

ScholarSuite adheres to a **Three-Tier Web Architecture**, separating concerns across presentation, application, and data layers.

❖ **Layers and Components**:

- **Presentation Layer**:
  - Built with **React + Next.js**
  - Handles routing, UI rendering, and client-side form logic
- **Application Layer**:
  - Powered by **Next.js API routes + Node.js**
  - Performs business logic like GPA calculation, attendance processing, and exam management
- **Data Layer**:
  - Consists of **PostgreSQL database**
  - Interfaced via **Prisma ORM** for safe and optimized access

❖ **Example Workflow – Student Fetch**:

1. Teacher opens the "View Students" page.
2. Browser sends GET /api/attendance?courseId=XYZ.
3. Server executes a Prisma query
4. SELECT * FROM Attendance WHERE courseId = 'XYZ';
5. Server returns the results in JSON format.
6. UI renders the Students list.

❖ **Communication**:

All interactions use **HTTP/HTTPS**, with **JSON-based data exchange** between the client and server.

4.5 Summary

ScholarSuite's design is:

- **Modular**: Independent and reusable components
- **Secure**: Strong authentication and access control
- **Data-Driven**: Clean schema enforces data integrity
- **Scalable**: Architecture supports horizontal growth
- **Maintainable**: Clear separation of concerns across layers

Chapter 5

RESULTS AND DISCUSSION

5.1 Introduction to Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

Who does Testing?

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in context of the given requirements. Moreover, developers also conduct testing which is called Unit Testing. In most cases, the following professionals are involved in testing a system within their respective capacities:

- Software Tester
- Software Developer
- Project Lead/Manager
- End Use

Levels of testing include different methodologies that can be used while conducting software testing. The main levels of software testing are:

- Functional Testing
- Non-functional Testing
- Functional Testing

This is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of a software is conducted on a complete, integrated system to evaluate the system's

compliance with its specified requirements.

Software Testing Life Cycle

The process of testing a software in a well-planned and systematic way is known as software testing lifecycle (STLC). Different organizations have different phases in STLC however generic Software Test Life Cycle (STLC) for waterfall development model consists of the following phases.

- Requirements Analysis
- Test Planning
- Test Analysis
- Test Design

Requirements Analysis

In this phase testers analyze the customer requirements and work with developers during the design phase to see which requirements are testable and how they are going to test those requirements. It is very important to start testing activities from the requirements phase itself because the cost of fixing defect is very less if it is found in requirements phase rather than in future phases

Test Planning

In this phase all the planning about testing is done like what needs to be tested, how the testing will be done, test strategy to be followed, what will be the test environment, what test methodologies will be followed, hardware and software availability, resources, risks etc. A high-level test plan document is created which includes all the planning inputs mentioned above and circulated to the stakeholders.

Test Analysis

After test planning phase is over test analysis phase starts, in this phase we need to dig deeper into project and figure out what testing needs to be carried out in each SDLC phase. Automation activities are also decided in this phase, if automation needs to be done for software product, how will the automation be done, how much time will it take to automate and which features need to be automated. Non-functional testing areas (Stress and performance testing) are also analyzed and defined in this phase.

Test case Design

In this phase various black-box and white-box test design techniques are used to design the test cases for testing, testers start writing test cases by following those design techniques, if automation testing needs to be done then automation scripts also needs to written in this phase.

5.2 Test Cases Design & Execution

Table 5.1 :Testcase 1

Test Scenario ID		ATT001		**Test case ID**	ATTTC01	
Test case description		Attendance Marking		**Test Priority**	HIGH	
Pre-Requisite		User logged in as teacher		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected Output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Mark attendance	Select class and students	Attendance saved	Marked present/absent as selected	Pass	NA
2	Submit with no student marked	Class selected but no students	Show warning	Warning “select at least one student”	Pass	NA
3	Try Unauthorized user	Logged in as student	Access denied	Access blocked	Pass	NA

Table 5.2 :Testcase 2

Test Scenario ID		AUTH001		**Test case ID**	AUTHTC01	
Test case description		Login		**Test Priority**	HIGH	
Pre-Requisite		User Account		**Post -Requisite**	Valid session created	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Valid login	Correct username and password	Correct Username And password	Dashboard loaded	Pass	NA
2	Invalid login	Wrong password	Wrong password	Error Message displayed	Pass	NA
3	Role mismatch	Student Tries Admin dashboard	Student Tries Admin dashboard	Redirected To correct Dashboard.	Pass	NA

Table 5.3 :Testcase 3

Test Scenario ID		EXAM001		**Test case ID**	EXAMTC01	
Test case description		Exam Creation and Marks Entry		**Test Prioriy**	MEDIUM	
Pre-Requisite		Admin or Teacher logged in		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Schedule new exam	Date, subject, class	Exam scheduled	Listed under upcoming exams	PASS	NA
2	Enter student marks	Valid score entered	Marks saved	Grades calculated and saved	PASS	NA
3	Enter out-of-range marks	Score > 100	Show validation	Error message shown	PASS	NA
4	Leave score empty	Field blank	Validation error	Prompt to enter marks	PASS	NA

Table 5.4 :Testcase 4

Test Scenario ID		CGPA001		**Test case ID**	CGPATC01	
Test case description		CGPA Calculation Functionality		**Test Priority**	HIGH	
Pre-Requisite		User must be logged in as a student		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Fill all fields	Valid grades and credits per subject	Show CGPA value	CGPA calculated and shown correctly	PASS	NA
2	Leave a field empty	Missing grade or credit	Show validation error	Validation error shown	PASS	NA
3	Enter invalid data	Enter text in numeric credit field	Show type error	Validation error shown	PASS	NA
4	Use decimal values	Grade: 8.5, Credit: 3	Accept decimals and calculate accurately	Accurate CGPA shown	PASS	NA

Table 5.5 :Testcase 5

Test Scenario ID			DASH001	**Test case ID**	DASHTC01	
Test case description			Dashboard Access Based on Role	**Test Priority**	HIGH	
Pre-Requisite			User logged in.	**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Access as Student	Role = Student	Load Student dashboard	Correct dashboard shown	PASS	NA
2	Access as Teacher	Role = Teacher	Load Teacher dashboard	Correct dashboard shown	PASS	NA
3	Access as Admin	Role = Admin	Load Admin dashboard	Correct dashboard shown	PASS	NA
4	Access dashboard without login	No session	Redirect to login	Redirected to login page	PASS	NA

Table 5.6 :Testcase 6

Test Scenario ID		PROF001		**Test case ID**	PROF01	
Test case description		Profile View and Edit		**Test Priority**	MEDIUM	
Pre-Requisite		User logged in.		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	View profile	Logged in user	Profile info displayed	All fields shown correctly	PASS	NA
2	Edit profile	Change name, email, etc.	Changes saved	Info updated in DB	PASS	NA
3	Submit invalid email	Invalid email format	Show validation error	Email format error shown	PASS	NA

Table 5.7 :Testcase 7

Test Scenario ID		RESULT001		**Test case ID**	RESULTTC01	
Test case description		Viewing Exam Results		**Test Priority**	HIGH	
Pre-Requisite		Student logged in		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	View marks list	Logged in as Student	Show subject-wise marks	Marks with subject shown	PASS	NA
2	No marks yet	Student without exam records	Show “No records found”	Appropriate message shown	PASS	NA
3	View by teacher	Logged in as Teacher	Show class-wise marks	Marks by student shown correctly	PASS	NA

Table 5.8 :Testcase 8

Test Scenario ID		SEC001		**Test case ID**	SEC_TC01	
Test case description		Access Control and Unauthorized Access		**Test Priority**	CRITICAL	
Pre-Requisite		User sessions active		**Post - Requisite**	NA	
S. No.	**Action**	**Inputs**	**Expected output**	**Actual Output Test Browser**	**Test Result**	**Test Comments**
1	Student opens admin URL	URL: /admin	Access Denied / Redirect to login	Access blocked	PASS	NA
2	Teacher tries to delete user	URL: /delete-user (admin only)	Access Denied	Redirect or error shown	PASS	NA
3	Unauthenticated user hits API	No token/session	Unauthorized response	401 Unauthorized shown	PASS	NA

5.3 Results

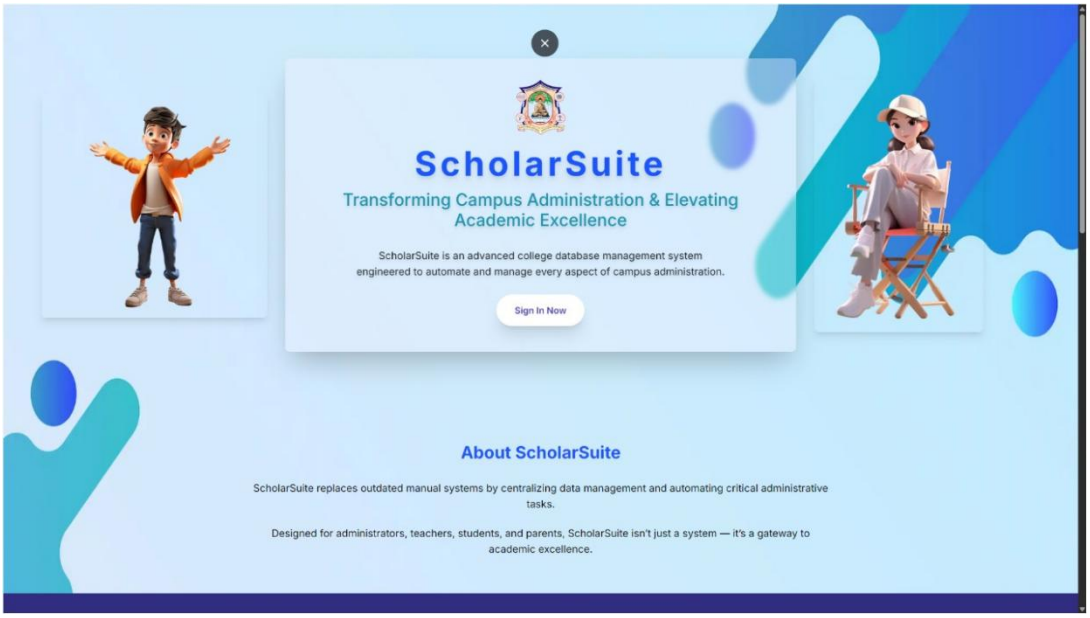


Figure 5.1: Landing Page

The above page introduces ScholarSuite, a college management system that centralizes data and automates campus administration tasks, with a sign-in option for users to access the platform.

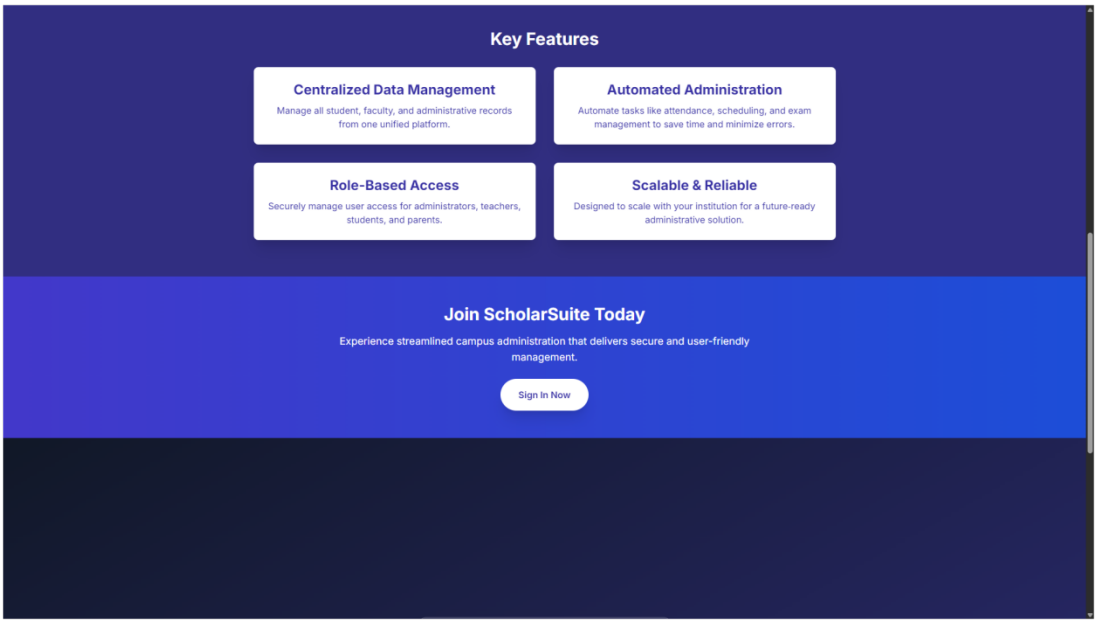


Figure 5.2: About Page

This Above page highlights ScholarSuite's key features—centralized data management, automated administration, role-based access, and scalability—while encouraging users to sign in and streamline campus operations.

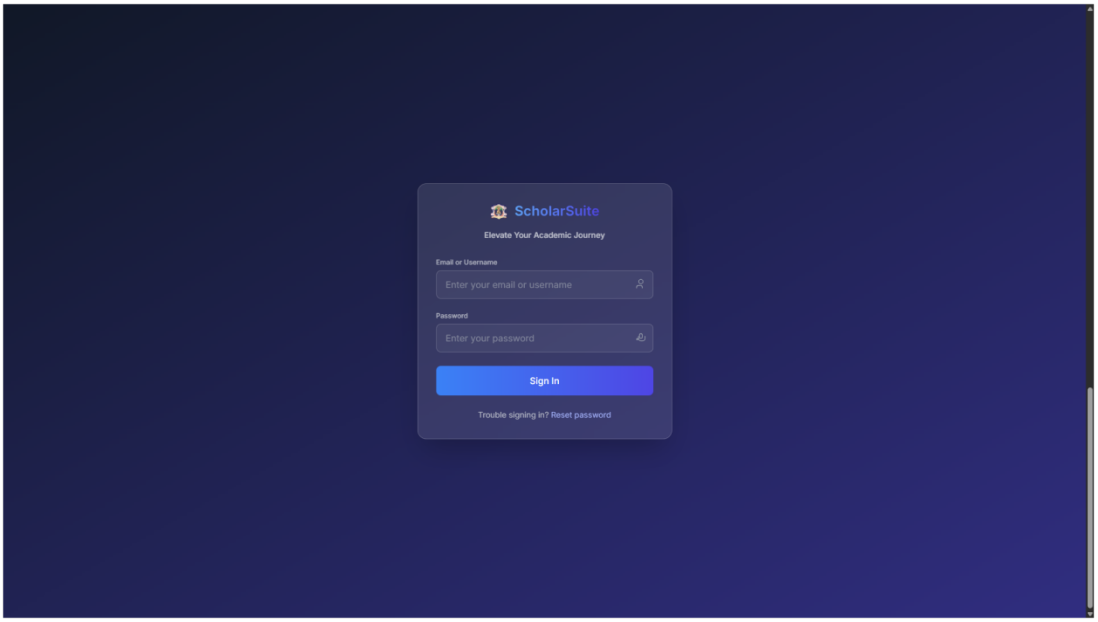


Figure 5.3: Sign-in page

This Sign-In page allows registered users to log into ScholarSuite by entering their credentials authenticated and processed by clerk, providing secure access to the management system.

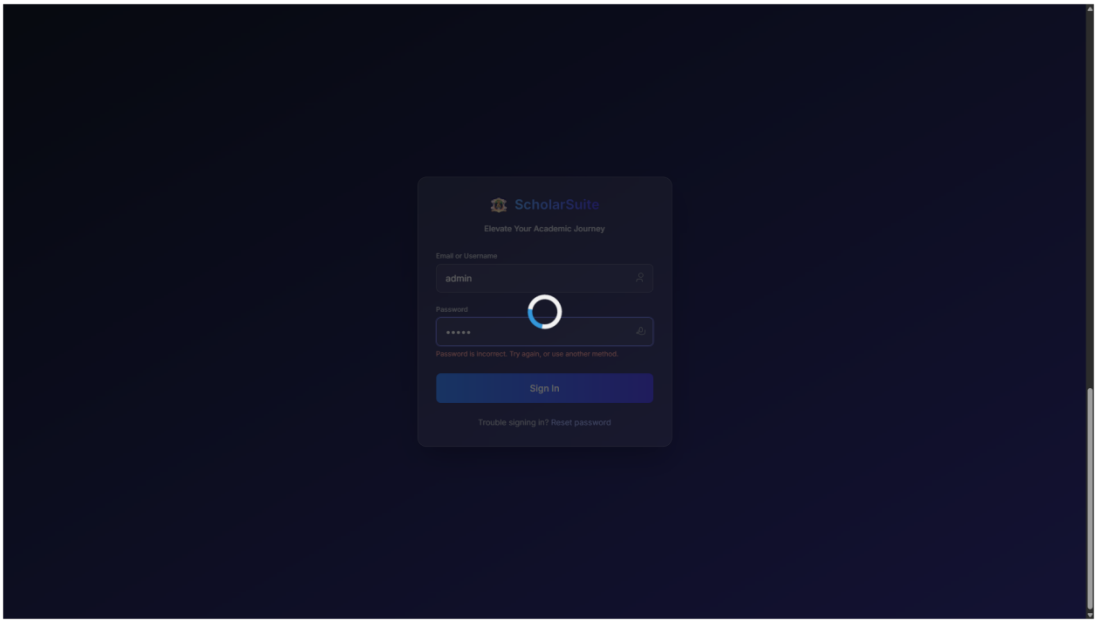


Figure 5.4: Incorrect Sign-in Page

This screen displays an incorrect login attempt on the ScholarSuite Sign-In page, prompting the user to retry or use another method for secure access.

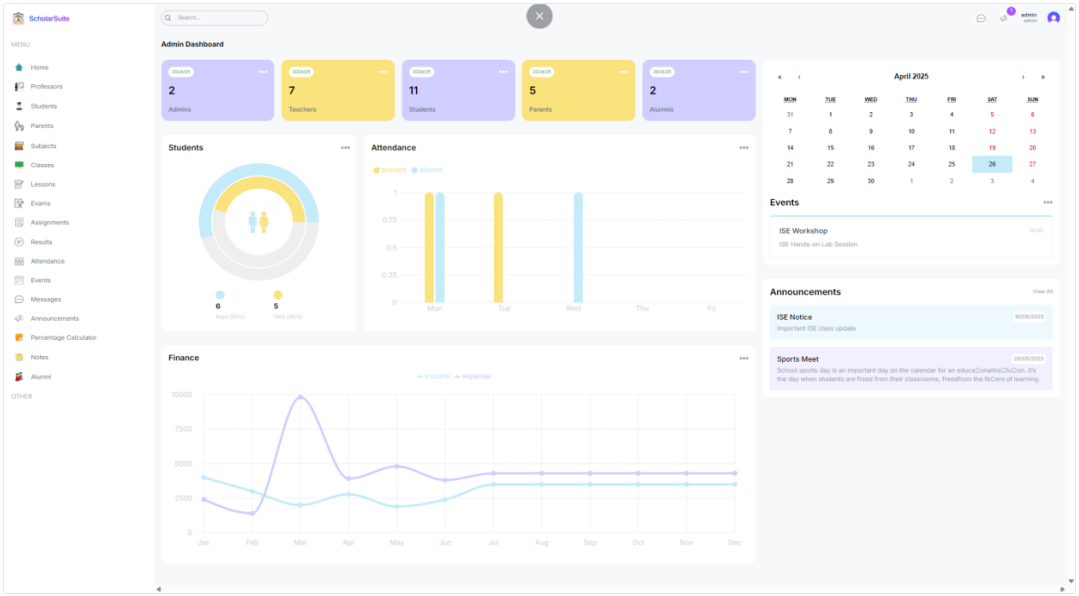


Figure 5.5: Admin Dashboard

This is the Admin Dashboard of ScholarSuite, displaying user statistics, student attendance, finance graphs, event calendar, and announcements to provide a comprehensive overview of campus operations.

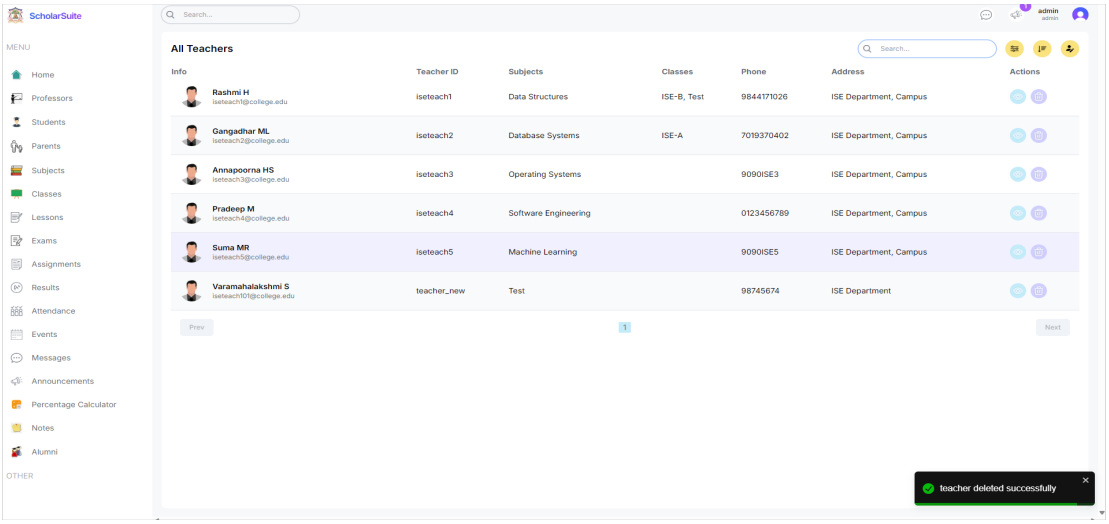


Figure 5.6:List of Teachers

This page displays a comprehensive list of teachers, including their names, IDs, subjects, assigned classes, contact information, and campus address. It helps administrators easily manage faculty records and perform actions like editing or removing entries, and notifications are developed by using React toastify library.

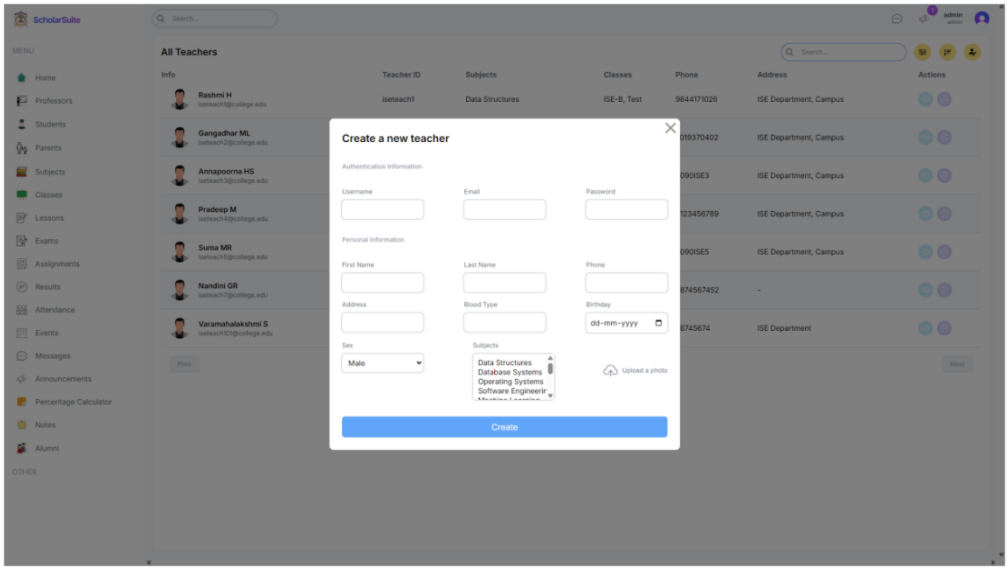


Figure 5.7:Form to create a new Teacher

This page shows a form used to create a new teacher profile by entering authentication and personal details, including subjects taught and a profile photo. It ensures proper onboarding and record-keeping of faculty within the ScholarSuite system.

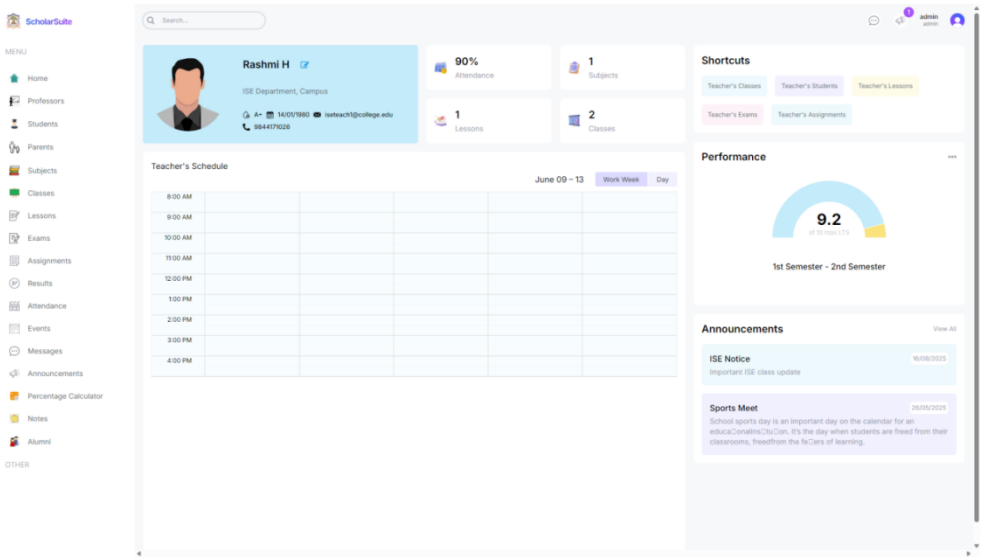


Figure 5.8:Teacher profile details

This page displays a teacher's profile with contact details, department, and assigned subjects, lessons, and classes. It also includes a weekly schedule, performance score, and recent announcements, offering a complete overview of the teacher's involvement.

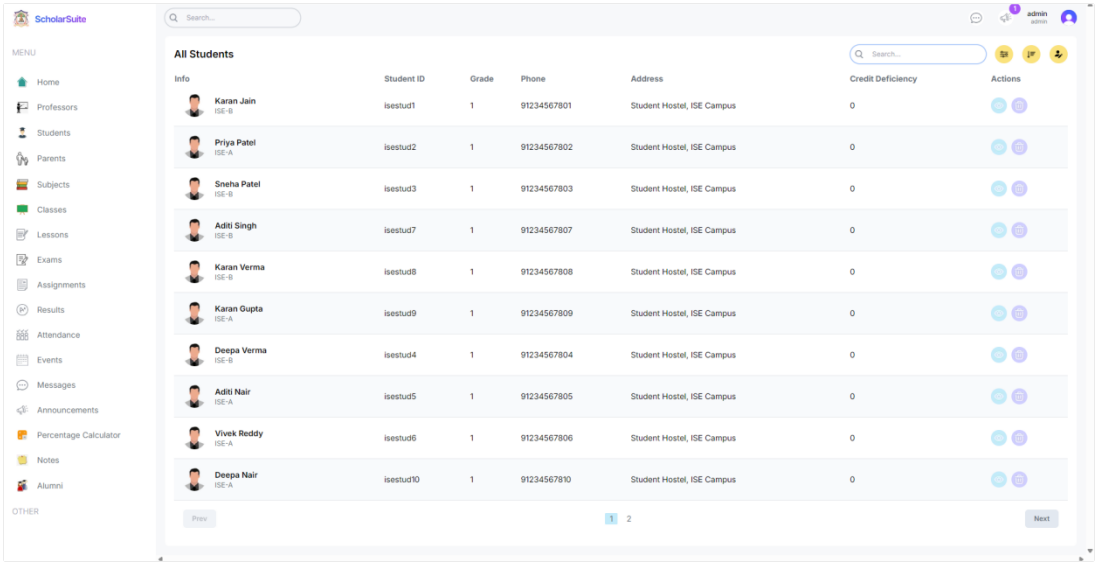


Figure 5.9:List of Students

This page presents a detailed list of all students, including their names, IDs, grade levels, contact information, addresses, and credit deficiency status. It allows easy access and management of student data within the ScholarSuite platform.

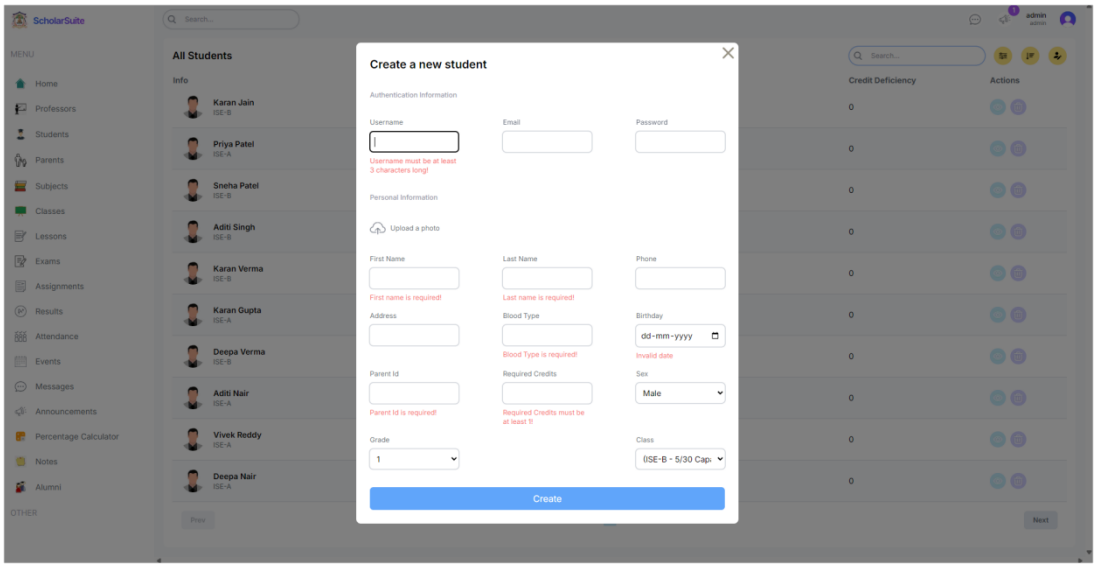


Figure 5.10:Form to create a new Student

This page features a form to register a new student by collecting login credentials and personal details such as name, contact info, parent ID, and class assignment it ensures form validation and validates data accordingly for. It ensures structured student onboarding into the system.

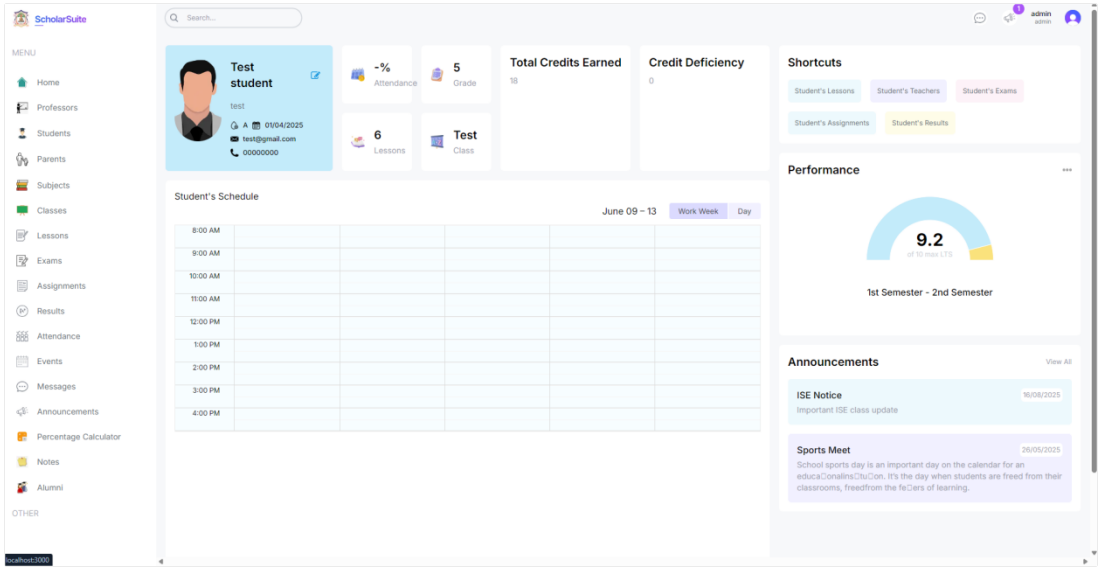


Figure 5.11:Student profile details

This page displays the student dashboard featuring a profile summary, class schedule, performance, and shortcuts to important sections like assignments and results. It also shows attendance, grades, credit status, and recent announcements for the student.

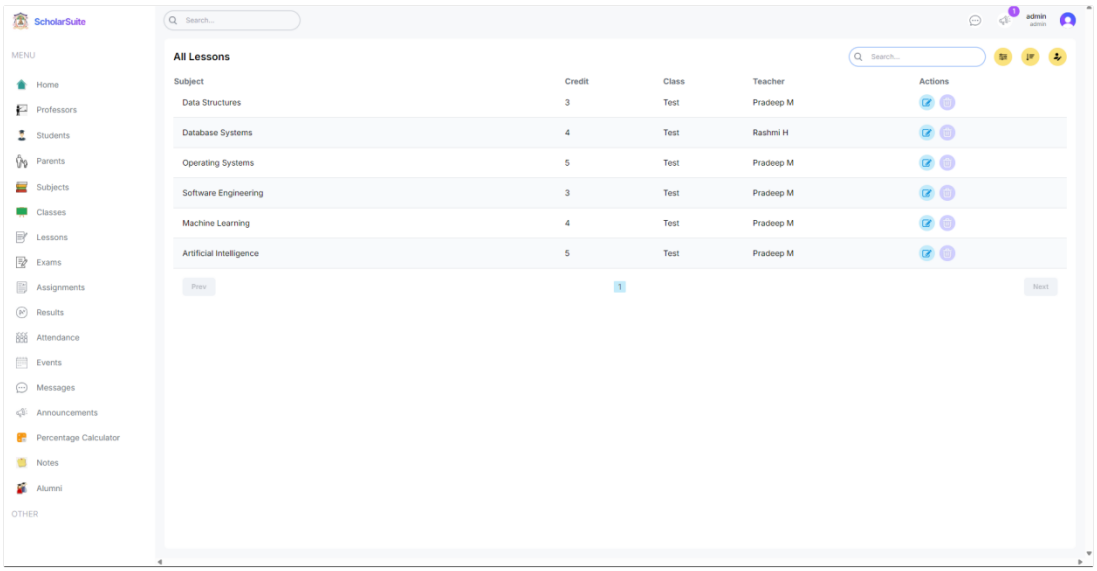


Figure 5.12:Lessons and its Details

This page displays a list of all lessons assigned to a student, including subject names, credit values, class name, and respective teachers. It also provides action buttons for viewing, editing, or managing each subject.

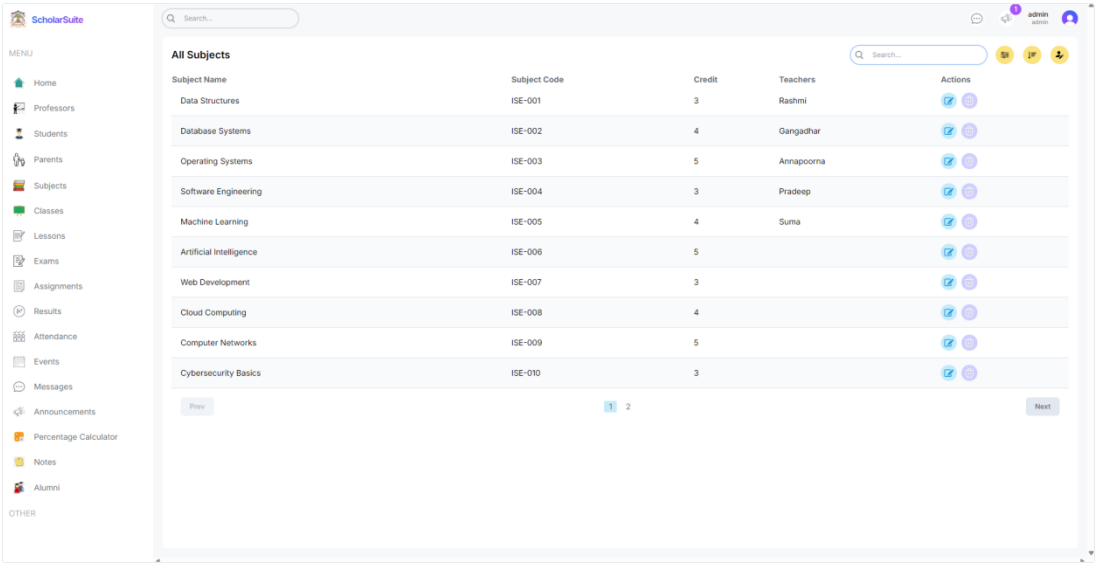


Figure 5.13:Subjects and its Details

This page presents a list of all subjects available, including their names, subject codes, credit values, and assigned teachers. Each subject row includes action buttons for viewing or editing subject details.

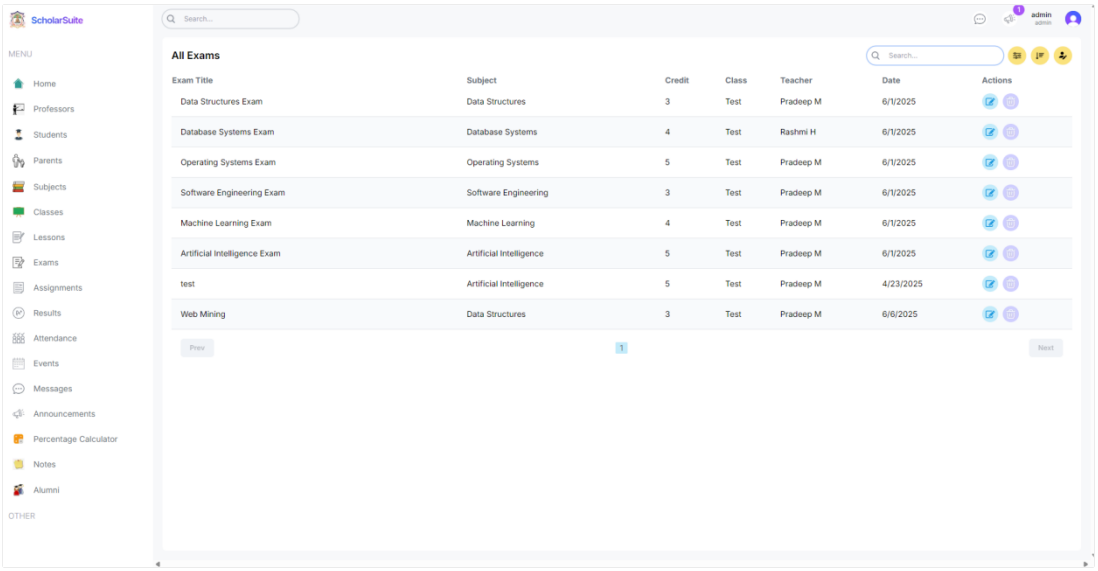


Figure 5.14:Exams and its Details

This page lists all exams along with details such as exam title, related subject, credit value, teacher name, class, and exam date. Action icons allow users to view or manage each exam entry.

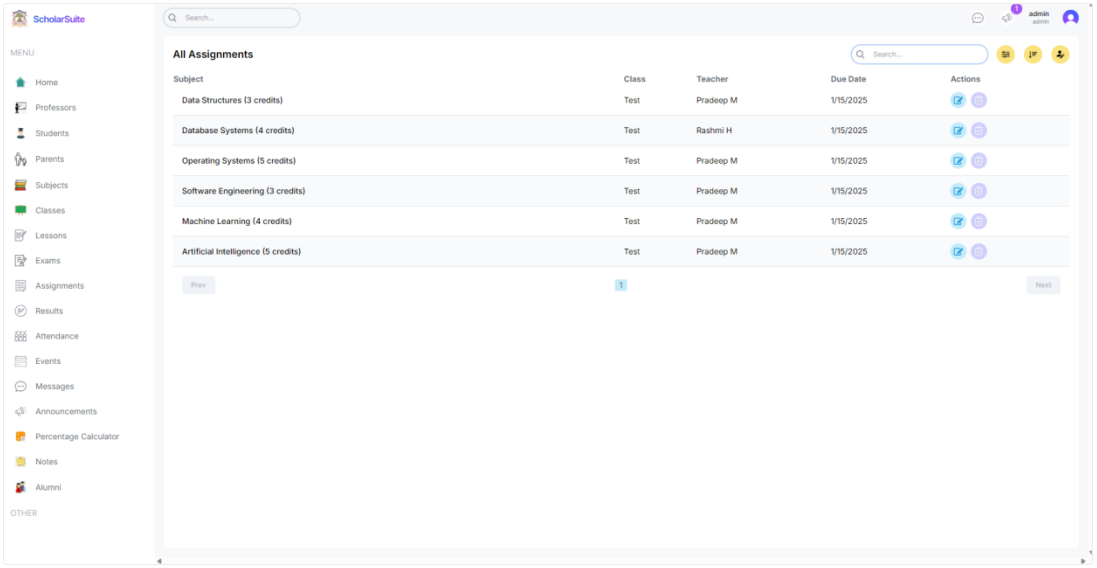


Figure 5.15:Assignments and its Details

This page displays a list of all assignments with details such as subject name (including credit info), assigned class, teacher, and due date. Each row offers action icons to view or manage assignment submissions.

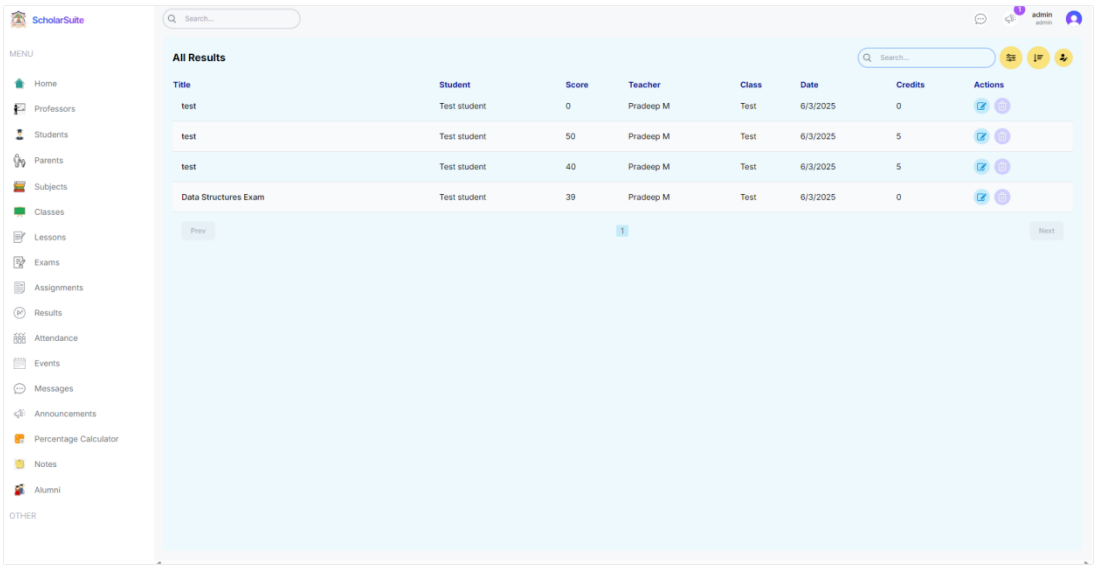


Figure 5.16:Results with details

This page shows the results of a student's assessments, including exam titles, scores, teachers, credits earned, and exam dates. Action buttons are available for viewing, editing, or managing each result entry.

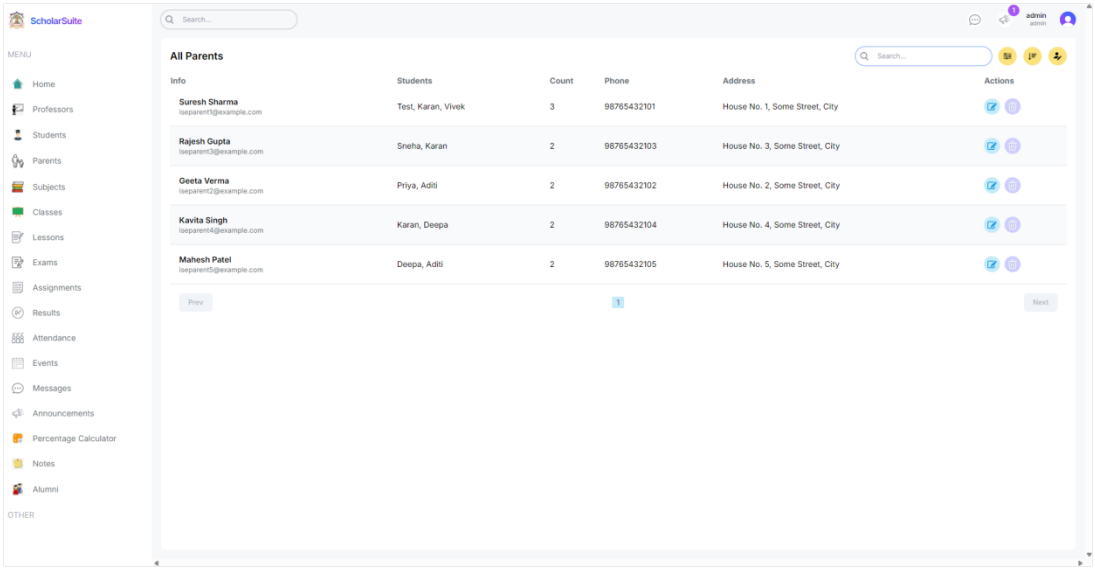


Figure 5.17:Parents and their details

This page lists parent details, including their name, email, associated students, number of children, contact number, and address. It also features action buttons to view or edit parent information.

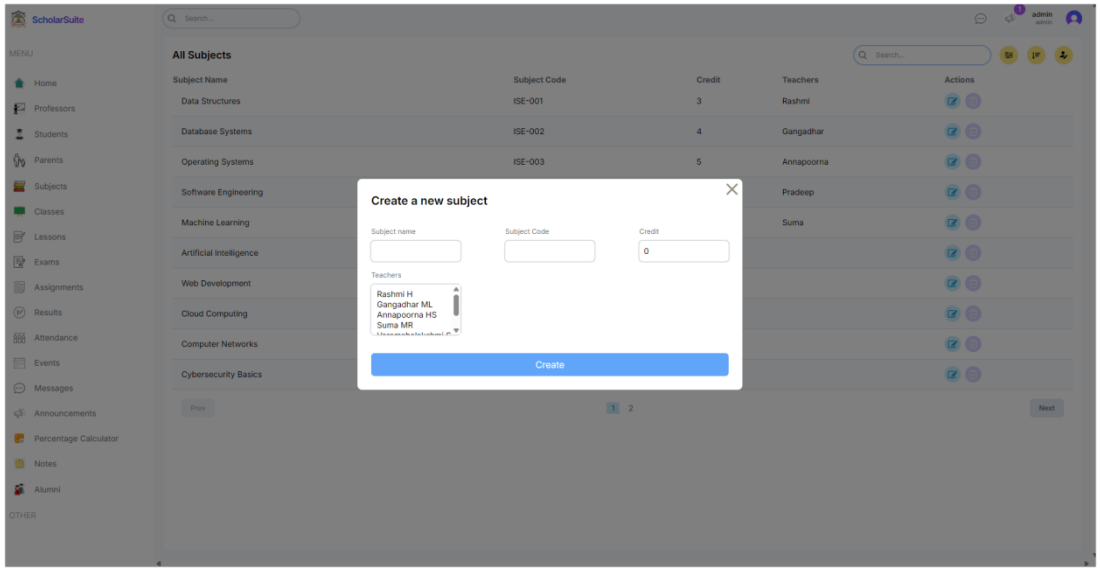


Figure 5.18:Form to add new subject

This pop-up form allows the admin to create a new subject by entering the subject name, code, credit value, and selecting one or more teachers. Once filled, clicking the "Create" button adds the subject to the subject list.

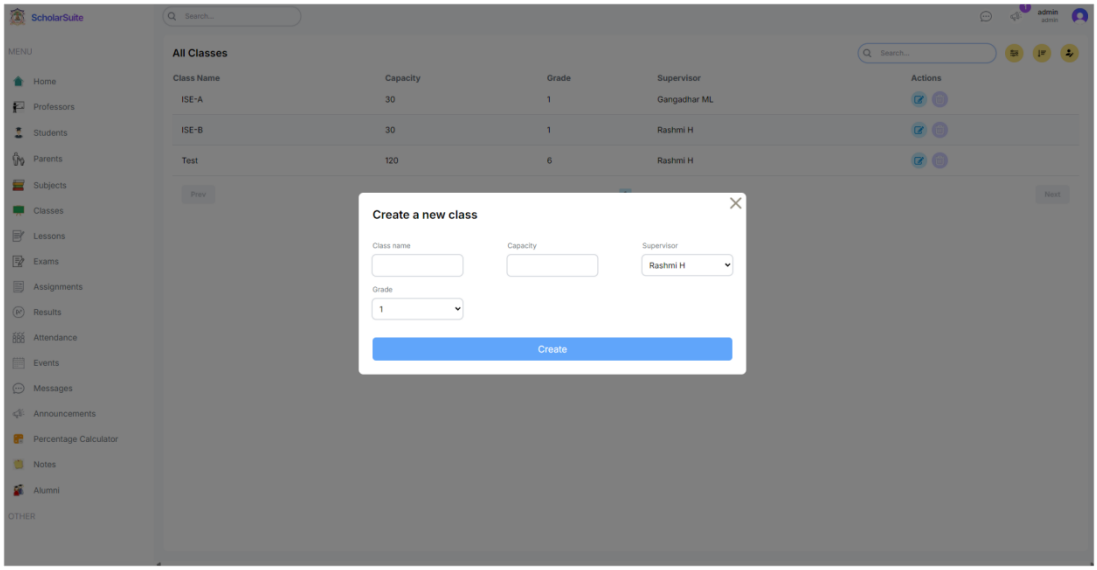


Figure 5.19:Form to add new class

This pop-up form is used to create a new class by entering the class name, capacity, grade, and selecting a supervisor. Once submitted via the “Create” button, the class gets added to the existing class list.

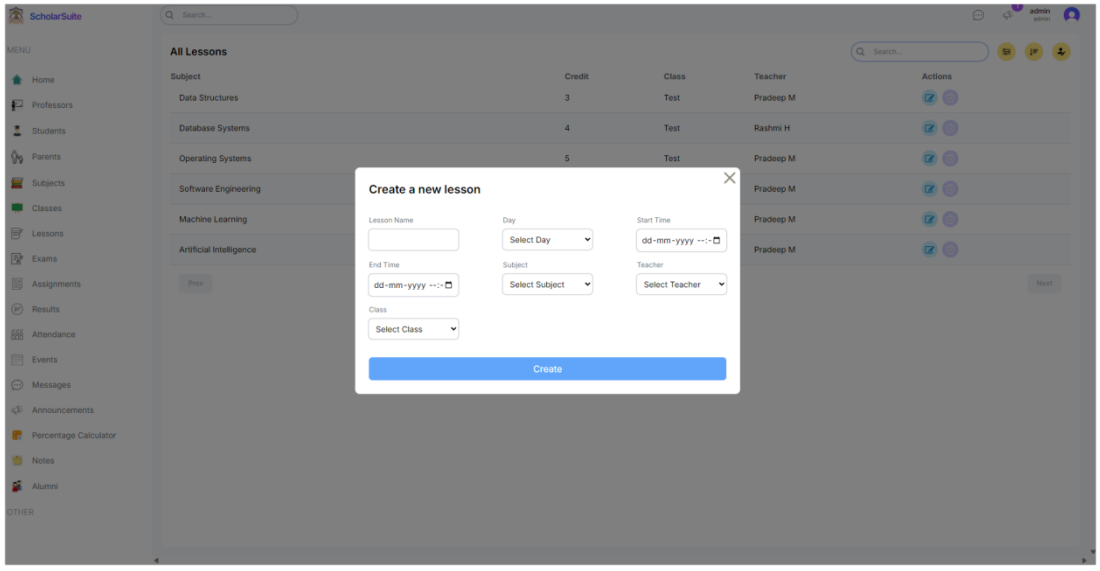


Figure 5.20:Form to create new Lesson

This page from the ScholarSuite platform displays a lesson management interface. It shows a list of existing lessons along with a popup form to create a new lesson by selecting details such as subject, teacher, class, and schedule.

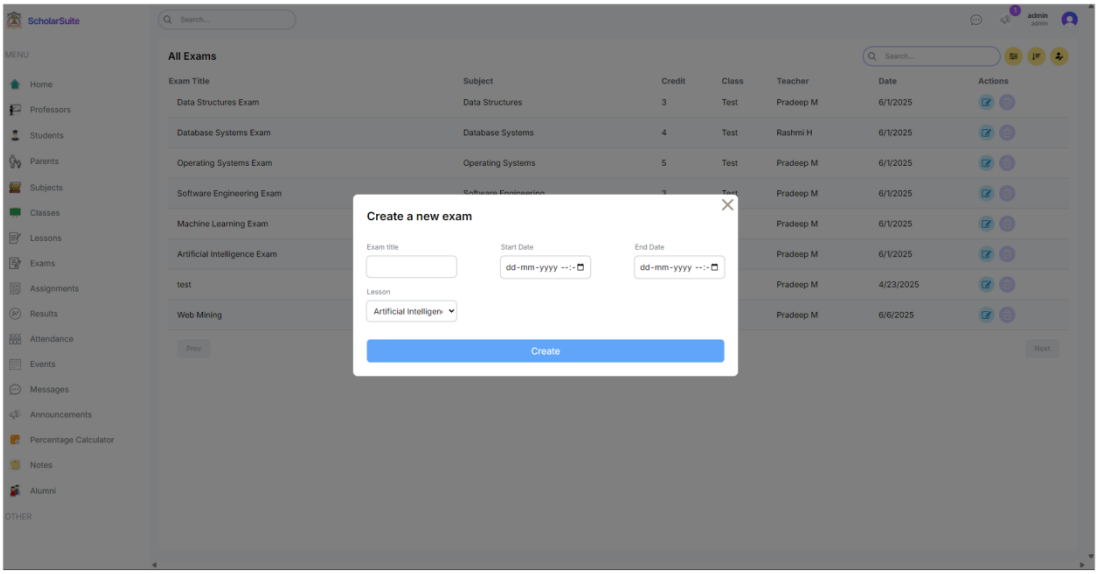


Figure 5.21:Form to create a new Exam

This page allows users to manage exams within the ScholarSuite platform. It displays a list of scheduled exams and includes a form to create a new exam by entering its title, dates, and selecting an associated lesson.

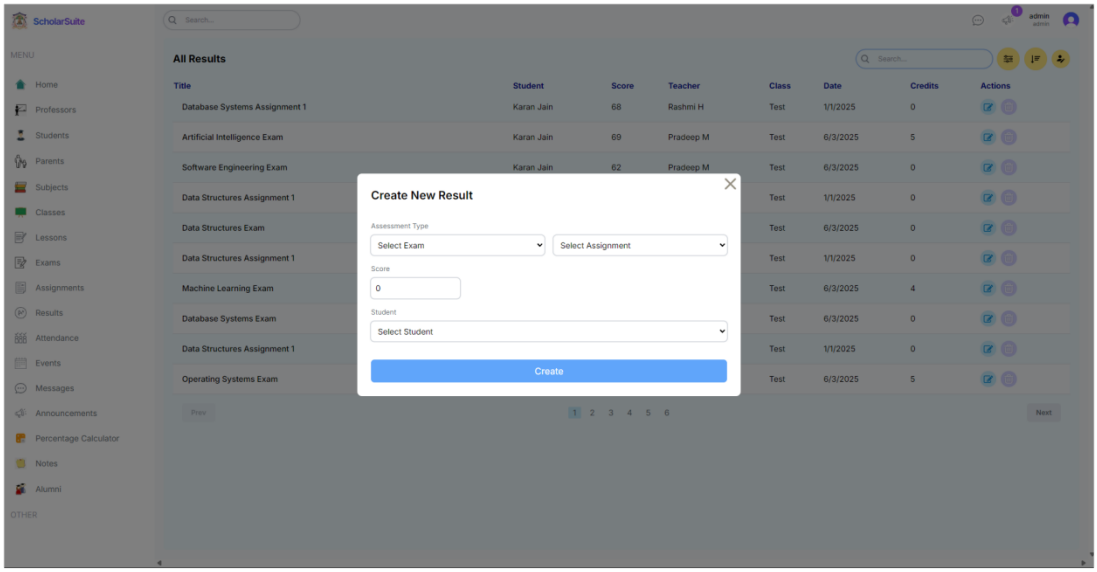


Figure 5.22:Form to create new result

This page shows the results management section of ScholarSuite, listing students' scores for various assessments. The pop-up form allows the creation of a new result by selecting the assessment type, exam or assignment, entering the score, and assigning it to a student.

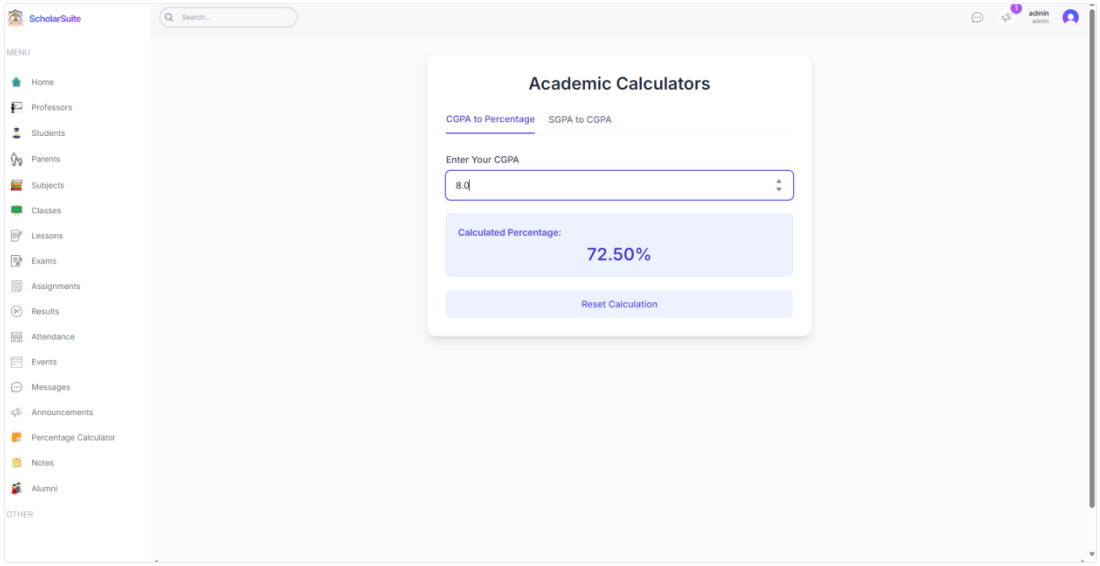


Figure 5.23:Calculator to calculate the percentage from CGPA

This page features an calculator that converts CGPA to percentage. By entering a CGPA value, users can instantly view the corresponding percentage, aiding in evaluations or form submissions.

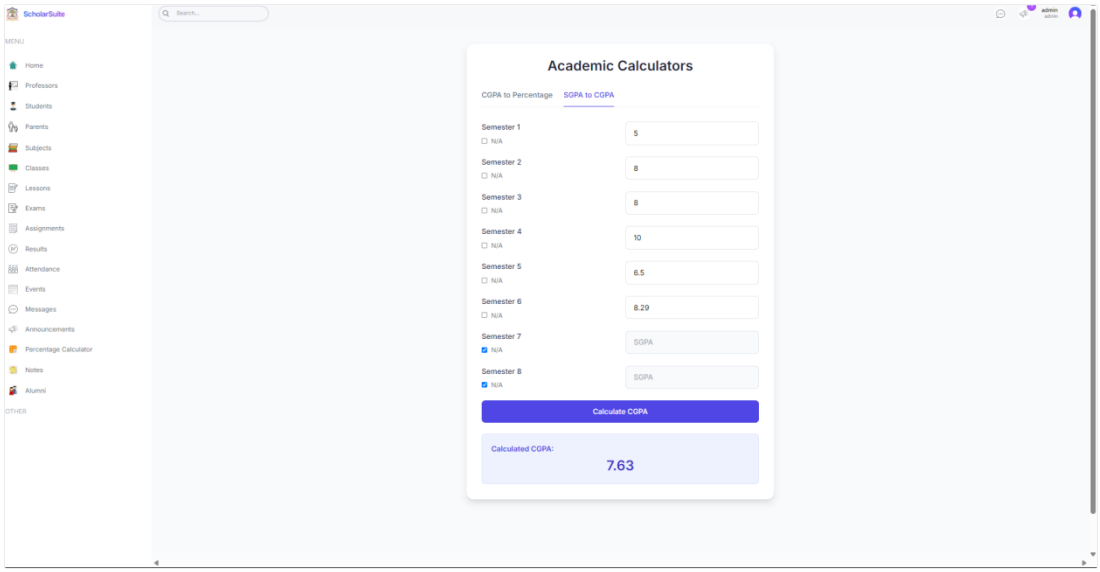


Figure 5.24:Calculator to calculate the CGPA from SGPA's

This page provides an calculator that converts SGPAs from multiple semesters into an overall CGPA. Users input individual semester GPAs, and the tool computes the average CGPA for tracking or reporting.

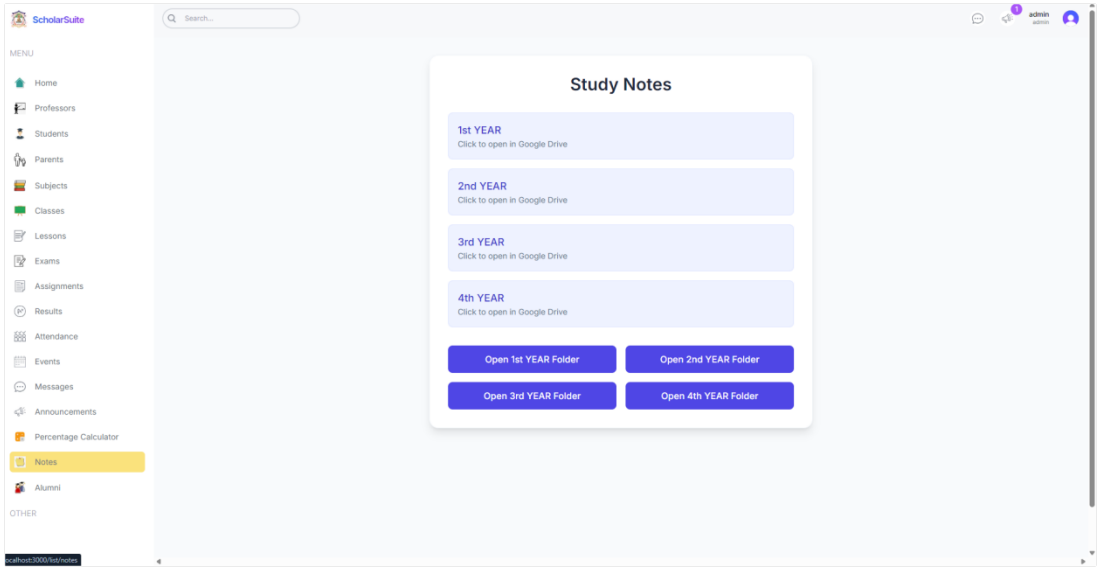


Figure 5.25:Digital Library

This page serves as a digital library for study notes, organized by year from 1st to 4th year. Users can access or open Google Drive folders containing relevant materials for each year with a single click.

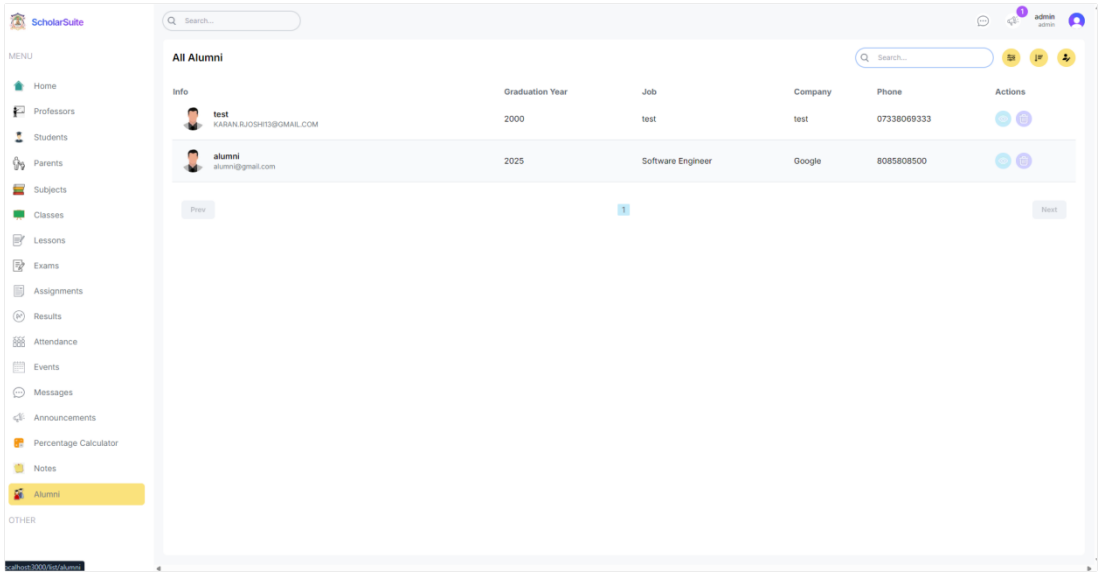


Figure 5.26:List of Alumni

This page displays a directory of alumni within the ScholarSuite system. It includes details such as name, graduation year, job title, company, and contact information, helping maintain connections with former students.

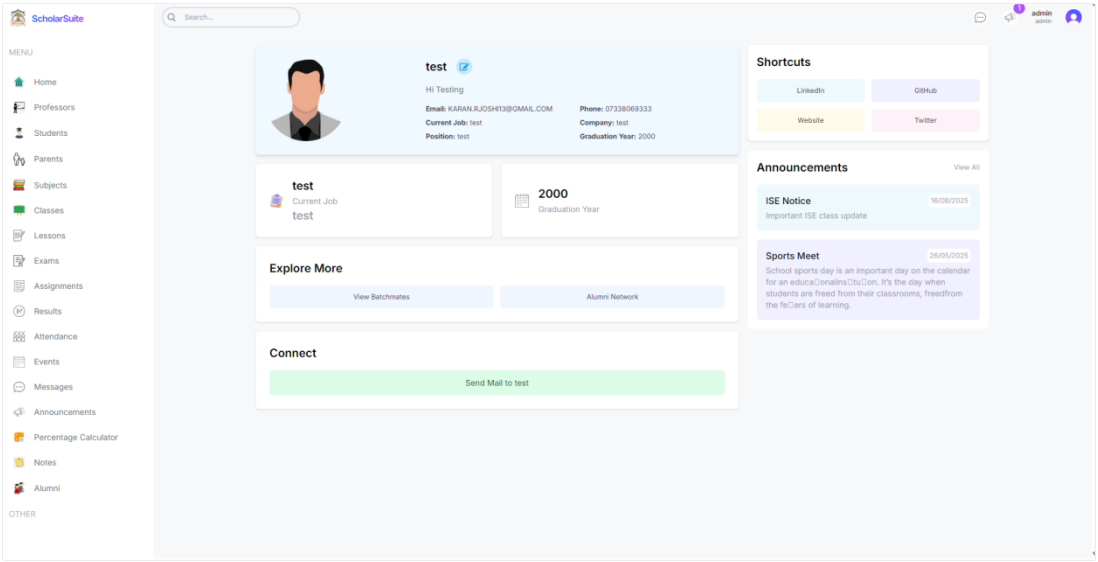


Figure 5.27:Profile details of Alumni

This page shows a detailed alumni profile within the ScholarSuite system. It includes personal and professional information, social shortcuts, announcements, and options to explore alumni networks or connect via email.

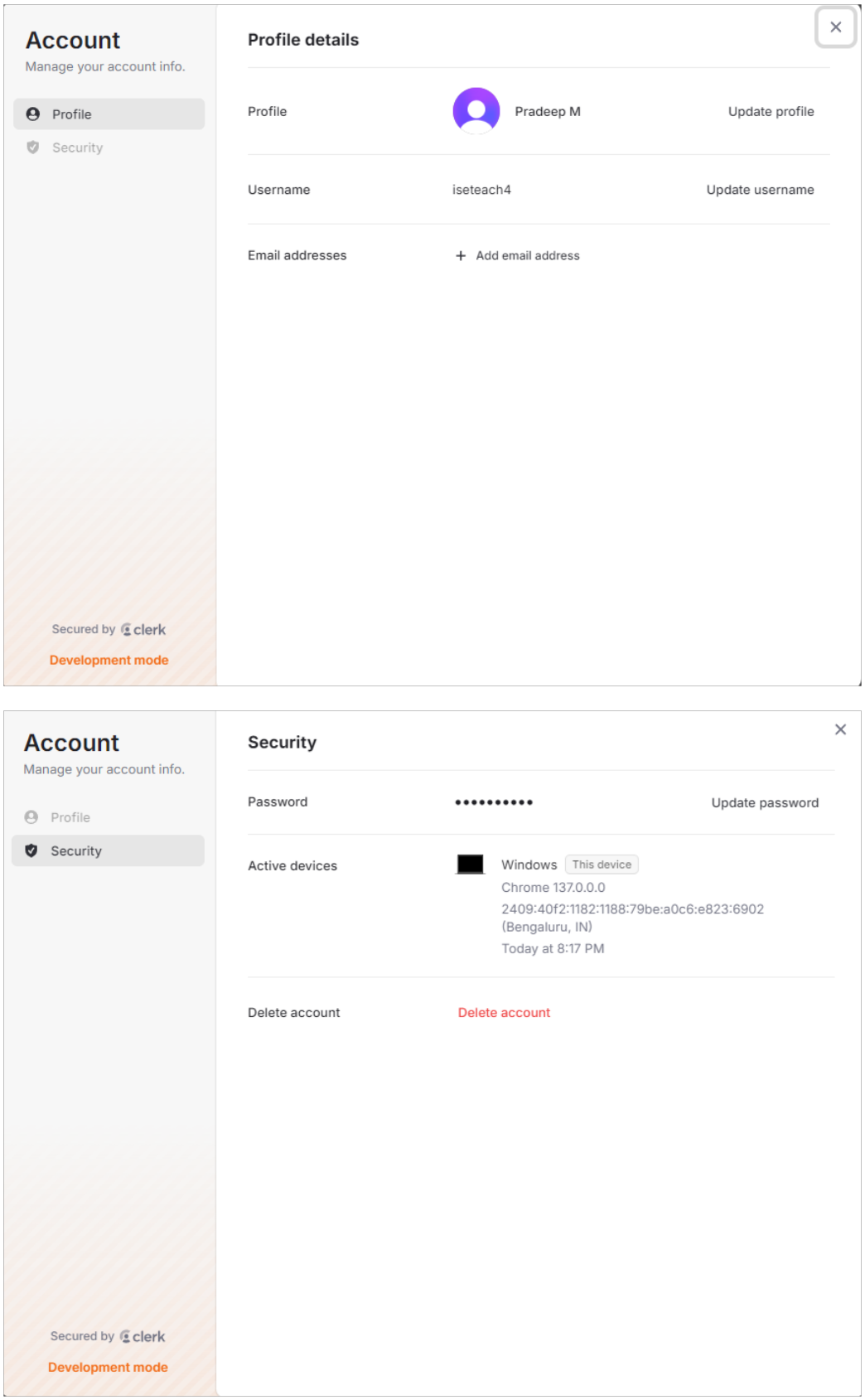


Figure 5.28:Account details and password update

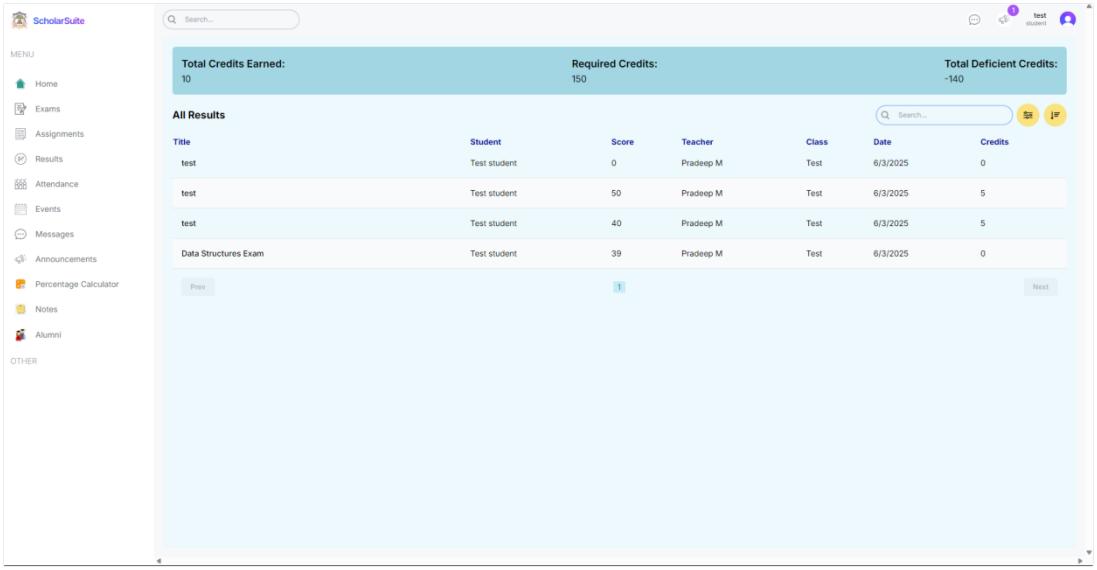


Figure 5.29:Credit deficiency details of a student

This page displays a student's credit summary, including total earned, required, and deficient credits. It also lists individual results with scores and credit values, helping track progress toward graduation requirements.

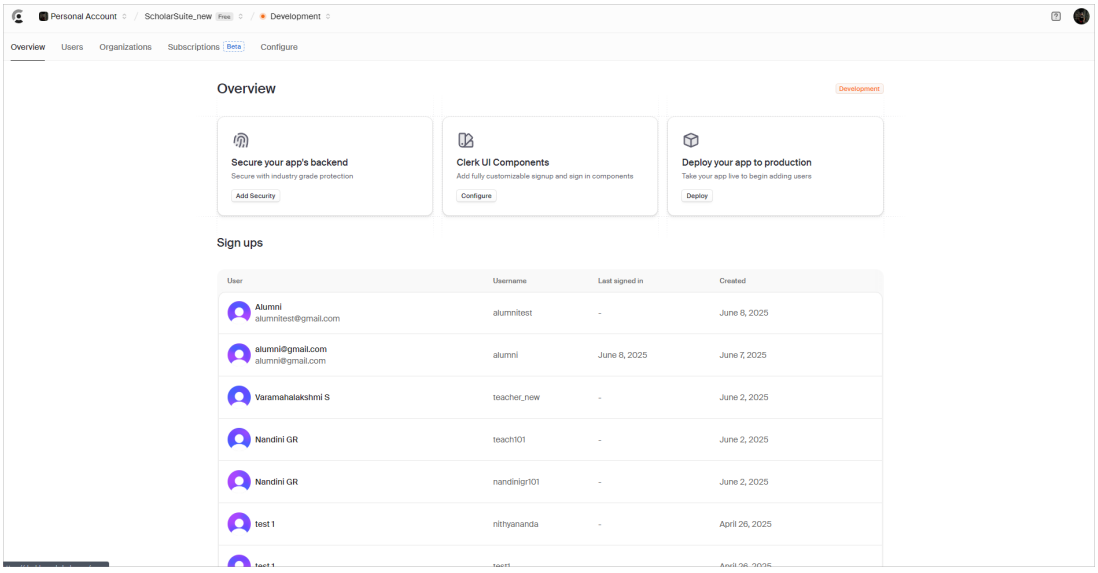
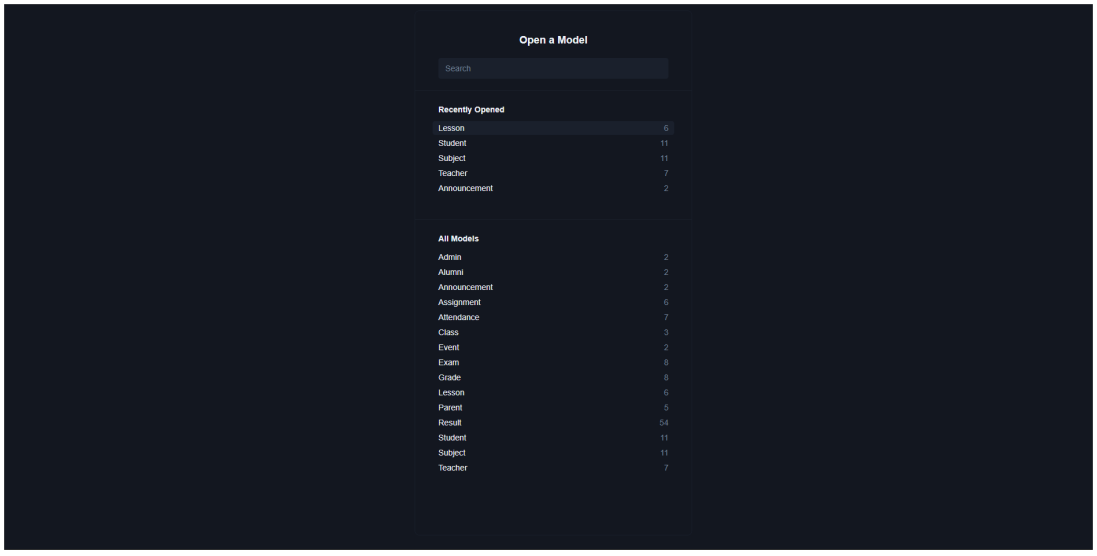
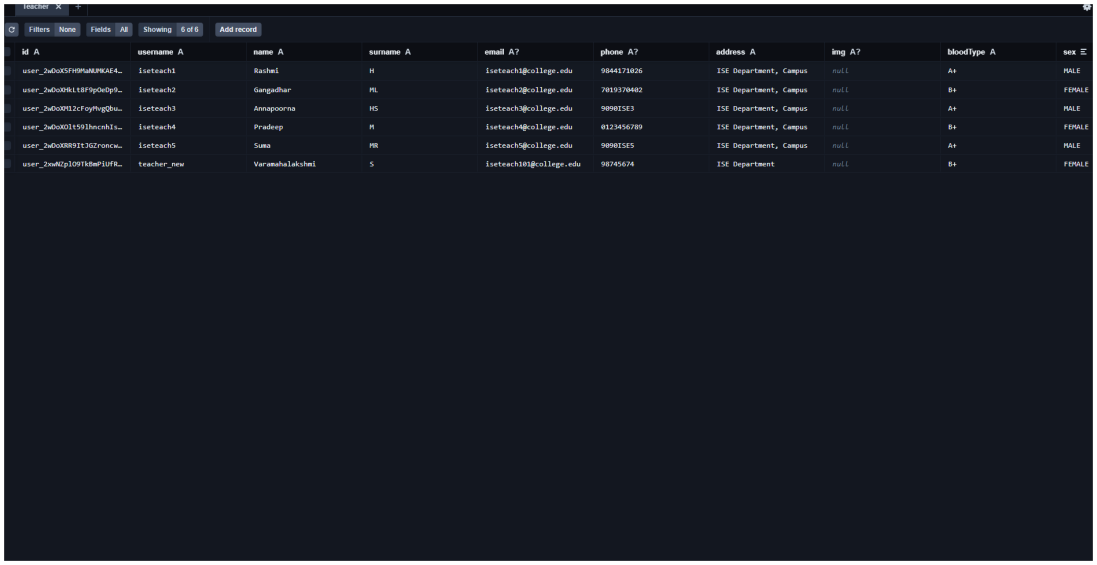


Figure 5.30:Login details at clerk security service provider

This is the Clerk dashboard showing user authentication details for the development environment. It lists users, their usernames, sign-in activity, and registration dates. The top section provides setup options for backend security, UI components, and deployment.





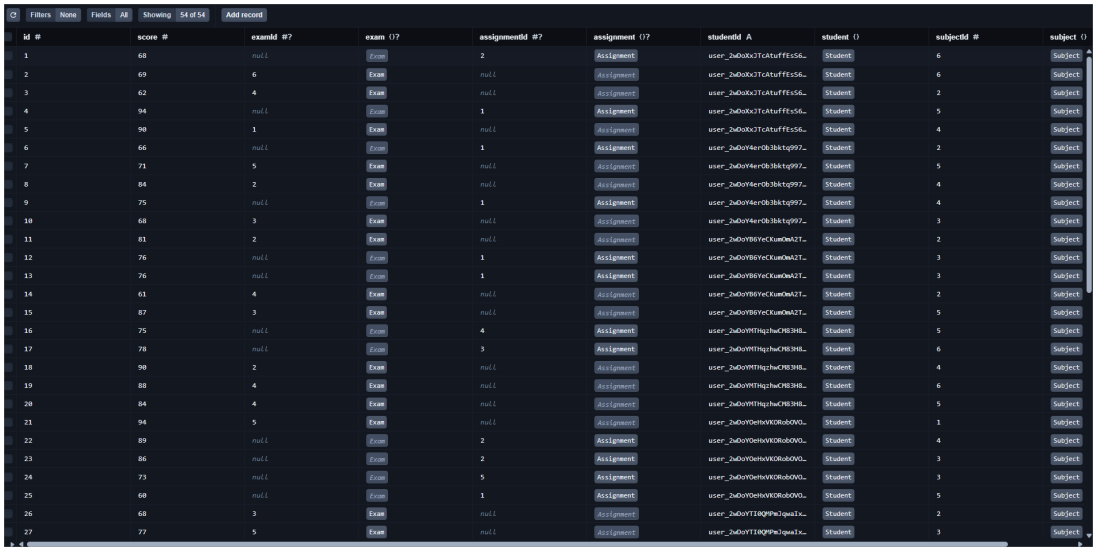


Figure 5.31:Prisma ORM

5.4 Result Summary

- All test cases were passed in both functional and integration testing.
- Role-based access control was enforced successfully.
- Error handling worked as expected in scenarios of wrong inputs or unauthorized access.
- Real-time UI updates and client-server communication were verified using fetch APIs.
- Calculations in CGPA module and form submissions in attendance/marks modules operated correctly without delay

5.5 Discussion on Outcomes

The project successfully demonstrated:

- Smooth user experience due to React + Tailwind UI.
- Security and validation using hashing and session-based login.
- Scalability of architecture via modular Next.js + Prisma implementation.
- Strong data integrity in attendance, exams, and marks modules.

Limitations observed:

- Real-time push updates (e.g., using WebSockets) are not yet implemented.
- Admin bulk upload/import features can be considered in future.

Chapter 6

CONCLUSION

ScholarSuite successfully demonstrates how modern web technologies can be leveraged to streamline and enhance campus administration. The system meets its core objectives by automating workflows such as CGPA calculation, attendance tracking, exam management, and marks entry—all within a single unified platform. With a modular architecture built using Next.js, TypeScript, and Prisma, ScholarSuite ensures long-term maintainability, scalability, and ease of development.

System testing confirmed improved usability, data accuracy, and role-specific accessibility, offering a seamless experience for students, teachers, and administrators. By reducing manual processes and centralizing data, ScholarSuite enhances transparency, efficiency, and user engagement.

Despite its success, the system has some limitations—such as the absence of real-time notifications, limited mobile responsiveness, and lack of large-scale performance testing. However, these can be addressed in future iterations.

The platform holds strong potential for expansion. Future enhancements include features like Parent logins, fee and library modules, single sign-on (SSO), cloud deployment with CI/CD, and analytics dashboards for insights. Such developments can evolve ScholarSuite into a comprehensive ERP system for educational institutions.

In conclusion, ScholarSuite not only fulfills its goal of transforming administration but also sets the foundation for broader digital transformation in education. Its thoughtful design, modern stack, and extensibility make it a valuable tool in advancing institutional efficiency and excellence.

REFERENCES

[1] Pandey, V., Anand, A., & Racherla, S. (2022). Student Profile Management System. International Research Journal of Engineering and Technology (IRJET). Retrieved from IRJET

[2] Singh Jaryal, V., & Goel, S. (2022). Student Management System. Jaypee Academic of Information Technology. Retrieved from IRJIT

[3] Selwyn, N., Hillman, T., Rensfeldt, A. B., & Perrotta, C. (2021). Digital Technologies and the Automation of Education — Key Questions and Concerns. Postdigital Science and Education. Retrieved from

[4] P, S. (2022). College ERP System. GitHub. Retrieved from GitHub

[5] P, S. (2022). Student Management System. GitHub

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.

(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL
(Ministry of Education Initiative)

TECHNODEA - 2025

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

BALAJI , ISE

for participating in "TECHNODEA-2025", A National Level Project Exhibition and Competition under UG/PG/Hobby section held at SSIT, Tumakuru on 2nd & 3rd June, 2025 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. Manu S
Convenor-Technodea

Associate Sponsors:

Canara Bank · V3 NOVUS · POWER TUNE · BENAKA ENGINEERS

Dr. M S Raviprakasha
Principal, SSIT

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.

(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL
(Ministry of Education Initiative)

TECHNODEA - 2025

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

BRUNDA K , ISE

for participating in "TECHNODEA-2025", A National Level Project Exhibition and Competition under UG/PG/Hobby section held at SSIT, Tumakuru on 2nd & 3rd June, 2025 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. Manu S
Convenor-Technodea

Associate Sponsors:

केनरा बैंक Canara Bank · V3 NOVUS · POWER TUNE · BENAKA ENGINEERS

Dr. M S Raviprakasha
Principal, SSIT

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.

(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL (Ministry of Education Initiative)

TECHNODEA - 2025

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

KARAN R JOSHI , ISE

for participating in "TECHNODEA-2025", A National Level Project Exhibition and Competition under UG/PG/Hobby section held at SSIT, Tumakuru on 2nd & 3rd June, 2025 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. Manu S
Convenor-Technodea

Associate Sponsors:

केनरा बैंक Canara Bank · V3 NOVUS · POWER TUNE · BENAKA ENGINEERS

Dr. M S Raviprakasha
Principal, SSIT

SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY, MARALUR, TUMAKURU.

(A Constituent College of Sri Siddhartha Academy of Higher Education, Deemed to be University under section 3 of UGC act 1956, Approved by AICTE, Accredited by NBA and NAAC with A+ Grade)

INSTITUTION'S INNOVATION COUNCIL
(Ministry of Education Initiative)

TECHNODEA - 2025

CERTIFICATE

OF PARTICIPATION

THIS CERTIFICATE IS AWARDED TO

NITHYANANDA R S , ISE

for participating in "TECHNODEA-2025", A National Level Project Exhibition and Competition under UG/PG/Hobby section held at SSIT, Tumakuru on 2nd & 3rd June, 2025 organized by Science and Technology Entrepreneurs Park (STEP), SSIT.

Dr. Manu S
Convenor-Technodea

Associate Sponsors:

केनरा बैंक Canara Bank · V3 NOVUS · POWER TUNE · BENAKA ENGINEERS

Dr. M S Raviprakasha
Principal, SSIT