ARYA HELI JITENDRA PRABH

Student management system

By Code Commanders

Contents

[Executive Summary 2](#_Toc127454286)

[Design Analysis Process 2](#_Toc127454287)

[Entity Relationship Diagram 3](#_Toc127454288)

[Table Designs – Data Dictionary 4](#_Toc127454289)

[Contributions 5](#_Toc127454290)

[References 5](#_Toc127454291)

# Executive Summary

Managing academic records manually is a time-consuming process. The Student Management System (SMS) is designed to digitize student enrolment, attendance tracking, grade management. This database-driven solution allows administrators, teachers, and students to efficiently manage and access academic information. The system ensures effective data handling, reducing errors in documentation while improving accessibility and automation.

# Design Analysis Process

Enables tracking of enrolments, attendance, marks, and academic progress.

Accessible to students, teachers, and administrators for effective data management.  
 Provides features such as timetable management, grading, and report generation.

**SDLC**

We have implemented the Agile Methodology to develop our Student Management System due to its iterative approach, which allows us to start with a basic framework and continuously improve the system through regular updates until the final release. This methodology enables users to actively participate in refining requirements and ensures a smooth development process by incorporating feedback at every stage.

We began by gathering requirements from students, teachers, ensuring that the system aligns with their needs. Throughout the development cycle, we iteratively enhanced features, making necessary modifications based on user feedback. Agile allowed us to start with an essential structure containing core functionalities, such as student enrollment and course management, and gradually expand the system with additional features, including attendance tracking and results management.

By following Agile, we maintained flexibility in planning, regularly evaluating and refining the system through phases like requirement analysis, design, development, testing, and deployment. This approach not only streamlined the development process but also ensured the system evolved efficiently, meeting the dynamic needs of the users while delivering continuous improvements.



[https://miro.medium.com/v2/resize:fit:1400/1\*qxcToTULQN7jD9g1WUu1gg.jpeg](https://miro.medium.com/v2/resize:fit:1400/1*qxcToTULQN7jD9g1WUu1gg.jpeg)

**Actors**

**Students**: In the student management system students are the primary users of this system. They can see their class schedule so they can reach their class on time, they could check their grades and see their progress, they can sign up for the other activities and upcoming events and they can update their personal information.

**Teachers**: In this system teachers can monitor and track the attendance of students in their classes, they can mark and update the assignments and provide feedback on students' progress.

Admin: The students’ admissions managed by the administrators, they can update and maintain students records or keep the system updated and running smoothly.

**User stories**

1. As a student, I want to check my grades for each course so that I can track my academic performance.
2. As a student, I want to register for courses online so that I can enroll without visiting the campus.
3. As a student, I want to update my personal information such as email and phone number to keep my records accurate.
4. As a student, I want to see the list of blocks I am currently enrolled in so that I can manage my course load.
5. As a student, I want to pay my tuition fees online so that I can complete my payments conveniently.
6. As a student, I want to access my class schedule so that I can plan and organize my study time and other activities accordingly.
7. As a student, I want to view my attendance record (absences, late arrivals) so that I can keep track of my attendance and stay informed about my standing.
8. As a teacher, I want to record attendance information (present, absent, late, sick leave) for each of my students so that I can maintain accurate attendance records.
9. As a teacher, I want to view a list of all students enrolled in the blocks I manage so that I can track their progress.
10. As a teacher, I want to record student grades and marks for each block so that students can track their academic performance.
11. As a teacher, I want to view and update the start and end dates of blocks so that the academic schedule stays consistent.
12. As a teacher, I want to assign grades to students based on their performance in the block.

### Functional and Non-Functional Requirements:

Functional: Student enrolment, attendance tracking, grade management, report generation.

Non-Functional: Secure data storage, user-friendly interface, scalable architecture.

**What We Did:**

* Designed and developed a comprehensive Student Management System that automates student enrolment, attendance tracking, grading.
* Integrated a relational database with well-structured tables for efficient data management.
* Built a user-friendly interface to enhance usability for students, teachers, and administrators.

**Why We Did It:**

* A centralized digital system improves accessibility and security.
* Automation of academic processes reduces administrative workload and improves data accuracy.

**Why Our Implementation Is Better:**

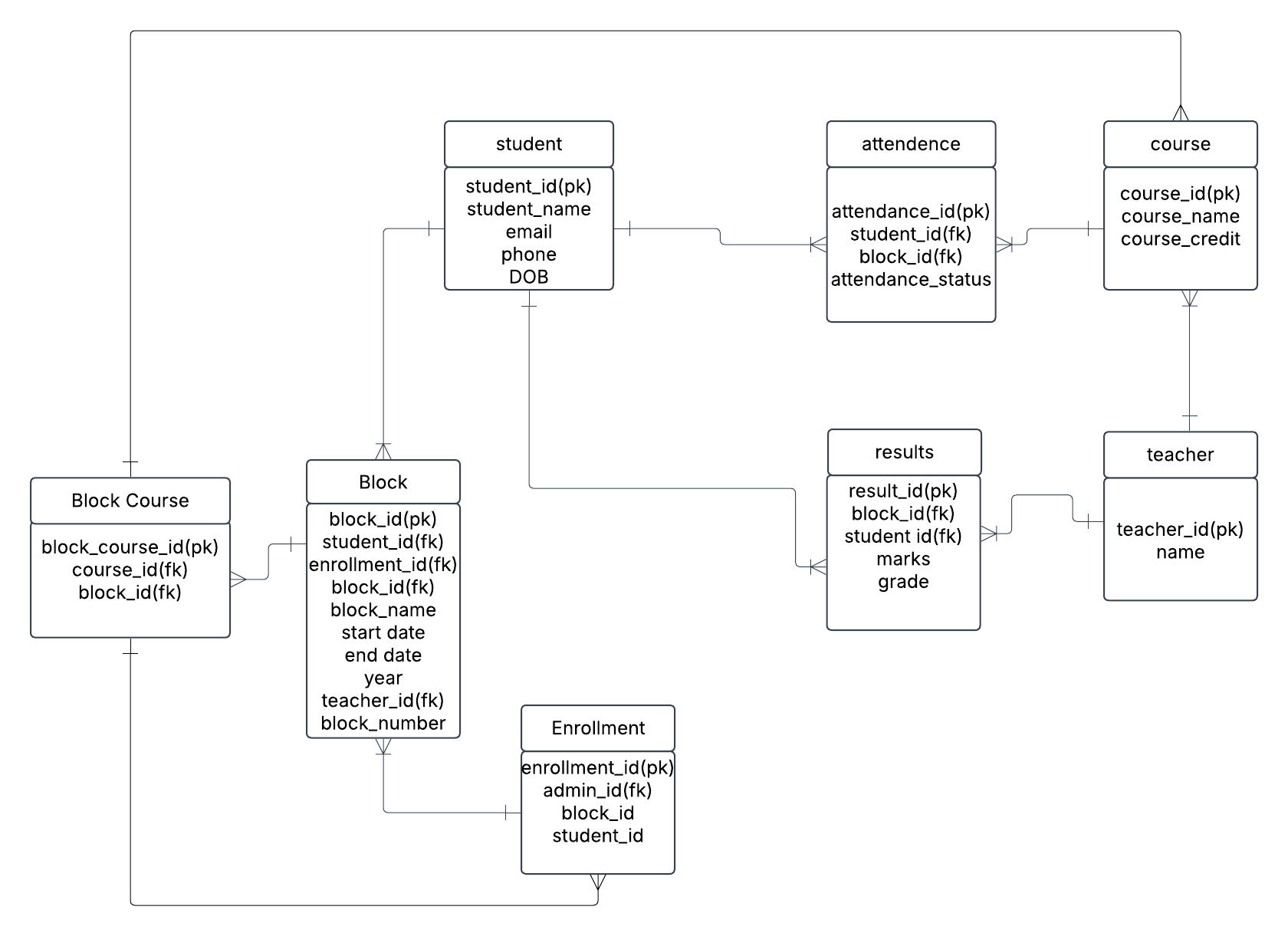
* Compared to other existing systems, our solution integrates billing and payment management, enhancing financial transparency.
* The design allows scalability, making it adaptable for future institutional growth.
* Provides real-time reporting and analytics for better decision-making.

Use Cases

1. **Student Enrolment** - A student can complete the enrolment process by selecting courses and submitting personal details.
2. **Student Timetable Access** - Students can view their class schedules, including lecture timings and exam dates.
3. **Student Information Access** - Students can access their personal information such as contact details and enrolled courses.
4. **Student Attendance Viewing** - Students can check their attendance records and see if they have been marked present, absent, or late.
5. **Student Grade Viewing** - Students can view their marks and academic progress for different courses.
6. **Teacher Attendance Management** - Teachers can mark student attendance with statuses like present, absent, or late.
7. **Teacher Grade Entry** - Teachers can enter and update student grades for different subjects.
8. **Teacher Course Management** - Teachers have access to their assigned course details and student enrolments.
9. **Teacher Class Scheduling** - Teachers can view and manage class schedules for their assigned subjects.
10. **Admin Fee Receipt Generation** - Administrators can generate and issue fee receipts for students.
11. **Admin Course Assignment** - Administrators can assign teachers to specific courses and update teaching schedules.
12. **Admin Report Generation** - Administrators can generate detailed reports on student attendance, performance, and enrolments.

# Entity Relationship Diagram

**Physical ERD :**



# Table Designs – Data Dictionary

# Student Management System - Data Dictionary

**Student Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| student\_id | Unique ID for each student | VARCHAR(10) | PK | NOT NULL | 12021, 12201 |
| student\_name | Name of each student | VARCHAR(50) |  | NOT NULL | Amit, Priya |
| email | Email of each student | VARCHAR(100) |  | NOT NULL | [amit@example.com](mailto:amit@example.com) |
| phone | Contact number of each student | BIGINT |  | NOT NULL | 9876543210 |
| DOB | Date of Birth | DATE |  | NULL | 2001-05-10 |

**Attendance Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| attendance\_id | Unique ID for attendance | VARCHAR(10) | PK | NOT NULL | AT1, AT2 |
| student\_id | Student ID | VARCHAR(10) | FK | NOT NULL | S1, S2 |
| block\_id | Block ID | VARCHAR(10) | FK | NOT NULL | B1, B2 |
| attendance\_status | Attendance Status (Present/Absent) | VARCHAR(10) |  | NOT NULL | Present, Absent |

**Course Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| course\_id | Unique ID for each course | VARCHAR(10) | PK | NOT NULL | C1, C2 |
| course\_name | Name of the course | VARCHAR(50) |  | NOT NULL | Mathematics, Physics |
| course\_credit | Credit for the course | INT |  | NOT NULL | 360, 390 |

**Teacher Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| teacher\_id | Unique ID for each teacher | VARCHAR(10) | PK | NOT NULL | 202, 101 |
| name | Name of the teacher | VARCHAR(50) |  | NOT NULL | Rajesh, Anjali |

**Block Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| block\_id | Unique ID for each block | VARCHAR(10) | PK | NOT NULL | B1, B2 |
| block\_name | Name of the block | VARCHAR(50) |  | NOT NULL | Semester 1, Semester 2 |
| start\_date | Start date of the block | DATE |  | NOT NULL | 2025-01-10 |
| end\_date | End date of the block | DATE |  | NOT NULL | 2025-06-15 |
| year | Year of the block | INT |  | NOT NULL | 2025 |
| teacher\_id | Teacher assigned to the block | VARCHAR(10) | FK | NOT NULL | 101, 202 |
| Block\_numbe | Numeber of the vlock | INT |  | NOT NULL | 1,2,3 |
| student\_id | Student ID | VARCHAR(10) | FK | NOT NULL | S1, S2 |

**Results Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| result\_id | Unique ID for result | VARCHAR(10) | PK | NOT NULL | R1, R2 |
| block\_id | Block ID | VARCHAR(10) | FK | NOT NULL | B1, B2 |
| student\_id | Student ID | VARCHAR(10) | FK | NOT NULL | 52121, 5412 |
| marks | Marks obtained | INT |  | NOT NULL | 85, 90 |
| grade | Grade obtained | VARCHAR(5) |  | NOT NULL | A, B |

**Enrollment Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| enrollment\_id | Unique ID for enrollment | VARCHAR(10) | PK | NOT NULL | E1, E2 |
| admin\_id | Admin who processed enrollment | VARCHAR(10) | FK | NOT NULL | A1, A2 |
| block\_id | Block in which student is enrolled | VARCHAR(10) | FK | NOT NULL | B1, B2 |
| student\_id | Student who is enrolled | VARCHAR(10) | FK | NOT NULL | 102001,11062 |

**Block course Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Key Field | Constraints | Example |
| block\_course\_id | Unique ID for block course | VARCHAR(10) | PK | NOT NULL | BC1, BC2 |
| course\_id | Course ID | VARCHAR(10) | FK | NOT NULL | C1, C2 |
| block\_id | Block ID | VARCHAR(10) | FK | NOT NULL | B1, B2 |

# 

# Contributions

Add the list of members and their contributions to the project.

https://github.com/arya4100/Code-Commanders

# References

Add references here if you have any.