SANJAY GHODAWAT INSTITUTE
Approved by A.I.C.T.E. New Delhi, and Recognized by DTE Mumbai, Govt. of Maharashtra

A

REPORT ON “School Timetable Management System”

SUBMITTED BY –

Sr. No.	Name of Student	Roll No.
1	Piyush Sachin Lohar.	33
2	Aditya Vikram Pawal.	31
3	Raj Chandrakant Dewarde.	28
4	Gautam Vishal Chavare.	78

UNDER THE GUIDANCE OF

Mr. A. G. Kalukhe

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING SANJAY GHODAWAT INSTITUTE, ATIGRE ACADEMIC YEAR: 2025-26

SANJAY GHODAWAT INSTITUTE

Approved by A.I.C.T.E. New Delhi, and Recognized by DTE Mumbai, Govt. of Maharashtra

Certificate

This is to certify that the Micro project work entitled

“School Timetable Management System

family”

Has been successfully completed by

In fulfillment for the

Diploma in Computer Science &Engineering

Maharashtra State Board of Technical Education

During the academic year 2025-26 under the guidance of

Mr. A. G. Kalukhe
Project Guide

Mr.S.V.Chavan
H.O.D

Dr. V. V. Giri
Principal

INDEX

Sr. No.	CONTENT	Page No.
1	Introduction	**4**
2	Problem Definition	**5,6**
3	System Analysis	**7**
4	Database Design	**8,9,10**
5	Implementation	**11,12,13**
6	Result	**14,15**
7	Conclusion	**16,17,18**
8	References	**19**

Introduction

The School Timetable Management System is a database-driven application designed to simplify and automate the process of creating, managing, and maintaining school timetables. Traditionally, preparing a timetable manually is a time-consuming and error-prone task that requires balancing subjects, teachers, classrooms, and time slots. This project uses a Database Management System (DBMS) to efficiently store, organize, and retrieve information related to teachers, subjects, classes, and schedules.

- To automate the process of timetable creation and management.
- To reduce manual effort and chances of human error.
- To maintain accurate data about teachers, subjects, classes, and timings.
- To ensure there are no clashes between teachers, subjects, or classrooms.
- To provide quick updates and modifications to the timetable.

Problem Definition

🎯 Real-World Problem for “School Timetable Management System”

Problem Statement:

In most schools, preparing and maintaining a timetable is done manually. It takes a lot of time to assign subjects, teachers, and classrooms while avoiding conflicts (like the same teacher in two classes at once). Making changes or updates also requires redoing the entire schedule. This process is inefficient, error-prone, and stressful for administrators.

Solution:

To solve this problem, the School Timetable Management System uses a Database Management System to store and manage all information — such as teacher details, subjects, classes, and periods. The system automatically generates a conflict-free timetable, allows easy modifications, and provides quick access to data.

Why We Do Need a Database for the School Timetable Management System

A database is essential for this project because it helps store, organize, and manage all the information related to classes, teachers, subjects, and schedules efficiently. Without a database, the data would have to be managed manually using paper or spreadsheets, which is time-consuming, error-prone, and difficult to update.

Using a Database Management System (DBMS) provides several key advantages:

✓ 1. Centralized Data Storage

All timetable-related data (teachers, subjects, classes, rooms, periods) is stored in one place, making access and management easy.

✓ 2. Avoids Data Redundancy

A DBMS ensures that information (like teacher or subject details) is not repeated unnecessarily, saving storage and preventing confusion.

✓ 3. Ensures Data Consistency

Any changes made in one table (like teacher details) are automatically reflected wherever needed.

✓ 4. Easy Data Retrieval

The database allows quick searches — for example, viewing a teacher's timetable or checking free classrooms.

System Analysis

Existing System

In many schools, the timetable is prepared **manually** by staff members using **paper-based methods** or **simple spreadsheets**. Teachers, classes, and subjects are scheduled by trial and error, which often leads to confusion and clashes.
Limitations of the existing system:

1. Time-consuming and requires a lot of manual effort.
2. High chances of human errors such as overlapping periods or incorrect teacher assignments.
3. Difficult to update or modify when changes are needed.
4. No centralized database to store class, teacher, and subject information.
5. Paper-based records are hard to maintain and not easily accessible.

Proposed System

The **School Timetable Management System** aims to automate the timetable creation and management process using a **database-driven application**.
Improvements and Features:

1. Automatic generation of timetables based on teacher and subject availability.
2. Centralized **DBMS** for storing all school, class, and subject data securely.
3. Easy modification and updates without redoing the entire timetable.
4. Reduces human errors and avoids period clashes.
5. Provides quick access to timetables for teachers and students.
6. Saves time and increases efficiency in timetable management.

Database Design

Entity–Relationship (ER) Diagram

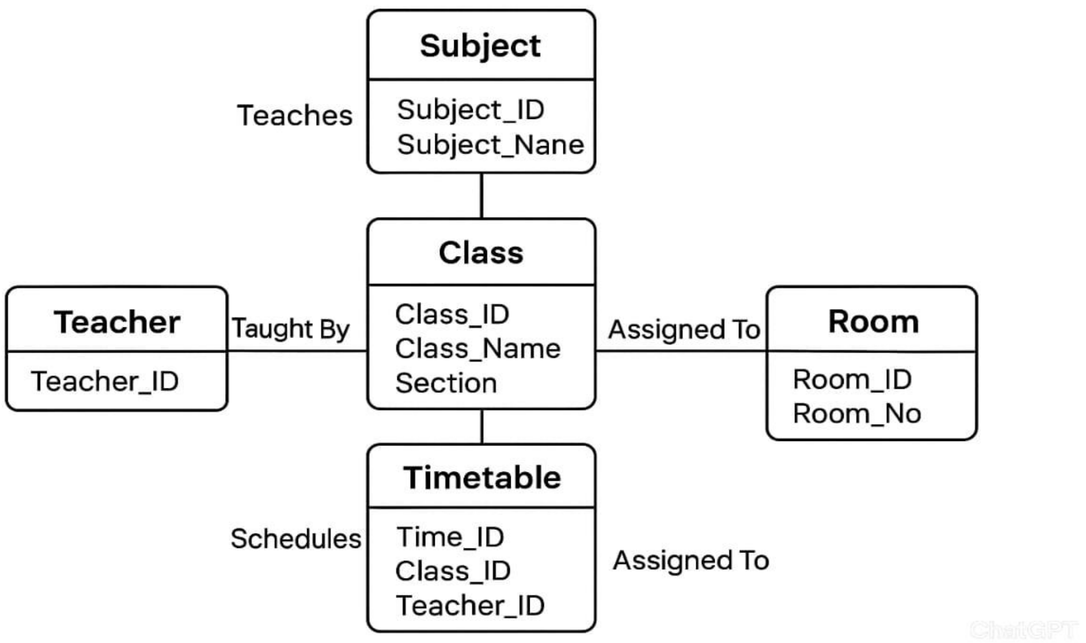
Entities and Relationships:

1. **Teacher** – teaches subjects and takes classes.
2. **Subject** – taught by teachers and assigned to classes.
3. **Class** – consists of multiple subjects and students.
4. **Timetable** – schedules subjects, teachers, and classes by day and period.
5. **Room** – assigned to a class for each period.

Relationships:

- A **Teacher** can teach many **Subjects**.
- A **Subject** can be taught to many **Classes**.
- A **Class** has many **Timetable entries**.
- Each **Timetable** entry links **Class**, **Teacher**, **Subject**, **Room**, and **Time**.

Entity–Relationship (ER) Diagram



Scheme Design

Table Name	Attributes	Primary Key (PK)	Foreign Key (FK)
Teacher	Teacher_ID, Teacher_Name, Qualification, Phone_No, Email	, Qualification, Phone_No, Email Teacher_ID	—
Subject	Subject_ID, Subject_Name, Teacher_ID	Subject_ID	Teacher_ID → Teacher(Teacher_ID)
Class	Class_ID, Class_Name, Section, Class_Teacher_ID	Class_ID	Class_Teacher_ID → Teacher(Teacher_ID)
Room	Room_ID, Room_No, Capacity	Room_ID	—
Timetable	Time_ID, Class_ID, Subject_ID, Teacher_ID, Room_ID, Day, Period	Time_ID	Class_ID → Class(Class_ID), Subject_ID → Subject(Subject_ID), Teacher_ID → Teacher(Teacher_ID), Room_ID → Room(Room_ID)

Normalization Steps

1NF (First Normal Form)

- All attributes contain **atomic values** (no repeating groups).
- Example: Each teacher's name, subject, or class is stored in separate columns.

2NF (Second Normal Form)

- All **non-key attributes** are fully dependent on the **primary key**.
- Example: In the **Timetable** table, data like `Day`, `Period`, `Class_ID`, `Subject_ID`, etc. depend on the primary key `Time_ID`.

3NF (Third Normal Form)

- There are **no transitive dependencies** (non-key attributes depending on other non-key attributes).
- Example: Teacher details are stored separately in the **Teacher** table instead of repeating in **Timetable**, ensuring no redundancy.

Implementation

📄 Project Section: SOFTWARE & HARDWARE REQUIREMENTS + SQL QUERIES

🔧 SOFTWARE REQUIREMENTS

- Back-end Database: MySQL Server (v5.7 or higher)
- Language (Back-end Logic): PHP / Python / Java (any one as per project)
- Front-end (optional): HTML, CSS, JavaScript (if web-based)
- Web Server (if PHP): XAMPP / WAMP
- IDE/Editor: VS Code / PyCharm / NetBeans / Sublime Text

🖥 HARDWARE REQUIREMENTS

- Processor: Intel i3 or higher
- RAM: 4 GB or more
- Hard Disk: Minimum 2 GB of free space
- Display: 1024x768 resolution or higher
- OS: Windows 7/8/10 or Linux

🧱 SQL QUERIES (DATABASE DESIGN)

1️⃣ Table Creation

```
CREATE TABLE Teacher (
Teacher_ID INT PRIMARY KEY,
Teacher_Name VARCHAR(100),
Qualification VARCHAR(100),
Phone_No VARCHAR(15),
Email VARCHAR(100)
);

CREATE TABLE Subject (
Subject_ID INT PRIMARY KEY,
Subject_Name VARCHAR(100),
Teacher_ID INT,
FOREIGN KEY (Teacher_ID) REFERENCES Teacher(Teacher_ID)
);

CREATE TABLE Class (
Class_ID INT PRIMARY KEY,
Class_Name VARCHAR(50),
Section VARCHAR(10),
Class_Teacher_ID INT,
FOREIGN KEY (Class_Teacher_ID) REFERENCES Teacher(Teacher_ID)
);

CREATE TABLE Room (
Room_ID INT PRIMARY KEY,
Room_No VARCHAR(20),
Capacity INT
);

CREATE TABLE Timetable (
Time_ID INT PRIMARY KEY,
Class_ID INT,
Subject_ID INT,
Teacher_ID INT,
Room_ID INT,
Day VARCHAR(20),
Period VARCHAR(20),
FOREIGN KEY (Class_ID) REFERENCES Class(Class_ID),
FOREIGN KEY (Subject_ID) REFERENCES Subject(Subject_ID),
FOREIGN KEY (Teacher_ID) REFERENCES Teacher(Teacher_ID),
FOREIGN KEY (Room_ID) REFERENCES Room(Room_ID)
);
```

2 Insert Records

```
INSERT INTO Teacher VALUES (1, 'Mr. Sharma', 'M.Sc, B.Ed', '9876543210',
'sharma@school.com');

INSERT INTO Subject VALUES (101, 'Mathematics', 1);

INSERT INTO Class VALUES (10, '10th Grade', 'A', 1);

INSERT INTO Room VALUES (1, 'Room-101', 40);
INSERT INTO Timetable VALUES (1001, 10, 101, 1, 1, 'Monday', '1st Period');
```

3 Retrieval (SELECT)

```
-- Get full timetable for a class:
SELECT T.Day, T.Period, S.Subject_Name, R.Room_No, Te.Teacher_Name
FROM Timetable T
JOIN Subject S ON T.Subject_ID = S.Subject_ID
JOIN Teacher Te ON T.Teacher_ID = Te.Teacher_ID
JOIN Room R ON T.Room_ID = R.Room_ID
WHERE T.Class_ID = 10;

-- List all teachers and their subjects:
SELECT T.Teacher_Name, S.Subject_Name
FROM Teacher T
JOIN Subject S ON T.Teacher_ID = S.Teacher_ID;
```

4 Update Operation

```
-- Change room for a class period
UPDATE Timetable
SET Room_ID = 2
WHERE Time_ID = 1001;
```

5 Delete Operation

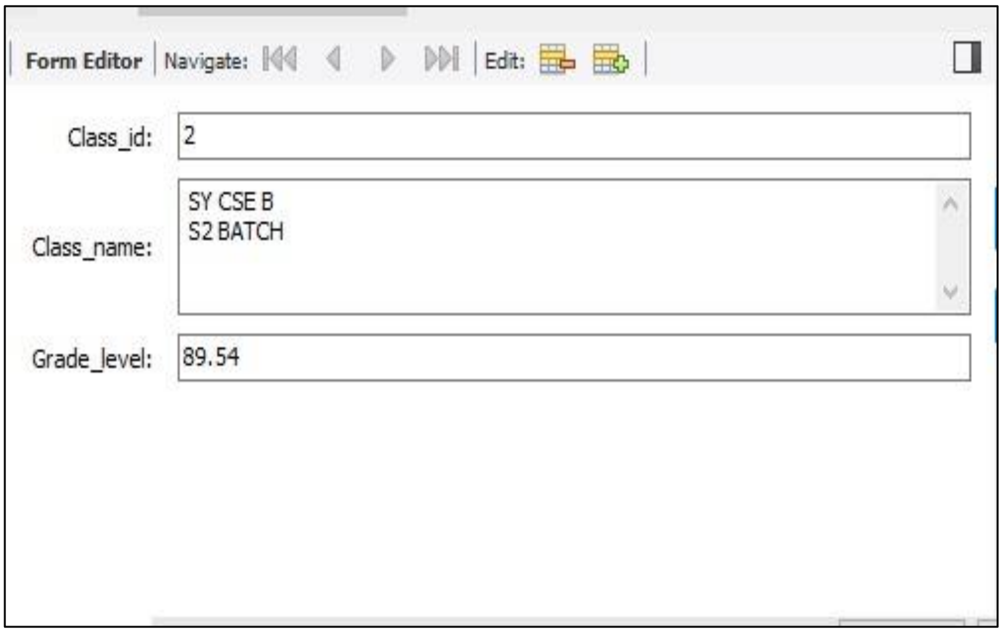
```
-- Remove a subject
DELETE FROM Subject
WHERE Subject_ID = 101;
```

SCREENSHOTS (if you used a front-end)

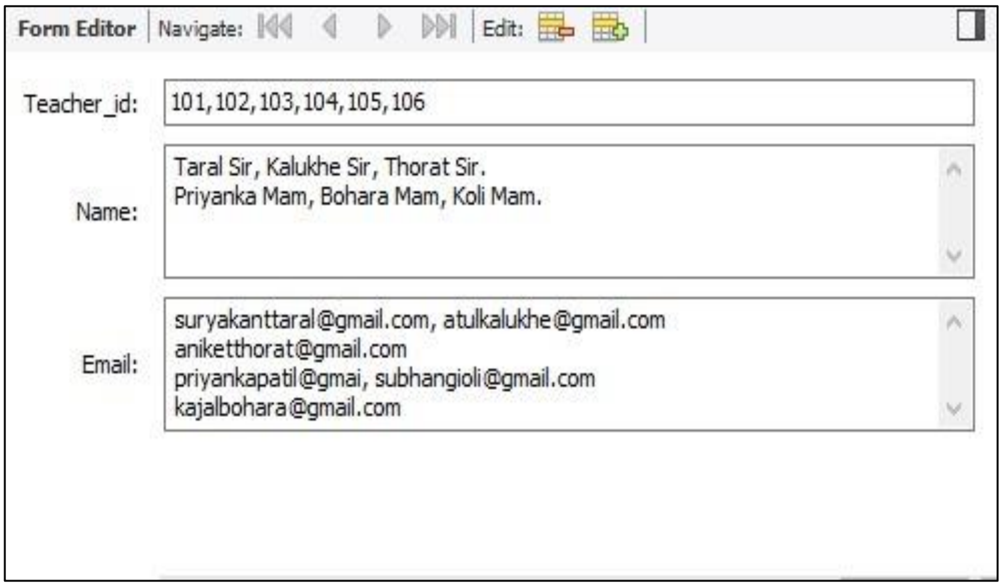
If you made a front-end using PHP or Python (Django/Flask), you can take screenshots of:

- Login Page
- Add Teacher / Add Subject Forms
- View Timetable Page
- Update/Delete record interface
- Class-wise Timetable Table Display

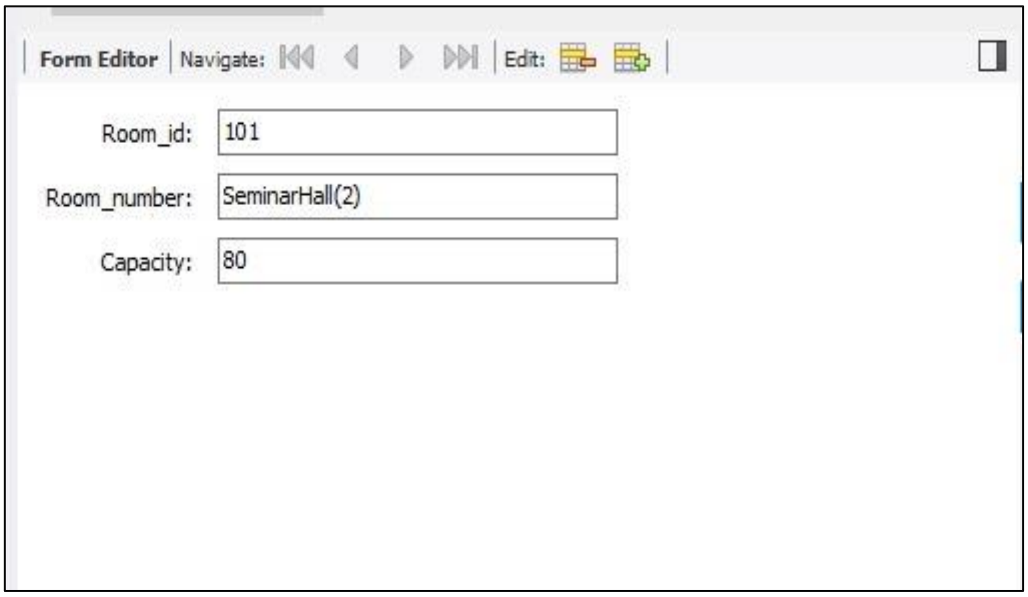
Result



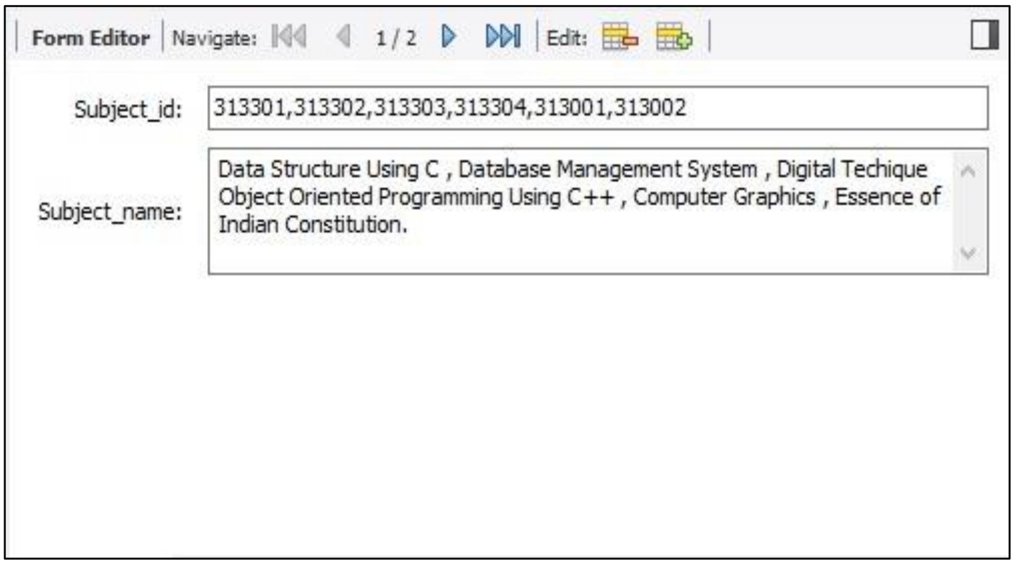
Class_id:	2
Class_name:	SY CSE B S2 BATCH
Grade_level:	89.54



Teacher_id:	101,102,103,104,105,106
Name:	Taral Sir, Kalukhe Sir, Thorat Sir. Priyanka Mam, Bohara Mam, Koli Mam.
Email:	suryakanttaral@gmail.com, atulkalukhe@gmail.com aniketthorat@gmail.com priyankapatil@gmai, subhangioli@gmail.com kajalbohara@gmail.com



Room_id:	101
Room_number:	SeminarHall(2)
Capacity:	80



Subject_id:	313301,313302,313303,313304,313001,313002
Subject_name:	Data Structure Using C , Database Management System , Digital Techique Object Oriented Programming Using C++ , Computer Graphics , Essence of Indian Constitution.

Conclusion

Why you learned

Working on this DBMS project helped me gain both theoretical knowledge and practical experience in designing and implementing real-world database applications. Below are the key things I learned during the course of this project:

1 Database Design & ER Modeling

- Learned how to identify entities, attributes, and relationships from a real-life scenario.
- Practiced converting a real-world problem (school timetable) into a proper Entity-Relationship (ER) diagram.
- Gained experience in designing normalized schemas to avoid redundancy and maintain data integrity.

2 SQL Query Writing

- Learned how to write SQL commands for creating tables, inserting data, and modifying records.
- Practiced SELECT queries with joins to retrieve meaningful reports like full class timetables or teacher schedules.
- Understood how to use constraints like PRIMARY KEY and FOREIGN KEY for ensuring data consistency.

3 Data Normalization

- Learned how to apply normalization rules (1NF, 2NF, 3NF) to ensure that the database is efficient, scalable, and free from anomalies.
- Understood the importance of separating data logically across multiple related tables.

4 Problem-Solving & Logical Thinking

- Understood how database systems can be used to solve real-world problems like avoiding scheduling conflicts or resource duplication.
- Learned to think critically about how to structure data and plan relationships between tables for accurate and fast retrieval.

5 Front-End and Integration (optional)

- If a front-end was used: Gained experience in connecting front-end tools (e.g., HTML, PHP, or Python Flask) to the MySQL database.
- Learned how to create user interfaces for adding, updating, and viewing records from the database.

6 Real-World Application of DBMS Concepts

- This project helped me move from textbook knowledge to practical application.
- I now better understand how schools, colleges, and institutions manage large volumes of data using database systems.

7 Project Planning & Documentation

- Learned how to document a full project including system analysis, design, ER diagrams, SQL queries, testing, and conclusions.
- Practiced writing formal reports that explain both technical and functional aspects of the project.

📌 Final Reflection:
This project enhanced my skills in database design, SQL, and system thinking. It also taught me the value of organizing data efficiently and solving problems with a structured, database-driven approach. Most importantly, it gave me confidence in building systems that are useful, reliable, and applicable in real life.

How project objective were achieved

The primary goal of the School Timetable Management System was to automate and simplify the scheduling of subjects, teachers, and classrooms, reducing manual effort and errors. The following points explain how each project objective was successfully achieved:

1. ✅ Automated Timetable Generation

2. ✅ Database Integration

3. ✅ Avoiding Clashes and Conflicts

Future Improvements

While the current version of the School Timetable Management System meets the basic scheduling and data management needs, there is scope for further enhancements to improve functionality, usability, and scalability.

Some possible future improvements include:

1. 📱 Mobile Application Integration
   Developing a mobile app version of the system for students and teachers would make timetable access more convenient and on-the-go.
2. 🌐 Web-Based Portal with Login System
   Implementing a secure web interface with role-based login (Admin, Teacher, Student) would allow personalized access to schedules and updates.
3. 🤖 AI-Based Auto-Scheduling
   Introducing AI or heuristic algorithms (like Genetic Algorithm or Constraint Satisfaction Problem solvers) can help generate optimized, clash-free timetables automatically with minimal human input.
4. 📤 Notifications & Alerts
   Adding features for sending email or SMS notifications to teachers/students about timetable changes, substitutions, or important notices.
5. 📈 Analytics Dashboard
   Creating visual dashboards that show teacher workload, classroom utilization, and free time slots to help in better planning and management.
6. ⏱️ Real-Time Substitution Handling
   Allowing real-time management of substitutions when a teacher is absent — automatically reallocating or notifying replacement teachers.
7. 🌍 Multi-School Support
   Expanding the system to support multiple schools or campuses under a single system (useful for educational groups or institutions with branches).

References

1. Books

- Elmasri, R., & Navathe, S. B. (2017). Fundamentals of Database Systems (7th Edition). Pearson Education.
  [ISBN: 978-0133970777]
- Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database System Concepts (7th Edition). McGraw Hill Education.
  [ISBN: 978-0078022159]

2. Websites

- GeeksforGeeks – DBMS Tutorial:
  https://www.geeksforgeeks.org/dbms/
- TutorialsPoint – DBMS Guide:
  https://www.tutorialspoint.com/dbms/
- W3Schools – SQL and Database Basics:
  https://www.w3schools.com/sql/

3. DBMS Manual

(Use your actual DBMS here — for example, if you're using MySQL or PostgreSQL)

- MySQL 8.0 Reference Manual:
  https://dev.mysql.com/doc/refman/8.0/en/
- SQLite Documentation:
  https://sqlite.org/docs.html

4. Lecture Notes

- NPTEL (National Programme on Technology Enhanced Learning) – Database Management Systems (Prof. P. P. Chakrabarti, IIT Kharagpur):
  https://nptel.ac.in/courses/106105175
- DBMS Notes by Gate Smashers (YouTube):
  https://www.youtube.com/playlist?list=PLjVLYmrlmjGdgJZAU3zwCq2pXHC4c7qcW