`

INTRODUCTION ABOUT THE PROJECT

In today’s rapidly evolving educational landscape, efficient administration and seamless communication have become critical for institutions aiming to provide a superior learning environment. The rise of digital education has transformed how academic data is managed, providing institutions with new tools but often leading to fragmented information silos. Despite the availability of various generic management tools and manual spreadsheet systems, many colleges struggle to find a centralized, user-friendly platform that integrates student enrollment, faculty management, and administrative communication into a single cohesive system. This is where the **College Management System** comes in.

This project bridges the gap between manual record-keeping and digital efficiency by offering a comprehensive, role-based dashboard for students, faculty, and administrators. Designed with a focus on data integrity, security, and usability, the platform provides a streamlined approach to managing campus operations. Administrators can oversee student admissions, manage course structures, and broadcast critical notices instantly. Faculty members can track their class schedules and manage academic records, while students gain direct access to their results, fee status, and campus updates. Unlike static websites, this system integrates proactive tools like automated GPA calculation and categorical notice boards to ensure timely information flow.

The goal of this project is to create a one-stop solution that empowers educational institutions to take control of their administrative workflow. By leveraging a robust database architecture and intuitive user interfaces, the platform not only provides a seamless data entry experience but also ensures that critical academic information is accessible, accurate, and secure.

As the necessity for digital campus management continues to rise, this system is built to meet these growing needs. Its responsive design, intuitive dashboards, and real-time data processing make it a powerful tool for users ranging from students checking exam schedules to administrators generating complex compliance reports.

`

From a technical perspective, the platform’s development is rooted in modern web development principles, utilizing **HTML5, Jinja2 templates, Bootstrap, and JavaScript** to create a responsive and interactive frontend. The application is powered by a robust backend built with **Python 3.x and the Flask framework**, employing the Application Factory pattern to ensure scalability and modularity. The data layer is managed using **MySQL with SQLAlchemy ORM**, ensuring secure and efficient relational data handling. The project follows a modular development approach, utilizing Flask Blueprints to organize routing and logic, ensuring maintainability and ease of future enhancements.

By combining ease of use, performance, and administrative depth, the College Management System seeks to stand out in the domain of educational software. Whether an institution is looking to reduce paperwork, streamline communication, or digitize academic records, this platform offers the tools and resources needed to succeed in today’s modern educational environment.

`

OBJECTIVE AND SCOPE OF PROJECT

The **College Management System** has the following objectives:

- To provide a centralized mechanism for managing **student and faculty profiles**, ensuring accurate and accessible records of academic and personal details.
- To digitize the **Notice Board** system with advanced categorization (Exam, Event, Academic) and priority pinning, ensuring critical announcements reach the right audience instantly.
- To streamline the **academic workflow** by managing Departments, Courses, and Sections effectively, allowing for structured curriculum planning.
- To enable efficient **Examination Management** by automating the recording of marks and the calculation of GPAs and aggregates, reducing manual errors.
- To provide a secure and user-friendly platform with **Role-Based Access Control (RBAC)**, ensuring that students, faculty, and admins can only access data relevant to their roles.
- To enhance administrative transparency by digitizing **Fee Structures and Payments**, allowing students to view their payment history and status remotely.

The scope for developing the **College Management System** includes:

- Managing **user authentication and session security**, including secure registration, login, and distinct dashboards for Admin, Faculty, and Student roles.
- Developing a **responsive user interface** using Bootstrap that adapts seamlessly to desktops, tablets, and smartphones for on-the-go access.
- Implementing **dynamic data visualization** and interactive elements, such as calendar-based schedules and real-time notice updates.
- Integrating **backend logic** for complex academic relationships, such as linking students to specific sections and faculty to specific courses.

`

- Creating a robust backend architecture using **Python (Flask) and MySQL** to handle data persistence, complex relational queries, and scalable application growth.

`

INTRODUCTION OF HTML AND CSS

HTML (Hypertext Mark-up Language) and CSS (Cascading Style Sheets) are essential technologies that form the foundation of web development, enabling the creation of visually engaging and well-structured web pages. Much like Visual Basic in application development, HTML and CSS are integral in defining the structure and presentation of web content.

HTML (Hypertext Mark-up Language):

HTML serves as the fundamental framework of web pages, providing a structured mark-up language to define content and layout. Developers use HTML to create documents by utilizing tags that categorize and organize different elements on a webpage, such as headings, paragraphs, images, links, forms, and more.

CSS (Cascading Style Sheets):

CSS complements HTML by allowing developers to manage the visual styling of web pages. It introduces style rules that dictate how HTML elements should be displayed across various devices. By separating content from design, CSS streamlines the creation of consistent and visually appealing user interfaces.

Together, HTML and CSS provide the essential building blocks for crafting an engaging and user-friendly web experience. While HTML defines the structure, CSS enhances the

`

design, offering a flexible and responsive layout. This synergy supports the principles of Rapid Application Development (RAD), simplifying the process of designing and styling web interfaces efficiently.

`

INTRODUCTION OF JAVASCRIPT

JavaScript is a versatile and powerful programming language primarily employed in web development. It is a key component of modern web browsers, empowering developers to create dynamic and interactive user interfaces. Often abbreviated as JS, JavaScript plays a pivotal role in enhancing web page functionality through client-side scripting.

Key Features of JavaScript:

Client-Side Scripting:

JavaScript is predominantly used for client-side scripting, meaning it executes directly within the user's web browser. This capability enables developers to craft dynamic content, validate forms, handle events, and manipulate the Document Object Model (DOM), allowing webpage content to be updated dynamically without the need for a page reload.

Object-Oriented:

JavaScript is an object-oriented language, supporting core concepts such as encapsulation, inheritance, and polymorphism. These features allow developers to organize and modularize their code effectively, facilitating better code management and scalability.

`

Interactivity and Dynamic Content:

JavaScript enhances web pages by enabling real-time responsiveness to user interactions. Developers can implement features like sliders, pop-ups, and form validations, while also facilitating dynamic content loading and updating, contributing to more engaging and user-friendly websites.

Event-Driven Programming:

JavaScript follows an event-driven programming model, wherein developers define functions that execute in response to specific events such as button clicks, mouseovers, or form submissions. This paradigm improves user experience by making web pages more interactive and responsive to user actions.

Cross-Browser Compatibility:

JavaScript is supported by all major web browsers, including Chrome, Firefox, Safari, and Edge, ensuring consistent functionality across diverse platforms and devices.

JavaScript is an indispensable technology in modern web development, frequently working alongside HTML and CSS to build interactive web applications. Over time, JavaScript has expanded beyond its initial use in web browsers, now also playing a significant role in server-side development (via Node.js), mobile app development, and other application domains.

`

INTRODUCTION OF PYTHON AND FLASK FRAMEWORK

Python is a high-level, interpreted, general-purpose programming language known for its emphasis on code readability and simplicity. Created by Guido van Rossum and first released in 1991, Python has grown to become one of the most popular programming languages in the world, powering everything from simple scripts to complex machine learning algorithms.

For this project, we utilize Flask, a micro web framework written in Python. Created by Armin Ronacher in 2010, Flask is classified as a "microframework" because it does not require particular tools or libraries to function. It has no default database abstraction layer, form validation, or other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.

Python code in a web context is executed via a WSGI (Web Server Gateway Interface) server. In our architecture, the Flask application factory (create_app) initializes the application, managing routes via Blueprints and handling database interactions through SQLAlchemy. Unlike older CGI scripts, this modern stack ensures that the application remains persistent and efficient in handling concurrent user requests.

While Python is often associated with data science and automation, its web capabilities are robust. Major platforms like Netflix, Uber, and Pinterest utilize Python for its scalability and ease of integration. The standard Python interpreter (CPython) is open-source and managed by the Python Software Foundation.

According to the TIOBE Index and GitHub Octoverse reports, Python consistently ranks as one of the top programming languages globally. Its ecosystem allows for the seamless integration of mathematical calculations (essential for our Compound Interest module) and

`

data processing, making it the ideal choice for a financial dashboard.

KEY FEATURES

- Microframework Architecture (Performance): Flask is lightweight and modular. Unlike monolithic frameworks that force a specific directory structure and dependencies, Flask allows the developer to keep the core application simple and extensible. This results in faster startup times and a codebase that is easier to navigate and maintain.
- Open Source: Both Python and Flask are open-source technologies with permissive licenses (BSD/MIT). This means the entire stack—from the interpreter to the web framework and ORM—is free to use, modify, and distribute, significantly reducing the development cost of the project.
- Readable Syntax: Python is often described as "executable pseudocode." Its syntax is clear, concise, and English-like, which reduces the cognitive load on developers. This readability ensures that the complex logic behind expense tracking and interest calculations remains understandable and maintainable.
- Jinja2 Templating (Embedded Logic): Flask utilizes the Jinja2 templating engine, which allows Python-like logic to be embedded within HTML files. This enables the server to render dynamic content—such as user-specific dashboards and expense tables—securely and efficiently before sending the final HTML to the client's browser.
- Platform Independence: Python is cross-platform by design. The Personal Finance Management System can be developed on Windows and deployed on Linux (e.g., Ubuntu with Nginx/Gunicorn) without requiring code changes. This portability ensures flexibility in hosting and deployment options.

- Database Support (SQLAlchemy): This project uses Flask-SQLAlchemy, an Object Relational Mapper (ORM) that provides a high-level abstraction for database interactions. It supports a wide range of databases, including the MySQL database used in our production environment. The ORM allows us to interact with the database using Python classes and objects rather than raw SQL queries, improving security and maintainability.
- Robust Error Reporting: Flask comes with a built-in debugger and logger. During development, the Werkzeug debugger provides interactive tracebacks in the browser if a crash occurs, allowing for rapid identification and resolution of bugs within the expense logic or routing.
- Dynamic Typing: Python is dynamically typed, meaning variable types are determined at runtime. This allows for rapid prototyping and flexibility when handling various data inputs, such as different currency formats or user profile details, without the strict boilerplate code required by statically typed languages.
- Security: Security is a core component of the Flask ecosystem. Our project leverages:
- Werkzeug: For secure password hashing (PBKDF2/Bcrypt).
- Flask-Login: For managing user sessions securely.
- Jinja2 Auto-escaping: To automatically protect against Cross-Site Scripting (XSS) attacks by escaping untrusted input in templates.
- Extensibility (Control): Flask gives developers granular control over the application. Features are added only when needed. For instance, we explicitly added Flask-Migrate for database schema management and Flask-Login for authentication. This "opt-in" approach prevents bloat and ensures the application remains optimized for its specific financial tracking purpose.
- Thriving Community: The Python and Flask communities are vast and active. Extensive documentation, third-party extensions (like Flask-WTF for forms), and

`

community support ensure that any technical challenge encountered during the development of the Personal Finance Management System can be resolved quickly.

`

INTRODUCTION OF MYSQL

What is a Database?

A database is a specialized application designed to store, manage, and organize data in a structured manner. It provides distinct APIs that allow users to create, access, manage, search, and replicate the data it contains. While other types of data storage systems, such as file systems or large hash tables in memory, can be used, they generally do not offer the same speed and ease of data retrieval and manipulation as a dedicated database system.

In modern applications, relational database management systems (RDBMS) are commonly used to handle large volumes of data. The term "relational" refers to the way data is organized into tables, where relationships between the data are established using primary keys and foreign keys.

Relational Database Management System (RDBMS)

An RDBMS is software that facilitates the implementation and management of relational databases. It enables the creation of databases with tables, columns, and indexes while ensuring the following functions:

- **Table Structure:** Allows the creation and organization of tables to store data.
- **Referential Integrity:** Maintains the integrity of relationships between tables using primary and foreign keys.

Index Management: Automatically updates indexes to optimize data retrieval.

- **SQL Query Processing:** Interprets SQL queries and retrieves data by combining information from various tables.

RDBMS Terminology

Before exploring specific database systems like MySQL, it is important to understand some key terminology used in RDBMS:

- **Database:** A collection of related tables that store data.
- **Table:** A table is a collection of rows and columns, representing data in a structured format similar to a spreadsheet.
- **Column:** A column contains data of a specific type or category, such as customer names, addresses, or product prices.
- **Row:** A row (also known as a tuple, entry, or record) is a set of related data within a table, often representing an individual record.
- **Redundancy:** The practice of storing data multiple times to improve system performance. While redundancy can speed up access, it may introduce challenges in data consistency.
- **Primary Key:** A primary key is a unique identifier for each record in a table. It ensures that each row in a table is distinguishable from others, and no two rows can have the same primary key value.
- **Foreign Key:** A foreign key is used to create a relationship between two tables. It refers to the primary key of another table, ensuring referential integrity between the two tables.

`

- **Compound Key (Composite Key):** A compound key is a primary key that consists of two or more columns, used when a single column cannot uniquely identify records in a table.

- **Index:** An index in a database is a data structure that allows for faster retrieval of data, similar to an index at the back of a book. It speeds up query execution by reducing the amount of data that needs to be scanned.

- **Referential Integrity:** Referential integrity ensures that foreign key values in a table always point to valid, existing rows in another table, preventing orphaned records and maintaining data consistency.

What is SQL?

SQL (Structured Query Language) is the standard language used for managing and manipulating data in a relational database management system (RDBMS). It provides an interface for accessing, updating, deleting, and managing database structures and data. SQL is the foundation of interacting with databases, enabling users to perform a wide range of operations on the data they store.

SQL is divided into four main subsets:

1. **DDL (Data Definition Language):** DDL is used to define and manage the structure of a database. It allows the creation, alteration, and deletion of database objects such as tables, views, and schemas. Key commands include:

    - **CREATE**: To create new database objects.

`

   - **ALTER**: To modify existing database objects.

   - **DROP**: To delete database objects.

2. **DML (Data Manipulation Language):** DML deals with the manipulation of data within the database. It allows for adding, updating, deleting, and querying data. Common DML commands include:

   - **SELECT**: To retrieve data from one or more tables.

   - **INSERT**: To add new data into a table.

   - **UPDATE**: To modify existing data.

   - **DELETE**: To remove data from a table.

3. **DCL (Data Control Language):** DCL is used to control access to data within the database. It helps manage user permissions, allowing for secure access and data management. Main DCL commands include:

   - **GRANT**: To give a user specific access rights to database objects.

   - **REVOKE**: To remove previously granted permissions.

4. **TCL (Transaction Control Language):** TCL is used to manage transactions in a database. A transaction is a sequence of operations performed as a single unit. TCL ensures that transactions are handled properly, preserving data integrity. Key TCL commands include:

   - **COMMIT**: To save all changes made during the transaction.

`

- **ROLLBACK**: To undo changes made during the transaction if an error occurs.
- **SAVEPOINT**: To set a point within a transaction to which you can roll back.
- **SET TRANSACTION**: To configure transaction properties.

MySQL

MySQL is a widely used relational database management system (RDBMS) developed and maintained by MySQL AB, a Swedish company. It is known for being fast, reliable, and easy to use, making it a popular choice for both small businesses and large enterprises. MySQL has several advantages that contribute to its growing popularity:

- **Open Source:** MySQL is released under an open-source license, meaning it is free to use and modify.

- **Powerful:** Despite being free, MySQL provides a range of features that rival some of the most expensive commercial database management systems. It can handle substantial databases and a variety of data processing tasks.

- **Standardized SQL Support:** MySQL uses a standard form of SQL, making it compatible with most database applications.

- **Cross-Platform Compatibility:** MySQL can run on various operating systems, including Windows, Linux, and macOS, and supports multiple programming languages like PHP, PERL, C, C++, and Java.

`

- **Speed:** MySQL is known for its quick performance, especially with large datasets. It is optimized for high-speed operations.

- **Scalability:** MySQL can handle large databases with ease. It supports up to 50 million rows per table, with the theoretical file size limit reaching 8 million terabytes.

- **Customization:** Being open source, MySQL can be customized and extended to fit specific needs. Developers can modify the software to suit their particular environment or requirements.

`

DEFINITION OF PROBLEM FOR COLLEGE MANAGEMENT SYSTEM

The College Management System faces several challenges related to its functionality, user accessibility, and backend operations. These problems span across multiple areas such as data redundancy, communication latency, security, and administrative efficiency, all of which impact the platform's overall effectiveness for an educational institution. Below are the primary problems:

1. **User Experience and Accessibility:**
   - Ensuring intuitive navigation for users with varying levels of technical proficiency (e.g., senior faculty, parents, or new students).
   - Addressing cluttered interface elements where academic data (timetables, fee structures) might overwhelm users, preventing them from finding critical information quickly.
2. **Data Visualization and Academic Analysis:**
   - Ensuring accurate, real-time rendering of performance charts (e.g., student attendance trends, semester-wise GPA growth) based on faculty input.
   - Optimizing filtering mechanisms to allow administrators to easily refine student lists by department, semester, or admission year.
3. **Data Entry and Integrity:**
   - Minimizing human error during manual data entry (e.g., entering incorrect marks or attendance status) to avoid skewed academic records.
   - Implementing a streamlined process for faculty to edit or correct entered marks without violating data consistency rules (e.g., changing marks after results are published).
4. **Database Management and Performance:**
   - Ensuring that the database can handle a growing volume of student records, historical results, and alumni data without compromising performance.

   - Optimizing SQL queries to avoid slowdowns during peak usage times, such as result declaration days or admission seasons.

5. **Security and Privacy:**
   - Protecting highly sensitive data like student contact details, academic grades, and fee payment history.
   - Ensuring secure session management to prevent unauthorized access, such as a student accessing faculty privileges or modifying their own grades.
6. **Calculation Accuracy:**
   - Addressing potential logic errors in academic calculations (e.g., GPA/CGPA aggregation, attendance percentage) to ensure students receive precise academic reports.
   - Ensuring the system handles diverse grading scales and credit systems accurately across different departments.
7. **Notification and Communication Reliability:**
   - Streamlining the "Digital Notice Board" to ensure students are alerted about upcoming exams, fee deadlines, or holiday announcements effectively.
   - Addressing issues where critical updates might be missed if the user does not log in frequently, highlighting the need for a persistent dashboard notification system.
8. **Mobile Compatibility:**
   - Ensuring a seamless mobile experience for students accessing their results or schedules via smartphones.
   - Adapting complex data tables (like mark sheets) to fit smaller screens without losing readability or formatting.
9. **System Scalability:**
   - Minimizing latency during concurrent operations, such as hundreds of students checking results simultaneously.

`

- Optimizing the Flask backend to handle high traffic loads during specific academic events without blocking requests.

10. **User Education and Onboarding:**
    - Providing clear guidance on how to navigate the system, such as how to apply for leave online or how to interpret the fee breakdown.
    - Reducing the learning curve for non-technical staff to ensure they can utilize the administrative modules immediately.

In sum, the **College Management System** needs to address the above problems in a systematic and comprehensive manner to provide a smooth, secure, and effective educational administration platform. These challenges require a balanced approach to enhance user experience, streamline academic operations, and ensure the scalability of the platform.

`

SYSTEM ANALYSIS

System analysis for the **College Management System** involves understanding the complex requirements of an educational institution comprising students, faculty, and administration. The system must provide a seamless user experience while ensuring data security and efficient management of academic records. The system analysis phase includes gathering requirements through analyzing common administrative bottlenecks, studying existing manual processes, and identifying key functionalities that will digitize the campus workflow. Here is the System Analysis for the College Management System in points:

1. **Problem Identification:**
    - Educational institutions often lack a centralized view of academic data, leading to fragmented records across departments (Library, Accounts, Exam Cell).
    - Manual record-keeping fails to provide real-time insights or quick retrieval of student history, leaving administration reactive rather than proactive.
2. **User Requirements:**
    - **Students:** Need a platform that provides quick access to results, fee status, attendance reports, and campus notices.
    - **Faculty:** Require tools to manage class attendance, upload marks, view student lists, and communicate with students.
    - **Administrators:** Require full control for managing user accounts, defining course structures, setting fee parameters, and monitoring overall system health.
3. **System Requirements:**
    - **Functional:**
        - **User Authentication:** Secure Registration/Login with Role-Based Access Control (Admin, Faculty, Student).

`

    - **Academic Management:** CRUD (Create, Read, Update, Delete) functionalities for Courses, Sections, and Departments.
    - **Examination Module:** Automated GPA calculation and Result generation.
    - **Interactive Dashboard:** Visualizing attendance and performance trends.
    - **Notice Board:** A digital CMS for posting and categorizing announcements.
  - **Non-Functional:**
    - **Scalability:** Ability to handle increasing amounts of student data year over year.
    - **Security:** High security to protect academic integrity and personal data using hashing and session protections.
    - **Performance:** Optimization to ensure instant feedback when uploading bulk data or searching for student records.

4. **System Architecture:**
   - The system is based on the **Model-View-Template (MVT)** architecture (standard for Flask), separating the business logic from the user interface for better maintainability.
   - The **Frontend** is developed using **HTML5, Jinja2 Templates, Bootstrap, and JavaScript**, providing a responsive and interactive experience.
   - The **Backend** is developed using **Python (Flask) and MySQL (via SQLAlchemy)**, enabling secure data storage, complex querying, and reliable logic execution.
5. **Data Flow:**
   - The system manages user input (student enrollment, marks entry, notice creation) through secure forms.
   - Flask routes process this data, validate it against academic rules, and interact with the MySQL database via SQLAlchemy models.
   - Dynamic responses update the dashboard to reflect changes (e.g., updated fee status) immediately.
6. **User Interface and Experience:**

`

   - The system emphasizes ease of use, with a role-specific dashboard design, sidebar navigation, and minimal steps to perform core tasks (like checking a result).
   - **Bootstrap** ensures the interface is uniform and accessible across different devices.

7. **Security Considerations:**
   - The system protects user data with **Werkzeug** security helpers for password hashing.
   - **Flask-Login** manages user sessions to ensure that a student cannot access faculty modules and vice versa.
   - Input validation is implemented at both the client-side and server-side to prevent SQL injection or data corruption.

`

SYSTEM REQUIREMENTS

Minimum Hardware Requirements for Development

- **CPU:** Intel Core i3 (10th Gen) or AMD Ryzen 3 equivalent (Modern multi-core processors are recommended for running the Python interpreter and database services simultaneously).
- **Memory (RAM):** 8 GB (4 GB is the absolute minimum, but 8 GB is recommended to run Visual Studio Code, a MySQL server, and a web browser smoothly).
- **Hard Disk:** 256 GB SSD (Solid State Drive is highly recommended over HDD for faster project indexing and local server startup).
- **System Type:** 64-bit Operating System (Required for modern Python versions and efficient memory management).

Minimum Software Requirements for Development

- **Operating System:**
    - **Windows:** Windows 10 or 11 (64-bit).
    - **Linux:** Ubuntu 20.04 LTS or later (Preferred for deployment simulation).
    - **macOS:** macOS Catalina or later.
- **Database:** MySQL 8.0 Community Server (or MariaDB equivalent).
- **Web Server:**
    - **Development:** Werkzeug (Built-in Flask development server).
    - **Production Readiness:** Gunicorn (WSGI Server) behind Nginx.
- **Programming Language:** Python 3.10 or later.
- **Frontend Technologies:**
    - HTML5, CSS3 (Bootstrap 5 Framework).
    - JavaScript (ES6 Modules, Chart.js for visualization, FullCalendar.js).

`

- **IDE / Text Editor:** Visual Studio Code (with Python Extension) or PyCharm Community Edition.
- **Version Control:** Git (for tracking changes).
- **Browser:** Google Chrome or Mozilla Firefox (latest versions with Developer Tools).

Minimum Hardware and Software Requirements for Running the Website

On PCs (Desktops and Laptops)

- **CPU:** Intel Core i3 / AMD Ryzen 3 or equivalent (Sufficient for processing client-side JavaScript charts).
- **Memory (RAM):** 4 GB or higher (Required to handle the DOM manipulation for the dashboard and charts).
- **Hard Disk:** Not applicable (Client-side storage relies on browser cache/cookies; the application is hosted on the server).
- **Operating System:**
  - Windows: Windows 10 or later.
  - macOS: macOS Big Sur or later.
  - Linux: Any modern distribution with a GUI.
- **Web Browser:** Modern browsers supporting ES6 JavaScript and Canvas API (e.g., Google Chrome, Mozilla Firefox, Microsoft Edge, Safari). **Internet Explorer is not supported.**

On Mobile Devices (Phones and Tablets)

- **CPU:** Quad-core processor (e.g., Snapdragon 600 series, Apple A12, or equivalent) for smooth chart rendering.
- **Memory (RAM):** 3 GB or higher (To ensure the dashboard remains responsive without reloading tabs).
- **Operating System:**

`

  - **Android:** Android 10.0 or later.
  - **iOS:** iOS 14 or later.
- **Browser:**
  - **Android:** Google Chrome (latest version).
  - **iOS:** Safari (latest version).
- **Screen Resolution:** 720x1280 pixels or higher (The Bootstrap grid system will adapt the layout, but HD resolution is recommended for reading financial tables).
- **Internet Connection:** 4G/5G or stable Wi-Fi (Required for fetching real-time data via AJAX and syncing expenses to the cloud database).

`

SYSTEM DESIGN AND CODING

INTRODUCTION: Software design is a critical phase in the software engineering process, laying the foundation for all subsequent development activities. It transcends development paradigms and application areas, serving as a common thread in the creation of every engineered system. For the **College Management System**, the design phase was initiated after the academic and administrative requirements were specified and analyzed. This step formed the core of the development phase, bridging the gap between the initial concept of "digital campus management" and the final **Python/Flask** implementation.

The central focus of this design is **data integrity and usability**. Through the design process, the quality of the final dashboard is ensured by translating user requirements—such as accurate GPA calculations, secure mark entry, and real-time notice broadcasting—into a structured framework. A robust design minimizes the risk of building unstable systems, ensuring that critical academic data is handled securely and that the system remains maintainable for future feature additions like alumni portals or library integration.

In this phase, detailed representations of the **MySQL database schema** (Users, Students, Courses, Results), **Flask Blueprint structures**, and **frontend interactive flows** were refined, reviewed, and documented. From both technical and project management perspectives, system design plays a pivotal role in guiding the engineering process toward a successful, bug-free launch.

INPUT DESIGN: Input design is the process of converting user-oriented input into a computer-readable format. As a key component of the overall system design, input design for an educational application must be executed with meticulous attention to detail. This is because the collection of input data—specifically **student records, examination marks, and attendance logs**—represents the most resource-intensive aspect of the system. Effective input design focuses on optimizing the methods of input collection while ensuring the highest possible level of accuracy to prevent academic discrepancies.

The primary objectives of input design for this project are as follows:

`

1. **Operational Efficiency:** Create an input process that is efficient, using intuitive forms that require minimal effort for faculty to upload marks or attendance.
2. **Accuracy:** Achieve the highest possible accuracy in input data (e.g., preventing marks greater than the maximum limit) to minimize errors in academic reports.
3. **User-Friendliness:** Ensure that the input process is intuitive, utilizing dropdowns for Course/Semester selection and date pickers for DOB to facilitate seamless interaction.

Input Data Considerations: When designing input data for the College Management System, the goal is to create a process that is easy, logical, and error-free. The data entry forms (managed via **HTML5 and Jinja2**) are designed so that users are clear on the required fields. For example, the "Add Student" form explicitly labels fields for **Name, Roll Number, Department, and Semester** to avoid ambiguity.

The input process involves the following stages in our context:

- **Data Recording:** Capturing raw academic data (e.g., "85" for "Data Structures") from faculty via web forms.
- **Data Validation:** Checking the accuracy of input data against predefined criteria (e.g., ensuring the Roll Number format is valid or that attendance dates are within the academic term).
- **Data Conversion:** Converting string inputs from forms into Python storage formats (e.g., integers for marks, datetime for admission dates).
- **Data Transmission:** Securely transmitting data from the client browser to the **Flask backend** using POST requests protected by CSRF tokens.

Input Types in the Project: A fundamental aspect of our input design is selecting data capture methods that reduce errors.

- **External Input:** Data sourced directly from the user, such as **Student Registration details, Faculty Mark Entries, and Notice Content**.

`

- **Internal Input:** Data generated by the system, such as `enrollment_date` timestamps or auto-generated user IDs.
- **Interactive Input:** Input provided by users in real-time, such as selecting a "Semester" from a dropdown to filter the student list immediately.

OUTPUT DESIGN: Output design refers to the process of determining how the results of processing will be communicated to the users. In an educational application, outputs play a crucial role in conveying meaningful insights, transforming raw data into visual trends for decision-making. Designing effective output involves organizing the presentation of information in a manner that is clear, actionable, and accessible.

The outputs of the **College Management System** are categorized as follows:

1. **External Outputs:** Information presented to the user on the dashboard, such as the **"Semester GPA"**, **"Attendance Percentage"**, or **"Active Notices"** list.
2. **Internal Outputs:** Logs generated within the system for debugging, such as database connection errors or failed login attempts.
3. **Interactive Outputs:** Dynamic charts (via **Chart.js**) that respond to user queries, such as visualizing "Class Performance" or "Pass/Fail Ratio" for a specific subject.
4. **Turnaround Outputs:** Outputs used for further processing, such as the **"Total Aggregate"** calculated by the system, which helps administration decide on student promotion eligibility.

The key to designing effective outputs is ensuring that the right academic metrics are presented in the right format. For instance, **attendance trends** are best shown as a Line Graph, while **subject-wise performance** is best shown as a Bar Chart. Outputs are structured to provide clear communication that facilitates academic planning.

`

The design of screens and interfaces plays a critical role. We utilize a **Responsive Dashboard Layout** (Sidebar + Main Content) to make outputs not only informative but also interactive. Users can navigate through tabs, request specific academic terms, and fulfill their needs by querying their academic history effectively.

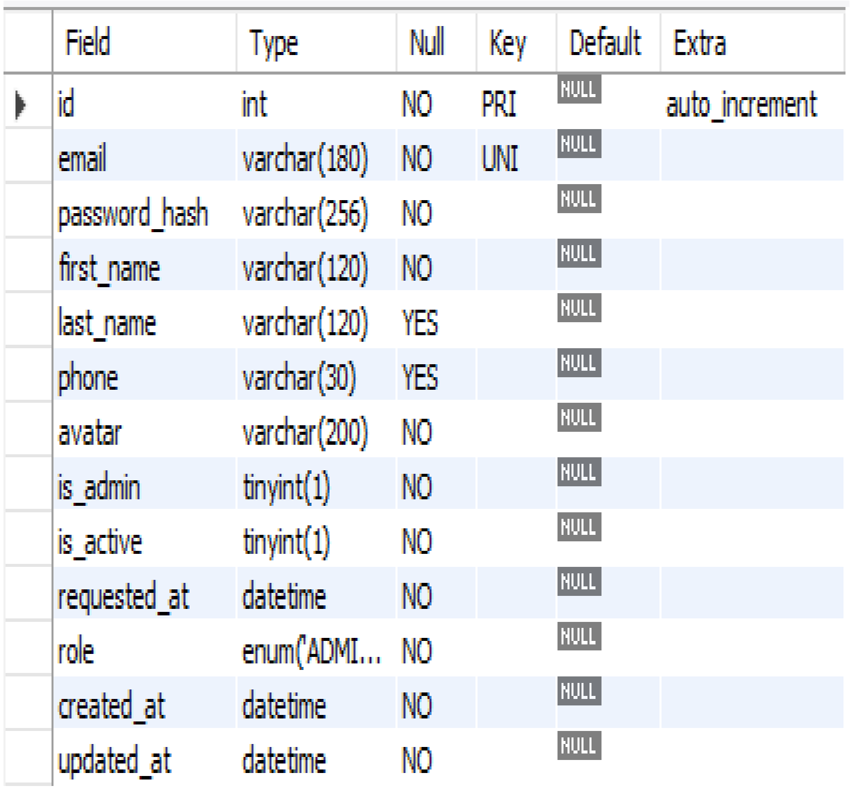
CONCLUSION: In the software engineering process, system design—including both input and output design—is foundational for building quality, reliable, and user-friendly educational tools. Input design ensures that sensitive academic data is captured accurately and efficiently, while output design guarantees that students and faculty receive the analytical insights they need in a well-structured, visual manner.

These steps are critical for the success of the **College Management System**, as they directly impact both the functionality (result accuracy) and usability (ease of administration) of the final product. By focusing on these design principles, we ensure that the system meets institutional needs, operates efficiently on the **Flask/MySQL stack**, and remains scalable for future enhancements.

`

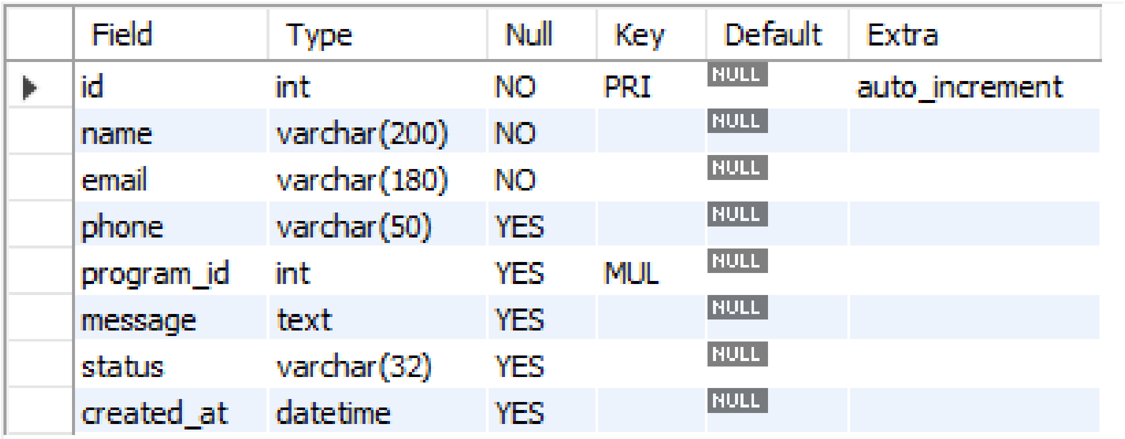
DATABASE DESIGN

TABLE 1: Users



Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
email	varchar(180)	NO	UNI	NULL	
password_hash	varchar(256)	NO		NULL	
first_name	varchar(120)	NO		NULL	
last_name	varchar(120)	YES		NULL	
phone	varchar(30)	YES		NULL	
avatar	varchar(200)	NO		NULL	
is_admin	tinyint(1)	NO		NULL	
is_active	tinyint(1)	NO		NULL	
requested_at	datetime	NO		NULL	
role	enum('ADMI...	NO		NULL	
created_at	datetime	NO		NULL	
updated_at	datetime	NO		NULL	

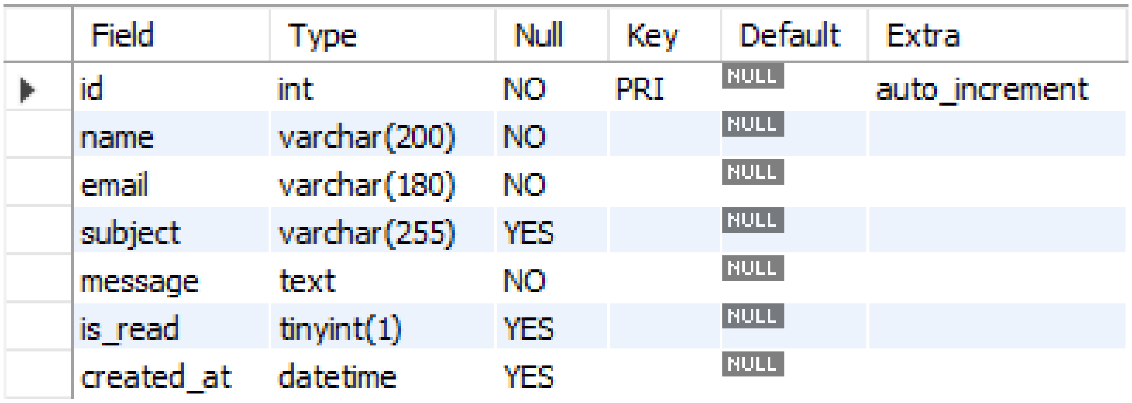
TABLE 2:Applications:



Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
name	varchar(200)	NO		NULL	
email	varchar(180)	NO		NULL	
phone	varchar(50)	YES		NULL	
program_id	int	YES	MUL	NULL	
message	text	YES		NULL	
status	varchar(32)	YES		NULL	
created_at	datetime	YES		NULL	

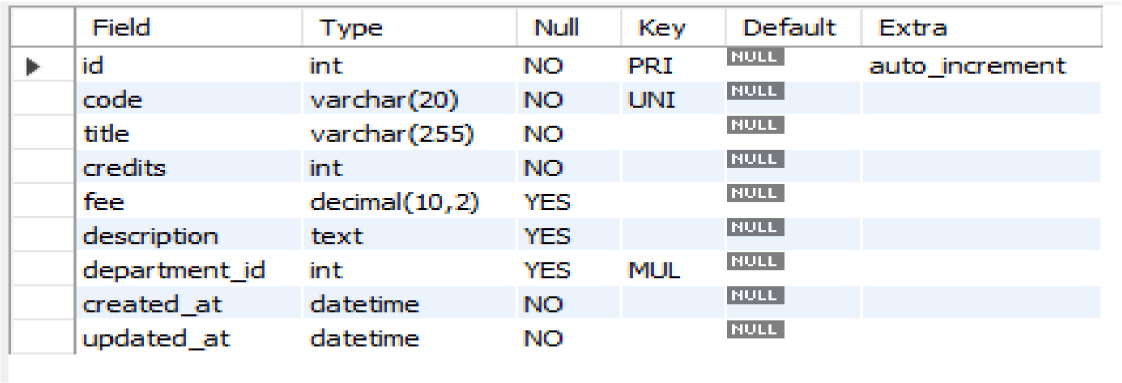
`

TABLE 3: Contact_Messages:



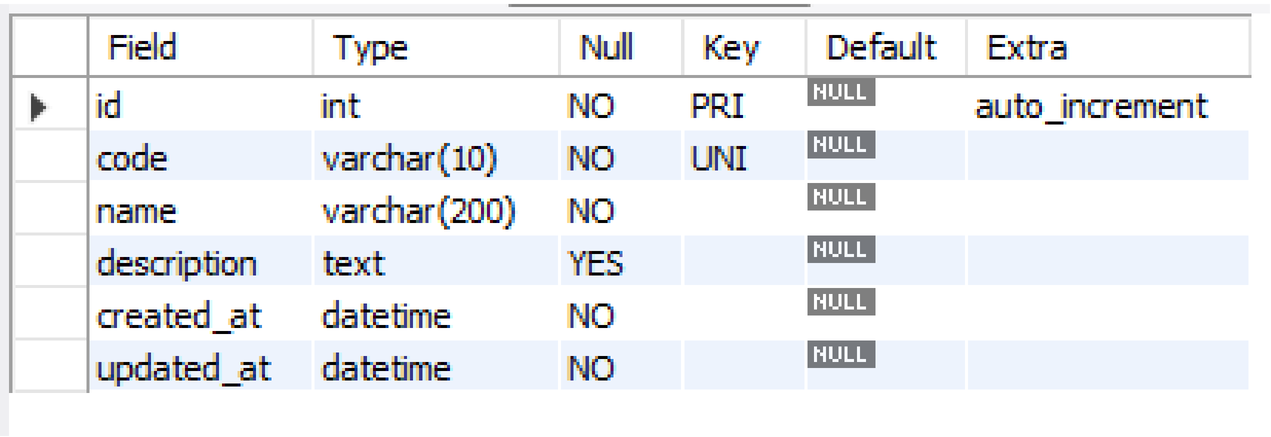
	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	name	varchar(200)	NO		NULL	
	email	varchar(180)	NO		NULL	
	subject	varchar(255)	YES		NULL	
	message	text	NO		NULL	
	is_read	tinyint(1)	YES		NULL	
	created_at	datetime	YES		NULL	

TABLE 4:Courses:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	code	varchar(20)	NO	UNI	NULL	
	title	varchar(255)	NO		NULL	
	credits	int	NO		NULL	
	fee	decimal(10,2)	YES		NULL	
	description	text	YES		NULL	
	department_id	int	YES	MUL	NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

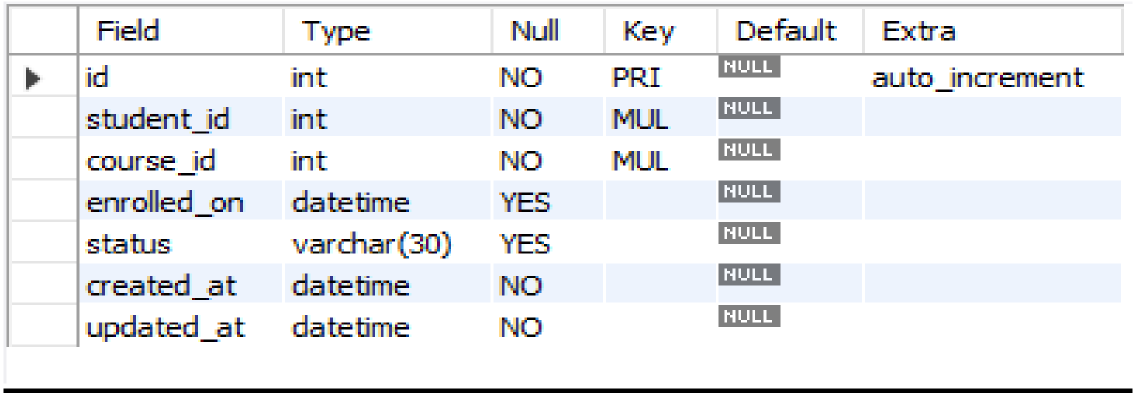
TABLE 6: Departments:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	code	varchar(10)	NO	UNI	NULL	
	name	varchar(200)	NO		NULL	
	description	text	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

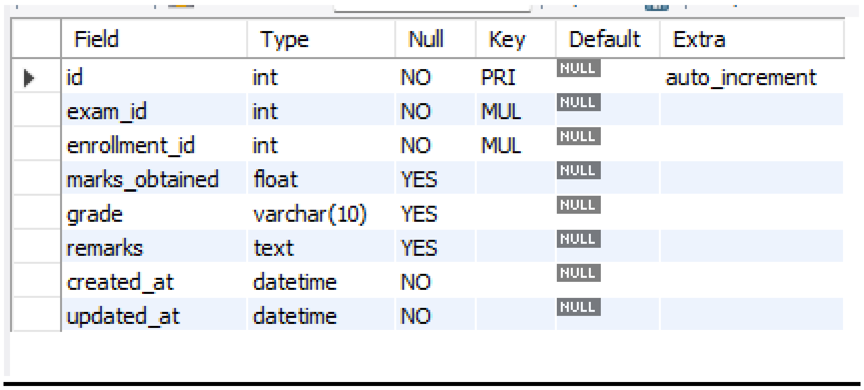
`

TABLE 7:Enrollments”



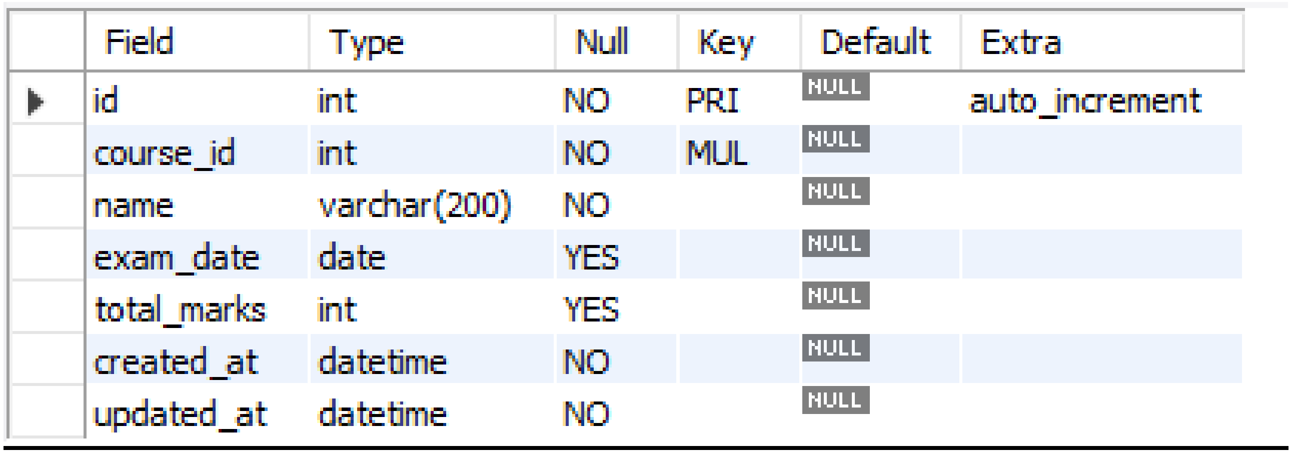
	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	student_id	int	NO	MUL	NULL	
	course_id	int	NO	MUL	NULL	
	enrolled_on	datetime	YES		NULL	
	status	varchar(30)	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

TABLE 8: Exam_Results:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	exam_id	int	NO	MUL	NULL	
	enrollment_id	int	NO	MUL	NULL	
	marks_obtained	float	YES		NULL	
	grade	varchar(10)	YES		NULL	
	remarks	text	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

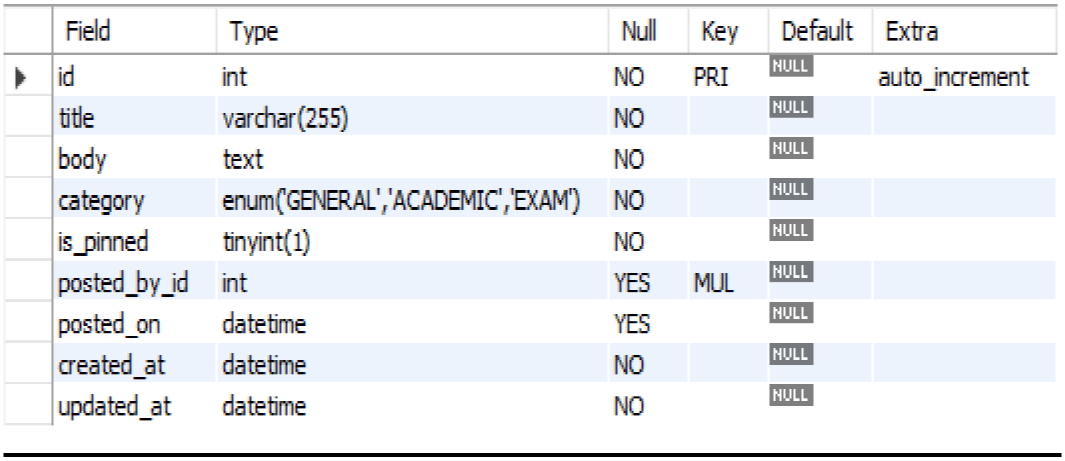
TABLE 9:Exams:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	course_id	int	NO	MUL	NULL	
	name	varchar(200)	NO		NULL	
	exam_date	date	YES		NULL	
	total_marks	int	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

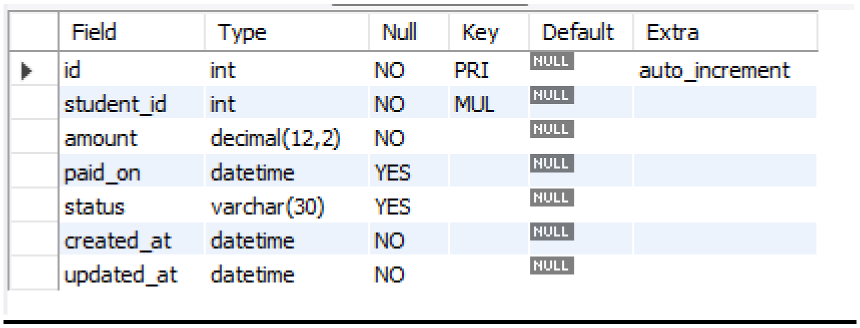
`

TABLE 10:Notices:



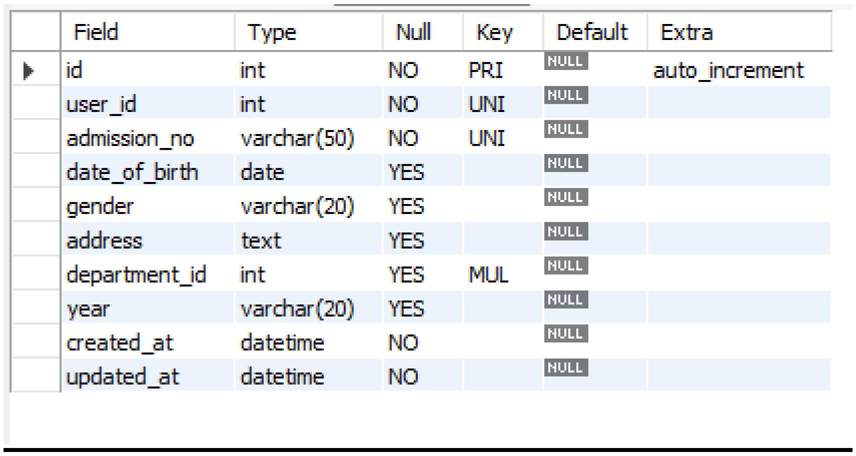
	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	title	varchar(255)	NO		NULL	
	body	text	NO		NULL	
	category	enum('GENERAL','ACADEMIC','EXAM')	NO		NULL	
	is_pinned	tinyint(1)	NO		NULL	
	posted_by_id	int	YES	MUL	NULL	
	posted_on	datetime	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

TABLE 11:Payment:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	student_id	int	NO	MUL	NULL	
	amount	decimal(12,2)	NO		NULL	
	paid_on	datetime	YES		NULL	
	status	varchar(30)	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

TABLE 12: Student Profile:



	Field	Type	Null	Key	Default	Extra
▶	id	int	NO	PRI	NULL	auto_increment
	user_id	int	NO	UNI	NULL	
	admission_no	varchar(50)	NO	UNI	NULL	
	date_of_birth	date	YES		NULL	
	gender	varchar(20)	YES		NULL	
	address	text	YES		NULL	
	department_id	int	YES	MUL	NULL	
	year	varchar(20)	YES		NULL	
	created_at	datetime	NO		NULL	
	updated_at	datetime	NO		NULL	

`

DATA FLOW DIAGRAM

It is a graphical tool used to describe and analyze the flow of data through a system. It focuses on the data flowing into the system, between processes, and in and out of data stores.

DFDs are of two types:

1. **Physical DFD:** The physical DFD is a model of the current system and is used to ensure that the current system has been clearly understood. It depicts *how* the system will be implemented (e.g., hardware, software, people).
2. **Logical DFD:** Logical DFDs are the model of the proposed system. They clearly show the requirements on which the new system should be built, focusing on *what* the system does rather than *how* it does it.

Notation Used In DFD:

There are four simple notations used to complete DFDs:

- **Dataflow:**

  Description: An arrow that shows the direction of data movement from one point to another.

- **External Entity:**

  Description: A square or rectangle that represents a source or destination of data outside the system (e.g., The User).

- **Process:**

  Description: A circle or rounded rectangle that represents a transformation or manipulation of data (e.g., "Calculate Interest" or "Verify Login").

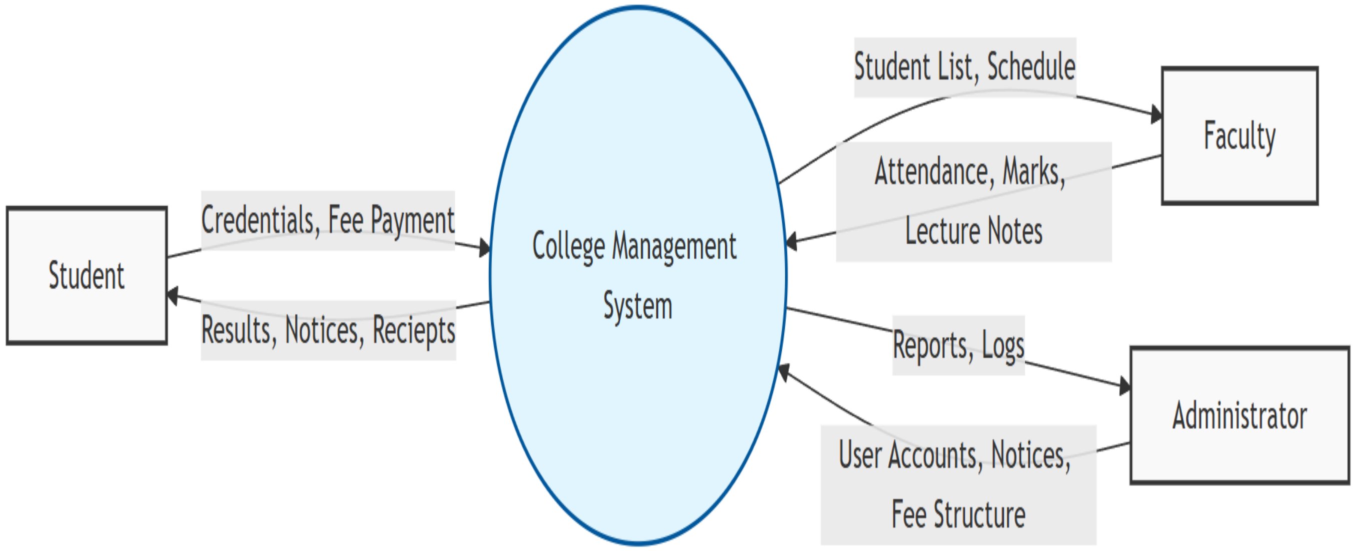
`

- **Data store:**

  Description: An open-ended rectangle representing a repository where data is stored for later use (e.g., The MySQL Database).

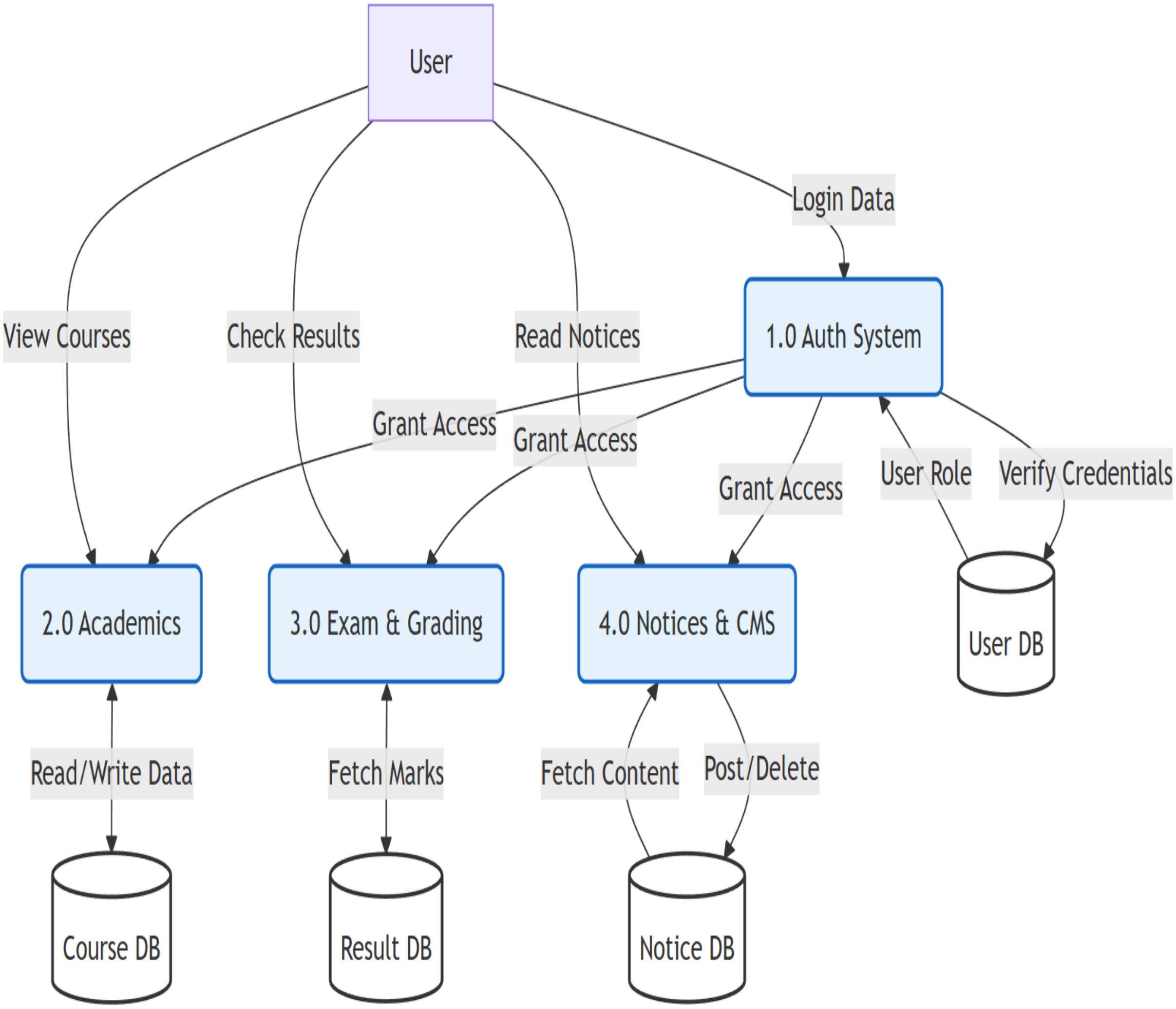
DFD FOR THE SYSTEM

Level 0 (Context Diagram):


Student
Credentials, Fee Payment
Results, Notices, Reciepts
College Management System
Student List, Schedule
Faculty
Attendance, Marks, Lecture Notes
Reports, Logs
Administrator
User Accounts, Notices, Fee Structure

`

Level 1 (Detailed Diagram):



`

ENTITY RELATIONSHIP DIAGRAM

Entity Relationship Diagram (ERD) can express the overall logical structure of a database graphically. It shows the relationships between different data sets.

The components of E-R Diagram are:

- **Entity:**

  Entity is a thing or object in a real world that is distinguishable from all other objects (e.g., *User*, *Expense*, *Category*).

- **Relationship:**

  It is an association among several entities (e.g., *User* **logs** *Expense*).

- **Weak Entity:**

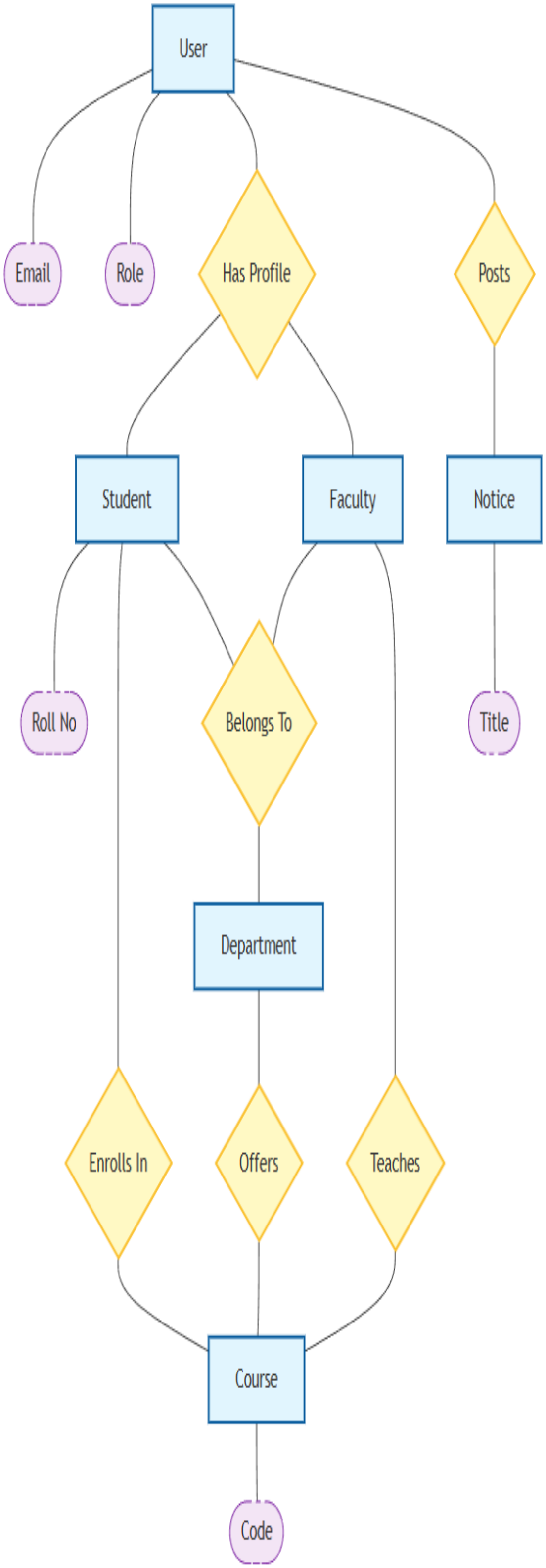
  A Weak entity is an entity that does not have any primary key of its own and depends on the existence of a strong entity (e.g., *Expense Category* might be weak if it relies entirely on a specific *User* context).

- **Attributes:**

  A property or characteristic of an entity. Often shown as an oval or circle (e.g., *Email*, *Amount*, *Date*).

`

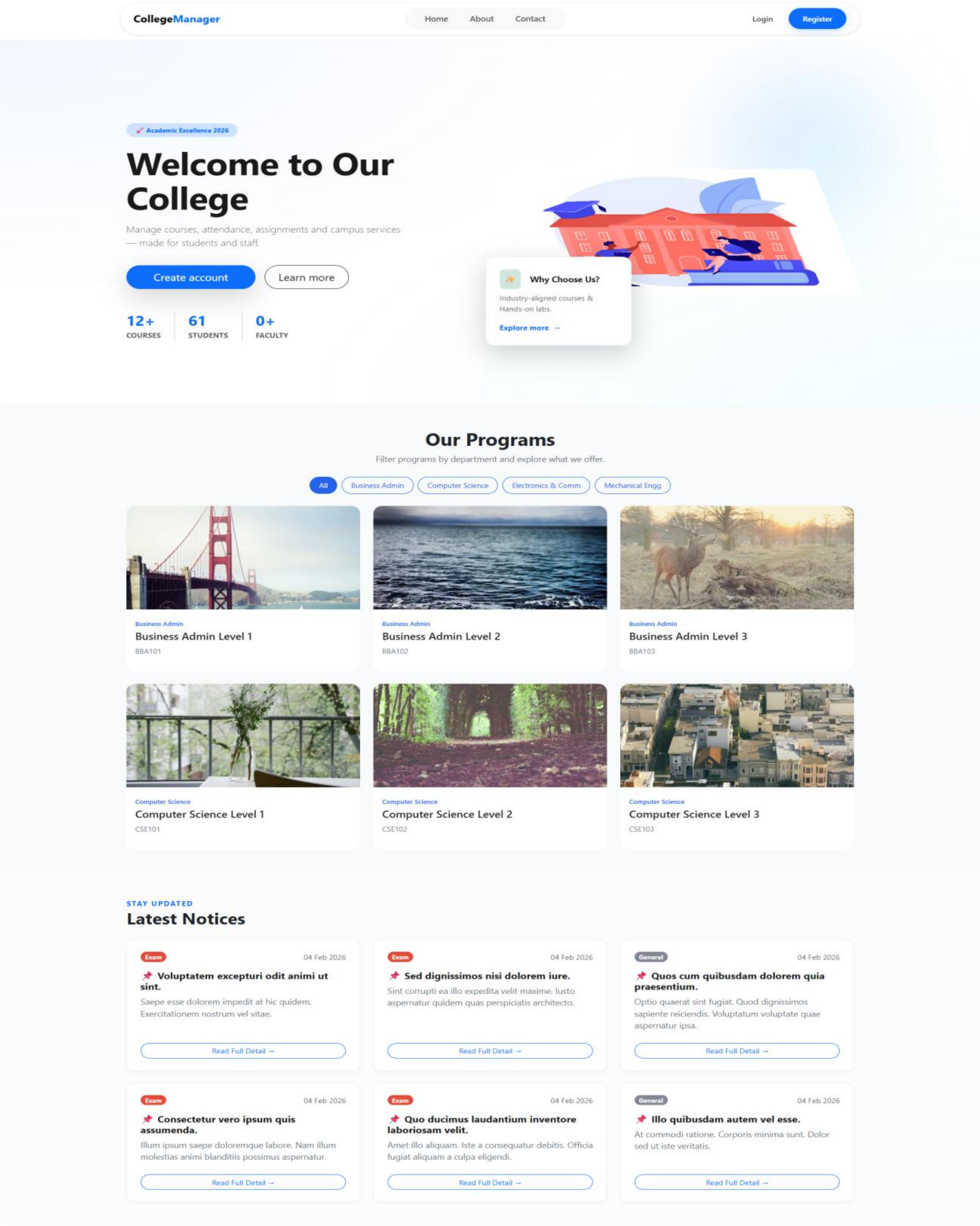
ENTITY RELATIONSHIP DIAGRAM (Visual)



`

INPUT FORMS, PAGES AND CODING

1. Home Page:



Academic Excellence 2026

Welcome to Our College

Manage courses, attendance, assignments and campus services — made for students and staff.

12+ COURSES

61 STUDENTS

0+ FACULTY

Why Choose Us?

Industry-aligned courses & Hands-on labs.

Explore more →

Our Programs

Filter programs by department and explore what we offer.

Business Admin
Business Admin Level 1
BBA101

Business Admin
Business Admin Level 2
BBA102

Business Admin
Business Admin Level 3
BBA103

Computer Science
Computer Science Level 1
CSE101

Computer Science
Computer Science Level 2
CSE102

Computer Science
Computer Science Level 3
CSE103

STAY UPDATED

Latest Notices

Exam 04 Feb 2026
📌 Voluptatem excepturi odit animi ut sint.
Saepe esse dolorem impedit at hic quidem. Exercitationem nostrum vel vitae.
Read Full Detail →

Exam 04 Feb 2026
📌 Sed dignissimos nisi dolorem iure.
Sint corrupti ea illo expedita velit maxime. Iusto aspernatur quidem quas perspiciatis architecto.
Read Full Detail →

General 04 Feb 2026
📌 Quos cum quibusdam dolorem quia praesentium.
Optio quaerat sint fugiat. Quod dignissimos sapiente reiciendis. Voluptatum voluptate quae aspernatur ipsa.
Read Full Detail →

Exam 04 Feb 2026
📌 Consectetur vero ipsum quis assumenda.
Illum ipsum saepe doloremque labore. Nam illum molestias animi blanditiis possimus aspernatur.
Read Full Detail →

Exam 04 Feb 2026
📌 Quo ducimus laudantium inventore laboriosam velit.
Amet illo aliquam. Iste a consequatur debitis. Officia fugiat aliquam a culpa eligendi.
Read Full Detail →

General 04 Feb 2026
📌 Illo quibusdam autem vel esse.
At commodi ratione. Corporis minima sunt. Dolor sed ut iste veritatis.
Read Full Detail →

`

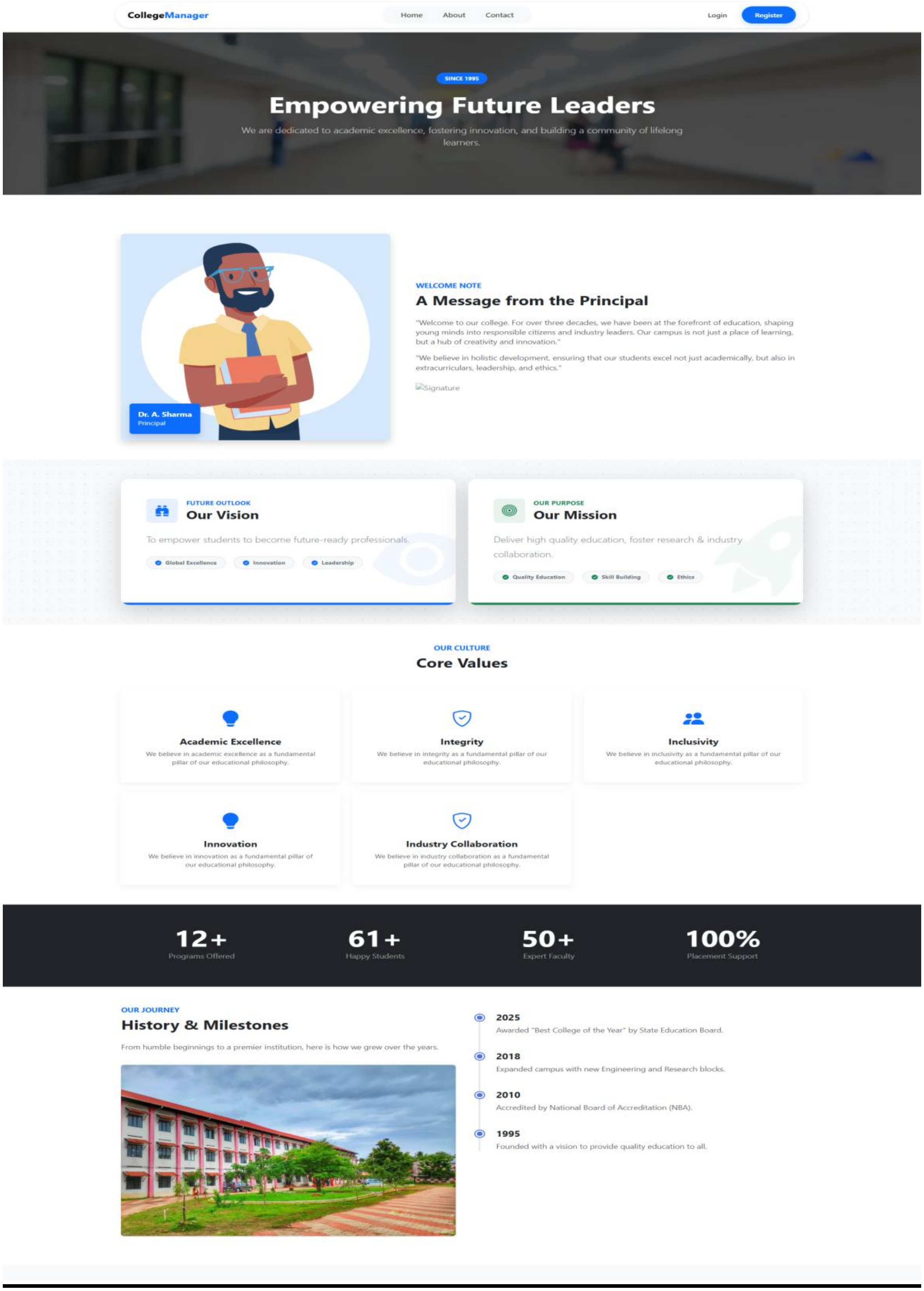
Code

```
{% extends "base.html" %}

{% block title %}Home - College Management System{% endblock %}

{% block content %}
<nav class="navbar navbar-expand-lg navbar-dark bg-primary">
  <div class="container">
    <a class="navbar-brand" href="#">
      <i class="fas fa-university"></i> CMS Portal
    </a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav">
      <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav ms-auto">
        <li class="nav-item">
          <a class="nav-link active" href="{{ url_for('main.index') }}">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('auth.login') }}">Login</a>
        </li>
        </ul>
    </div>
  </div>
</nav>

<header class="hero-section text-center py-5 bg-light">
  <div class="container">
    <h1 class="display-4 fw-bold">Welcome to College Management System</h1>
    <p class="lead text-muted mb-4">
      A centralized platform for Students, Faculty, and Administration.
      Manage attendance, results, and notices efficiently.
    </p>

    <div class="d-flex justify-content-center gap-3">
      <a href="{{ url_for('auth.login') }}" class="btn btn-primary btn-lg">
        <i class="fas fa-sign-in-alt"></i> Student Login
      </a>
      <a href="{{ url_for('auth.faculty_login') }}" class="btn btn-outline-dark btn-lg">
        <i class="fas fa-chalkboard-teacher"></i> Faculty Login
      </a>
    </div>
  </div>
</header>

<section class="py-5">
  <div class="container">
    <div class="row text-center">

      <div class="col-md-4 mb-4">
        <div class="card h-100 shadow-sm">
          <div class="card-body">
            <i class="fas fa-user-graduate fa-3x text-primary mb-3"></i>
            <h5 class="card-title">Student Portal</h5>
```

```
`
      <p class="card-text">View results, check attendance...</p>
     </div>
    </div>
   </div>

   <div class="col-md-4 mb-4">
    </div>

   <div class="col-md-4 mb-4">
    </div>

  </div>
 </div>
</section>

<footer class="bg-dark text-white text-center py-3 mt-auto">
 <div class="container">
  <p class="mb-0">&copy; 2026 College Management System. All Rights Reserved.</p>
 </div>
</footer>
{% endblock %}
```

`

2. About Page:



SINCE 1995

Empowering Future Leaders

We are dedicated to academic excellence, fostering innovation, and building a community of lifelong learners.

Dr. A. Sharma
Principal

WELCOME NOTE

A Message from the Principal

"Welcome to our college. For over three decades, we have been at the forefront of education, shaping young minds into responsible citizens and industry leaders. Our campus is not just a place of learning, but a hub of creativity and innovation."

"We believe in holistic development, ensuring that our students excel not just academically, but also in extracurriculars, leadership, and ethics."

Signature

FUTURE OUTLOOK

Our Vision

To empower students to become future-ready professionals.

✓ Global Excellence ✓ Innovation ✓ Leadership

OUR PURPOSE

Our Mission

Deliver high quality education, foster research & industry collaboration.

✓ Quality Education ✓ Skill Building ✓ Ethics

OUR CULTURE

Core Values

Academic Excellence
We believe in academic excellence as a fundamental pillar of our educational philosophy.

Integrity
We believe in integrity as a fundamental pillar of our educational philosophy.

Inclusivity
We believe in inclusivity as a fundamental pillar of our educational philosophy.

Innovation
We believe in innovation as a fundamental pillar of our educational philosophy.

Industry Collaboration
We believe in industry collaboration as a fundamental pillar of our educational philosophy.

12+ Programs Offered

61+ Happy Students

50+ Expert Faculty

100% Placement Support

OUR JOURNEY

History & Milestones

From humble beginnings to a premier institution, here is how we grew over the years.

2025
Awarded "Best College of the Year" by State Education Board.

2018
Expanded campus with new Engineering and Research blocks.

2010
Accredited by National Board of Accreditation (NBA).

1995
Founded with a vision to provide quality education to all.

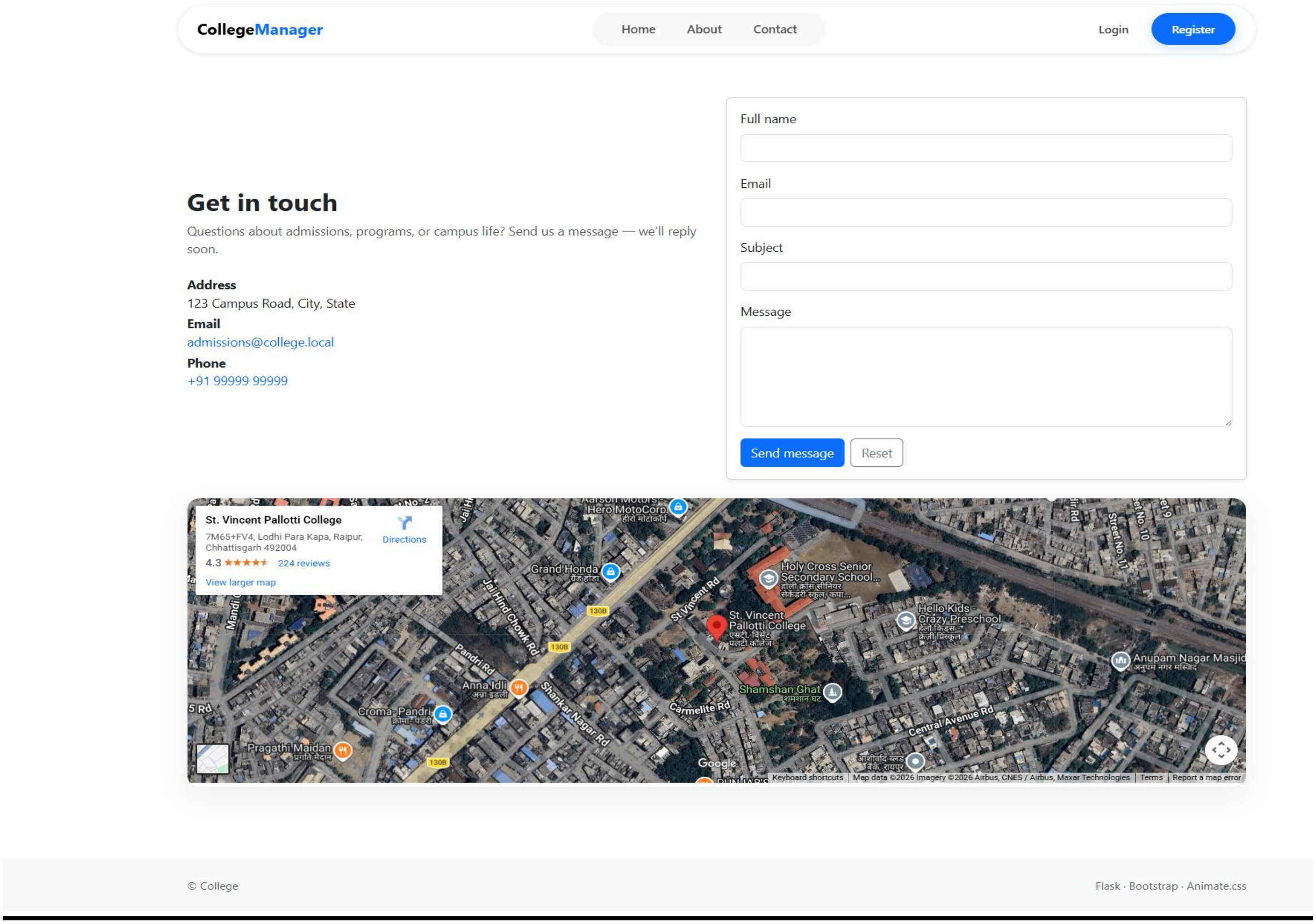
`

Code:

```
{% extends "base.html" %}

{% block title %}About Us - College Management System{% endblock %}

{% block content %}
<div class="container py-5">
   <div class="row mb-5">
      <div class="col-lg-8 mx-auto text-center">
         <h1 class="display-4">About Our Institute</h1>
         <p class="lead text-muted">
            Empowering education through technology and innovation since 1995.
         </p>
      </div>
   </div>
   <div class="row align-items-center mb-5">
      <div class="col-md-6">
         <img src="{{ url_for('static', filename='img/campus.jpg') }}" class="img-fluid rounded shadow" alt="Campus">
      </div>
      <div class="col-md-6">
         <h3>Our Mission</h3>
         <p>
            To provide a centralized platform that bridges the gap between students,
            faculty, and administration, fostering an environment of transparency
            and academic excellence.
         </p>

         <h3>Our Vision</h3>
         <p>
            To become a global leader in educational management, ensuring that
            every student has access to real-time academic resources.
         </p>
      </div>
   </div>
   <div class="row text-center bg-light py-4 rounded">
      <div class="col-md-4">
         <h2 class="text-primary fw-bold">5000+</h2>
         <p>Students Enrolled</p>
      </div>
      <div class="col-md-4">
         <h2 class="text-primary fw-bold">200+</h2>
         <p>Expert Faculty</p>
      </div>
      <div class="col-md-4">
         <h2 class="text-primary fw-bold">50+</h2>
         <p>Courses Offered</p>
      </div>
   </div>

</div>
{% endblock %}
```

`

3. Contact Page:



CollegeManager

Home About Contact

Login Register

Get in touch

Questions about admissions, programs, or campus life? Send us a message — we'll reply soon.

Address
123 Campus Road, City, State
Email
admissions@college.local
Phone
+91 99999 99999

Full name

Email

Subject

Message

Send message Reset

St. Vincent Pallotti College
7M65+FV4, Lodhi Para Kapa, Raipur, Chhattisgarh 492004
4.3 ★★★★★ 224 reviews
View larger map

Directions
Grand Honda
Jai Hind Chowk Rd
St Vincent Rd
Holy Cross Senior Secondary School
St. Vincent Pallotti College
Hello Kids - Crazy Preschool
Anupam Nagar Masjid
Pandri Rd
Anna Idli
Croma, Pandri
Shankar Nagar Rd
Carmelite Rd
Shamshan Ghat
Central Avenue Rd
Pragathi Maidan
Google
Keyboard shortcuts Map data ©2026 Imagery ©2026 Airbus, CNES / Airbus, Maxar Technologies Terms Report a map error

© College

Flask · Bootstrap · Animate.css

Code:

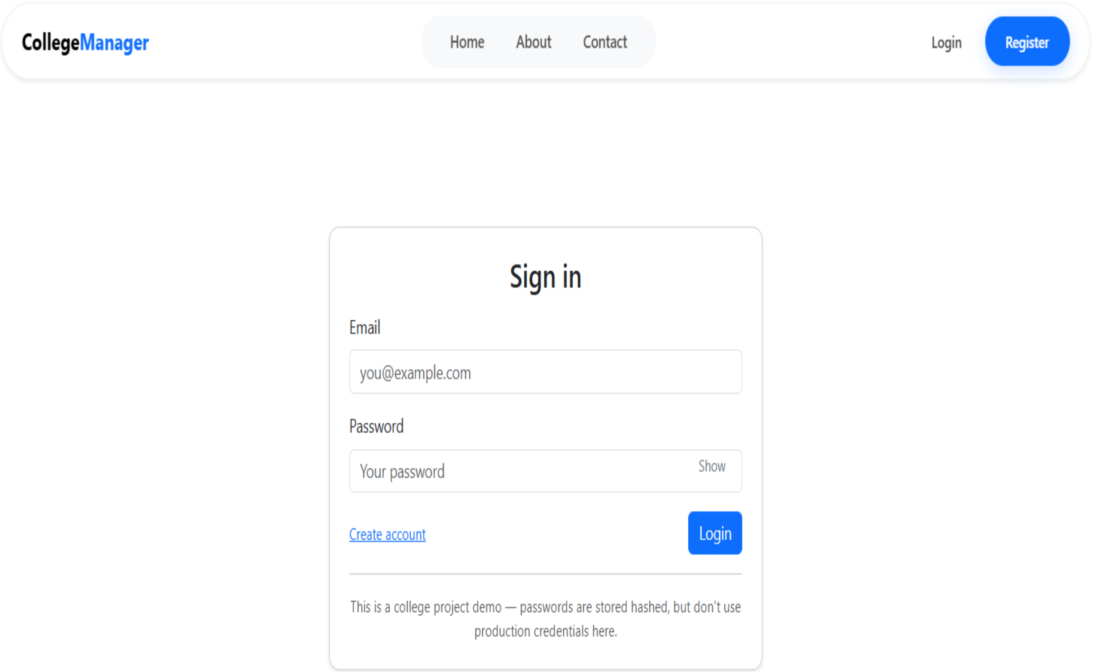
```
{% extends "base.html" %}

{% block title %}Contact Us - College Management System{% endblock %}

{% block content %}
<div class="container py-5">

<h2 class="text-center mb-5">Get in Touch</h2>

<div class="row g-5">

<div class="col-md-7">
<div class="card shadow-sm">
<div class="card-body p-4">
<form action="{{ url_for('main.contact') }}" method="POST">
<input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
```

`

```
<div class="mb-3">
<label for="name" class="form-label">Full Name</label>
<input type="text" class="form-control" id="name" name="name" required>
</div>

<div class="mb-3">
<label for="email" class="form-label">Email Address</label>
<input type="email" class="form-control" id="email" name="email" required>
</div>
<div class="mb-3">
<label for="message" class="form-label">Message</label>
<textarea class="form-control" id="message" name="message" rows="5"
required></textarea>
</div>
<button type="submit" class="btn btn-primary w-100">Send Message</button>
</form>
</div>
</div>
</div>
<div class="col-md-5">
<div class="h-100 p-4 bg-light rounded border">
<h4>Contact Information</h4>
<p class="text-muted mb-4">
Reach out to the administration for admissions or technical support.
</p>
<div class="d-flex align-items-start mb-3">
<i class="bi bi-geo-alt-fill text-primary me-3 fs-5"></i>
<div>
<h5>Address</h5>
<p class="mb-0">123 University Road, Knowledge City,<br>Raipur, Chhattisgarh -
492001</p>
</div>
</div>
<div class="d-flex align-items-start mb-3">
<i class="bi bi-envelope-fill text-primary me-3 fs-5"></i>
<div>
<h5>Email</h5>
<p class="mb-0">admin@collegems.edu.in</p>
</div>
</div>
<div class="d-flex align-items-start mb-3">
<i class="bi bi-telephone-fill text-primary me-3 fs-5"></i>
<div>
<h5>Phone</h5>
<p class="mb-0">+91 98765 43210</p>
</div>
</div>
<div class="map-container mt-4">
<iframe src="https://www.google.com/maps/embed?..."
width="100%" height="200" style="border:0;" allowfullscreen></iframe>
</div>
</div>
</div>
</div>
</div>
{% endblock %}
```

`

4. Login Page:



This is a college project demo — passwords are stored hashed, but don't use production credentials here.

Code:

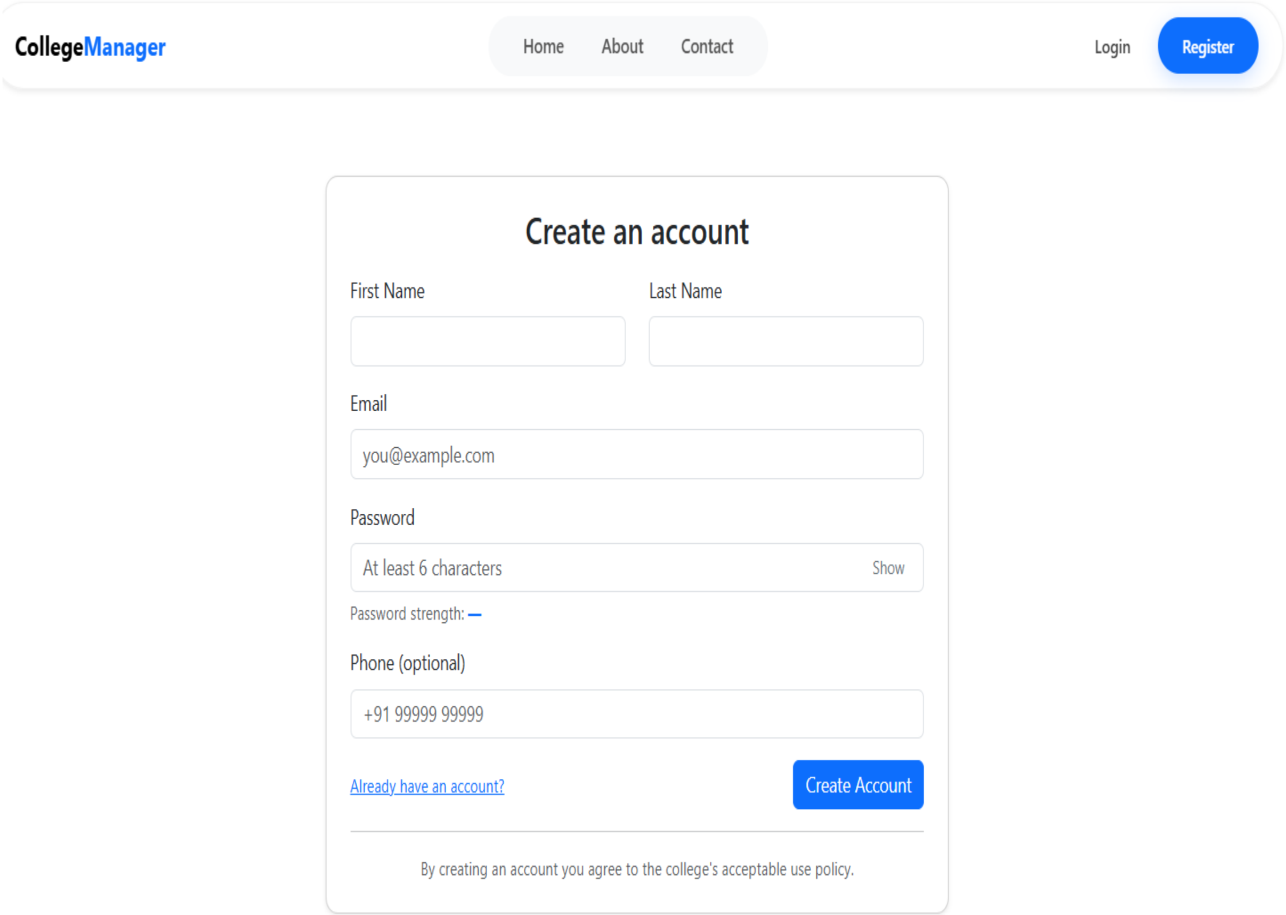
```
{% extends "base.html" %}

{% block title %}Login - College Management System{% endblock %}

{% block content %}
<div class="container d-flex justify-content-center align-items-center vh-100">
    <div class="card shadow-lg" style="width: 400px;">
        <div class="card-header bg-primary text-white text-center">
            <h4><i class="fas fa-user-lock"></i> User Login</h4>
        </div>

        <div class="card-body p-4">
            {% with messages = get_flashed_messages(with_categories=true) %}
                {% if messages %}
                    {% for category, message in messages %}
                        <div class="alert alert-{{ category }}">{{ message }}</div>
                    {% endfor %}
                {% endif %}
            {% endwith %}

            <form method="POST" action="{{ url_for('auth.login') }}">
                <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">

                <div class="mb-3">
                    <label for="email" class="form-label">Email Address</label>
```

`

```
                    <div class="input-group">
                        <span class="input-group-text"><i class="fas fa-envelope"></i></span>
                        <input type="email" class="form-control" name="email" required
placeholder="student@college.edu">
                    </div>
                </div>

                <div class="mb-3">
                    <label for="password" class="form-label">Password</label>
                    <div class="input-group">
                        <span class="input-group-text"><i class="fas fa-key"></i></span>
                        <input type="password" class="form-control" name="password" required>
                    </div>
                </div>

                <div class="d-grid gap-2">
                    <button type="submit" class="btn btn-primary">Login</button>
                </div>
            </form>
        </div>

        <div class="card-footer text-center">
            <small>Don't have an account? <a href="{{ url_for('auth.register') }}">Register
Here</a></small>
        </div>
    </div>
</div>
{% endblock %}
```

`

5. Register Page:



Create an account

First Name

Last Name

Email

you@example.com

Password

At least 6 characters

Password strength: —

Phone (optional)

+91 99999 99999

By creating an account you agree to the college's acceptable use policy.

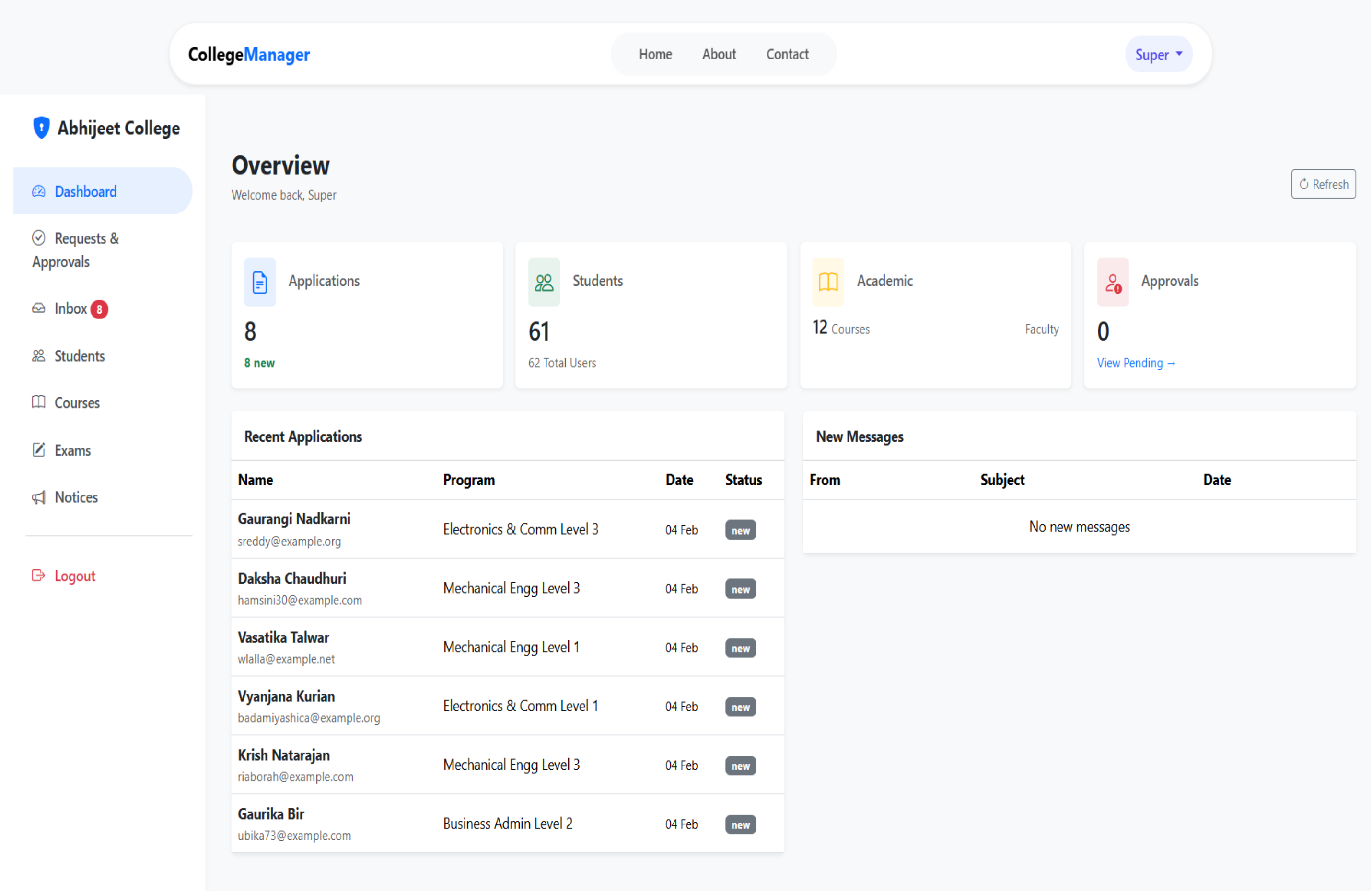
Code:

```
{% extends "base.html" %}

{% block title %}Register - College Management System{% endblock %}

{% block content %}
<div class="container py-5">
    <div class="row justify-content-center">
        <div class="col-md-6">
            <div class="card shadow">
                <div class="card-header bg-dark text-white">
                    <h4 class="mb-0"><i class="fas fa-user-plus"></i> Create Account</h4>
                </div>

                <div class="card-body">
                    <form method="POST" action="{{ url_for('auth.register') }}">
                        <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
```

```
`
                        <div class="mb-3">
                            <label class="form-label">Full Name</label>
                            <input type="text" class="form-control" name="name" required>
                        </div>

                        <div class="mb-3">
                            <label class="form-label">Email Address</label>
                            <input type="email" class="form-control" name="email" required>
                            <div class="form-text">Use your official college email ID.</div>
                        </div>

                        <div class="mb-3">
                            <label class="form-label">Register As</label>
                            <select class="form-select" name="role" required>
                                <option value="student">Student</option>
                                <option value="faculty">Faculty</option>
                            </select>
                        </div>

                        <div class="row">
                            <div class="col-md-6 mb-3">
                                <label class="form-label">Password</label>
                                <input type="password" class="form-control" name="password"
required>
                            </div>
                            <div class="col-md-6 mb-3">
                                <label class="form-label">Confirm Password</label>
                                <input type="password" class="form-control" name="confirm_password"
required>
                            </div>
                        </div>

                        <button type="submit" class="btn btn-success w-100">
                            Register Account
                        </button>
                    </form>
                </div>

                <div class="card-footer text-center">
                    <small>Already registered? <a href="{{ url_for('auth.login') }}">Login
here</a>.</small>
                </div>
            </div>
        </div>
    </div>
</div>
{% endblock %}
```

`

6. Admin Dashboard:



Overview

Welcome back, Super

Applications
8
8 new

Students
61
62 Total Users

Academic
12 Courses
Faculty

Approvals
0
View Pending →

Recent Applications

Name	Program	Date	Status
Gaurangi Nadkarni sreddy@example.org	Electronics & Comm Level 3	04 Feb	new
Daksha Chaudhuri hamsini30@example.com	Mechanical Engg Level 3	04 Feb	new
Vasatika Talwar wlalla@example.net	Mechanical Engg Level 1	04 Feb	new
Vyanjana Kurian badamiyashica@example.org	Electronics & Comm Level 1	04 Feb	new
Krish Natarajan riaborah@example.com	Mechanical Engg Level 3	04 Feb	new
Gaurika Bir ubika73@example.com	Business Admin Level 2	04 Feb	new

New Messages

From	Subject	Date
	No new messages	

Code:

```
{% extends "admin/base_admin.html" %}

{% block title %}Admin Dashboard - CMS{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

  <div class="d-flex justify-content-between align-items-center mb-4">
    <h2><i class="fas fa-tachometer-alt"></i> Admin Overview</h2>
    <span class="text-muted">Welcome, {{ current_user.name }}</span>
  </div>

  <div class="row g-3 mb-4">

    <div class="col-md-3">
      <div class="card text-white bg-primary h-100">
        <div class="card-body d-flex align-items-center">
          <i class="fas fa-user-graduate fa-3x opacity-50 me-3"></i>
          <div>
```

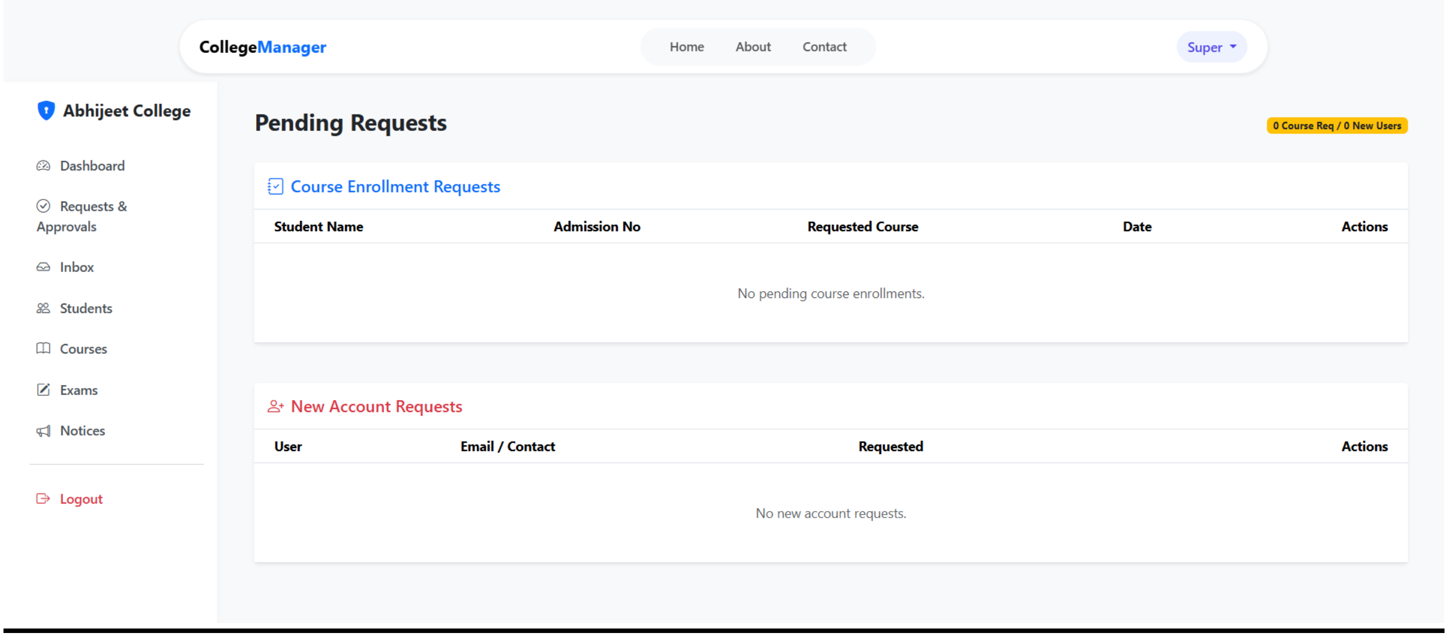
`

```
                        <h5 class="card-title">Students</h5>
                        <h2 class="mb-0">{{ stats.total_students }}</h2>
                    </div>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card text-white bg-success h-100">
                <div class="card-body d-flex align-items-center">
                    <i class="fas fa-chalkboard-teacher fa-3x opacity-50 me-3"></i>
                    <div>
                        <h5 class="card-title">Faculty</h5>
                        <h2 class="mb-0">{{ stats.total_faculty }}</h2>
                    </div>
                </div>
            </div>
        </div>

        <div class="col-md-3">
            <div class="card text-white bg-warning h-100">
                <div class="card-body d-flex align-items-center">
                    <i class="fas fa-user-clock fa-3x opacity-50 me-3"></i>
                    <div>
                        <h5 class="card-title">Pending</h5>
                        <h2 class="mb-0">{{ stats.pending_users }}</h2>
                    </div>
                </div>
                <a href="{{ url_for('admin.approvals') }}" class="card-footer text-white small">
                    View Requests <i class="fas fa-arrow-right"></i>
                </a>
            </div>
        </div>

    </div>

    <div class="row">
        <div class="col-md-6">
            <div class="card shadow-sm">
                <div class="card-header bg-dark text-white">
                    <h5>Quick Actions</h5>
                </div>
                <div class="card-body d-grid gap-2">
                    <a href="{{ url_for('admin.add_student') }}" class="btn btn-outline-primary text-
start">
                        <i class="fas fa-plus-circle"></i> Admit New Student
                    </a>
                    <a href="{{ url_for('admin.create_notice') }}" class="btn btn-outline-info text-
start">
                        <i class="fas fa-bullhorn"></i> Post New Notice
                    </a>
                    <a href="{{ url_for('admin.manage_fees') }}" class="btn btn-outline-success
text-start">
                        <i class="fas fa-file-invoice-dollar"></i> Update Fee Structure
                    </a>
                </div>
            </div>
        </div>
```

```
`
             </div>

          </div>
          {% endblock %}
```

`

7. Admin Requests & Approvals:



Pending Requests

0 Course Req / 0 New Users

Course Enrollment Requests

Student Name	Admission No	Requested Course	Date	Actions

No pending course enrollments.

New Account Requests

User	Email / Contact	Requested	Actions

No new account requests.

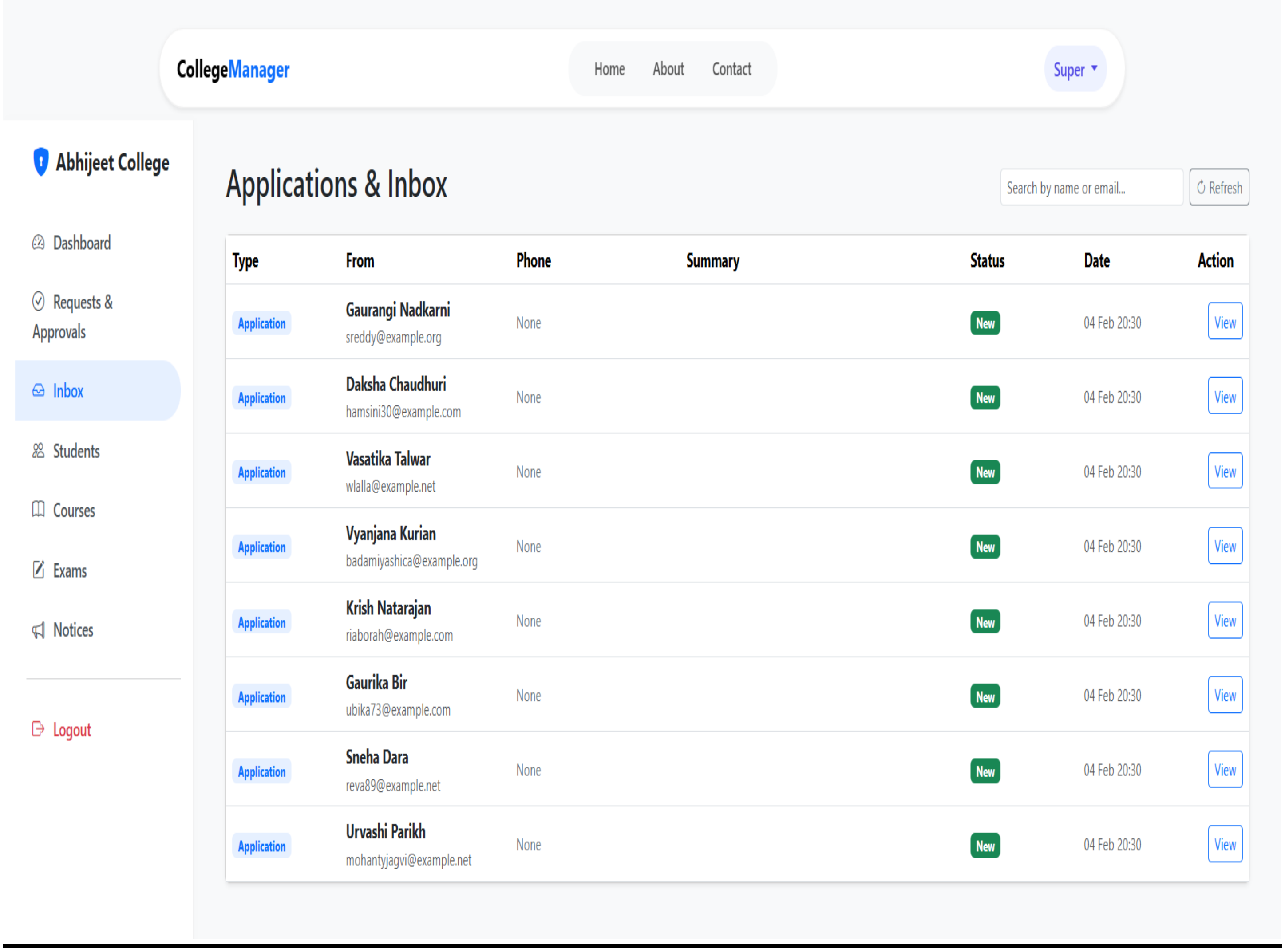
Code:

```
{% extends "admin/base_admin.html" %}

{% block title %}Pending Approvals - CMS{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

    <h3 class="mb-4"><i class="fas fa-check-double"></i> Pending User Approvals</h3>

    {% if pending_users %}
    <div class="card shadow-sm">
        <div class="card-body p-0">
            <div class="table-responsive">
                <table class="table table-hover align-middle mb-0">
                    <thead class="bg-light">
                        <tr>
                            <th>Name</th>
                            <th>Email</th>
                            <th>Role</th>
                            <th>Reg. Date</th>
                            <th>Actions</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for user in pending_users %}
                        <tr>
                            <td>
                                <div class="fw-bold">{{ user.name }}</div>
```

```
`
                            </td>
                            <td>{{ user.email }}</td>
                            <td>
                                <span class="badge bg-secondary">{{ user.role|capitalize }}</span>
                            </td>
                            <td>{{ user.created_at.strftime('%Y-%m-%d') }}</td>
                            <td>
                                <div class="btn-group btn-group-sm">
                                    <form action="{{ url_for('admin.approve_user', user_id=user.id) }}"
method="POST" class="d-inline">
                                        <input type="hidden" name="csrf_token" value="{{ csrf_token()
}}">
                                        <button type="submit" class="btn btn-success" title="Approve">
                                            <i class="fas fa-check"></i> Approve
                                        </button>
                                    </form>

                                    <form action="{{ url_for('admin.reject_user', user_id=user.id) }}"
method="POST" class="d-inline ms-1">
                                        <input type="hidden" name="csrf_token" value="{{ csrf_token()
}}">
                                        <button type="submit" class="btn btn-danger" title="Reject"
onclick="return confirm('Are you sure?');">
                                            <i class="fas fa-times"></i> Reject
                                        </button>
                                    </form>
                                </div>
                            </td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
    </div>
    {% else %}
        <div class="alert alert-info text-center">
            <i class="fas fa-info-circle"></i> No pending approvals at this time.
        </div>
    {% endif %}

</div>
{% endblock %}
```

`

8. Admin Inbox:



Applications & Inbox

Type	From	Phone	Summary	Status	Date	Action
Application	Gaurangi Nadkarni sreddy@example.org	None		New	04 Feb 20:30	View
Application	Daksha Chaudhuri hamsini30@example.com	None		New	04 Feb 20:30	View
Application	Vasatika Talwar wlalla@example.net	None		New	04 Feb 20:30	View
Application	Vyanjana Kurian badamiyashica@example.org	None		New	04 Feb 20:30	View
Application	Krish Natarajan riaborah@example.com	None		New	04 Feb 20:30	View
Application	Gaurika Bir ubika73@example.com	None		New	04 Feb 20:30	View
Application	Sneha Dara reva89@example.net	None		New	04 Feb 20:30	View
Application	Urvashi Parikh mohantyjagvi@example.net	None		New	04 Feb 20:30	View

Code:

```
{% extends "admin/base_admin.html" %}

{% block title %}Inbox - Admin Dashboard{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

    <div class="d-flex justify-content-between align-items-center mb-4">
        <h3><i class="fas fa-inbox"></i> Admin Inbox</h3>
        <span class="badge bg-primary">{{ messages|length }} New Messages</span>
    </div>

    <div class="card shadow-sm">
        <div class="card-body p-0">
            <div class="table-responsive">
```

`

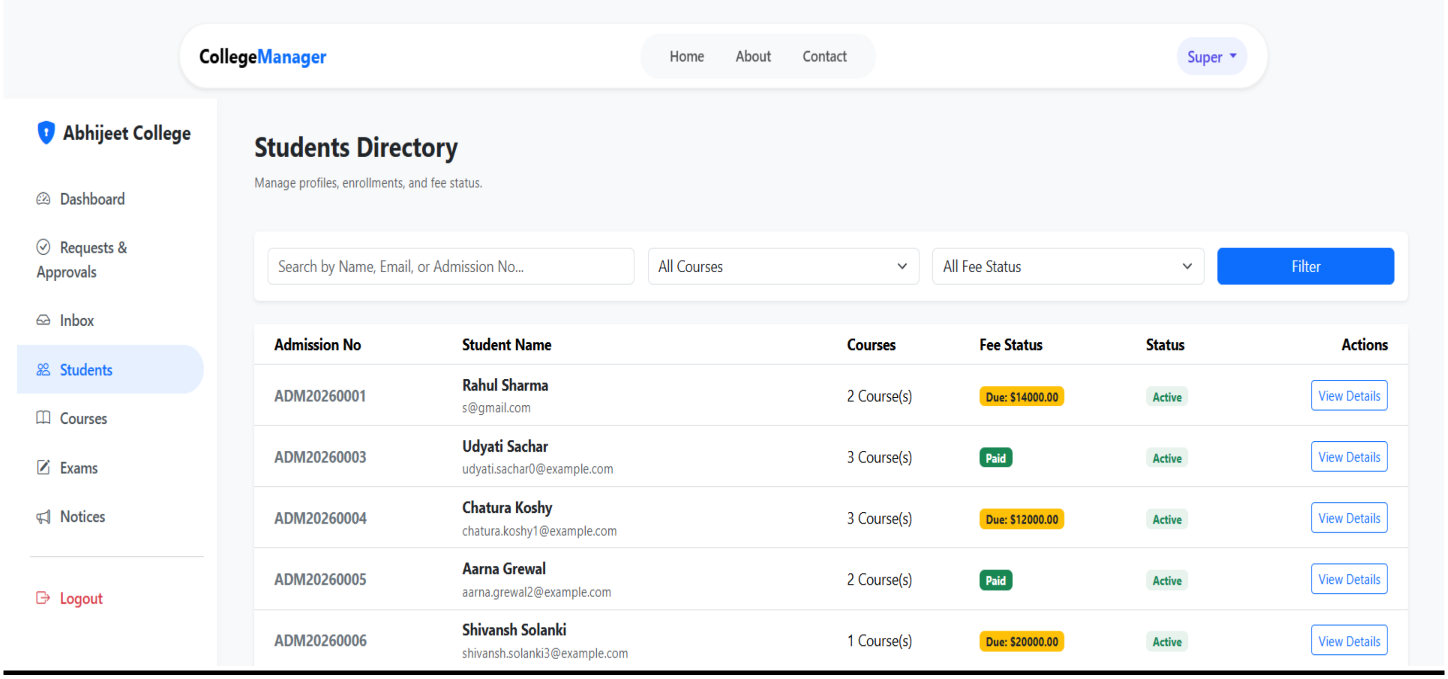
```
                <table class="table table-hover align-middle mb-0">
                    <thead class="bg-light">
                        <tr>
                            <th>Date</th>
                            <th>Sender</th>
                            <th>Subject / Message</th>
                            <th>Actions</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for msg in messages %}
                        <tr class="{% if not msg.is_read %}fw-bold bg-white{% endif %}">
                            <td class="text-muted small">{{ msg.timestamp.strftime('%d %b, %Y')
}}</td>
                            <td>
                                <div>{{ msg.name }}</div>
                                <small class="text-muted">{{ msg.email }}</small>
                            </td>
                            <td>
                                <div class="text-truncate" style="max-width: 300px;">
                                    {{ msg.message }}
                                </div>
                            </td>
                            <td>
                                <div class="btn-group btn-group-sm">
                                    <button class="btn btn-outline-primary" data-bs-toggle="modal" data-
bs-target="#viewMsg{{ msg.id }}">
                                        <i class="fas fa-eye"></i> View
                                    </button>
                                    <form action="{{ url_for('admin.delete_message', msg_id=msg.id) }}"
method="POST" class="d-inline">
                                        <input type="hidden" name="csrf_token" value="{{ csrf_token()
}}">
                                        <button class="btn btn-outline-danger" onclick="return
confirm('Delete this message?');">
                                            <i class="fas fa-trash"></i>
                                        </button>
                                    </form>
                                </div>

                                <div class="modal fade" id="viewMsg{{ msg.id }}" tabindex="-1">
                                    <div class="modal-dialog">
                                        <div class="modal-content">
                                            <div class="modal-header">
                                                <h5 class="modal-title">Message Details</h5>
                                                <button type="button" class="btn-close" data-bs-
dismiss="modal"></button>
                                            </div>
                                            <div class="modal-body fw-normal">
                                                <p><strong>From:</strong> {{ msg.name }} ({{ msg.email
}})</p>
                                                <p><strong>Message:</strong><br>{{ msg.message }}</p>
                                            </div>
                                            <div class="modal-footer">
                                                <a href="mailto:{{ msg.email }}" class="btn btn-primary">
                                                    <i class="fas fa-reply"></i> Reply via Email
                                                </a>
                                            </div>
```

`

```
                    </div>
                  </div>
                </div>
              </td>
            </tr>
            {% endfor %}
          </tbody>
        </table>
      </div>
    </div>
  </div>

</div>
{% endblock %}
```

`

9. Admin Students:



Students Directory

Manage profiles, enrollments, and fee status.

Admission No	Student Name	Courses	Fee Status	Status	Actions
ADM20260001	Rahul Sharma s@gmail.com	2 Course(s)	Due: $14000.00	Active	View Details
ADM20260003	Udyati Sachar udyati.sachar0@example.com	3 Course(s)	Paid	Active	View Details
ADM20260004	Chatura Koshy chatura.koshy1@example.com	3 Course(s)	Due: $12000.00	Active	View Details
ADM20260005	Aarna Grewal aarna.grewal2@example.com	2 Course(s)	Paid	Active	View Details
ADM20260006	Shivansh Solanki shivansh.solanki3@example.com	1 Course(s)	Due: $20000.00	Active	View Details

Code:

```
{% extends "admin/base_admin.html" %}

{% block title %}Manage Students - Admin Dashboard{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

    <div class="d-flex justify-content-between align-items-center mb-4">
        <h3><i class="fas fa-user-graduate"></i> Manage Students</h3>
        <a href="{{ url_for('admin.add_student') }}" class="btn btn-success">
            <i class="fas fa-plus"></i> Add New Student
        </a>
    </div>

    <div class="card mb-4">
        <div class="card-body">
            <form method="GET" action="{{ url_for('admin.manage_students') }}" class="row g-3">
                <div class="col-md-4">
                    <input type="text" class="form-control" name="search" placeholder="Search by Name or Roll No..." value="{{ request.args.get('search', '') }}">
                </div>
                <div class="col-md-3">
                    <select class="form-select" name="dept">
                        <option value="">All Departments</option>
                        <option value="CSE">Computer Science</option>
                        <option value="ME">Mechanical</option>
```

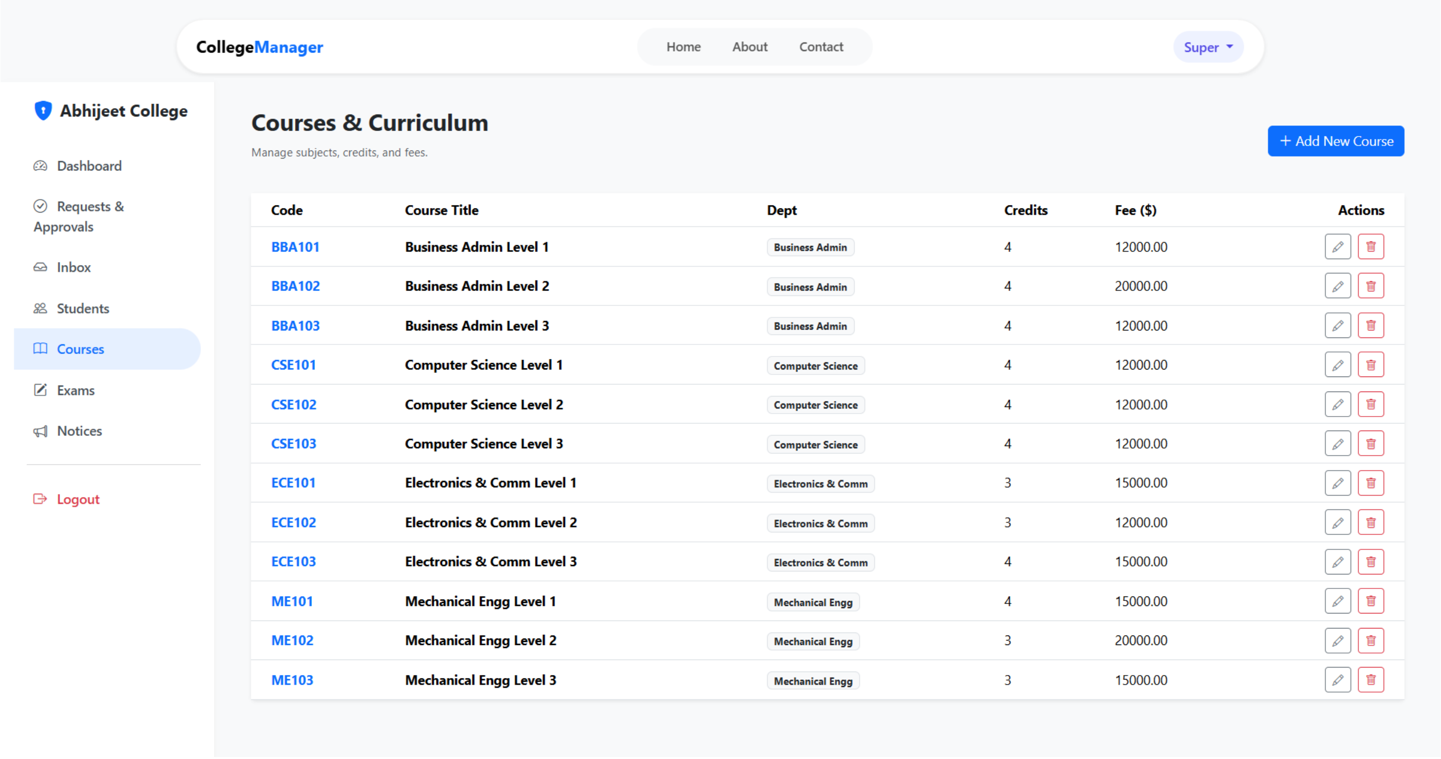
```
`
                        </select>
                    </div>
                    <div class="col-md-2">
                        <button type="submit" class="btn btn-primary w-100">Filter</button>
                    </div>
                </form>
            </div>
        </div>

        <div class="card shadow-sm">
            <div class="card-body p-0">
                <div class="table-responsive">
                    <table class="table table-striped align-middle mb-0">
                        <thead class="bg-dark text-white">
                            <tr>
                                <th>Roll No</th>
                                <th>Student Name</th>
                                <th>Department</th>
                                <th>Semester</th>
                                <th>Status</th>
                                <th>Actions</th>
                            </tr>
                        </thead>
                        <tbody>
                            {% for student in students %}
                            <tr>
                                <td><code>{{ student.roll_no }}</code></td>
                                <td>
                                    <div class="d-flex align-items-center">
                                        <img src="{{ student.avatar }}" class="rounded-circle me-2"
width="30">
                                        {{ student.name }}
                                    </div>
                                </td>
                                <td>{{ student.department }}</td>
                                <td>{{ student.semester }}</td>
                                <td>
                                    {% if student.is_active %}
                                        <span class="badge bg-success">Active</span>
                                    {% else %}
                                        <span class="badge bg-danger">Inactive</span>
                                    {% endif %}
                                </td>
                                <td>
                                    <div class="btn-group btn-group-sm">
                                        <a href="{{ url_for('admin.edit_student', id=student.id) }}" class="btn
btn-warning" title="Edit">
                                            <i class="fas fa-edit"></i>
                                        </a>
                                        <a href="{{ url_for('admin.view_student', id=student.id) }}"
class="btn btn-info" title="View Profile">
                                            <i class="fas fa-id-card"></i>
                                        </a>
                                        <form action="{{ url_for('admin.delete_student', id=student.id) }}"
method="POST" class="d-inline">
                                            <input type="hidden" name="csrf_token" value="{{ csrf_token()
}}">
                                            <button type="submit" class="btn btn-danger" onclick="return
```

`

```
confirm('Delete this student record?');">
                                            <i class="fas fa-trash-alt"></i>
                                        </button>
                                    </form>
                                </div>
                            </td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>

            <div class="card-footer py-3">
                <nav>
                    <ul class="pagination justify-content-center mb-0">
                        <li class="page-item disabled"><a class="page-link"
href="#">Previous</a></li>
                        <li class="page-item active"><a class="page-link" href="#">1</a></li>
                        <li class="page-item"><a class="page-link" href="#">Next</a></li>
                    </ul>
                </nav>
            </div>

        </div>
    </div>

</div>
{% endblock %}
```

`

10. Admin Courses Management:



Courses & Curriculum

Manage subjects, credits, and fees.

Code	Course Title	Dept	Credits	Fee ($)	Actions
BBA101	Business Admin Level 1	Business Admin	4	12000.00	✎ 🗑
BBA102	Business Admin Level 2	Business Admin	4	20000.00	✎ 🗑
BBA103	Business Admin Level 3	Business Admin	4	12000.00	✎ 🗑
CSE101	Computer Science Level 1	Computer Science	4	12000.00	✎ 🗑
CSE102	Computer Science Level 2	Computer Science	4	12000.00	✎ 🗑
CSE103	Computer Science Level 3	Computer Science	4	12000.00	✎ 🗑
ECE101	Electronics & Comm Level 1	Electronics & Comm	3	15000.00	✎ 🗑
ECE102	Electronics & Comm Level 2	Electronics & Comm	3	12000.00	✎ 🗑
ECE103	Electronics & Comm Level 3	Electronics & Comm	4	15000.00	✎ 🗑
ME101	Mechanical Engg Level 1	Mechanical Engg	4	15000.00	✎ 🗑
ME102	Mechanical Engg Level 2	Mechanical Engg	3	20000.00	✎ 🗑
ME103	Mechanical Engg Level 3	Mechanical Engg	3	15000.00	✎ 🗑

Code:

```
{% extends "admin/base_admin.html" %}

{% block title %}Manage Courses - CMS{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

    <div class="d-flex justify-content-between align-items-center mb-4">
        <h3><i class="fas fa-book"></i> Course Management</h3>
        <button class="btn btn-primary" data-bs-toggle="modal" data-bs-
target="#addCourseModal">
            <i class="fas fa-plus"></i> Add New Course
        </button>
    </div>

    <div class="card shadow-sm">
        <div class="card-body p-0">
            <table class="table table-hover align-middle mb-0">
                <thead class="bg-light">
                    <tr>
                        <th>Code</th>
                        <th>Course Name</th>
                        <th>Department</th>
                        <th>Credits</th>
```

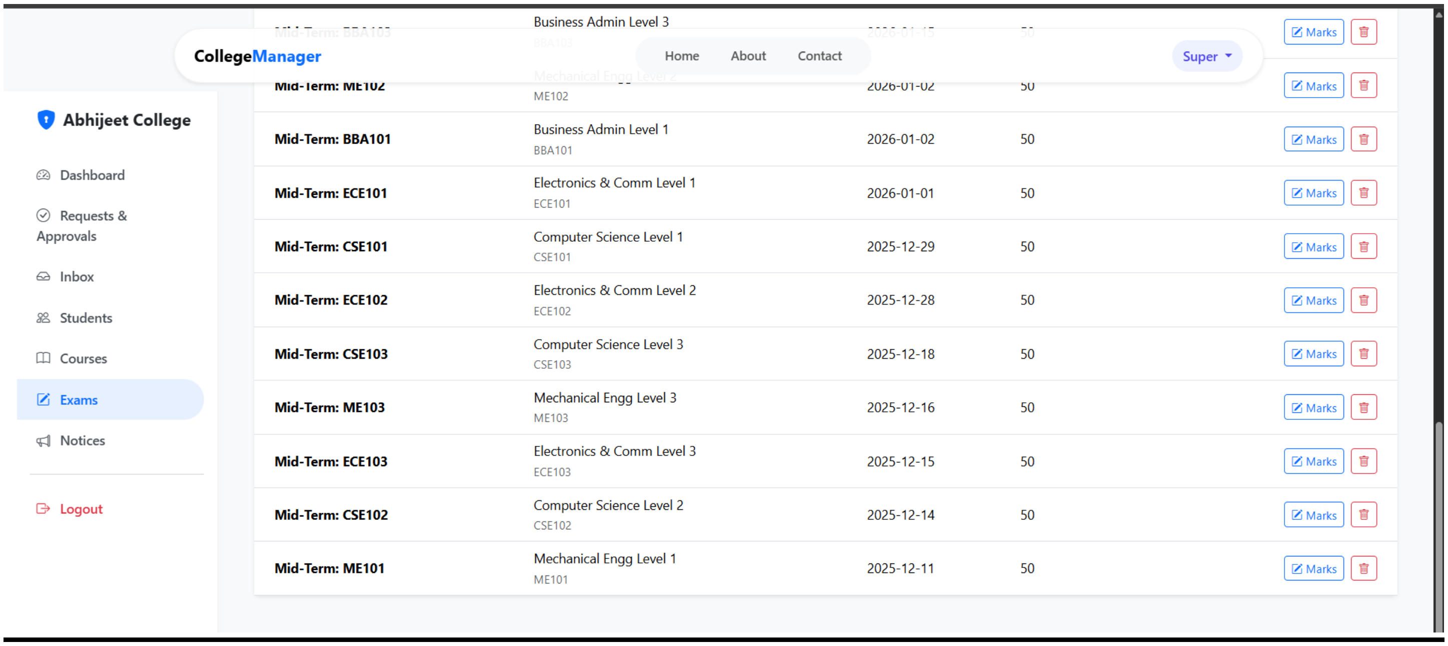
```
`
                        <th>Faculty In-Charge</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    {% for course in courses %}
                    <tr>
                        <td><span class="badge bg-secondary">{{ course.code }}</span></td>
                        <td class="fw-bold">{{ course.name }}</td>
                        <td>{{ course.department }}</td>
                        <td>{{ course.credits }}</td>
                        <td>
                            {% if course.faculty %}
                                <i class="fas fa-user-tie text-muted me-1"></i> {{ course.faculty.name }}
                            {% else %}
                                <span class="text-danger small">Unassigned</span>
                            {% endif %}
                        </td>
                        <td>
                            <div class="btn-group btn-group-sm">
                                <a href="{{ url_for('admin.edit_course', id=course.id) }}" class="btn btn-outline-primary">
                                    <i class="fas fa-edit"></i>
                                </a>
                                <form action="{{ url_for('admin.delete_course', id=course.id) }}" method="POST" class="d-inline">
                                    <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
                                    <button type="submit" class="btn btn-outline-danger" onclick="return confirm('Delete this course?');">
                                        <i class="fas fa-trash"></i>
                                    </button>
                                </form>
                            </div>
                        </td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>

    <div class="modal fade" id="addCourseModal" tabindex="-1">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="{{ url_for('admin.add_course') }}" method="POST">
                    <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
                    <div class="modal-header">
                        <h5 class="modal-title">Add New Course</h5>
                        <button type="button" class="btn-close" data-bs-dismiss="modal"></button>
                    </div>
                    <div class="modal-body">
                        <div class="mb-3">
                            <label>Course Name</label>
                            <input type="text" name="name" class="form-control" required>
                        </div>
                        <div class="row">
                            <div class="col-md-6 mb-3">
```

`

```
                                <label>Course Code</label>
                                <input type="text" name="code" class="form-control" placeholder="e.g. CS101" required>
                            </div>
                            <div class="col-md-6 mb-3">
                                <label>Credits</label>
                                <input type="number" name="credits" class="form-control" min="1" max="5" required>
                            </div>
                        </div>
                        <div class="mb-3">
                            <label>Department</label>
                            <select name="department" class="form-select" required>
                                <option value="CSE">Computer Science</option>
                                <option value="ME">Mechanical</option>
                                </select>
                        </div>
                    </div>
                    <div class="modal-footer">
                        <button type="submit" class="btn btn-success">Save Course</button>
                    </div>
                </form>
            </div>
        </div>
    </div>

</div>
{% endblock %}
```

`

11. Admin Exams:



Mid-Term: BBA103	Business Admin Level 3 BBA103	2026-01-15	50	Marks 🗑
Mid-Term: ME102	Mechanical Engg Level 2 ME102	2026-01-02	50	Marks 🗑
Mid-Term: BBA101	Business Admin Level 1 BBA101	2026-01-02	50	Marks 🗑
Mid-Term: ECE101	Electronics & Comm Level 1 ECE101	2026-01-01	50	Marks 🗑
Mid-Term: CSE101	Computer Science Level 1 CSE101	2025-12-29	50	Marks 🗑
Mid-Term: ECE102	Electronics & Comm Level 2 ECE102	2025-12-28	50	Marks 🗑
Mid-Term: CSE103	Computer Science Level 3 CSE103	2025-12-18	50	Marks 🗑
Mid-Term: ME103	Mechanical Engg Level 3 ME103	2025-12-16	50	Marks 🗑
Mid-Term: ECE103	Electronics & Comm Level 3 ECE103	2025-12-15	50	Marks 🗑
Mid-Term: CSE102	Computer Science Level 2 CSE102	2025-12-14	50	Marks 🗑
Mid-Term: ME101	Mechanical Engg Level 1 ME101	2025-12-11	50	Marks 🗑

Code:

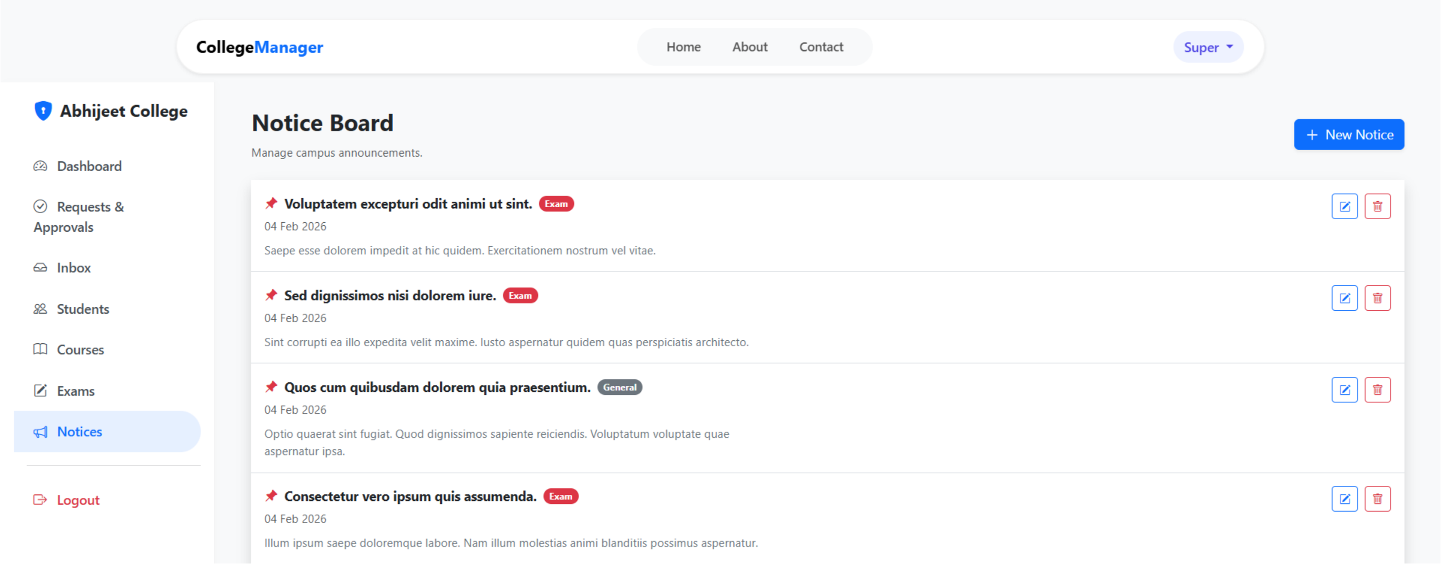
```
{% extends "admin/base_admin.html" %}

{% block title %}Examination Schedule - CMS{% endblock %}

{% block admin_content %}
<div class="container-fluid py-4">

    <div class="d-flex justify-content-between align-items-center mb-4">
        <h3><i class="fas fa-clock"></i> Examination Schedule</h3>
        <a href="{{ url_for('admin.schedule_exam') }}" class="btn btn-warning text-dark">
            <i class="fas fa-calendar-plus"></i> Schedule New Exam
        </a>
    </div>

    <div class="card shadow-sm">
        <div class="card-body p-0">
            <table class="table table-striped mb-0">
                <thead class="bg-dark text-white">
                    <tr>
                        <th>Date</th>
                        <th>Time</th>
                        <th>Subject / Course</th>
                        <th>Semester</th>
                        <th>Room No</th>
                        <th>Status</th>
                        <th>Action</th>
                    </tr>
                </thead>
```

`

```
                <tbody>
                    {% for exam in exams %}
                    <tr>
                        <td>{{ exam.date.strftime('%d %b, %Y') }}</td>
                        <td>{{ exam.start_time.strftime('%I:%M %p') }}</td>
                        <td>
                            <strong>{{ exam.course.name }}</strong><br>
                            <small class="text-muted">{{ exam.course.code }}</small>
                        </td>
                        <td>{{ exam.semester }}</td>
                        <td><span class="badge bg-info text-dark">{{ exam.room_no }}</span></td>
                        <td>
                            {% if exam.is_completed %}
                                <span class="badge bg-secondary">Completed</span>
                            {% else %}
                                <span class="badge bg-success">Upcoming</span>
                            {% endif %}
                        </td>
                        <td>
                            <div class="btn-group btn-group-sm">
                                <a href="#" class="btn btn-outline-secondary" title="Edit"><i class="fas fa-pen"></i></a>
                                <a href="#" class="btn btn-outline-danger" title="Cancel"><i class="fas fa-times"></i></a>
                            </div>
                        </td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>

</div>
{% endblock %}
```

`

12. Admin Notices:



Notice Board

Manage campus announcements.

Voluptatem excepturi odit animi ut sint. Exam

04 Feb 2026

Saepe esse dolorem impedit at hic quidem. Exercitationem nostrum vel vitae.

Sed dignissimos nisi dolorem iure. Exam

04 Feb 2026

Sint corrupti ea illo expedita velit maxime. Iusto aspernatur quidem quas perspiciatis architecto.

Quos cum quibusdam dolorem quia praesentium. General

04 Feb 2026

Optio quaerat sint fugiat. Quod dignissimos sapiente reiciendis. Voluptatum voluptate quae aspernatur ipsa.

Consectetur vero ipsum quis assumenda. Exam

04 Feb 2026

Illum ipsum saepe doloremque labore. Nam illum molestias animi blanditiis possimus aspernatur.

Code:

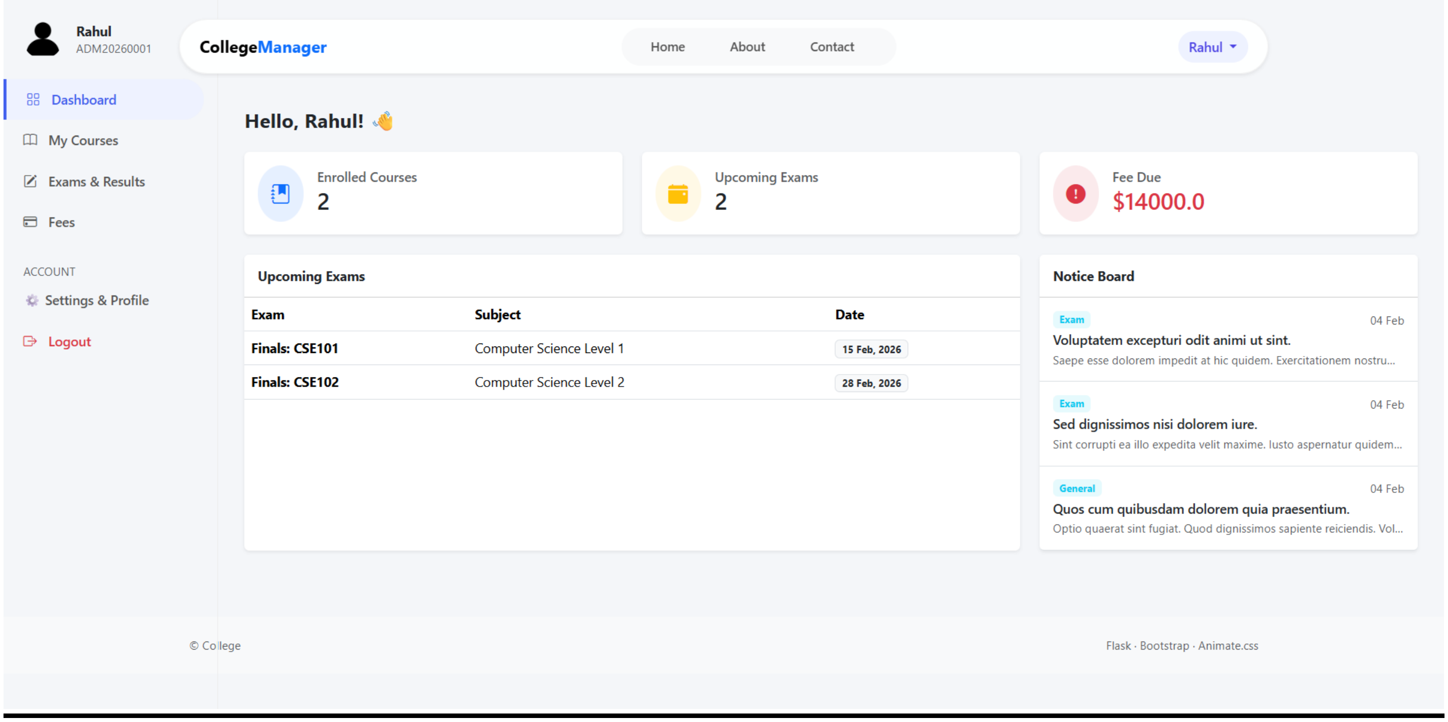
```
{% extends "base.html" %}

{% block title %}Notice Board - CMS{% endblock %}

{% block content %}
<div class="container py-5">

    <div class="text-center mb-5">
        <h2 class="fw-bold"><i class="fas fa-bullhorn text-primary"></i> Digital Notice
Board</h2>
        <p class="text-muted">Stay updated with the latest campus announcements.</p>
    </div>

    {% if current_user.is_admin %}
    <div class="mb-4 text-end">
        <a href="{{ url_for('admin.create_notice') }}" class="btn btn-primary">
            <i class="fas fa-plus-circle"></i> Post New Notice
        </a>
    </div>
    {% endif %}

    <div class="row">
        <div class="col-md-10 mx-auto">

            {% for notice in notices %}
            <div class="card mb-3 shadow-sm border-start border-5 {% if notice.priority == 'high'
%}border-danger{% else %}border-primary{% endif %}">
                <div class="card-body">
                    <div class="d-flex justify-content-between align-items-start">
                        <div>
```

`

```
                            <h5 class="card-title mb-1">
                                {% if notice.priority == 'high' %}
                                    <i class="fas fa-exclamation-circle text-danger me-2"></i>
                                {% endif %}
                                {{ notice.title }}
                            </h5>
                            <small class="text-muted">
                                <i class="fas fa-calendar-alt me-1"></i> {{ notice.posted_at.strftime('%d
%b, %Y') }}
                                &bull;
                                <span class="badge bg-light text-dark border">{{ notice.category
}}</span>
                            </small>
                        </div>

                        {% if current_user.is_admin %}
                        <form action="{{ url_for('admin.delete_notice', id=notice.id) }}"
method="POST">
                            <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
                            <button type="submit" class="btn btn-sm btn-outline-danger"
onclick="return confirm('Delete this notice?');">
                                <i class="fas fa-trash"></i>
                            </button>
                        </form>
                        {% endif %}
                    </div>

                    <p class="card-text mt-3">
                        {{ notice.content | truncate(200) }}
                    </p>

                    {% if notice.attachment %}
                    <a href="{{ url_for('static', filename='uploads/' + notice.attachment) }}"
class="btn btn-sm btn-link px-0 text-decoration-none">
                        <i class="fas fa-paperclip"></i> Download Attachment
                    </a>
                    {% endif %}
                </div>
            </div>
            {% else %}
                <div class="alert alert-secondary text-center py-5">
                    <h4>No active notices found.</h4>
                </div>
            {% endfor %}

        </div>
    </div>
</div>
{% endblock %}
```

`

13. Student Dashboard:



Hello, Rahul! 👋

Enrolled Courses
2

Upcoming Exams
2

Fee Due
$14000.0

Upcoming Exams

Exam	Subject	Date
Finals: CSE101	Computer Science Level 1	15 Feb, 2026
Finals: CSE102	Computer Science Level 2	28 Feb, 2026

Notice Board

Exam 04 Feb
Voluptatem excepturi odit animi ut sint.
Saepe esse dolorem impedit at hic quidem. Exercitationem nostru...

Exam 04 Feb
Sed dignissimos nisi dolorem iure.
Sint corrupti ea illo expedita velit maxime. Iusto aspernatur quidem...

General 04 Feb
Quos cum quibusdam dolorem quia praesentium.
Optio quaerat sint fugiat. Quod dignissimos sapiente reiciendis. Vol...

© College

Flask · Bootstrap · Animate.css

Code:

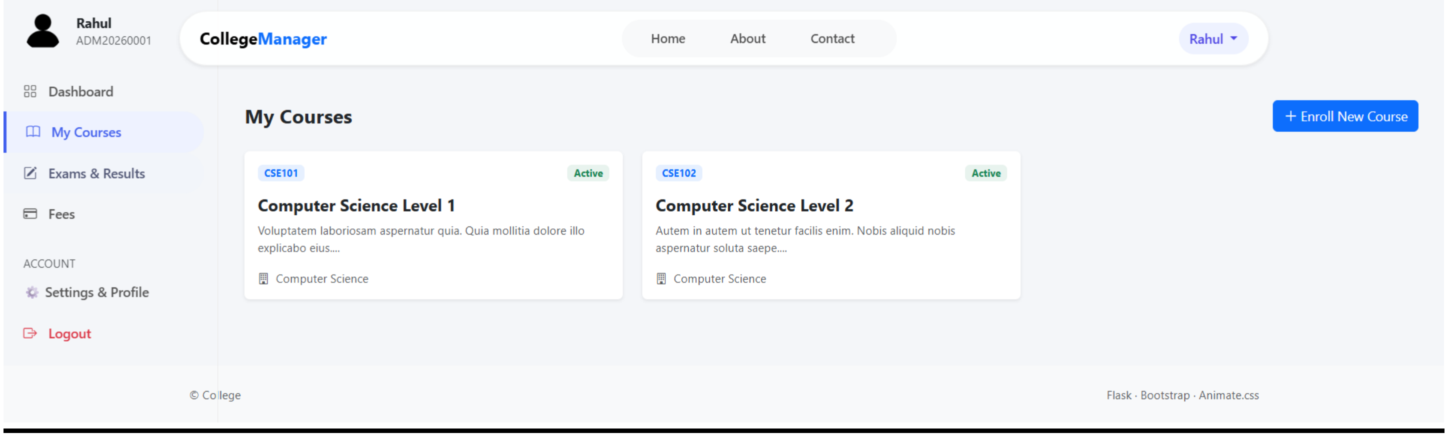
```
{% extends "student/base_student.html" %}

{% block title %}Student Dashboard - CMS{% endblock %}

{% block student_content %}
<div class="container-fluid py-4">

    <div class="d-flex justify-content-between align-items-center mb-4">
        <div>
            <h2 class="fw-bold">Welcome, {{ current_user.name }}</h2>
            <p class="text-muted">Roll No: {{ student.roll_no }} | Sem: {{ student.semester
}}</p>
        </div>
        <span class="badge bg-primary fs-6">{{ student.department }} Department</span>
    </div>

    <div class="row g-3 mb-4">

        <div class="col-md-4">
            <div class="card text-white bg-info h-100 shadow-sm">
                <div class="card-body d-flex align-items-center">
                    <i class="fas fa-user-clock fa-3x opacity-50 me-3"></i>
                    <div>
                        <h5>Attendance</h5>
                        <h2 class="mb-0">{{ student.attendance_percentage }}%</h2>
                    </div>
```

`

```
                </div>
            </div>
        </div>

        <div class="col-md-4">
            <div class="card text-white bg-success h-100 shadow-sm">
                <div class="card-body d-flex align-items-center">
                    <i class="fas fa-chart-line fa-3x opacity-50 me-3"></i>
                    <div>
                        <h5>Current CGPA</h5>
                        <h2 class="mb-0">{{ student.cgpa }}</h2>
                    </div>
                </div>
            </div>
        </div>

        <div class="col-md-4">
            <div class="card text-white {% if student.fee_due > 0 %}bg-danger{% else %}bg-secondary{% endif %} h-100 shadow-sm">
                <div class="card-body d-flex align-items-center">
                    <i class="fas fa-wallet fa-3x opacity-50 me-3"></i>
                    <div>
                        <h5>Fee Status</h5>
                        <h3 class="mb-0">
                            {% if student.fee_due > 0 %}
                                Due: ₹{{ student.fee_due }}
                            {% else %}
                                Paid
                            {% endif %}
                        </h3>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <div class="card shadow-sm">
        <div class="card-header bg-white">
            <h5 class="mb-0"><i class="fas fa-bullhorn text-warning"></i> Recent Notices</h5>
        </div>
        <div class="list-group list-group-flush">
            {% for notice in notices[:3] %}
            <a href="{{ url_for('student.view_notice', id=notice.id) }}" class="list-group-item list-group-item-action">
                <div class="d-flex w-100 justify-content-between">
                    <h6 class="mb-1">{{ notice.title }}</h6>
                    <small class="text-muted">{{ notice.date }}</small>
                </div>
                <p class="mb-1 small text-muted">{{ notice.content | truncate(80) }}</p>
            </a>
            {% endfor %}
        </div>
    </div>

</div>
{% endblock %}
```

`

14. Student Courses:



My Courses

CSE101 Active

Computer Science Level 1

Voluptatem laboriosam aspernatur quia. Quia mollitia dolore illo explicabo eius....

Computer Science

CSE102 Active

Computer Science Level 2

Autem in autem ut tenetur facilis enim. Nobis aliquid nobis aspernatur soluta saepe....

Computer Science

Code:

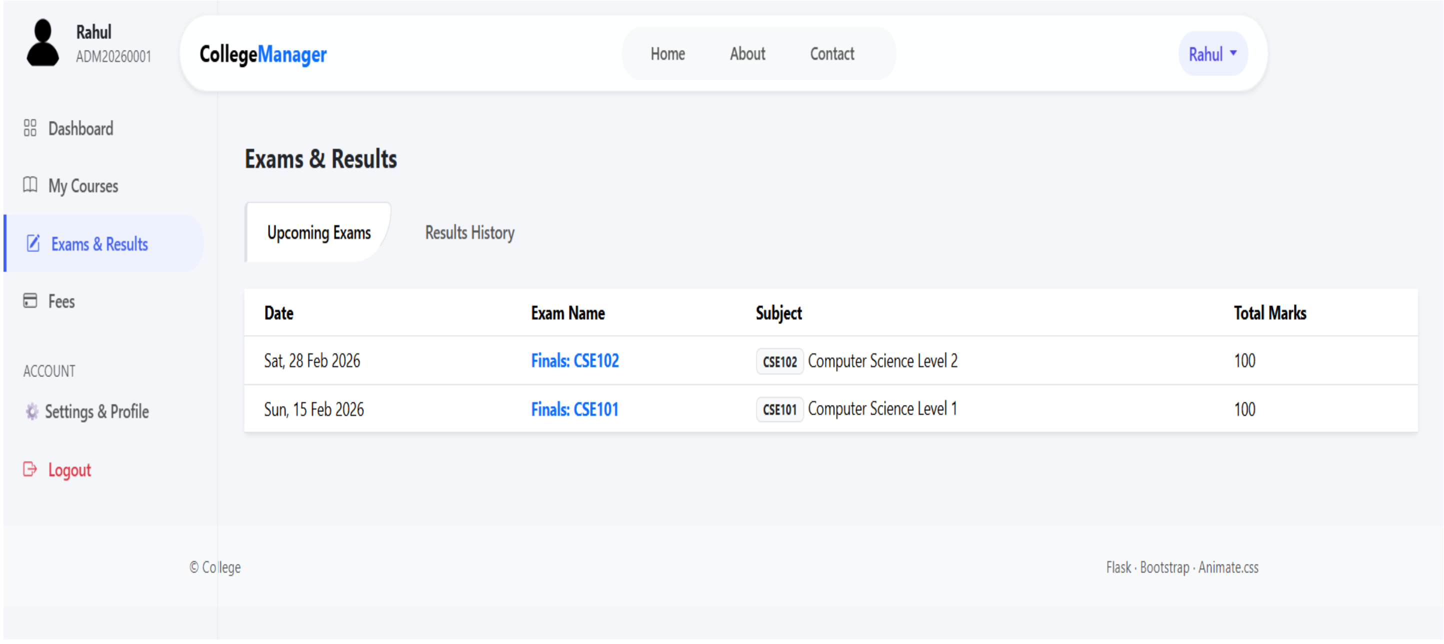
```
{% extends "student/base_student.html" %}

{% block title %}My Courses - CMS{% endblock %}

{% block student_content %}
<div class="container-fluid py-4">

    <h3 class="mb-4"><i class="fas fa-book-open"></i> Enrolled Courses</h3>

    <div class="card shadow-sm border-0">
        <div class="card-body p-0">
            <div class="table-responsive">
                <table class="table table-hover align-middle mb-0">
                    <thead class="bg-light">
                        <tr>
                            <th>Course Code</th>
                            <th>Subject Name</th>
                            <th>Credits</th>
                            <th>Faculty</th>
                            <th>Status</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for enrollment in enrollments %}
                        <tr>
                            <td><span class="badge bg-secondary">{{ enrollment.course.code
}}</span></td>
                            <td class="fw-bold">{{ enrollment.course.name }}</td>
                            <td>{{ enrollment.course.credits }}</td>
                            <td>
                                {% if enrollment.course.faculty %}
                                    <div class="d-flex align-items-center">
                                        <img src="{{ enrollment.course.faculty.avatar }}" class="rounded-
circle me-2" width="25">
                                        {{ enrollment.course.faculty.name }}
```

`

```
                                    </div>
                                {% else %}
                                    <span class="text-muted">--</span>
                                {% endif %}
                            </td>
                            <td>
                                <span class="badge bg-success">Enrolled</span>
                            </td>
                        </tr>
                        {% else %}
                        <tr>
                            <td colspan="5" class="text-center py-4">
                                No courses found for the current semester.
                            </td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
    </div>

</div>
{% endblock %}
```

`

15. Student Exams and Results:



Exams & Results

Date	Exam Name	Subject	Total Marks
Sat, 28 Feb 2026	Finals: CSE102	CSE102 Computer Science Level 2	100
Sun, 15 Feb 2026	Finals: CSE101	CSE101 Computer Science Level 1	100

Code:

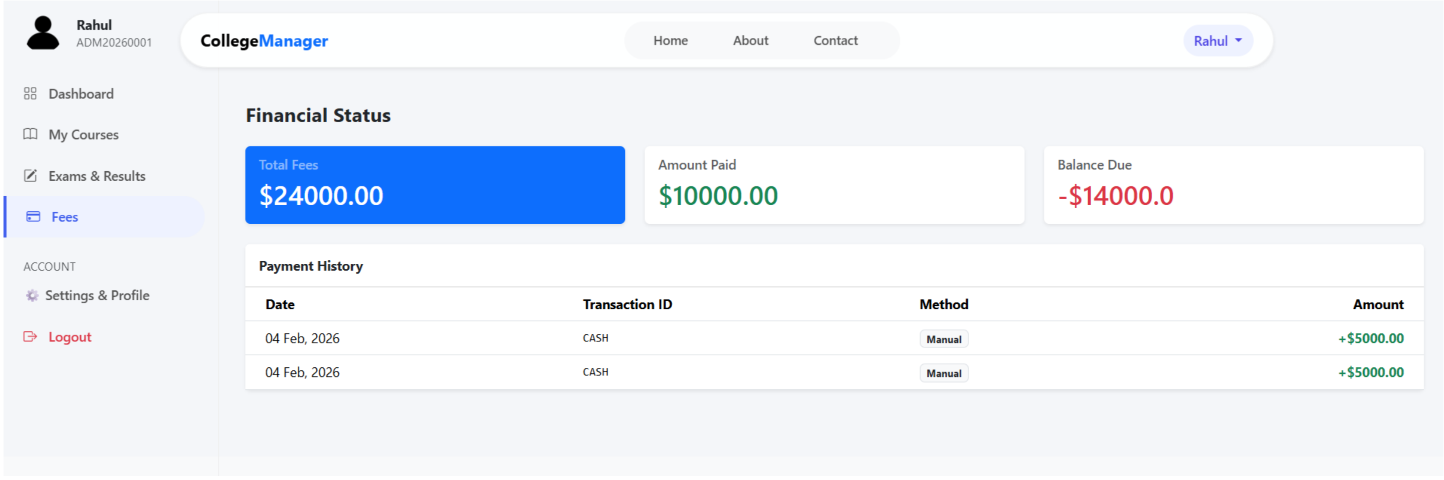
```
{% extends "student/base_student.html" %}

{% block title %}Exams & Results - CMS{% endblock %}

{% block student_content %}
<div class="container-fluid py-4">

    <div class="card mb-5 shadow-sm">
        <div class="card-header bg-warning text-dark">
            <h5 class="mb-0"><i class="fas fa-calendar-alt"></i> Upcoming Exams</h5>
        </div>
        <div class="card-body p-0">
            <table class="table table-striped mb-0">
                <thead>
                    <tr>
                        <th>Date</th>
                        <th>Time</th>
                        <th>Subject</th>
                        <th>Room No</th>
                    </tr>
                </thead>
                <tbody>
                    {% for exam in upcoming_exams %}
                    <tr>
                        <td>{{ exam.date }}</td>
                        <td>{{ exam.time }}</td>
                        <td class="fw-bold">{{ exam.course_name }}</td>
```

`

```
                        <td><span class="badge bg-dark">{{ exam.room_no }}</span></td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>

    <div class="card shadow-sm">
        <div class="card-header bg-success text-white">
            <h5 class="mb-0"><i class="fas fa-trophy"></i> Published Results</h5>
        </div>
        <div class="card-body p-0">
            <div class="table-responsive">
                <table class="table table-bordered align-middle mb-0 text-center">
                    <thead class="bg-light">
                        <tr>
                            <th>Subject</th>
                            <th>Credits</th>
                            <th>Marks Obtained</th>
                            <th>Grade</th>
                            <th>Result</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for result in results %}
                        <tr>
                            <td class="text-start">{{ result.course.name }}</td>
                            <td>{{ result.course.credits }}</td>
                            <td>
                                <strong>{{ result.marks_obtained }}</strong> / {{ result.total_marks }}
                            </td>
                            <td><span class="fw-bold">{{ result.grade }}</span></td>
                            <td>
                                {% if result.is_passed %}
                                    <span class="badge bg-success">PASS</span>
                                {% else %}
                                    <span class="badge bg-danger">FAIL</span>
                                {% endif %}
                            </td>
                        </tr>
                        {% endfor %}

                        <tr class="table-active fw-bold">
                            <td colspan="2" class="text-end">Semester Performance:</td>
                            <td colspan="3" class="text-start">
                                SGPA: {{ sgpa }} | Attendance: {{ attendance }}%
                            </td>
                        </tr>
                    </tbody>
                </table>
            </div>
        </div>
    </div>

</div>
{% endblock %}
```

`

16. Student Fees



Financial Status

Total Fees
$24000.00

Amount Paid
$10000.00

Balance Due
-$14000.0

Payment History

Date	Transaction ID	Method	Amount
04 Feb, 2026	CASH	Manual	+$5000.00
04 Feb, 2026	CASH	Manual	+$5000.00

Code:

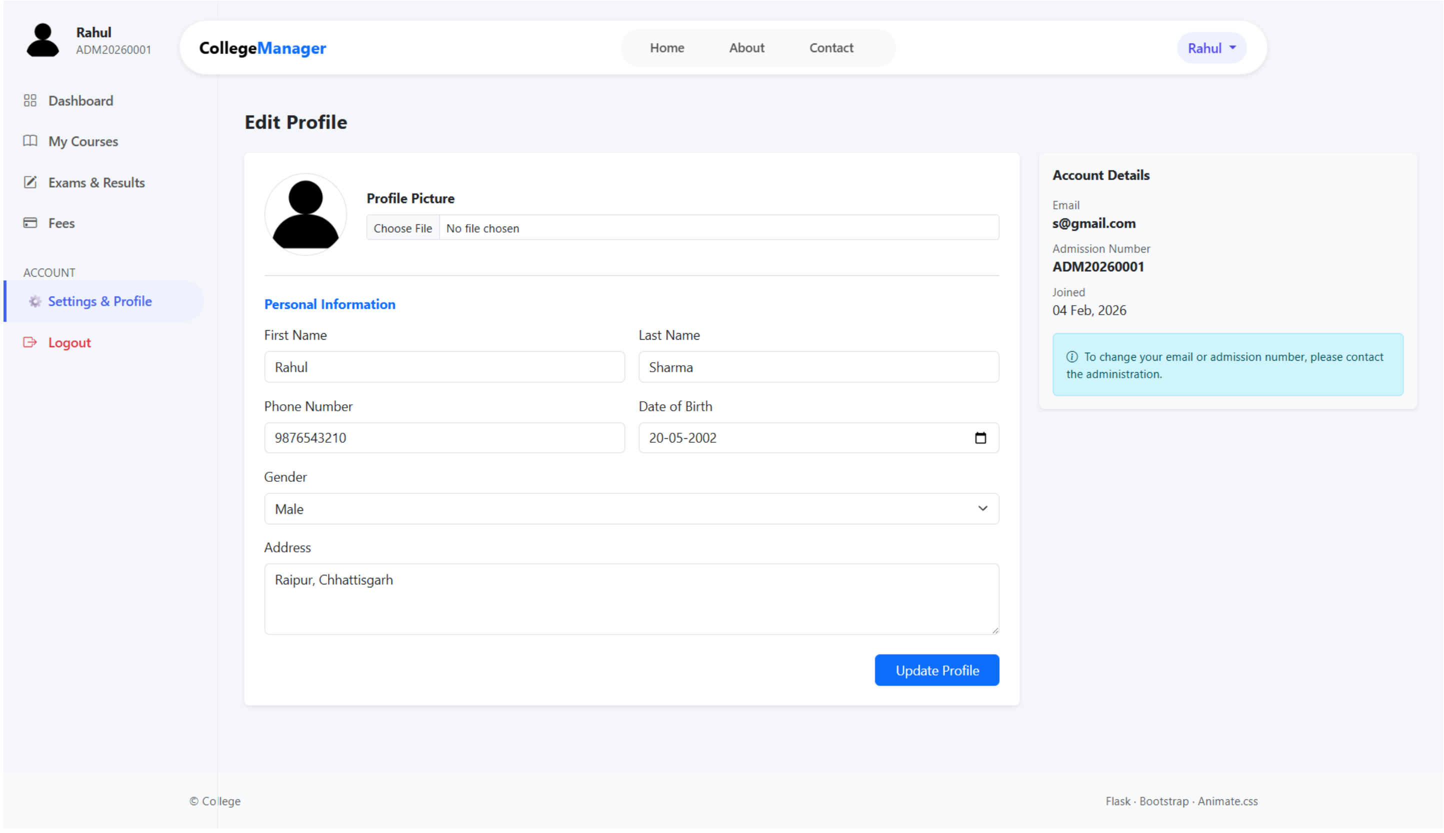
```
{% extends "student/base_student.html" %}

{% block title %}My Fees - CMS{% endblock %}

{% block student_content %}
<div class="container-fluid py-4">

    <h3 class="mb-4"><i class="fas fa-file-invoice-dollar"></i> Fee Details</h3>

    <div class="row mb-4">
        <div class="col-md-6">
            <div class="card shadow-sm border-start border-5 border-primary">
                <div class="card-body">
                    <h5 class="card-title text-muted">Total Semester Fee</h5>
                    <h2 class="mb-0">₹{{ fee_structure.total_amount }}</h2>
                </div>
            </div>
        </div>
        <div class="col-md-6">
            <div class="card shadow-sm border-start border-5 {% if fee_structure.due_amount > 0 %}border-danger{% else %}border-success{% endif %}">
                <div class="card-body">
                    <h5 class="card-title text-muted">Pending Due</h5>
                    <h2 class="mb-0">₹{{ fee_structure.due_amount }}</h2>
                    {% if fee_structure.due_amount > 0 %}
                        <button class="btn btn-sm btn-danger mt-2">
                            <i class="fas fa-credit-card"></i> Pay Online
                        </button>
                    {% else %}
                        <span class="badge bg-success mt-2">Paid in Full</span>
                    {% endif %}
                </div>
```

`

```
            </div>
        </div>
    </div>

    <div class="card shadow-sm">
        <div class="card-header bg-light">
            <h5 class="mb-0"><i class="fas fa-history"></i> Transaction History</h5>
        </div>
        <div class="card-body p-0">
            <div class="table-responsive">
                <table class="table table-hover mb-0">
                    <thead>
                        <tr>
                            <th>Date</th>
                            <th>Transaction ID</th>
                            <th>Amount Paid</th>
                            <th>Method</th>
                            <th>Status</th>
                            <th>Receipt</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for payment in payments %}
                        <tr>
                            <td>{{ payment.date.strftime('%d %b, %Y') }}</td>
                            <td><code>{{ payment.transaction_id }}</code></td>
                            <td class="fw-bold">₹{{ payment.amount }}</td>
                            <td>{{ payment.method }}</td> <td>
                                <span class="badge bg-success">Success</span>
                            </td>
                            <td>
                                <a href="{{ url_for('student.download_receipt', id=payment.id) }}"
class="btn btn-sm btn-outline-dark">
                                    <i class="fas fa-download"></i> PDF
                                </a>
                            </td>
                        </tr>
                        {% else %}
                        <tr>
                            <td colspan="6" class="text-center py-3">No payment records found.</td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div>
        </div>
    </div>

</div>
{% endblock %}
```

`

17. Student Settings & Profile:



Edit Profile

Profile Picture

No file chosen

Personal Information

First Name: Rahul

Last Name: Sharma

Phone Number: 9876543210

Date of Birth: 20-05-2002

Gender: Male

Address: Raipur, Chhattisgarh

Account Details

Email
s@gmail.com

Admission Number
ADM20260001

Joined
04 Feb, 2026

ⓘ To change your email or admission number, please contact the administration.

Code:

```
{% extends "base.html" %}

{% block title %}Profile Settings - CMS{% endblock %}

{% block content %}
<div class="container py-5">

    <div class="row">
        <div class="col-md-4 mb-4">
            <div class="card shadow-sm text-center p-4">
                <div class="mb-3 position-relative d-inline-block">
                    <img src="{{ url_for('static', filename='uploads/avatars/' + current_user.avatar)
}}"
                         class="rounded-circle img-thumbnail"
                         style="width: 150px; height: 150px; object-fit: cover;">

                    <form action="{{ url_for('user.update_avatar') }}" method="POST"
enctype="multipart/form-data" class="mt-3">
                        <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">
                        <label class="btn btn-sm btn-outline-primary">
                            <i class="fas fa-camera"></i> Change Photo
```

`

```
                            <input type="file" name="avatar" class="d-none"
onchange="this.form.submit()">
                        </label>
                    </form>
                </div>

                <h4>{{ current_user.name }}</h4>
                <p class="text-muted mb-1">{{ current_user.role | capitalize }}</p>
                <small class="text-muted">Joined: {{ current_user.created_at.strftime('%Y')
}}</small>
            </div>
        </div>

        <div class="col-md-8">
            <div class="card shadow-sm">
                <div class="card-header bg-white">
                    <ul class="nav nav-tabs card-header-tabs" id="profileTabs">
                        <li class="nav-item">
                            <a class="nav-link active" data-bs-toggle="tab" href="#details">Personal
Details</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" data-bs-toggle="tab" href="#security">Security</a>
                        </li>
                    </ul>
                </div>

                <div class="card-body tab-content">

                    <div class="tab-pane fade show active" id="details">
                        <form action="{{ url_for('user.update_profile') }}" method="POST">
                            <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">

                            <div class="row mb-3">
                                <div class="col-md-6">
                                    <label class="form-label">Full Name</label>
                                    <input type="text" class="form-control" name="name" value="{{
current_user.name }}" readonly>
                                    <small class="text-muted">Contact Admin to change name.</small>
                                </div>
                                <div class="col-md-6">
                                    <label class="form-label">Email Address</label>
                                    <input type="email" class="form-control" name="email" value="{{
current_user.email }}" readonly>
                                </div>
                            </div>

                            <div class="mb-3">
                                <label class="form-label">Phone Number</label>
                                <input type="text" class="form-control" name="phone" value="{{
current_user.phone }}">
                            </div>

                            <div class="mb-3">
                                <label class="form-label">Address</label>
                                <textarea class="form-control" name="address" rows="3">{{
current_user.address }}</textarea>
                            </div>
```

`

```
                            <button type="submit" class="btn btn-primary">
                                <i class="fas fa-save"></i> Save Changes
                            </button>
                        </form>
                    </div>

                    <div class="tab-pane fade" id="security">
                        <form action="{{ url_for('user.change_password') }}" method="POST">
                            <input type="hidden" name="csrf_token" value="{{ csrf_token() }}">

                            <div class="mb-3">
                                <label class="form-label">Current Password</label>
                                <input type="password" class="form-control" name="current_password"
required>
                            </div>

                            <div class="row mb-3">
                                <div class="col-md-6">
                                    <label class="form-label">New Password</label>
                                    <input type="password" class="form-control" name="new_password"
required>
                                </div>
                                <div class="col-md-6">
                                    <label class="form-label">Confirm New Password</label>
                                    <input type="password" class="form-control"
name="confirm_password" required>
                                </div>
                            </div>

                            <button type="submit" class="btn btn-danger">
                                <i class="fas fa-key"></i> Update Password
                            </button>
                        </form>
                    </div>

                </div>
            </div>
        </div>
    </div>
</div>
{% endblock %}
```

`

LIMITATIONS

- **Lack of Biometric/RFID Integration (Manual Attendance):** The current system relies entirely on manual data entry for logging student attendance. Faculty must physically mark present/absent for every class, which can be time-consuming for large batches and prone to human error. Unlike hardware-integrated systems, this project does not currently connect to biometric fingerprint scanners or RFID card readers for automated attendance tracking.
- **No Multi-Branch/University Support:** The platform currently operates as a standalone instance for a single college campus. It does not support multi-tenant architecture to manage multiple branches or affiliated colleges under a single university umbrella. This limits usability for large educational trusts that require centralized data aggregation across different geographical locations.
- **Static Timetable Management:** The "Class Schedule" module displays timetables based on static, pre-defined entries managed by the admin. It lacks a real-time conflict detection algorithm or integration with faculty availability calendars. Consequently, rescheduling a class requires manual coordination, and the system does not automatically suggest available slots or rooms.

FUTURE SCOPE OF THE PROJECT

- **AI-Driven Academic Insights:** The system can integrate artificial intelligence (AI) and machine learning algorithms to analyze student performance trends over semesters. AI-driven insights could automatically detect "at-risk students" (e.g., those with declining attendance or grades) and suggest personalized remedial classes or interventions to improve retention rates.
- **Biometric & RFID Hardware Integration:** Implementing IoT (Internet of Things) integration would allow the system to communicate with physical biometric devices. This would enable automated attendance marking as soon as a student enters the classroom or library, significantly reducing manual administrative effort and eliminating "proxy" attendance.

`

- **Online Payment Gateway Integration:** Collaborating with Payment Gateways (like Razorpay, Stripe, or PayPal) would enable secure, real-time fee collection directly through the student dashboard. This would automate the fee reconciliation process, instantly generating digital receipts and updating the "Fee Status" to 'Paid' without manual admin intervention.
- **Alumni & Placement Portal:** The platform can extend its services to include an "Alumni Network" and "Placement Cell" module. This would allow former students to stay connected, view job postings, and mentor current students, fostering a strong community and improving the college's placement statistics.
- **Mobile Application Development:** While the current web dashboard is responsive, developing a dedicated mobile application (iOS/Android) using React Native or Flutter would provide a better native experience. Features like push notifications for urgent notices (e.g., "Exam Postponed") and offline result viewing would significantly enhance usability.

`

CONCLUSION

The **College Management System** project successfully addresses the key objectives of providing a comprehensive and user-friendly platform for educational administration. By offering specific tools for **student enrollment, result management, and digital communication** (via the Notice Board), the platform empowers institutions to take control of their academic workflow.

The scope of the project includes creating a secure system that isolates user data, with a focus on **Flask-based authentication**, **SQLAlchemy ORM** for relational data integrity, and a dynamic **JavaScript/Bootstrap frontend**. The Application Factory pattern ensures that the system is modular and maintainable, while the **MySQL database** allows for the efficient storage of thousands of student records and historical results. The project also adheres to security best practices, such as **password hashing** and **CSRF protection**, to safeguard sensitive personal and academic information.

In conclusion, this project provides an efficient, scalable, and secure solution for replacing manual registers and fragmented spreadsheets with a smart digital dashboard. It meets the platform’s goals by ensuring ease of use, instant data retrieval, and streamlined workflows for faculty and administration. With future enhancements like **hardware integration** and **AI analytics**, this system is poised to become a valuable asset for modern campus management.

`

BIBLIOGRAPHY

The Bibliography contains references to all the documents and resources that were referred to for the creation and successful completion of the project. It contains the names of the referred software engineering documents, Python/Flask documentation, and frontend library standards.

- **Book:** *Flask Web Development: Developing Web Applications with Python* by Miguel Grinberg.
- **Book:** *Python Crash Course* by Eric Matthes (Section on Web Apps).
- **Documentation:** Flask Official Documentation (https://flask.palletsprojects.com/).
- **Documentation:** Jinja2 Template Designer Documentation.
- **Resource:** https://getbootstrap.com (Bootstrap 5 Documentation).
- **Resource:** https://www.chartjs.org (Chart.js Documentation for Academic Data Visualization).
- **Resource:** https://sqlalchemy.org (SQLAlchemy ORM Tutorial).
- **Tool:** https://fontawesome.com (Icons for Dashboard).
- **Tool:** https://mermaid.live (For System Diagrams: ER, DFD).
- **Tutorials:** "Corey Schafer - Python Flask Tutorials" on YouTube.