STUDENT MANAGEMENT SYSTEM - FULL CRUD PROJECT DOCUMENTATION

1. Introduction

- This project is a console-based Student Management System developed using Java and-MySQL.

- It allows users to perform complete CRUD operations (Create, Read, Update, Delete) on student data such as name, age, and grade.

- The system is built using Core Java, JDBC (Java Database Connectivity), and MySQL as the backend database.

2. Objective

- The main objective of this project is to build a robust backend system that connects Java with MySQL using JDBC and allows users to manage student records.

- This helps demonstrate database connectivity, SQL query handling, exception management, and object-oriented design in Java.

3. Technologies Used

- Core Java
- JDBC (Java Database Connectivity)
- MySQL (Database)
- Eclipse IDE

4. Database Design

- The database used in this project is named 'student_db'. It contains a single table named 'students' with the following schema:

```
CREATE TABLE students (
   id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(100),
   age INT,
   grade VARCHAR(10)
);
```

5. Project Structure

The project contains the following Java classes:

- **Student.java:** A POJO class that represents the student model, including both constructors for insert and update operations.
- **DBUtil.java:** Manages the database connection using JDBC.
- **StudentDAO.java:** Contains methods to perform CRUD operations on the database.
- **Main.java:** The entry point of the application that provides a menu-driven user interface.

6. Step-by-Step Execution

Step 1: Create the MySQL database and table using the following commands:

```
CREATE DATABASE student_db;
USE student_db;
CREATE TABLE students (
   id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(100),
   age INT,
   grade VARCHAR(10)
);
```

Step 2: Set up the project in Eclipse IDE and add the MySQL Connector JAR to the build path.

Step 3: Implement the following Java classes:

- Student.java with constructors for both add and update

- DBUtil.java to establish the connection

- StudentDAO.java with methods: addStudent(), displayAllStudents(), updateStudent(), deleteStudent()

- Main.java to provide the console-based menu interface

Step 4: Run the program. Use the menu to perform Create, Read, Update, and Delete operations.

7. Sample Output

==== Student Management System ====
1. Add Student
2. View All Students
3. Update Student
4. Delete Student
5. Exit
Choose: 1
Enter Name: Avinash
Enter Age: 21
Enter Grade: B.Tech
 Student added successfully

8. CRUD Methods Overview

- addStudent(Student s): Inserts a new student record into the database.
- displayAllStudents(): Fetches and displays all student records.
- updateStudent(Student s): Updates student record based on ID.
- deleteStudent(int id): Deletes a student record by ID.

9. Future Enhancements

- Add login authentication for admin.
- Build a GUI using Java Swing or JavaFX.
- Deploy it as a web application using JSP/Servlets.
- Add input validation and better error handling.

10. Conclusion

This full CRUD Student Management System demonstrates a complete Java-MySQL backend workflow. It offers hands-on experience with JDBC, SQL, object-oriented programming, and modular design suitable for resume projects and interview preparation.