Experiment-7

1. Aim:

Create Java applications with JDBC for database connectivity, CRUD operations, and MVC architecture.

1. Create a Java program to connect to a MySQL database and fetch data from a single table.
   Program should:
   Use DriverManager and Connection objects.
   Retrieve and display all records from a table named Employee with column EMpID, Name, Salary.

2. Build a program to perform CRUD operations (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.
   Transaction handling to ensure data integrity.

3. Develop a Java application using JDBC and MVC architecture to manage student data.
   The application should:
   Use a Student class as the model with fields like StudentID, Name, Department, and Marks.
   Include a database table to store student data.
   Allow the user to perform CRUD operations through a simple menu-driven view.
   Implement database operations in a separate controller class.

2. Objective:

1. Connecting to a MySQL database: Using JDBC to establish a connection, retrieve all records from the Employee table, and display EmpID, Name, and Salary using DriverManager and ResultSet..

2. Performing CRUD operations: Implementing Create, Read, Update, and Delete functionalities on the Product table using a menu-driven approach while ensuring data integrity through transaction handling.

3. Building an MVC-based application: Developing a Java application that follows the Model-View-Controller architecture to manage student data, where the Student class represents the model, a database stores student records, and a separate controller handles database operations with a menu-driven interface.

3. Implementation/Code:

a.)

1. Sql Code:

```
CREATE DATABASE company_db;

USE company_db;

CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    Name VARCHAR(50),
    Salary DECIMAL(10,2)
);

INSERT INTO Employee (EmpID, Name, Salary) VALUES
(1, 'Manish La', 70000.00),
(2, 'Amit Sharma', 60000.00),
(3, 'Ramesh Gupta', 55000.00),
(4, 'Pooja Verma', 62000.00),
(5, 'Vikram Singh', 58000.00);
```

2. Java Code:

```
import java.sql.*;

public class FetchEmployee {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/company_db"; // Database URL
        String user = "root"; // MySQL username (change if needed)
        String password = "your_password"; // Your MySQL password
```

```
            try {
                // 1. Load MySQL JDBC Driver
                Class.forName("com.mysql.cj.jdbc.Driver");

                // 2. Establish Connection
                Connection con = DriverManager.getConnection(url, user, password);

                // 3. Create Statement
                Statement stmt = con.createStatement();

                // 4. Execute Query
                ResultSet rs = stmt.executeQuery("SELECT * FROM Employee");

                // 5. Display Results
                System.out.println("EmpID | Name        | Salary");
                System.out.println("-----------------------");
                while (rs.next()) {
                    System.out.println(rs.getInt("EmpID") + " | " +
                                       rs.getString("Name") + " | " +
                                       rs.getDouble("Salary"));
                }

                // 6. Close Connection
                rs.close();
                stmt.close();
                con.close();

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

b.)

1. Sql Code:

```
CREATE DATABASE store_db;
USE store_db;
```

```
CREATE TABLE Product (
    ProductID INT PRIMARY KEY AUTO_INCREMENT,
    ProductName VARCHAR(100),
    Price DECIMAL(10,2),
    Quantity INT
);

INSERT INTO Product (ProductName, Price, Quantity) VALUES
('Laptop', 50000.00, 10),
('Smartphone', 20000.00, 20),
('Tablet', 15000.00, 15),
('Headphones', 3000.00, 25),
('Smartwatch', 8000.00, 18);
```

2. Java Code:

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

public class ProductCRUD {
    static final String URL = "jdbc:mysql://localhost:3306/store_db"; // Database URL
    static final String USER = "root"; // MySQL username
    static final String PASSWORD = "your_password"; // Your MySQL password

    public static void main(String[] args) {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver"); // Load MySQL JDBC Driver
            Connection con = DriverManager.getConnection(URL, USER, PASSWORD);
            Scanner scanner = new Scanner(System.in);
            int choice;

            do {
                System.out.println("\nProduct CRUD Operations:");
                System.out.println("1. Insert 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 your choice: ");
                choice = scanner.nextInt();

                switch (choice) {
                    case 1:
                        insertProduct(con, scanner);
                        break;
                    case 2:
                        viewProducts(con);
                        break;
                    case 3:
                        updateProduct(con, scanner);
                        break;
                    case 4:
                        deleteProduct(con, scanner);
                        break;
                    case 5:
                        System.out.println("Exiting...");
                        break;
                    default:
                        System.out.println("Invalid choice. Try again.");
                }
            } while (choice != 5);

            scanner.close();
            con.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    // Method to Insert Product
    private static void insertProduct(Connection con, Scanner scanner) throws SQLException
{
        System.out.print("Enter Product Name: ");
        scanner.nextLine();
        String name = scanner.nextLine();
        System.out.print("Enter Price: ");
        double price = scanner.nextDouble();
        System.out.print("Enter Quantity: ");
        int quantity = scanner.nextInt();

        String sql = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?,
?)";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, name);
        pstmt.setDouble(2, price);
        pstmt.setInt(3, quantity);
        pstmt.executeUpdate();
        System.out.println("Product inserted successfully.");
    }

    // Method to View Products
    private static void viewProducts(Connection con) throws SQLException {
        String sql = "SELECT * FROM Product";
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery(sql);

        System.out.println("\nProductID | Product Name  | Price  | Quantity");
        System.out.println("-----------------------------");
        while (rs.next()) {
            System.out.println(rs.getInt("ProductID") + " | " +
                               rs.getString("ProductName") + " | " +
                               rs.getDouble("Price") + " | " +
                               rs.getInt("Quantity"));
        }
    }

    // Method to Update Product
    private static void updateProduct(Connection con, Scanner scanner) throws SQLException
{
        System.out.print("Enter ProductID to update: ");
        int productId = scanner.nextInt();
        System.out.print("Enter new Price: ");
        double newPrice = scanner.nextDouble();
        System.out.print("Enter new Quantity: ");
        int newQuantity = scanner.nextInt();
        String sql = "UPDATE Product SET Price = ?, Quantity = ? WHERE ProductID = ?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setDouble(1, newPrice);
        pstmt.setInt(2, newQuantity);
        pstmt.setInt(3, productId);
        int rowsAffected = pstmt.executeUpdate();

        if (rowsAffected > 0) {
            System.out.println("Product updated successfully.");
        } else {
            System.out.println("Product not found.");
        }
    }

    // Method to Delete Product
    private static void deleteProduct(Connection con, Scanner scanner) throws SQLException
{
        System.out.print("Enter ProductID to delete: ");
        int productId = scanner.nextInt();

        String sql = "DELETE FROM Product WHERE ProductID = ?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setInt(1, productId);
        int rowsAffected = pstmt.executeUpdate();

        if (rowsAffected > 0) {
            System.out.println("Product deleted successfully.");
        } else {
            System.out.println("Product not found.");
        }
    }
}
```

c.)

1. Sql Code:

```
CREATE DATABASE university_db;
USE university_db;
```

```
CREATE TABLE Student (
    StudentID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Department VARCHAR(50),
    Marks INT
);

INSERT INTO Student (Name, Department, Marks) VALUES
('Manish lalwani', 'Computer Science', 85),
('Amit Sharma', 'Mechanical Engineering', 78),
('Ramesh Gupta', 'Civil Engineering', 82),
('Pooja Verma', 'Electrical Engineering', 90),
('Vikram Singh', 'Information Technology', 88);
```

2. Java Code:

i. **For Model:**

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

    public Student(int studentID, String name, String department, int 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 int getMarks() { return marks; }

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

ii. **For Controller:**

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

public class StudentController {
   private static final String URL = "jdbc:mysql://localhost:3306/university_db";
   private static final String USER = "root";
   private static final String PASSWORD = "your_password";

   public Connection connect() throws SQLException {
       return DriverManager.getConnection(URL, USER, PASSWORD);
   }

   // Insert Student
   public void insertStudent(String name, String department, int marks) {
       String sql = "INSERT INTO Student (Name, Department, Marks) VALUES (?,
?, ?)";
       try (Connection con = connect(); PreparedStatement pstmt =
con.prepareStatement(sql)) {
           pstmt.setString(1, name);
           pstmt.setString(2, department);
           pstmt.setInt(3, marks);
           pstmt.executeUpdate();
           System.out.println("Student inserted successfully.");
       } catch (SQLException e) {
           e.printStackTrace();
       }
   }

   // Fetch All Students
   public List<Student> getAllStudents() {
       List<Student> students = new ArrayList<>();
       String sql = "SELECT * FROM Student";
       try (Connection con = connect(); Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(sql)) {
```

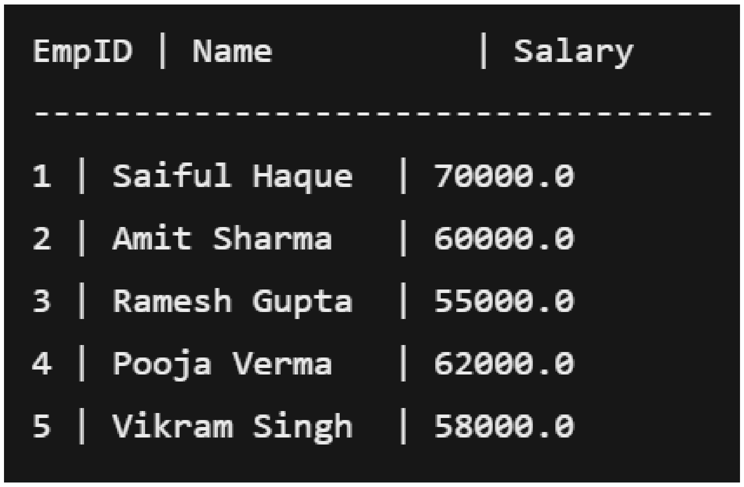
```
            while (rs.next()) {
                students.add(new Student(rs.getInt("StudentID"), rs.getString("Name"),
rs.getString("Department"), rs.getInt("Marks")));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return students;
    }

    // Update Student Marks
    public void updateStudent(int studentID, int newMarks) {
        String sql = "UPDATE Student SET Marks = ? WHERE StudentID = ?";
        try (Connection con = connect(); PreparedStatement pstmt =
con.prepareStatement(sql)) {
            pstmt.setInt(1, newMarks);
            pstmt.setInt(2, studentID);
            int rowsUpdated = pstmt.executeUpdate();
            if (rowsUpdated > 0) System.out.println("Student updated successfully.");
            else System.out.println("Student not found.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Delete Student
    public void deleteStudent(int studentID) {
        String sql = "DELETE FROM Student WHERE StudentID = ?";
        try (Connection con = connect(); PreparedStatement pstmt =
con.prepareStatement(sql)) {
            pstmt.setInt(1, studentID);
            int rowsDeleted = pstmt.executeUpdate();
            if (rowsDeleted > 0) System.out.println("Student deleted successfully.");
            else System.out.println("Student not found.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }}
```

iii. For View:

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

public class StudentApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        StudentController controller = new StudentController();
        int choice;

        do {
            System.out.println("\nStudent Management System:");
            System.out.println("1. Insert Student");
            System.out.println("2. View Students");
            System.out.println("3. Update Student Marks");
            System.out.println("4. Delete Student");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter Student Name: ");
                    scanner.nextLine();
                    String name = scanner.nextLine();
                    System.out.print("Enter Department: ");
                    String department = scanner.nextLine();
                    System.out.print("Enter Marks: ");
                    int marks = scanner.nextInt();
                    controller.insertStudent(name, department, marks);
                    break;

                case 2:
                    List<Student> students = controller.getAllStudents();
                    System.out.println("\nStudentID | Name | Department | Marks");
                    System.out.println("-------------------------");
                    for (Student s : students) {
                        System.out.println(s);
                    }
                    break;

                case 3:
                    System.out.print("Enter StudentID to update: ");
                    int studentID = scanner.nextInt();
                    System.out.print("Enter new Marks: ");
                    int newMarks = scanner.nextInt();
                    controller.updateStudent(studentID, newMarks);
                    break;

                case 4:
                    System.out.print("Enter StudentID to delete: ");
                    int deleteID = scanner.nextInt();
                    controller.deleteStudent(deleteID);
                    break;

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

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

        scanner.close();
    }
}
```

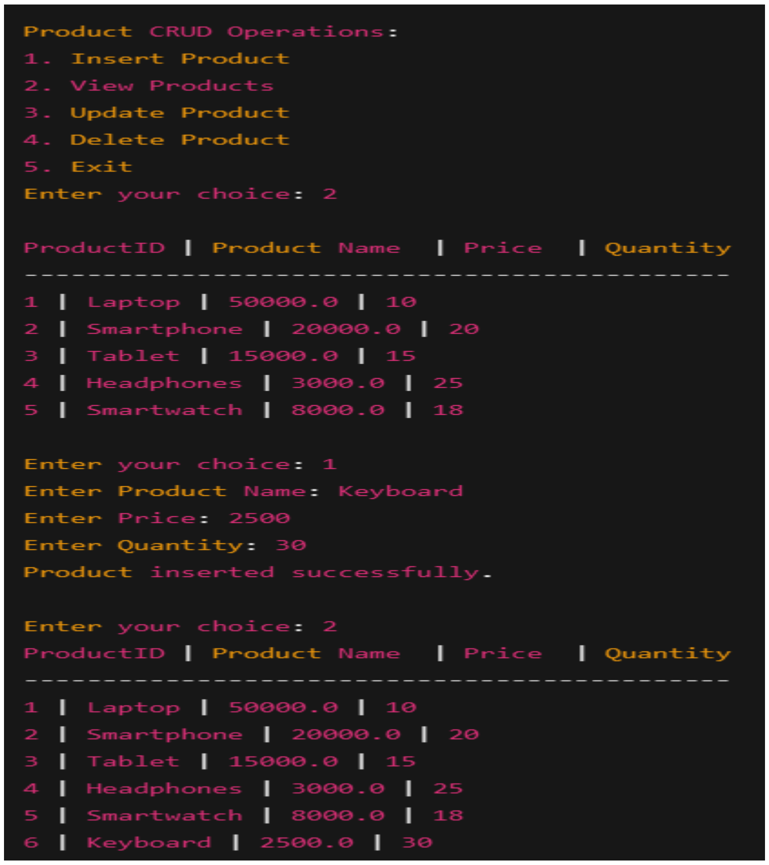
4. Output:

a.)



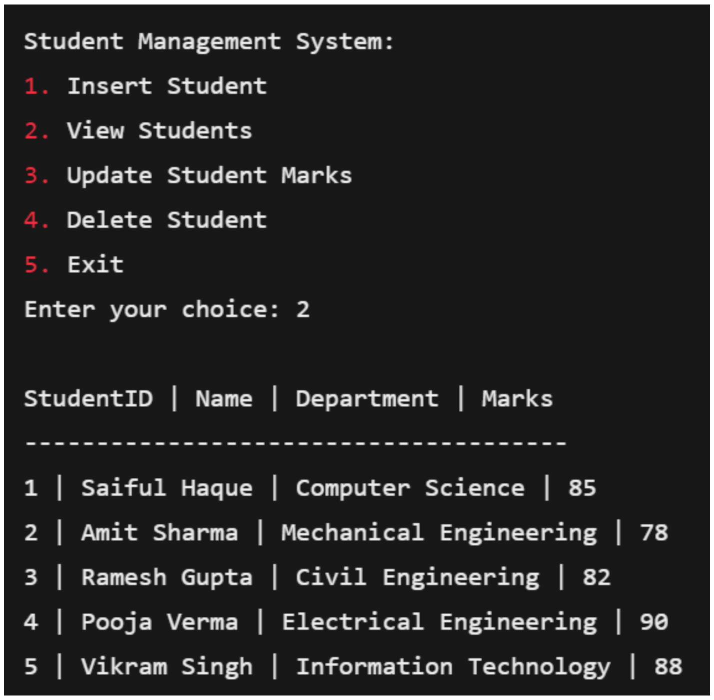
```
EmpID | Name            | Salary
-----------------------------------
1 | Saiful Haque  | 70000.0
2 | Amit Sharma   | 60000.0
3 | Ramesh Gupta  | 55000.0
4 | Pooja Verma   | 62000.0
5 | Vikram Singh  | 58000.0
```

b.)



```
Product CRUD Operations:
1. Insert Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 2

ProductID | Product Name  | Price   | Quantity
---------------------------------------------
1 | Laptop | 50000.0 | 10
2 | Smartphone | 20000.0 | 20
3 | Tablet | 15000.0 | 15
4 | Headphones | 3000.0 | 25
5 | Smartwatch | 8000.0 | 18

Enter your choice: 1
Enter Product Name: Keyboard
Enter Price: 2500
Enter Quantity: 30
Product inserted successfully.

Enter your choice: 2
ProductID | Product Name  | Price   | Quantity
---------------------------------------------
1 | Laptop | 50000.0 | 10
2 | Smartphone | 20000.0 | 20
3 | Tablet | 15000.0 | 15
4 | Headphones | 3000.0 | 25
5 | Smartwatch | 8000.0 | 18
6 | Keyboard | 2500.0 | 30
```

c.)



```
Student Management System:
1. Insert Student
2. View Students
3. Update Student Marks
4. Delete Student
5. Exit
Enter your choice: 2

StudentID | Name | Department | Marks
--------------------------------------
1 | Saiful Haque | Computer Science | 85
2 | Amit Sharma | Mechanical Engineering | 78
3 | Ramesh Gupta | Civil Engineering | 82
4 | Pooja Verma | Electrical Engineering | 90
5 | Vikram Singh | Information Technology | 88
```

5. Learning Outcome:

1. Established a connection between Java and MySQL using JDBC for efficient database interaction, including retrieving, updating, and managing records.
2. Implemented CRUD (Create, Read, Update, Delete) operations on MySQL tables using SQL queries and a user-friendly menu-driven approach.
3. Applied transaction handling techniques to ensure data integrity during insert, update, and delete operations.
4. Developed a structured MVC-based student management system, improving modularity and maintainability by separating model, view, and controller components.
5. Utilized PreparedStatement to prevent SQL injection and ResultSet to efficiently process query results.
6. Enhanced problem-solving and database management skills by integrating Java programming concepts with real-world database operations.