Experiment 7

Student Name: Agraj Chandra **UID: 22BCS16738**
Branch: CSE – General **Section/Group:22BCSIOT_617_B**
Semester: VIth **Date of Performance: 20/03/25**
Subject Name: Project based learning java **Subject Code:22CSH-359**

Aim

To develop Java applications using JDBC (Java Database Connectivity) for database connectivity, CRUD operations, and MVC architecture.

Objectives

· **Understand Database Connectivity:** Learn how to connect Java applications to a **MySQL database** using JDBC.

· **Perform CRUD Operations:** Implement **Create, Read, Update, and Delete (CRUD)** operations on database tables.

· **Use Different Levels of Complexity:**

- **Easy:** Fetch and display records from a database.
- **Medium:** Implement a menu-driven program with transaction handling.
- **Hard:** Develop a **full MVC-based** student management system.

· **Enhance Java Skills:** Gain experience in **SQL queries, Prepared Statements, and Transaction Management**.

· **Implement MVC Architecture:** Separate **Model, View, and Controller** for better maintainability and scalability.

· **Ensure Data Integrity:** Use **transaction handling** to prevent data inconsistencies.

Conduct:

EASSY : Problem Statement:
Create a Java program to connect to a MySQL database and fetch data from a single table. The program should:

- Use **DriverManager** and **Connection** objects.
- Retrieve and display all records from a table named **Employee** with columns **EmpID, Name, and Salary**.

CODE:

```
import java.sql.*;


public class EmployeeData {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/your_database";
        String user = "your_username";
        String password = "your_password";

        String query = "SELECT * FROM Employee";

        try (Connection conn = DriverManager.getConnection(url, user, password);
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            System.out.println("EmpID | Name | Salary");
            while (rs.next()) {
                int id = rs.getInt("EmpID");
                String name = rs.getString("Name");
                double salary = rs.getDouble("Salary");
                System.out.println(id + " | " + name + " | " + salary);
            }
        } catch (SQLException e) {
```

```
            e.printStackTrace();
        }
    }
}
```

Steps:

This Java program connects to a MySQL database and retrieves all records from the **Employee** table.

Steps to follow:

1. Install MySQL and create a database.
2. Create an **Employee** table with columns: EmpID, Name, Salary.
3. Use **JDBC** to connect to MySQL and fetch the records.

Medium Level

Aim:

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.

CODE:

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


public class ProductCRUD {
    static final String URL = "jdbc:mysql://localhost:3306/your_database";
    static final String USER = "your_username";
    static final String PASSWORD = "your_password";

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

```
    do {
        System.out.println("\nMenu:");
        System.out.println("1. 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();

        switch (choice) {
            case 1 -> addProduct();
            case 2 -> viewProducts();
            case 3 -> updateProduct();
            case 4 -> deleteProduct();
            case 5 -> System.out.println("Exiting...");
            default -> System.out.println("Invalid choice! Try again.");
        }
    } while (choice != 5);
    scanner.close();
}

private static void addProduct() {
    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         Scanner scanner = new Scanner(System.in)) {
        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 sql = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)";
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, name);
            stmt.setDouble(2, price);
            stmt.setInt(3, quantity);
            stmt.executeUpdate();
            System.out.println("Product added successfully!");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private static void viewProducts() {
    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         Statement stmt = conn.createStatement();
         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"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```
private static void updateProduct() {
    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         Scanner scanner = new Scanner(System.in)) {
        System.out.print("Enter Product ID to update: ");
        int id = scanner.nextInt();
        System.out.print("Enter new Price: ");
        double price = scanner.nextDouble();

        String sql = "UPDATE Product SET Price = ? WHERE ProductID = ?";
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setDouble(1, price);
            stmt.setInt(2, id);
            stmt.executeUpdate();
            System.out.println("Product updated successfully!");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

private static void deleteProduct() {
    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         Scanner scanner = new Scanner(System.in)) {
        System.out.print("Enter Product ID to delete: ");
        int id = scanner.nextInt();

        String sql = "DELETE FROM Product WHERE ProductID = ?";
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setInt(1, id);
            stmt.executeUpdate();
```

```
            System.out.println("Product deleted successfully!");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

Steps:

This program performs **Create, Read, Update, and Delete (CRUD)** operations on the **Product** table.

Steps to follow:

1. Create a **Product** table with columns: ProductID, ProductName, Price, Quantity.
2. Implement a menu-driven program for CRUD operations.
3. Ensure **transaction handling** for data integrity.

Hard Level

Aim:

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.**

Steps and CODE:

Steps to follow:

1. **Create a database table** Student with columns: StudentID, Name, Department, Marks.
2. **Define a** Student **class** (Model).
3. **Implement a** StudentController to handle database operations.
4. **Create a** StudentView to interact with users.
5. **Use JDBC** for database connectivity.

Code for Hard Level:

1. Student Model (Student.java)

```
java
CopyEdit
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; }
}
```

2. Student Controller (StudentController.java)

```
java
CopyEdit
import java.sql.*;
public class StudentController {
    private static final String URL = "jdbc:mysql://localhost:3306/your_database";
    private static final String USER = "your_username";
    private static final String PASSWORD = "your_password";

    public void addStudent(Student student) throws SQLException {
        String sql = "INSERT INTO Student (StudentID, Name, Department, Marks) VALUES (?, ?, ?, ?)";
        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
```

```
            PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setInt(1, student.getStudentID());
            stmt.setString(2, student.getName());
            stmt.setString(3, student.getDepartment());
            stmt.setInt(4, student.getMarks());
            stmt.executeUpdate();
        }
    }
}
```

3. Student View (Main.java)

```
java
CopyEdit
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        StudentController controller = new StudentController();

        System.out.print("Enter Student ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Department: ");
        String dept = scanner.nextLine();
        System.out.print("Enter Marks: ");
        int marks = scanner.nextInt();

        Student student = new Student(id, name, dept, marks);
        try {
            controller.addStudent(student);
            System.out.println("Student added successfully!");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```