EXPERIMENT- 7

Student Name: Shivam **UID: 23BCS80044**

Branch: CSE **Section/Group: 642/B**

Semester: 6th **Date of Performance: 17/03/25**

Subject Name: PBLJ **Subject Code:22CSH-359**

EASY LEVEL

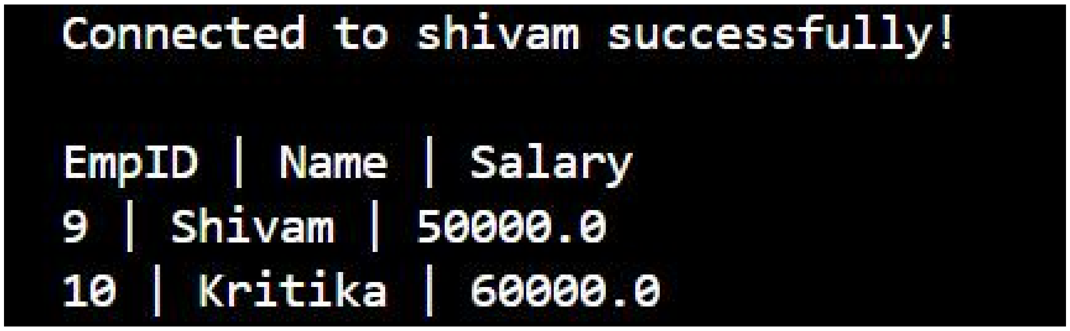
1. **Aim**: Create a Java program to connect to a MySQL database and fetch data from a single table.

2. **Objective:** To retrieve and display all records from a table named Employee with columns EmpID, Name, and Salary.

3. **Implementation/Code:**

```
import java.sql.*;

public class App {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/?serverTimezone=UTC"; // Connect without selecting a DB first
        String dbUrl = "jdbc:mysql://localhost:3306/shivam?serverTimezone=UTC"; // URL with DB selected
        String user = "root";
        String password = ""; // Update with actual password if required

        try {
            // Step 1: Connect to MySQL without specifying a database
            Connection conn = DriverManager.getConnection(url, user, password);
            Statement stmt = conn.createStatement();

            // Step 2: Create Database if it does not exist
            String createDbSQL = "CREATE DATABASE IF NOT EXISTS shivam";
            stmt.executeUpdate(createDbSQL);

            // Close first connection (optional but recommended)
            stmt.close();
            conn.close();

            // Step 3: Reconnect, now specifying the 'shivam' database
            conn = DriverManager.getConnection(dbUrl, user, password);
            stmt = conn.createStatement();
            System.out.println("Connected to shivam successfully!\n");

            // Step 4: Create Employee Table (if it doesn't exist)
            String createTableSQL = "CREATE TABLE IF NOT EXISTS Employee ("
                    + "EmpID INT PRIMARY KEY AUTO_INCREMENT, "
                    + "Name VARCHAR(255), "
                    + "Salary DOUBLE)";
            stmt.executeUpdate(createTableSQL);

            // Step 5: Insert Sample Data
            String insertSQL = "INSERT INTO Employee (Name, Salary) VALUES ('Shivam', 50000), ('Kritika', 60000)";
            stmt.executeUpdate(insertSQL);

            // Step 6: Retrieve and Display Data
            ResultSet rs = stmt.executeQuery("SELECT * FROM Employee");
            System.out.println("EmpID | Name | Salary");
            while (rs.next()) {
                System.out.println(rs.getInt("EmpID") + " | " + rs.getString("Name") + " | " + rs.getDouble("Salary"));
            }

            // Step 7: Close resources
            rs.close();
            stmt.close();
            conn.close();

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

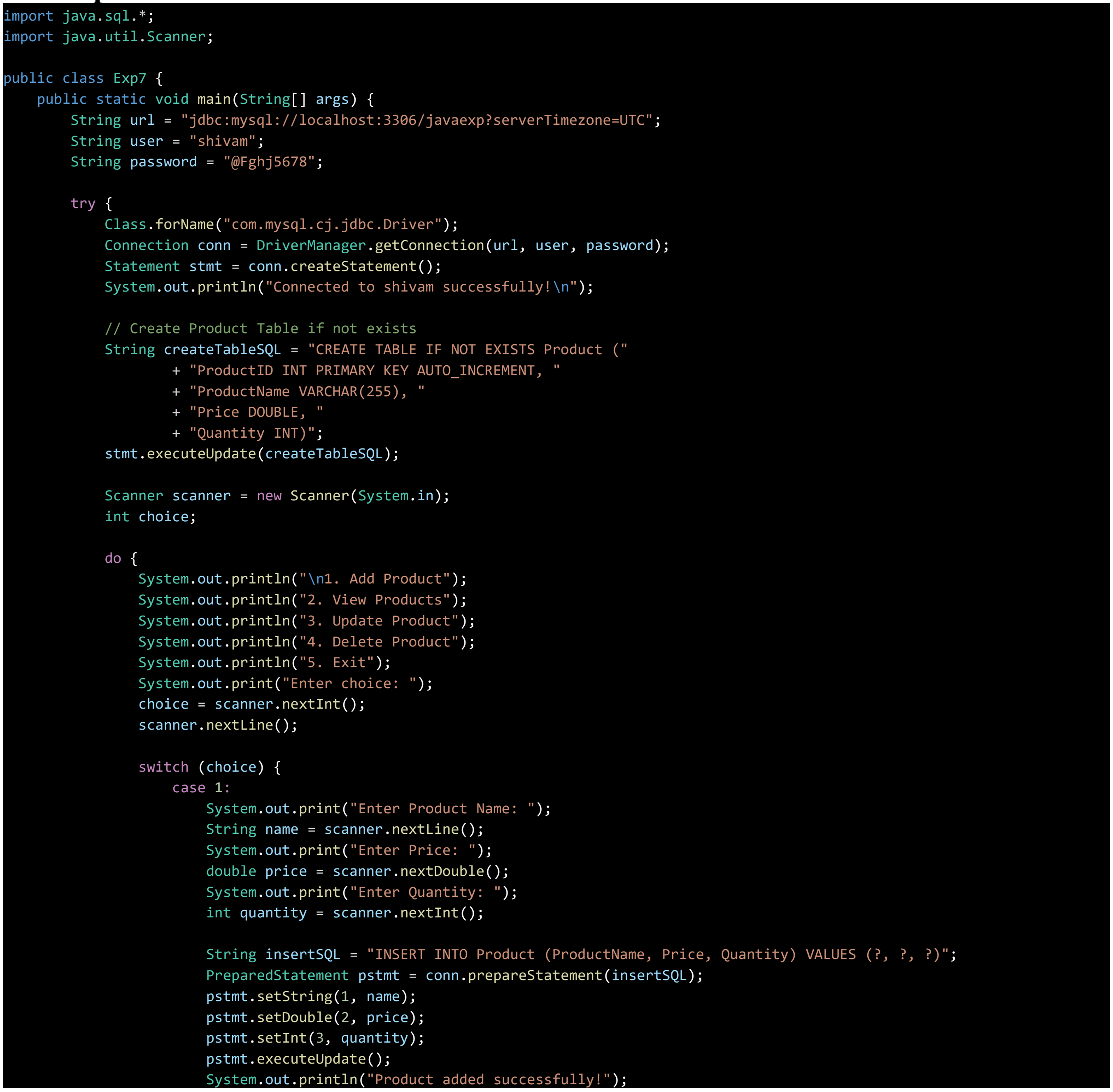
4. Output:



```
Connected to shivam successfully!

EmpID | Name | Salary
9 | Shivam | 50000.0
10 | Kritika | 60000.0
```

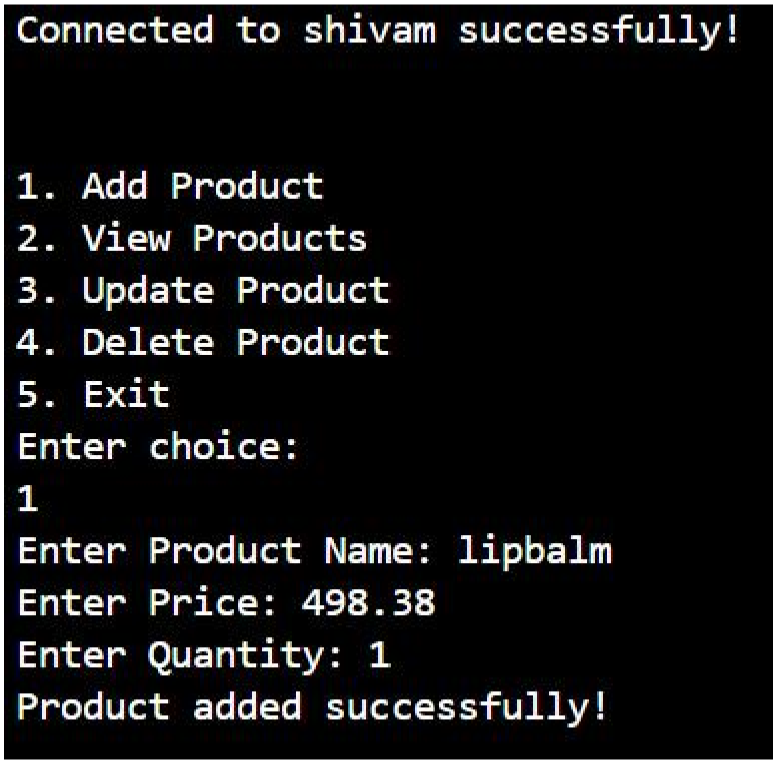
MEDIUM LEVEL

1. **Aim**: Build a program to perform CRUD operations
2. **Objective:** To perform Create, Read, Update, Delete on a database table Product with columns: ProductID, ProductName, Price, and Quantity. The program should include menu-driven options for each operation.
3. **Implementation/Code:**

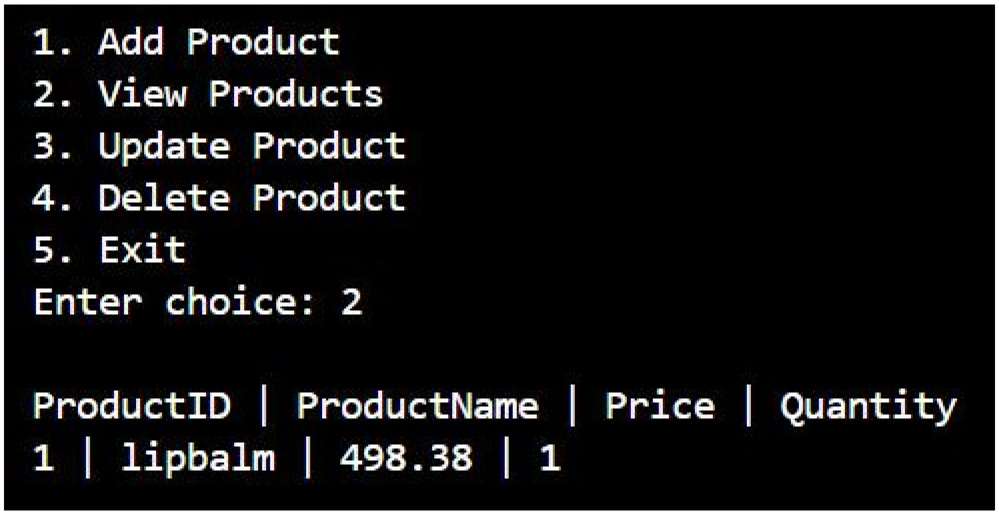


```java
import java.sql.*;
import java.util.Scanner;

public class Exp7 {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/javaexp?serverTimezone=UTC";
        String user = "shivam";
        String password = "@Fghj5678";

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection conn = DriverManager.getConnection(url, user, password);
            Statement stmt = conn.createStatement();
            System.out.println("Connected to shivam successfully!\n");

            // Create Product Table if not exists
            String createTableSQL = "CREATE TABLE IF NOT EXISTS Product ("
                    + "ProductID INT PRIMARY KEY AUTO_INCREMENT, "
                    + "ProductName VARCHAR(255), "
                    + "Price DOUBLE, "
                    + "Quantity INT)";
            stmt.executeUpdate(createTableSQL);

            Scanner scanner = new Scanner(System.in);
            int choice;

            do {
                System.out.println("\n1. Add Product");
                System.out.println("2. View Products");
                System.out.println("3. Update Product");
                System.out.println("4. Delete Product");
                System.out.println("5. Exit");
                System.out.print("Enter choice: ");
                choice = scanner.nextInt();
                scanner.nextLine();

                switch (choice) {
                    case 1:
                        System.out.print("Enter Product Name: ");
                        String name = scanner.nextLine();
                        System.out.print("Enter Price: ");
                        double price = scanner.nextDouble();
                        System.out.print("Enter Quantity: ");
                        int quantity = scanner.nextInt();

                        String insertSQL = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)";
                        PreparedStatement pstmt = conn.prepareStatement(insertSQL);
                        pstmt.setString(1, name);
                        pstmt.setDouble(2, price);
                        pstmt.setInt(3, quantity);
                        pstmt.executeUpdate();
                        System.out.println("Product added successfully!");
                        break;

                    case 2:
                        ResultSet rs = stmt.executeQuery("SELECT * FROM Product");
                        System.out.println("\nProductID | ProductName | Price | Quantity");
                        while (rs.next()) {
                            System.out.println(rs.getInt("ProductID") + " | " + rs.getString("ProductName") + " | " +
rs.getDouble("Price") + " | " + rs.getInt("Quantity"));
                        }
                        rs.close();
                        break;

                    case 3:
                        System.out.print("Enter Product ID to update: ");
                        int updateID = scanner.nextInt();
                        scanner.nextLine();
                        System.out.print("Enter New Name: ");
                        String newName = scanner.nextLine();
                        System.out.print("Enter New Price: ");
                        double newPrice = scanner.nextDouble();
                        System.out.print("Enter New Quantity: ");
                        int newQuantity = scanner.nextInt();

                        String updateSQL = "UPDATE Product SET ProductName=?, Price=?, Quantity=? WHERE ProductID=?";
                        pstmt = conn.prepareStatement(updateSQL);
                        pstmt.setString(1, newName);
                        pstmt.setDouble(2, newPrice);
                        pstmt.setInt(3, newQuantity);
                        pstmt.setInt(4, updateID);
                        pstmt.executeUpdate();
                        System.out.println("Product updated successfully!");
                        break;

                    case 4:
                        System.out.print("Enter Product ID to delete: ");
                        int deleteID = scanner.nextInt();

                        String deleteSQL = "DELETE FROM Product WHERE ProductID=?";
                        pstmt = conn.prepareStatement(deleteSQL);
                        pstmt.setInt(1, deleteID);
                        pstmt.executeUpdate();
                        System.out.println("Product deleted successfully!");
                        break;

                    case 5:
                        System.out.println("Exiting...");
                        break;

                    default:
                        System.out.println("Invalid choice. Try again.");
                }
            } while (choice != 5);

            scanner.close();
            conn.close();
        } catch (ClassNotFoundException e) {
            System.out.println("MySQL JDBC Driver not found!");
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

4. Output:



```
Connected to shivam successfully!


1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter choice:
1
Enter Product Name: lipbalm
Enter Price: 498.38
Enter Quantity: 1
Product added successfully!
```



```
1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter choice: 2

ProductID | ProductName | Price | Quantity
1 | lipbalm | 498.38 | 1
```

HARD LEVEL

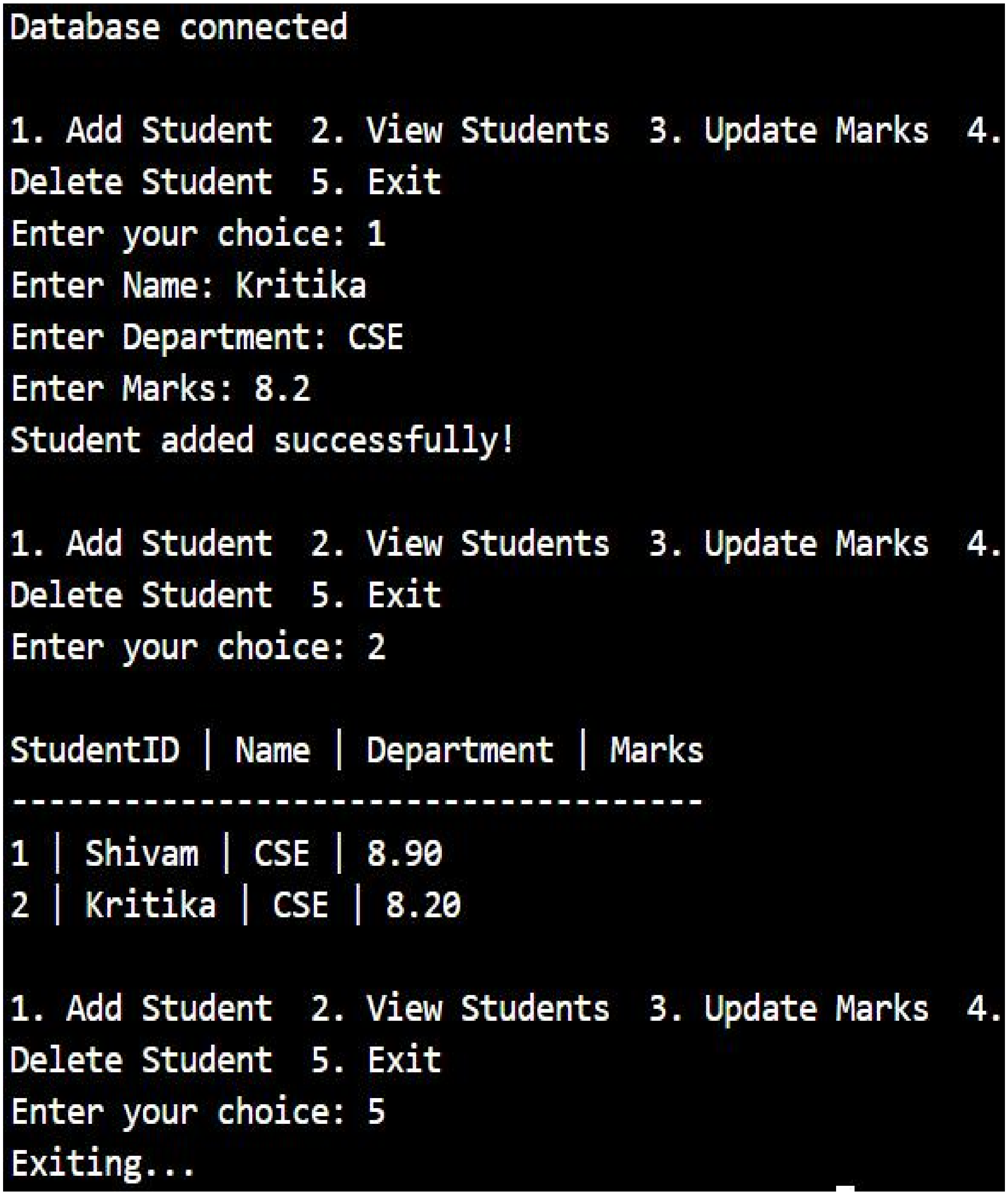
1. **Aim**: Develop a Java application using JDBC and MVC architecture to manage student data.

2. **Objective:** To Use a Student class as the model with fields like StudentID, Name, Department, and Marks. Include a database table to store student data.

3. **Implementation/Code:**

```java
import java.sql.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Exp2 {
    // Database Credentials
    private static final String URL = "jdbc:mysql://localhost:3306/StudentDB";
    private static final String USER = "root"; // Change as needed
    private static final String PASSWORD = ""; // Change as needed

    // Student Model
    static class Student {
        private int studentID;
        private String name;
        private String department;
        private double marks;

        public Student(int studentID, String name, String department, double marks) {
            this.studentID = studentID;
            this.name = name;
            this.department = department;
            this.marks = marks;
        }

        public int getStudentID() { return studentID; }
        public String getName() { return name; }
        public String getDepartment() { return department; }
        public double getMarks() { return marks; }

        public void setStudentID(int studentID) { this.studentID = studentID; }
        public void setName(String name) { this.name = name; }
        public void setDepartment(String department) { this.department = department; }
        public void setMarks(double marks) { this.marks = marks; }
    }

    // DAO (Database Access Object)
    static class StudentDAO {
        public Connection connect() throws SQLException {
            return DriverManager.getConnection(URL, USER, PASSWORD);
        }

        public void addStudent(Student student) throws SQLException {
            String sql = "INSERT INTO Student (Name, Department, Marks) VALUES (?, ?, ?)";
            try (Connection conn = connect(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
                pstmt.setString(1, student.getName());
                pstmt.setString(2, student.getDepartment());
                pstmt.setDouble(3, student.getMarks());
                pstmt.executeUpdate();
            }
        }

        public List<Student> getStudents() throws SQLException {
            List<Student> students = new ArrayList<>();
            String sql = "SELECT * FROM Student";
            try (Connection conn = connect(); Statement stmt = conn.createStatement(); ResultSet rs = stmt.executeQuery(sql)) {
                while (rs.next()) {
                    students.add(new Student(rs.getInt("StudentID"), rs.getString("Name"),
                                             rs.getString("Department"), rs.getDouble("Marks")));
                }
            }
            return students;
        }

        public void updateStudentMarks(int studentID, double newMarks) throws SQLException {
            String sql = "UPDATE Student SET Marks = ? WHERE StudentID = ?";
            try (Connection conn = connect(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
                pstmt.setDouble(1, newMarks);
                pstmt.setInt(2, studentID);
                pstmt.executeUpdate();
            }
        }

        public void deleteStudent(int studentID) throws SQLException {
            String sql = "DELETE FROM Student WHERE StudentID = ?";
            try (Connection conn = connect(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
                pstmt.setInt(1, studentID);
                pstmt.executeUpdate();
            }
        }
    }

    // Controller
    static class StudentController {
        private StudentDAO studentDAO = new StudentDAO();

        public void addStudent(Student student) throws SQLException {
            studentDAO.addStudent(student);
        }

        public List<Student> getStudents() throws SQLException {
            return studentDAO.getStudents();
        }

        public void updateStudentMarks(int studentID, double newMarks) throws SQLException {
            studentDAO.updateStudentMarks(studentID, newMarks);
        }

        public void deleteStudent(int studentID) throws SQLException {
            studentDAO.deleteStudent(studentID);
        }
    }

    // Main Menu (View)
    public static void main(String[] args) {
        try {
            StudentController controller = new StudentController();
            Scanner sc = new Scanner(System.in);
            System.out.println("Database connected");

            while (true) {
                System.out.println("\n1. Add Student  2. View Students  3. Update Marks  4. Delete Student  5. Exit");
                System.out.print("Enter your choice: ");
                int choice = sc.nextInt();
                sc.nextLine(); // Consume newline

                switch (choice) {
                    case 1:
                        System.out.print("Enter Name: ");
                        String name = sc.nextLine();

                        System.out.print("Enter Department: ");
                        String dept = sc.nextLine();
                        System.out.print("Enter Marks: ");
                        double marks = sc.nextDouble();

                        controller.addStudent(new Student(0, name, dept, marks));
                        System.out.println("Student added successfully!");
                        break;

                    case 2:
                        List<Student> students = controller.getStudents();
                        System.out.println("\nStudentID | Name | Department | Marks");
                        System.out.println("---------------------------------------");

                        for (Student s : students) {
                            System.out.printf("%d | %s | %s | %.2f\n",
                                    s.getStudentID(), s.getName(), s.getDepartment(), s.getMarks());
                        }
                        break;

                    case 3:
                        System.out.print("Enter StudentID to update: ");
                        int updateId = sc.nextInt();

                        System.out.print("Enter new Marks: ");
                        double newMarks = sc.nextDouble();

                        controller.updateStudentMarks(updateId, newMarks);
                        System.out.println("Student marks updated successfully!");
                        break;

                    case 4:
                        System.out.print("Enter StudentID to delete: ");
                        int deleteId = sc.nextInt();

                        controller.deleteStudent(deleteId);
                        System.out.println("Student deleted successfully!");
                        break;

                    case 5:
                        System.out.println("Exiting...");
                        sc.close();
                        return;

                    default:
                        System.out.println("Invalid choice. Please try again.");
                }
            }
        } catch (SQLException e) {
            System.err.println("Database error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

4. **Output:**



```
Database connected

1. Add Student  2. View Students  3. Update Marks  4.
Delete Student  5. Exit
Enter your choice: 1
Enter Name: Kritika
Enter Department: CSE
Enter Marks: 8.2
Student added successfully!

1. Add Student  2. View Students  3. Update Marks  4.
Delete Student  5. Exit
Enter your choice: 2

StudentID | Name | Department | Marks
-------------------------------------
1 | Shivam | CSE | 8.90
2 | Kritika | CSE | 8.20

1. Add Student  2. View Students  3. Update Marks  4.
Delete Student  5. Exit
Enter your choice: 5
Exiting...
```

5. Learning Outcomes:

(i) Learn how to **establish a connection** between a Java application and a MySQL database using **JDBC**.

(ii) Understand the use of **DriverManager and Connection objects** to interact with the database.

(iii) Learn to use **PreparedStatement** to securely execute SQL queries.