NAME	Prakash Joshi
UID	23BCS12438
CLASS	622-A

Experiment – 6 (Part – c) :

Title

Student Management Application Using JDBC and MVC Architecture.

Objective

To build a Java application that manages student data using JDBC and follows the **Model-View-Controller (MVC)** design pattern.

Task Description

This task focuses on building a well-structured Java application that performs database operations while adhering to software design principles:

- **Model (Student Class)**:
  - Fields: StudentID, Name, Department, Marks.
  - Represents the student data structure.
- **View (User Interface)**:

- Menu-driven system for:
  - Adding a new student.
  - Viewing all students.
  - Updating student details.
  - Deleting a student record.

- **Controller (Database Operations Class)**:
  - Contains JDBC logic for performing all CRUD operations.
  - Uses PreparedStatement for executing SQL commands securely.

This question integrates:

- Object-oriented programming with JDBC.
- MVC architecture for better separation of concerns.
- Robust input handling and database operations.

Code :

1. Model: Student.java

```
public class Student {
    private int studentID;
    private String name;
    private String department;
    private double marks;

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

```
    }

    // Getters and Setters
    public int getStudentID() { return studentID; }
    public void setStudentID(int studentID) { this.studentID = studentID; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getDepartment() { return department; }
    public void setDepartment(String department) { this.department =
department; }

    public double getMarks() { return marks; }
    public void setMarks(double marks) { this.marks = marks; }

    @Override
    public String toString() {
        return studentID + "\t" + name + "\t" + department + "\t" + marks;
    }
}
```

2. Controller: StudentController.java

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

public class StudentController {
    private final String url = "jdbc:mysql://localhost:3306/companydb";
    private final String username = "root";
    private final String password = "your_password"; // replace with your
MySQL password

    public StudentController() {
```

```
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            System.out.println("❌ MySQL JDBC Driver not found.");
        }
    }

    // CREATE
    public void addStudent(Student student) {
        String sql = "INSERT INTO Student (StudentID, Name, Department,
Marks) VALUES (?, ?, ?, ?)";
        try (Connection con = DriverManager.getConnection(url, username,
password);
             PreparedStatement ps = con.prepareStatement(sql)) {

            ps.setInt(1, student.getStudentID());
            ps.setString(2, student.getName());
            ps.setString(3, student.getDepartment());
            ps.setDouble(4, student.getMarks());

            int rows = ps.executeUpdate();
            System.out.println(rows + " student(s) added successfully!");

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

    // READ
    public List<Student> getAllStudents() {
        List<Student> students = new ArrayList<>();
        String sql = "SELECT * FROM Student";
        try (Connection con = DriverManager.getConnection(url, username,
password);
             Statement stmt = con.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")
            ));
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
    return students;
}

// UPDATE
public void updateStudent(Student student) {
    String sql = "UPDATE Student SET Name=?, Department=?, Marks=?
WHERE StudentID=?";
    try (Connection con = DriverManager.getConnection(url, username,
password);
         PreparedStatement ps = con.prepareStatement(sql)) {

        ps.setString(1, student.getName());
        ps.setString(2, student.getDepartment());
        ps.setDouble(3, student.getMarks());
        ps.setInt(4, student.getStudentID());

        int rows = ps.executeUpdate();
        if (rows > 0)
            System.out.println("☑ Student updated successfully!");
        else
            System.out.println("❌ Student not found.");
```

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

    // DELETE
    public void deleteStudent(int studentID) {
        String sql = "DELETE FROM Student WHERE StudentID=?";
        try (Connection con = DriverManager.getConnection(url, username,
password);
             PreparedStatement ps = con.prepareStatement(sql)) {

            ps.setInt(1, studentID);
            int rows = ps.executeUpdate();
            if (rows > 0)
                System.out.println("☑ Student deleted successfully!");
            else
                System.out.println("❌ Student not found.");

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

3. View: StudentView.java

```
import java.util.List;
import java.util.Scanner;

public class StudentView {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        StudentController controller = new StudentController();
```

```
        int choice;
        do {
            System.out.println("\n====== STUDENT MANAGEMENT MENU
======");
            System.out.println("1. Add Student");
            System.out.println("2. View All Students");
            System.out.println("3. Update Student");
            System.out.println("4. Delete Student");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();
            sc.nextLine(); // consume newline

            switch (choice) {
                case 1 -> {
                    System.out.print("Enter Student ID: ");
                    int id = sc.nextInt();
                    sc.nextLine();
                    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(id, name, dept, marks));
                }
                case 2 -> {
                    List<Student> students = controller.getAllStudents();
                    System.out.println("\nID\tName\tDepartment\tMarks");
                    System.out.println("----------------------------------------");
                    for (Student s : students) {
                        System.out.println(s);
                    }
                }
```

```
            case 3 -> {
                System.out.print("Enter Student ID to update: ");
                int id = sc.nextInt();
                sc.nextLine();
                System.out.print("Enter new Name: ");
                String name = sc.nextLine();
                System.out.print("Enter new Department: ");
                String dept = sc.nextLine();
                System.out.print("Enter new Marks: ");
                double marks = sc.nextDouble();

                controller.updateStudent(new Student(id, name, dept, marks));
            }
            case 4 -> {
                System.out.print("Enter Student ID to delete: ");
                int id = sc.nextInt();
                controller.deleteStudent(id);
            }
            case 5 -> System.out.println("👋 Exiting program...");
            default -> System.out.println("❌ Invalid choice. Try again.");
          }

        } while (choice != 5);

        sc.close();
    }
}
```

4. My SQL Table Sample :

```
CREATE DATABASE companydb;
USE companydb;

CREATE TABLE Student (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(50),
    Department VARCHAR(50),
    Marks DOUBLE
);
```

Output :