Data Base Management System

Project Proposal

School Management System

Submitted by:

Name:	M.Kamran	2024-SE-24
Name:	M.Ahsan Sadiq	2024-SE-40
Name:	Aaon Muhammad	2024-SE-45

Submitted to:

Miss. Rimsha Noreen

Dated: 14th April 2024

Department of Computer Science

University of Engineering and Technology Lahore, New Campus.

SCHOOL MANAGEMENT SYSTEM

1. PROJECT OVERVIEW

The goal of this project is to design and implement a robust SQL Server database for a School Management System that effectively manages student records, staff, courses, grades, attendance, and administrative operations. This system will centralize all school-related data to enhance accessibility, consistency, and efficiency.

---

2. OBJECTIVES

- To store and manage information on students, teachers, courses, and classes.
- To track student performance, attendance, and grades.
- To automate administrative tasks such as admissions, scheduling, and reporting.
- To ensure secure access and maintain data integrity.

---

3. KEY MODULES

The proposed system will consist of the following core modules:

1. **Student Management**
2. **Staff/Teacher Management**
3. **Class and Subject Management**
4. **Attendance Tracking**
5. **Grading and Results**
6. **Timetable and Scheduling**
7. **User Roles and Authentication**

---

4. TECHNOLOGY STACK

- **Database**: Microsoft SQL Server

---

5. PROPOSED DATABASE SCHEMA

5.1 TABLES OVERVIEW

Table Name	**Description**
Students	Stores student personal data
Teachers	Stores teacher personal data
Classes	Holds class information
Subjects	List of subjects offered
Enrolments	Links students to subjects
Attendance	Tracks student attendance
Grades	Stores exam and assessment results
Users	Login credentials and roles
Timetable	Class schedule per subject/teacher

5.2 EXAMPLE TABLE STRUCTURES

STUDENTS

```
CREATE TABLE Students (

  StudentID INT PRIMARY KEY IDENTITY,

  FirstName VARCHAR(50),

  LastName VARCHAR(50),

  DateOfBirth DATE,

  Gender VARCHAR(10),

  ClassID INT FOREIGN KEY REFERENCES Classes(ClassID),

  AdmissionDate DATE,
  Email VARCHAR(100),
  Phone VARCHAR(20),
  Address VARCHAR(255)
);
```

TEACHERS

```
CREATE TABLE Teachers (
    TeacherID INT PRIMARY KEY IDENTITY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    Phone VARCHAR(20),
    SubjectID INT FOREIGN KEY REFERENCES Subjects(SubjectID)
);
```

SUBJECTS

```
CREATE TABLE Subjects (
    SubjectID INT PRIMARY KEY IDENTITY,
    SubjectName VARCHAR(100),
    ClassLevel VARCHAR(50)
);
```

ENROLMENT

```
CREATE TABLE Enrollments (
    EnrollmentID INT PRIMARY KEY IDENTITY,
    StudentID INT,
    SubjectID INT,
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (SubjectID) REFERENCES Subjects(SubjectID)
);
```

GRADES

```
CREATE TABLE Grades (
    GradeID INT PRIMARY KEY IDENTITY,
    StudentID INT FOREIGN KEY REFERENCES Students(StudentID),
    SubjectID INT FOREIGN KEY REFERENCES Subjects(SubjectID),
    ExamScore DECIMAL(5,2),
    Term VARCHAR(50),
    Year INT
);
```

ATTENDANCE

```
CREATE TABLE Attendance (
    AttendanceID INT PRIMARY KEY IDENTITY,
    StudentID INT FOREIGN KEY REFERENCES Students(StudentID),
    Date DATE,
    Status VARCHAR(10) CHECK (Status IN ('Present', 'Absent', 'Late')),
    RecordedBy INT FOREIGN KEY REFERENCES Teachers(TeacherID)
);
```

USERS

```
CREATE TABLE Users (
    UserID INT PRIMARY KEY IDENTITY,
    Username VARCHAR(50) UNIQUE,
    PasswordHash VARCHAR(255),
    Role VARCHAR(20),
    LinkedID INT
);
```

TIME TABLE

```
CREATE TABLE Timetable (
    TimetableID INT PRIMARY KEY IDENTITY,
    ClassID INT,
    SubjectID INT,
    TeacherID INT,
    DayOfWeek VARCHAR(20),
    StartTime TIME,
    EndTime TIME,
    FOREIGN KEY (ClassID) REFERENCES Classes(ClassID),
    FOREIGN KEY (SubjectID) REFERENCES Subjects(SubjectID),
    FOREIGN KEY (TeacherID) REFERENCES Teachers(TeacherID)
);
```

6. DETAILED ENTITY RELATIONSHIP DESCRIPTION

- **Students** are linked to **Classes** (Many-to-One).
- **Teachers** teach **Subjects** (Many-to-Many through Timetable).
- **Students** enroll in **Subjects** (Many-to-Many via Enrollments).
- **Grades** are assigned to **Students** for **Subjects** (Many-to-Many with attributes).
- **Attendance** tracks daily student presence, linked to **Students** and recorded by **Teachers**.
- **Users** can be linked to either a **Student** or **Teacher** based on their role.
- **Timetable** schedules subjects taught by **Teachers** to **Classes**.

9. HOW VIEWS AND TRIGGERS ARE USED IN THE PROJECT

VIEWS IN ACTION:

- **Access Control:** Teachers and students can access simplified reports through views like vw_StudentGradesSummary, rather than querying tables directly.
- **Reports and Dashboards:** Admin dashboards can use vw_ClassAttendance to monitor overall attendance without repeating joins.

TRIGGERS IN ACTION:

- **Data Integrity:** trg_PreventClassDelete protects against accidental deletion of classes that are still assigned to students.
- **Automation:** Triggers ensure that routine checks and log entries happen automatically, minimizing human error and administrative overhead.

7. SECURITY CONSIDERATIONS

- Implement proper user roles and access control.
- Use password hashing and data encryption where necessary.
- Ensure regular backups and disaster recovery protocols.

---

8. BENEFITS

- Centralized and organized data management.
- Improved tracking of student performance and attendance.
- Efficient administrative processes.
- Easy reporting and analytics.
