SUNWAY COLLEGE KATHMANDU

CREATING AI LEADERS

IN ACADEMIC PARTNERSHIP WITH

BIRMINGHAM CITY University

**Report Title: ERM Report of CMS**

**Student Name: Bishwas Chaudhary**

**Date: Dec 29, 2025**

**Student ID: 25123796**

**Module code: DIG5127**

**Module Title**
**Database and Web Application Development**

# COLLEGE MANAGEMENT SYSTEM

## 1. Project Title:

Web-Based College Management System (CMS) Using PHP and MySQL

## 2. Introduction:

The current education system makes it clear that managing college operations manually isn't feasible. It takes unnecessary time, effort, and leads to mistakes. Colleges and other educational institutions are already trying to avoid dealing with large amounts of data that involve students, faculty, programs, attendance, exams, and finances. The old ways of managing data with paper systems and disjointed software are increasingly insecure and uncentralized.

The **College Management System (CMS)** is one such way of managing data. The system is an application that helps colleges automate and integrate their core administrative systems. Moreover, the CMS creates an administrative system where other faculty and role-based students can manage data. This will provide data in an efficient way, erasing redundancy, and increasing productivity in the institution.

Using the tools CMS created a project with **PHP** backend logic, **MySQL** to manage the databases, and a front end of **HTML, CSS and Javascript**. The project is designed as a complete project that can scale for real world use.

## 3. Problem Statement

Most colleges face several challenges due to manual or semi-automated systems, including:

- Record keeping of student and staff data is done manually
- Tracking attendance and academic performance is done manually
- Errors in examination marks and fee management
- Information is not available in a centralized manner and not accessible in real time
- Data security is inadequate, as is access control

All of the above lead to heightened levels of administrative work, and a decrease in accuracy and negatively affects the overall quality of decision making.

# 4. Objectives of the Project

The goals of the College Management System include:

- The digitization of academic and administrative systems
- The development of an organized and centralized database
- The improvement of accuracy in tracking attendance and managing exams and fees
- The provision of role-based access controls
- The optimization of data security, consistency, and integrity

# 5. Why This Project

The following reasons were taken into consideration when selecting this project:

- It addresses real-world problems faced by educational institutions
- It shows good comprehension of database architecture and normalization
- It is good for assessing academic performance
- It embodies concepts of full-stack web development
- It is flexible and can be improved later on

# 6. Scope of the Project

The scope of CMS includes:

- User authorization and authentication.
- Student management
- Faculty management
- Management of courses and subjects.
- Attendance tracking
- Management of examination and marks.
- Fee and payment management
- Notification system
- Dashboards and reports

More sophisticated functions including AI-related attendance systems and chatbots can be discussed as improvements of the future.

# 7. System Users

The system supports three primary users:

- Admin - Maintains system configuration.
- Faculty - Deals with teaching related tasks.
- Student - Accesses academic and financial information

Each user is provided role-based access to system functionalities.

# 8. Functional Requirements

## *1. Admin Module*

- Login and logout
- Manage students and faculty
- Create courses and subjects
- Assign subjects to faculty
- Define fee structures
- View attendance, marks, and payment reports

## *2. Faculty Module*

- Login and logout
- View assigned subjects
- Mark student attendance
- Enter examination marks
- View attendance

## *3. Student Module*

- Login and logout
- View personal profile
- View attendance percentage
- View examination marks
- View fee payment status

## 9. Non-Functional Requirements:

- Secure authentication using sessions
- Password hashing
- Role-based access control
- Data consistency and integrity
- Responsive and user-friendly interface
- Reliable database operations

## 10. Technology Stack:

### *1. Backend*

- PHP

### *2. Database*

- MySQL

### *3. Server Environment*

- XAMPP

### *4. Frontend*

- HTML
- CSS
- JavaScript

### *5. Tools*

- phpMyAdmin
- Web Browser

## 11. System Architecture:

### *1. Presentation Layer*

- User interface developed using HTML, CSS, and JavaScript

### *2. Application Layer*

- PHP scripts handling logic and validation

### *3. Database Layer*

- MySQL database storing structured data

# 12. Database Design

The CMS database is designed following **Third Normal Form (3NF)** to eliminate redundancy and ensure data consistency.

**Main Tables:**

- Users
- Students
- Faculty
- Courses
- Subjects
- Attendance
- Exams
- Marks
- Fees
- Payments
- Notifications

Junction tables such as **enrollments**, **faculty_subject**, and **attendance_details** are used in the physical database to resolve many-to-many relationships.

# 13. Entity Relationship Model (ERM)

The Entity Relationship Model represents the logical structure of CMS by identifying entities, their attributes, and relationships.

**Key Entities:**

- Users (authentication details)
- Students (academic profiles)
- Faculty (teaching profiles)
- Courses (academic programs)
- Subjects (course units)
- Attendance (class sessions)
- Exams (assessment events)
- Marks (exam results)
- Fees (fee structures)
- Payments (financial transactions)
- Notifications (system messages)

**Entity Attributes:**

1. **Users**
    - id (Primary Key): Unique identifier for each system user
    - email(unique) : Email address used for login
    - password: Hashed password for secure authentication
    - role: Defines user role (Admin, Faculty, Student)
    - created_at: Timestamp indicating when the user account was created

2. **Students**
    - id (Primary Key): Unique identifier for each student
    - name: Full name of the student
    - roll_number: Unique academic roll number
    - course_id (Foreign Key): References the course in which the student is enrolled
    - semester: Current academic semester
    - email: Contact email address
    - phone: Contact phone number
    - created_at: Timestamp indication

3. **Faculty**
    - id (Primary Key): Unique identifier for each faculty member
    - user_id (Foreign Key): References the corresponding user account in the Users table
    - name: Full name of the faculty member
    - department: Department to which the faculty belongs
    - phone: Contact phone number
    - created_at: Timestamp indication
4. **Courses**
    - id (Primary Key): Unique identifier for each course
    - course_name: Name of the academic program
    - duration: Duration of the course in years
    - created_at: Timestamp indication
5. **Subjects**
    - id (Primary Key): Unique identifier for each subject
    - subject_name: Name of the subject
    - course_id (Foreign Key): References the course offering the subject
    - semester: Semester in which the subject is taug**ht**
    - created_at: Timestamp indication

6. **Attendance**
    - id (Primary Key): Unique identifier for each attendance record
    - subject_id (Foreign Key): References the Subject associated with the attendance session
    - course_id (Foreign Key): References the Course associated with the attendance session
    - semester: The academic semester in which the attendance was recorded
    - date: The date of the class session
    - created_at: Timestamp indication

7. **Exams**
    - id (Primary Key): Unique identifier for each exam
    - exam_name: Name of the examination
    - course_id (Foreign Key): Course associated with the exam
    - semester: Semester in which the exam is conducted
    - created_at: Timestamp indication

8. **Marks**
    - id (Primary Key): Unique identifier for each marks record
    - exam_id (Foreign Key): Exam associated with the marks
    - student_id (Foreign Key): Student who obtained the marks
    - subject_id (Foreign Key): Subject for which marks are awarded
    - marks: Marks obtained by the student
    - created_at: Timestamp indication
9. **Fees**
    - id (Primary Key): Unique identifier for each fee record
    - course_id (Foreign Key): References the Course for which the fee is defined
    - amount: The total fee amount for the course and semester
    - semester: The semester to which the fee applies (e.g., Semester 1, Semester 2)
    - created_at: Timestamp indication
10. **Payments**
    - id (Primary Key): Unique identifier for each payment transaction
    - student_id (Foreign Key): Student who made the payment
    - fee_id (Foreign Key): Fee structure associated with the payment
    - status: Payment status (Paid / Pending)
    - payment_date: Date of payment
    - created_at: Timestamp indication

**11. Notifications**

- id (Primary Key): Unique identifier for each notification
- user_id (Foreign Key): User who receives the notification
- message: Notification content
- created_at: Timestamp indicating when the notification was generated

# 14. Relationships Overview

- One user is associated with one student or faculty
- One course has many students
- One course offers many subjects
- Faculty teach multiple subjects (M:N)
- Students enroll in multiple subjects (M:N)
- Students participate in multiple attendance sessions (M:N)
- Attendance status is stored as a relationship attribute
- One course defines multiple fee structures
- Students make multiple payments
- Each payment corresponds to a specific fee
- Users receive multiple notifications

# 15. Logical vs Physical ER Design

The **logical ER diagram** is the representation of the real-world objects and conceptual relations without implementation restrictions. Most of the many-to-many relationships are presented directly and other attributes like the attendance status is modeled at the relationship level.

The **physical ER diagram** is the implementation of the database. Junction tables are used to resolve many-to-many relationships and primary and foreign keys are explicitly defined. The normalization, integrity and effective data storage is guaranteed by the physical design.

# 16. Data Dictionary

1. **Table: Users**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each system user (Primary Key) | 101 |
| email | VARCHAR | 100 | User email address (Unique) | admin@gmail.com |
| password | VARCHAR | 255 | Hashed password for secure authentication | $2y$10$... |
| role | ENUM | 20 | User role (Admin, Faculty, Student) | Admin |
| created_at | DATETIME | | Timestamp indicating when the user account was created | 2025-01-10 10:30 |

2. **Table: Students**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each student (Primary Key) | 102 |
| user_id | INT | 11 | Foreign Key to Users table | 105 |
| name | VARCHAR | 100 | Full name of the student | Bishwas Chaudhary |
| roll_number | VARCHAR | 50 | Unique academic roll number | CS-25123796 |
| course_id | INT | 11 | Foreign Key to Courses table | 3 |
| semester | VARCHAR | 10 | Current academic semester | A3 |
| phone | VARCHAR | 20 | Student contact phone number | 9876543210 |

| | | | | |
|---|---|---|---|---|
| created_at | DATETIME | | Timestamp when the student record was created | 2025-01-01 09:00 |

3. **Table: Faculty**

| **Field Name** | **Data Type** | **Size** | **Description** | **Example** |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each faculty member (Primary Key) | 201 |
| user_id | INT | 11 | Foreign Key to Users table | 105 |
| name | VARCHAR | 100 | Full name of the faculty member | Dr.Mehta |
| department | VARCHAR | 100 | Department to which the faculty belongs | Computer Science |
| phone | VARCHAR | 20 | Contact phone number | 9123456780 |
| created_at | DATETIME | | Timestamp indicating when the faculty record was created | 2025-01-10 12:00 |

4. **Table: Courses**

| **Field Name** | **Data Type** | **Size** | **Description** | **Example** |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each course (Primary Key) | 3 |
| course_name | VARCHAR | 100 | Name of the academic program | BCA |
| duration | INT | 2 | Duration of the course in years | 3 |
| created_at | DATETIME | | Timestamp when the course record was created | 2025-01-10 10:00 |

5. **Table: Subjects**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each subject (Primary Key) | 45 |
| subject_name | VARCHAR | 100 | Name of the subject | DBMS |
| course_id | INT | 11 | Foreign Key to Courses table | 3 |
| semester | VARCHAR | 10 | Semester in which the subject is taught | A4 |
| created_at | DATETIME | | Timestamp when the subject record was created | 2025-01-01 12:00 |

6. **Table: Faculty_Subject (Junction Table)**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for the record (Primary Key) | 101 |
| faculty_id | INT | 11 | Foreign Key to Faculty table | 201 |
| created_at | DATETIME | | Timestamp when the faculty-subject relationship was created | 2025-01-01 13:00 |

7. **Table: Enrollments (Junction Table)**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for the enrollment record (Primary Key) | 301 |
| student_id | INT | 11 | Foreign Key to Students table | 501 |

| | | | | |
|---|---|---|---|---|
| subject_id | INT | 11 | Foreign Key to Subjects table | 45 |
| created_at | DATETIME | | Timestamp when the enrollment record was created | 2025-01-10 10:30 |

8. **Table: Attendance**

| **Field Name** | **Data Type** | **Size** | **Description** | **Example** |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for the enrollment record (Primary Key) | 101 |
| subject_id | INT | 11 | Foreign Key to Subjects table | 45 |
| course_id | INT | 11 | Foreign Key to Courses table | 3 |
| semester | VARCHAR | 10 | The academic semester in which the attendance was recorded | A4 |
| date | date | | Date of the class session | 2025-02-10 |
| created_at | DATETIME | | Timestamp when the attendance record was created | 2025-02-10 10:00 |

9. **Table: Attendance_Details (Junction Table)**

| **Field Name** | **Data Type** | **Size** | **Description** | **Example** |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each attendance detail (Primary Key) | 201 |
| attendance_id | INT | 11 | Foreign Key to Attendance table | 101 |
| student_id | INT | 11 | Foreign Key to Students table | 501 |
| status | ENUM | 20 | Attendance status (present/absent) | present |
| created_at | DATETIME | | Timestamp when the attendance status was created | 2025-02-10 10:00 |

**10. Table: Exams**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each attendance detail (Primary Key) | 201 |
| exam_name | VARCHAR | 100 | Name of the exam | Midterm |
| course_id | INT | 11 | Foreign Key to Courses table | 3 |
| semester | VARCHAR | 10 | Semester in which the exam is conducted | A3 |
| created_at | DATETIME | | Timestamp when the exam record was created | 2025-01-01 12:30 |

**11. Table: Marks**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each marks record (Primary Key) | 501 |
| exam_id | INT | 11 | Foreign Key to Exams table | 12 |
| student_id | INT | 11 | Foreign Key to Students table | 501 |
| subject_id | INT | 11 | Foreign Key to Subjects table | 45 |
| marks | INT | 3 | Marks obtained by the student | 75 |
| created_at | DATETIME | | Timestamp when the marks were recorded | 2025-01-10 10:45 |

12. **Table: Fees**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each fee record (Primary Key) | 301 |
| course_id | INT | 11 | Foreign Key to Courses table | 3 |
| semester | VARCHAR | 10 | The semester for which the fee applies | A1 |
| amount | DECIMAL | 10,2 | Total fee amount for the course and semester | 45000.00 |
| created_at | DATETIME | | Timestamp when the fee record was created | 2025-01-01 12:00 |

**13. Table: Payments**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each payment record (Primary Key) | 601 |
| student_id | INT | 11 | Foreign Key to Students table | 501 |
| fee_id | INT | 11 | Foreign Key to Fees table | 301 |
| status | ENUM | 20 | Payment status (Paid / Pending) | Paid |
| payment_date | DATE | | Date of payment | 2025-01-15 |
| created_at | DATETIME | | Timestamp when the payment record was created | 2025-01-15 14:00 |

**14. Table: Notifications**

| Field Name | Data Type | Size | Description | Example |
|---|---|---|---|---|
| id | INT | 11 | Unique identifier for each notification (Primary Key) | 801 |
| user_id | INT | 11 | Foreign Key to Users table | 101 |
| message | TEXT | | Notification content | Fee due reminder |
| created_at | DATETIME | | Timestamp when the notification was created | 2025-01-10 15:00 |

# 17. Limitations:

- Requires internet connectivity
- No mobile application support
- Advanced AI-based features not implemented
- Designed for small to medium-sized institutions

# 18. Conclusion:

College Management System (CMS) is a multi-service web-based application that is effective in automating administrative and academic procedures. The project has been used to show good application of PHP and MySQL to develop a database-driven system that is normalized, secure and scaled. A clear distinction between logical and physical design will provide the clarity of concepts and efficient database implementation that CMS guarantees. The system minimizes the workload of people working on it manually, enhances accuracy, and gives a stable platform to the management of modern colleges.

# 19. Physical ER Diagram

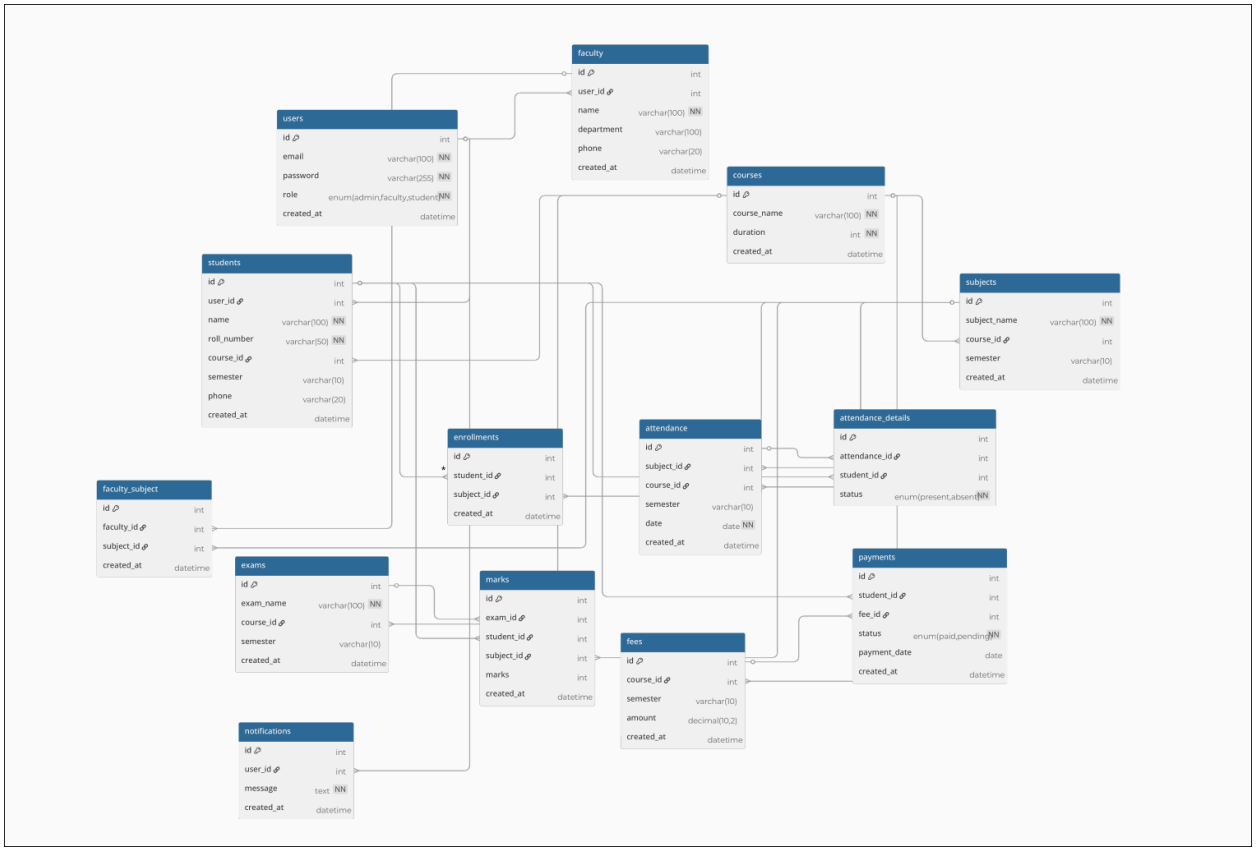


Fig: Physical ER Diagram

# 20. Logical ER Diagram

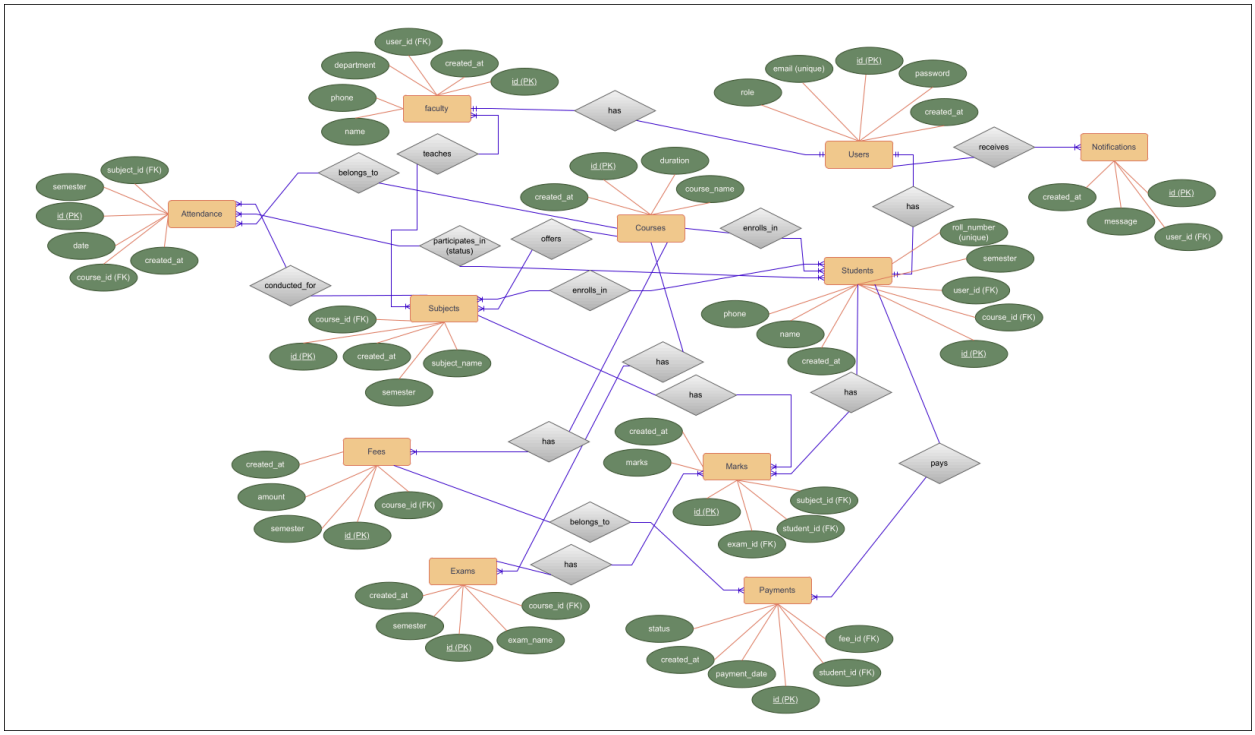


Fig: Logical ER Diagram