PROJECT REPORT

Project Title

College Management System (CMS)

Submitted By

MD SHAKLAIN
Python Full-Stack Intern

Internship Organization

Inno Fortune IT Pvt. Ltd.

Under the Guidance of

Ms. Nikita Madam

Date

23-02-2026

1.Table of Contents

1.Abstract

The **College Management System (CMS)** is a web-based application developed to automate and digitize the academic and administrative operations of a college. The system is designed to provide a centralized platform for managing student records, faculty details, course information, attendance tracking, and examination results in an efficient and structured manner.

Traditional college management processes rely heavily on manual documentation and record-keeping, which often leads to data redundancy, human errors, time delays, and difficulties in generating accurate reports. The College Management System addresses these issues by implementing a secure and reliable digital solution that enhances operational efficiency and ensures data accuracy.

The system incorporates multiple integrated modules, including student management, teacher management, course management, attendance monitoring, and result processing. It follows a role-based access control mechanism where administrators manage the overall system, teachers handle academic records, and students can access their personal academic information.

The project is developed using modern web technologies such as Python and the Django framework for backend development, along with HTML, CSS, and JavaScript for the frontend interface. A relational database system is used to store and manage structured data effectively. The system follows industry-standard software development practices, ensuring scalability, security, and maintainability.

2. Introduction

In the modern digital era, educational institutions are increasingly adopting information technology solutions to improve administrative efficiency and academic management. Colleges and universities handle a large volume of data related to students, faculty members, courses, attendance, examinations, and results. Managing this data manually through paper-based systems or basic spreadsheets can be time-consuming, error-prone, and inefficient.

The **College Management System (CMS)** is developed as a comprehensive web-based solution to address these challenges. The system aims to computerize and streamline all major academic and administrative activities within a college environment. By integrating various functional modules into a unified platform, the CMS enables smooth coordination between administration, faculty, and students.

The primary goal of the system is to centralize data management and provide real-time access to information. With the help of this system, administrators can efficiently manage student admissions, faculty records, and course details. Teachers can maintain attendance records and upload examination marks, while students can easily access their academic information such as attendance status and results.

The system is designed using modern web technologies and follows a modular architecture, ensuring scalability and flexibility. It implements secure authentication mechanisms to protect sensitive data and maintain user privacy. The use of a relational database ensures structured data storage and integrity.

3.Scope of the Project

The College Management System (CMS) is designed to provide a centralized, automated, and secure platform for managing academic and administrative activities of an educational institution. The scope of this project defines the boundaries, features, and operational coverage of the system.

1. Academic Management Scope

The system will manage complete academic records including:

- Student registration and profile management
- Course creation and assignment
- Teacher allocation to courses
- Semester-wise subject management
- Result preparation and grade calculation
- Attendance tracking and monitoring

This ensures smooth handling of academic data in a structured format.

2. Administrative Management Scope

The system will help in automating administrative tasks such as:

- Maintaining student personal details
- Managing faculty records
- Generating attendance reports
- Generating academic performance reports
- Monitoring course enrolment

This reduces manual paperwork and improves operational efficiency.

3. Role-Based Access Scope

The system will support multiple user roles with different permissions:

Admin:

- Full access to all modules
- Add, update, delete student/teacher/course records
- Generate reports

Teacher:

- Mark attendance
- Enter student marks
- View assigned courses

Student:

- View personal details
- Check attendance
- View results

This ensures secure and structured access control.

4. Data Management Scope

- Centralized database storage
- Secure data handling
- Backup capability
- Data consistency using relational database
- Prevention of duplicate records

5. Reporting and Monitoring Scope

The system will provide:

- Student attendance reports
- Result summaries
- Course-wise performance analysis
- Dashboard statistics

This helps management in decision-making.

6. Technical Scope

The system will be:

- Web-based application
- Developed using Django framework
- Accessible via modern web browsers
- Designed using modular architecture
- Scalable for future enhancements

7. Future Expandable Scope

The system is designed in such a way that it can be expanded to include:

- Online examination system
- Fee payment integration
- SMS/Email notification system
- Mobile application version
- Library management module
- Hostel management module

4. Requirements

The requirements of the College Management System (CMS) define the functional and non-functional needs necessary for the successful implementation and operation of the system. These requirements ensure that the system performs efficiently, securely, and reliably.

4.1 Functional Requirements

Functional requirements describe the specific features and operations the system must perform.

4.1.1 User Authentication

- The system must provide secure login functionality.
- Users must log in using valid credentials (username and password).
- The system must support role-based access control (Admin, Teacher, Student).
- Unauthorized users must not access restricted data.

4.1.2 Student Management

- The Admin must be able to add, update, delete, and view student records.
- The system must store student details such as name, email, phone number, admission date, and assigned course.
- The system must maintain unique student identification numbers.

4.1.3 Teacher Management

- The Admin must be able to manage teacher records.
- Teacher details must include name, email, department, and assigned courses.

4.1.4 Course Management

- The admin must be able to create and manage courses.
- Each course must be assigned to a teacher.
- Course details must include course name and duration.

4.1.5 Attendance Management

- Teachers must be able to mark attendance for students.
- Attendance records must include date and status (Present/Absent).
- Students must be able to view their attendance records.

4.1.6 Result Management

- Teachers must be able to upload student marks.
- The system must calculate grades if required.
- Students must be able to view their results securely.

4.1.7 Report Generation

- The system must generate reports for students, attendance, and results.
- Reports should be viewable and printable.

4.2 Non-Functional Requirements

Non-functional requirements describe the quality attributes of the system.

4.2.1 Security

- The system must use authentication and authorization mechanisms.
- User passwords must be securely stored.
- Access to sensitive data must be restricted based on roles.

4.2.2 Performance

- The system must respond quickly to user requests.
- Data retrieval operations should be efficient.

4.2.3 Reliability

- The system must maintain data integrity.
- The database must store accurate and consistent data.

4.2.4 Scalability

- The system must support increasing numbers of users and records.
- The architecture should allow future enhancements.

4.2.5 Usability

- The interface must be simple and user-friendly.
- Users should easily navigate between modules.

4.2.6 Maintainability

- The code should follow modular design principles.
- Future developers should be able to update the system easily.

4.3 Hardware Requirements

- Computer/Laptop
- Minimum 4GB RAM
- Internet Connection

4.4 Software Requirements

- Operating System: Windows / Linux / macOS
- Python
- Django Framework
- Web Browser (Chrome, Firefox, etc.)
- Database: SQLite / MySQL
- Git (for version control)
- The interface must be simple and user-friendly.
- Users should easily navigate between modules.

4.2.6 Maintainability

- The code should follow modular design principles.
- Future developers should be able to update the system easily.

4.3 Hardware Requirements

- Computer/Laptop
- Minimum 4GB RAM
- Internet Connection

4.4 Software Requirements

- Operating System: Windows / Linux / macOS
- Python
- Django Framework
- Web Browser (Chrome, Firefox, etc.)
- Database: SQLite / MySQL
- Git (for version control)

5. User Roles

In the College Management System (CMS), different user roles are defined to ensure secure and structured access to the system. Each role has specific permissions and responsibilities.

5.1 Admin

The Admin has full control over the system.

Responsibilities:

- Add, update, and delete student records
- Add and manage teacher records
- Create and manage courses
- Assign teachers to courses
- View attendance and result reports
- Manage user accounts

Admin acts as the system controller.

5.2 Teacher

The Teacher role is responsible for academic activities.

Responsibilities:

- View assigned courses
- Mark student attendance
- Enter student marks
- Update academic records
- View student details (limited access)

Teachers can only access data related to their assigned courses.

5.3 Student

The Student role has limited access.

Responsibilities:

- View personal profile
- Check attendance records
- View results and grades
- View enrolled courses

Students cannot modify system data.

6. Module Finalization

After completing requirement analysis and user role identification, the system modules were finalized. Each module was designed to perform specific tasks while maintaining proper integration with other modules. The modular approach ensures better organization, maintainability, and scalability of the system.

The following core modules were finalized for the College Management System:

6.1 Authentication Module

The Authentication Module ensures secure access to the system.

Features:

- User login with username and password
- Role-based access control (Admin, Teacher, Student)
- Secure password storage (hashed passwords)
- Logout functionality
- Session management

This module restricts unauthorized access and ensures that users can only access features assigned to their role.

6.2 Student Management Module

This module manages all student-related data.

Features:

- Add new student records
- Update student details
- Delete student records
- View student list
- Search student by ID or name
- Store student enrolment information

The module ensures centralized storage of student academic and personal data.

6.3 Teacher Management Module

The Teacher Management Module handles faculty records.

Features:

- Add teacher details
- Update teacher information
- Delete teacher records
- Assign teachers to courses
- View teacher profile

This module maintains organized faculty information within the system.

6.4 Course Management Module

This module manages academic courses offered by the college.

Features:

- Create new courses
- Update course details
- Delete courses
- Assign teacher to course
- Enrol students in courses
- View course list

It ensures structured academic planning and course allocation.

6.5 Attendance Management Module

This module records and monitors student attendance.

Features:

- Mark attendance (Present/Absent)
- Store attendance with date
- View attendance history
- Generate attendance reports
- Course-wise attendance tracking

This module improves monitoring of student participation.

6.6 Result Management Module

The Result Management Module handles examination and grading processes.

Features:

- Enter student marks
- Update marks
- Calculate total and grade automatically
- Generate result report
- View semester-wise performance

This module ensures accurate and transparent academic evaluation.

6.7 Dashboard Module

The Dashboard provides a summarized view of system data.

Features:

- Role-based dashboard display
- Display total students, teachers, courses
- Show attendance summary
- Quick navigation to modules

The dashboard improves user experience and accessibility.

7. Use Case Diagram

7.1 Actors

👤 **Admin**

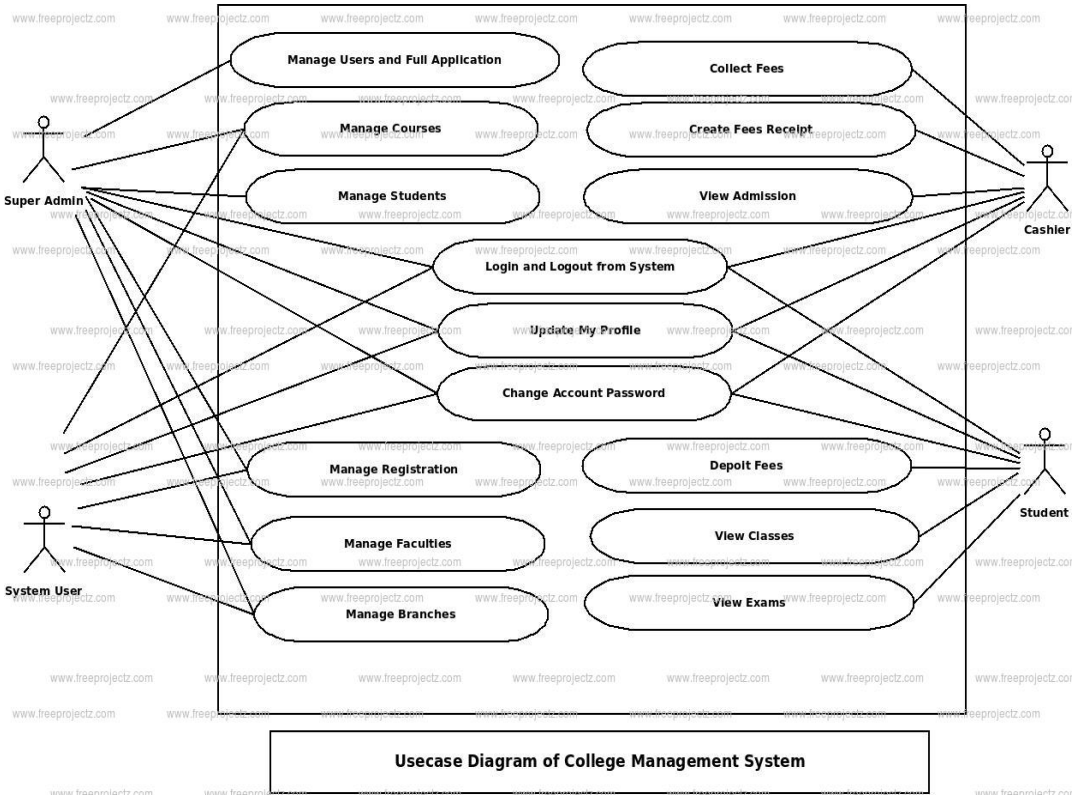
👩‍🏫 **Teacher**

👨‍🎓 **Student**

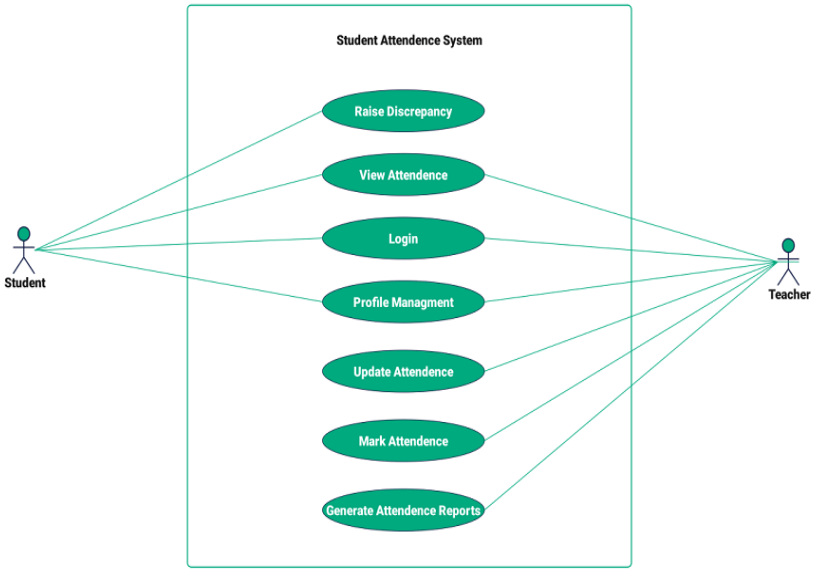
7.2 Admin Use Cases

- Login
- Manage Students
- Manage Teachers
- Manage Staff
- Manage Courses
- Manage Fees Structure
- Manage Exams
- Generate Results
- View Attendance Reports
- Generate Reports
- System Management

Admin has full control over academic and administrative operations.

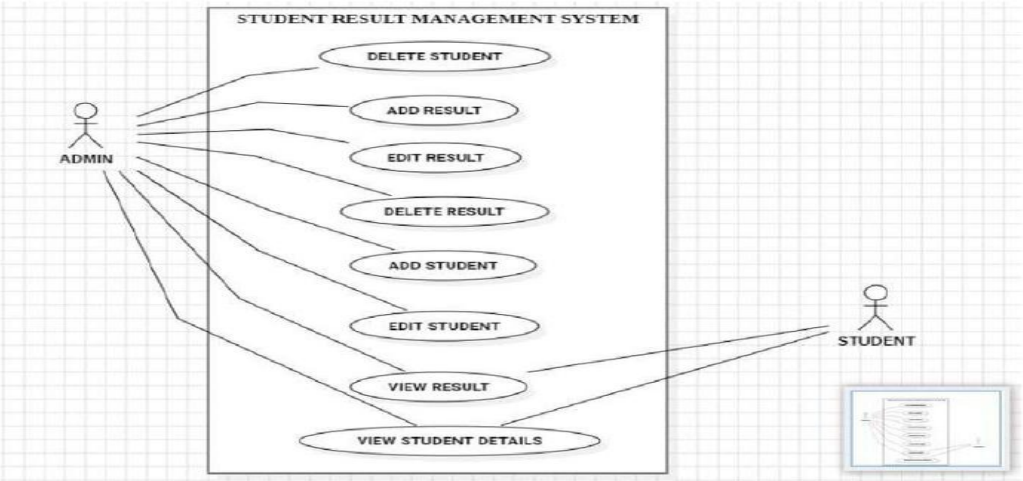


Use Case Diagram for Collage Management System



Use Case Diagram for Student Attendance

Use Case Diagram Of Student Result Management system



Use Case Diagram for Student Result

7.3 Student Use Cases

- Login
- View Profile
- View Attendance
- View Results
- View Exam Schedule
- View Fee Details
- Enrol in Courses

Students have limited (view-based) access.

7.4 Staff Use Cases

- Login
- Manage Fee Records
- Maintain Student Documents
- Assist in Exam Management
- Update Administrative Records

Staff supports administrative activities.

7.5 Fees Management Module

- Define fee structure
- Collect fees
- Generate receipts
- Track pending payments

7.6 Exam Management Module

- Schedule exams
- Assign exam dates
- Manage question papers
- Generate result sheets

7.7 Purpose

- Defines system boundaries
- Clarifies role-based responsibilities
- Helps in requirement validation
- Supports structured development

8.Module Mapping

Module Mapping connects user roles with system modules and their permissions.

Role vs Module Mapping Table

Module	Admin	Teacher	Student
Authentication	✅	✅	✅
Student Management	Full Access	View Only	View Own
Teacher Management	Full Access	View Own	No Access
Course Management	Full Access	View Assigned	View Enrolled
Attendance Module	View Reports	Mark & View	View Own
Result Module	View Reports	Enter & update	View Own
Dashboard	Full View	Limited View	Limited View

Explanation

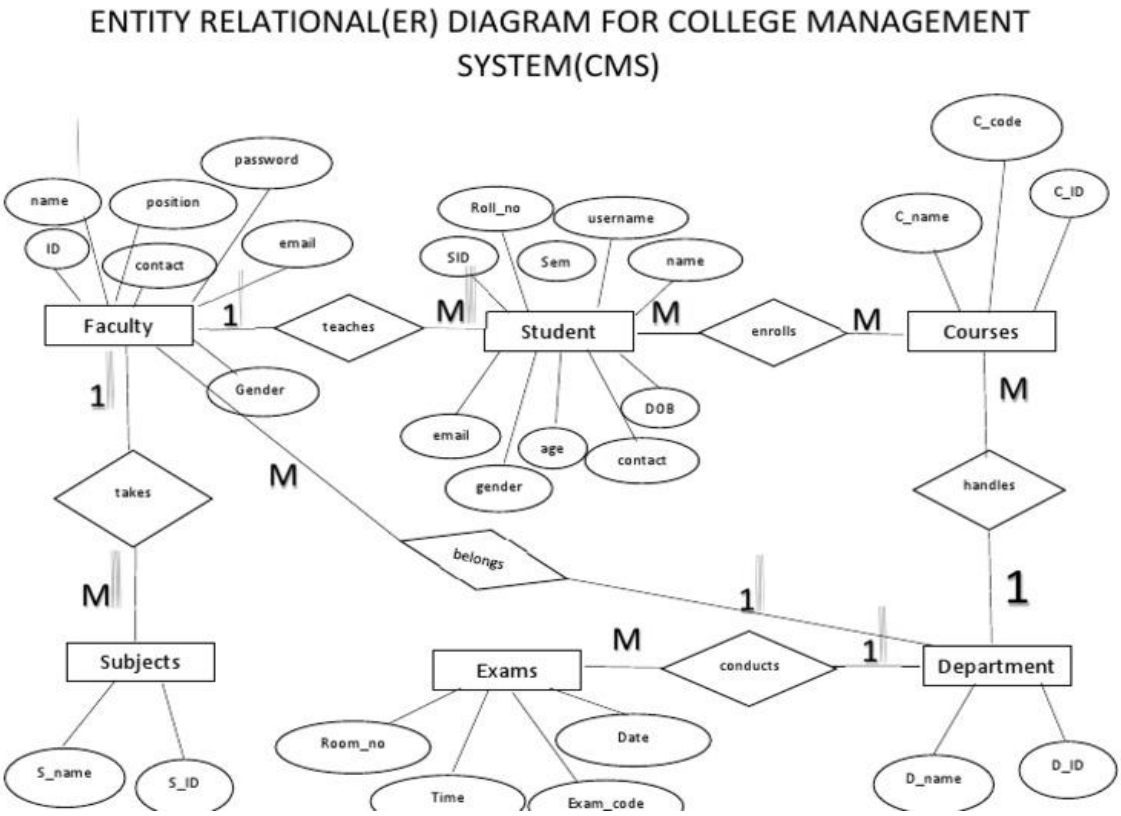
- **Admin** has full system control.
- **Teacher** has academic-related access.
- **Student** has read-only access to personal data.

This mapping ensures:

- Secure role-based access
- Clear responsibility distribution
- Structured system control
- Reduced data misuse

9. ER Diagram

An **Entity Relationship (ER) Diagram** is a graphical representation of entities, their attributes, and the relationships between them in a database system. It is used to design and visualize the structure of a database before implementation.



ER Diagram For CMS

- ## Entities and Attributes

9.1 Student

- **student_id** (Primary Key)
- name
- email
- phone
- address
- enrollment_date

9.2 Teacher

- **teacher_id** (Primary Key)
- name
- email
- phone
- qualification

9.3 Course

- **course_id** (Primary Key)
- course_name
- course_code
- credits
- teacher_id (Foreign Key)

9.4 Attendance

- **attendance_id** (Primary Key)
- student_id (Foreign Key)
- course_id (Foreign Key)
- date
- status (Present/Absent)

9.5 Exam

- **exam_id** (Primary Key)
- exam_name
- exam_date
- course_id (Foreign Key)

9.6 Result

- **result_id** (Primary Key)
- student_id (Foreign Key)
- exam_id (Foreign Key)
- marks
- grade

9.7 Fees

- **fee_id** (Primary Key)
- student_id (Foreign Key)
- total_amount
- paid_amount
- due_amount
- payment_date

10. Database Schema (Tables & Fields)

The database schema defines the structure of tables, their fields, primary keys, and foreign key relationships used in the College Management System.

10.1 Student Table

Field Name	Data Type	Description
student_id (PK)	Integer	Unique ID of student
name	Varchar (100)	Student full name
email	Varchar (100)	Student email
phone	Varchar (15)	Contact number
address	Text	Residential address
enrollment_date	Date	Admission date

10.2 Teacher Table

Field Name	Data Type	Description
teacher_id (PK)	Integer	Unique ID of teacher
name	Varchar (100)	Teacher name
email	Varchar (100)	Teacher email
phone	Varchar (15)	Contact number
qualification	Varchar (100)	Academic qualification

10.3 Course Table

Field Name	Data Type	Description
course_id (PK)	Integer	Unique course ID
course_name	Varchar(100)	Course name
course_code	Varchar(20)	Unique course code
credits	Integer	Course credits
teacher_id (FK)	Integer	Linked teacher

10.4 Attendance Table

Field Name	Data Type	Description
attendance_id (PK)	Integer	Unique attendance ID
student_id (FK)	Integer	Linked student
course_id (FK)	Integer	Linked course
date	Date	Attendance date
status	Varchar (10)	Present/Absent

10.5 Exam Table

Field Name	Data Type	Description
exam_id (PK)	Integer	Unique exam ID
exam_name	Varchar(100)	Exam title
exam_date	Date	Exam date
course_id (FK)	Integer	Related course

10.6Result Table

Field Name	Data Type	Description
result_id (PK)	Integer	Unique result ID
student_id (FK)	Integer	Linked student
exam_id (FK)	Integer	Linked exam
marks	Integer	Marks obtained
grade	Varchar(5)	Grade

10.7 Fees Table

Field Name	Data Type	Description
fee_id (PK)	Integer	Unique fee ID
student_id (FK)	Integer	Linked student
total_amount	Decimal (10,2)	Total fee
paid_amount	Decimal (10,2)	Paid amount
due_amount	Decimal (10,2)	Remaining amount
payment_date	Date	Payment date

Key Relationships

- Student → Attendance (1:M)
- Student → Result (1:M)
- Student → Fees (1:M)
- Teacher → Course (1:M)
- Course → Exam (1:M)
- Course → Attendance (1:M)