Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)
School of Computing
B.Tech. – Computer Science and Engineering

VTR UGE2021- (CBCS)

Academic Year: 2025–2026

SDG 4: Quality Education
Course Code : 10211CS207

Course Name : Database Management Systems

Slot No : S4

DBMS PROJECT REPORT

Title: **STUDENT ATTENDENCE MANGEMENT SYSTEM**

Submitted by:

VTUNO	REGISTER NUMBER	STUDENT NAME
29186	24UECS0218	M.RAMANJANEYULU
29353	24UECS1009	Y.GOWTHAM
29255	24UECS0866	R.KARUNAKAR REDDY
29314	24UECS1070	O.HARSHITH KUMAR
29175	24UECS1021	Y.KOUSHIK

Under the guidance of:
Dr V Senthil kumar

INDEX **PAGE**

1. Introduction

In educational institutions, maintaining student attendance is an essential but repetitive task. Traditionally, attendance is recorded manually on paper registers, which is time-consuming and prone to human error.
The **Student Attendance Management System (SAMS)** aims to automate this process. It provides a structured and reliable way to record, manage, and analyze attendance data digitally.

This system enables faculty to mark attendance quickly, retrieve attendance history at any time, and generate accurate reports. It also provides a foundation for further integration with other academic systems such as marks management or student portals.

2.Problem Statement

Manual attendance systems lead to:

- Data loss or misplacement of attendance records.
- Difficulty in calculating attendance percentage.
- Errors during report generation.
- Inefficiency in managing large student datasets.

Hence, there is a need for a **computerized attendance management system** that is reliable, fast, and accurate, with both relational (SQL) and NoSQL database support.

.

3.Objectives

The main objectives of this project are:

- To design and implement a **database-driven attendance management system**.
- To store and retrieve attendance records efficiently using **Oracle (SQL)**.
- To demonstrate integration with **MongoDB (NoSQL)** for flexible document-based storage.
- To minimize manual work and ensure data accuracy.
- To generate daily and summary reports easily.

4.System Requirements

Hardware Requirements

Component	Specification
Processor	Intel i3 or higher
RAM	4 GB or above
Storage	Minimum 250 GB HDD / SSD
Display	14” or above
Input Devices	Keyboard and Mouse

Software Requirements

Component	Specification
Operating System	Windows / Linux
Programming Language	SQL / Python
Database	Oracle, MongoDB
Tools	SQL Developer / VS Code
Connectivity	Internet for MongoDB integration

5. System Analysis and Design

System Overview

The system consists of two main entities:

- **Student:** Contains student details such as Roll Number, Name, Class, and Department.
- **Attendance:** Stores the daily attendance status of each student.

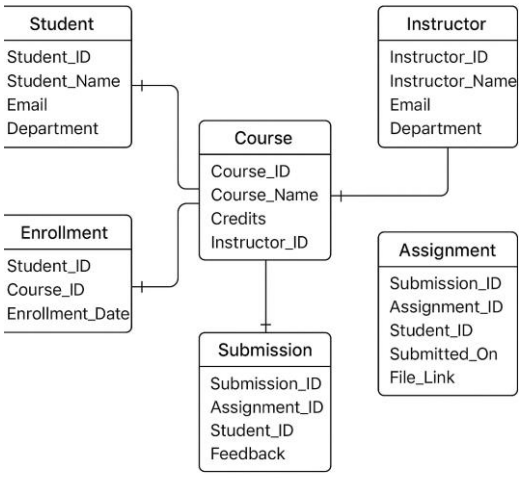
Functionalities

1. Add new student records.
2. Mark attendance for each student daily.
3. View attendance reports by date or student.

4. Integrate relational (Oracle) and non-relational (MongoDB) data for better scalability.

Design Approach

- **Input:** Student details and attendance status.
- **Process:** Validation, storage, and report generation.
- **Output:** Attendance summary and individual reports.

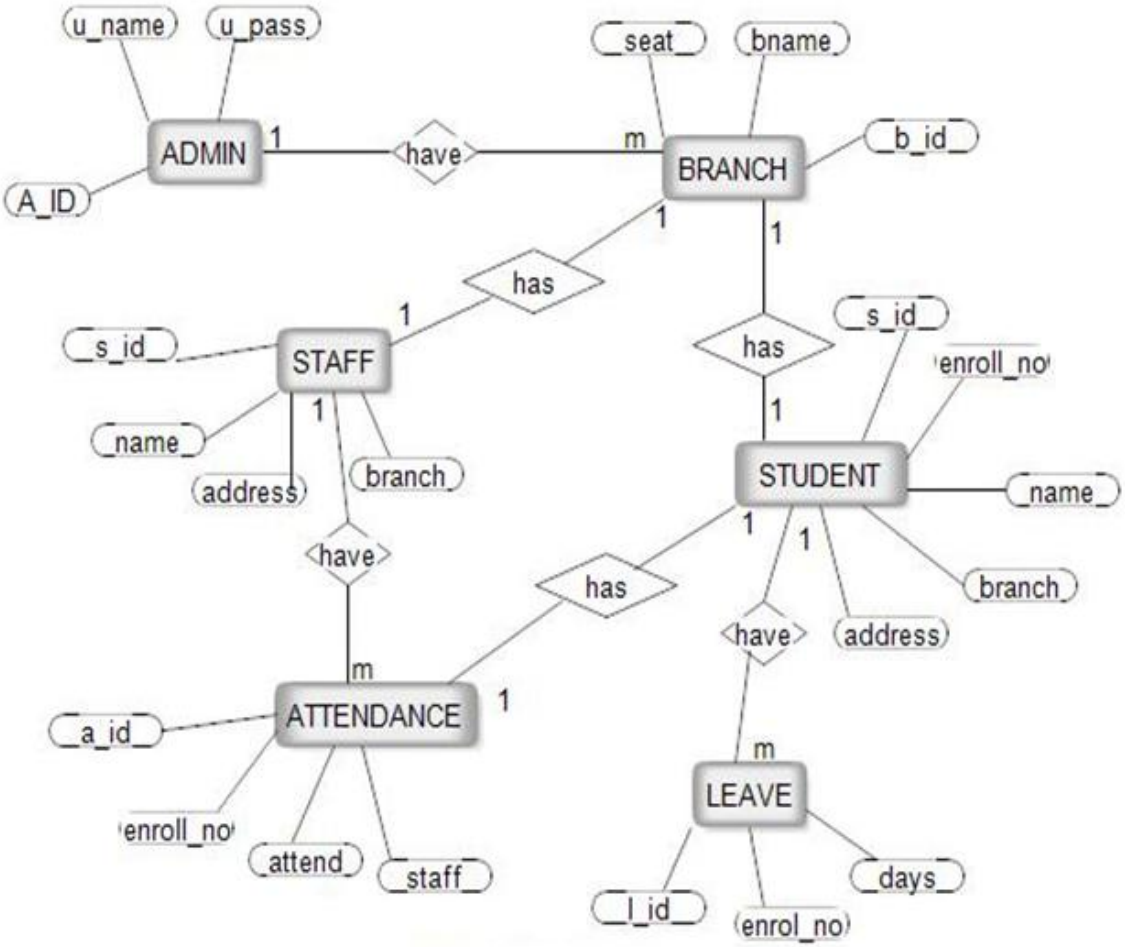


6. ER Diagram (Conceptual Design)

Relationships:

- A Student can enroll in multiple Courses.
- Each Course is handled by one Instructor.
- Each Course contains multiple Assignments.
- Each Assignment receives multiple Submissions.
- Each Submission is evaluated and assigned a Grade.

Figure: ER Diagram



E- R Diagram

Student Attendance Management System

Entities and Attributes

- **Student**
  - StudentID (Primary Key)
  - Name

- FirstName
- LastName
- DateOfBirth
- ContactNumber
- ClassID (Foreign Key)

- **Teacher**
  - TeacherID (Primary Key)
  - Name
  - Email
  - PhoneNumber
- **Course/Subject**
  - SubjectID (Primary Key)
  - SubjectName
  - SubjectCode
  - Credit
- **Class**
  - ClassID (Primary Key)
  - ClassName
  - Department
  - Year
- **Attendance**
  - AttendanceID (Primary Key)
  - StudentID (Foreign Key)

- SubjectID (Foreign Key)
- ClassID (Foreign Key)
- Date
- Status (e.g., Present, Absent)
- Timestamp
- TeacherID (Foreign Key)

7. Schema Design (Oracle)

```
-- Create Student Table
CREATE TABLE STUDENT (
   Roll_No NUMBER PRIMARY KEY,
   Name VARCHAR2(50),
   Class VARCHAR2(10),
   Department VARCHAR2(20)
);

-- Create Attendance Table
CREATE TABLE ATTENDANCE (
   A_ID NUMBER PRIMARY KEY,
   Roll_No NUMBER REFERENCES STUDENT(Roll_No),
   A_Date DATE,
```

Status VARCHAR2(10)

);

SAMPLE DATA

INSERT INTO STUDENT VALUES (101, 'Ravi', 'III-CSE', 'CSE');

INSERT INTO STUDENT VALUES (102, 'Kiran', 'III-CSE', 'CSE');

INSERT INTO ATTENDANCE VALUES (1, 101, TO_DATE('2025-10-26','YYYY-MM-DD'), 'Present');

INSERT INTO ATTENDANCE VALUES (2, 102, TO_DATE('2025-10-26','YYYY-MM-DD'), 'Absent');

8. NORMALIZATION

Normal Form	Description	Achieved By
1NF	Each cell contains atomic values	Dividing student and attendance info into two tables
2NF	All non-key attributes depend on the whole primary key	Primary keys properly defined

Normal Form	Description	Achieved By
3NF	No transitive dependency	Each table depends only on its key

Result: Database is normalized up to **Third Normal Form (3NF)** ensuring data integrity and minimal redundancy.

9.Implementation (SQL Queries)

a) Inserting Data

```
INSERT INTO STUDENT VALUES (103, 'Ramu', 'III-CSE', 'CSE');
INSERT INTO ATTENDANCE VALUES (3, 103, SYSDATE, 'Present');
```

b) Viewing Attendance

```
SELECT S.Roll_No, S.Name, A.A_Date, A.Status
FROM STUDENT S
JOIN ATTENDANCE A ON S.Roll_No = A.Roll_No;
```

c) Updating Attendance

```
UPDATE ATTENDANCE
SET Status = 'Absent'
WHERE Roll_No = 103 AND A_Date = SYSDATE;
```

d) Generating Attendance Summary

```
SELECT Roll_No, COUNT(Status) AS Total_Days,
SUM(CASE WHEN Status='Present' THEN 1 ELSE 0 END) AS Present_Days
FROM ATTENDANCE
GROUP BY Roll_No;
```

---

10. Input and Output

Input Screens

- Student Registration Form

- Attendance Marking Page (date-wise)

Sample Input

Roll_No	Name	Date	Status
101	Ravi	26-Oct-2025	Present
102	Kiran	26-Oct-2025	Absent

Expected Output

Roll_No	Name	Present_Days	Total_Days	Attendance %
101	Ravi	5	6	83.3%
102	Kiran	4	6	66.7%

11. Integration with MongoDB (NoSQL)

MongoDB Collection Structure

```
{
 "_id": "101",
 "Name": "Ravi",
 "Department": "CSE",
 "Attendance": [
   { "Date": "2025-10-26", "Status": "Present" },
   { "Date": "2025-10-27", "Status": "Absent" }
```

```
  ]
}
```

MongoDB Commands

```
// Insert student record
db.students.insertOne({
  Roll_No: 101,
  Name: "Ravi",
  Attendance: [{ Date: "2025-10-26", Status: "Present" }]
});

// View all attendance
db.students.find({}, { _id: 0, Name: 1, Attendance: 1 });
```

MongoDB allows flexible storage, making it easier to handle dynamic attendance records for various subjects or semesters.

---

12. Results and Discussion

The system was tested with multiple student records. The Oracle backend efficiently handled structured data, while MongoDB allowed for flexible NoSQL integration.

- Attendance records were accurately stored and retrieved.
- Reports were generated instantly.

- The hybrid database model improved data scalability and accessibility.

This system successfully demonstrated automation, data consistency, and user-friendliness.

13. Conclusion

The **Student Attendance Management System** efficiently automates the attendance process, reducing manual workload and improving accuracy. Integration of SQL (Oracle) and NoSQL (MongoDB) demonstrates modern database handling. The project can be extended with a **web or mobile interface** in the future to enable remote attendance tracking and real-time analytics.

14. References

- Oracle Database Documentation – Oracle Corporation
- MongoDB Manual – MongoDB Inc.
- W3Schools SQL and Python Tutorials
- Database System Concepts – Korth & Silberschatz
- Vel Tech University DBMS Lab Notes