Experiment -7

Student Name: Rishu Raj **UID:22BCS15617**
Branch: BE-CSE **Section/Group: IOT_631/B**
Semester: 6th **DOP:03/04/2025**
Subject Name: PBLJ **Subject Code: 22CSH-359**

1. **Aim:** 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..

2. **Objective:** To develop a Java program that connects to a MySQL database, retrieves data from the Employee table, and displays all records, demonstrating basic JDBC connectivity and data retrieval operations.

3. **Code:**

```
import java.sql.*;
public class FetchEmployeeData {
public static void main(String[] args) {
        String url =  "jdbc:mysql://localhost:3306/testdb";
        String user = "root";
        String password = "password";
        String query = "SELECT EmpID, Name, Salary FROM Employee";

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

           // Establish connection
           Connection con = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to the database!");

           // Create statement and execute query
           Statement stmt = con.createStatement();
           ResultSet rs = stmt.executeQuery(query);

           // Display results
```

```
        System.out.println("\nEmployee Records:");     System.out.println("-----------------------------------");
        System.out.printf("%-10s %-20s %-10s%n", "EmpID", "Name", "Salary");
        System.out.println("-----------------------------------");

        while (rs.next()) {
        int empID = rs.getInt("EmpID");
        String name = rs.getString("Name");
        double salary = rs.getDouble("Salary");

        System.out.printf("%-10d %-20s %-10.2f%n", empID, name, salary);
        }

        // Close resources
        rs.close();
        stmt.close();
        con.close();
        System.out.println("\nConnection closed.");
        }
        catch (ClassNotFoundException e)
        {
        System.out.println("MySQL Driver not found: " + e.getMessage());
        }
        catch (SQLException e) {
        System.out.println("SQL Error: " + e.getMessage());
        }
    }
}
```

4. Output:

```
(base) PS C:\Users\virat\OneDrive\Desktop\java exp7> java -cp ".;lib/mysql-connector-j-9.2.0.jar" FetchEmployeeD
ata
>>
Connected to the database!

Employee Records:
-------------------------------------
EmpID      Name                 Salary
-------------------------------------
1          Alice                50000.00
2          Bob                  60000.00
3          Charlie              55000.00

Connection closed.
(base) PS C:\Users\virat\OneDrive\Desktop\java exp7>
```

7.2.1 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. Transaction handling to ensure data integrity.

7.2.2 Objective: To develop a Java program that connects to a MySQL database and performs CRUD operations (Create, Read, Update, Delete) on the Product table. The program ensures data integrity by using transaction handling and provides a menu-driven interface for user-friendly interaction.

7.2.3 Code:

```
import java.sql.*;
import java.util.Scanner;
public class ProductCRUD {

private static final String URL = "jdbc:mysql://localhost:3306/ProductDB";
private static final String USER = "root";
private static final String PASSWORD = "password";

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

    try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD)) {
        Class.forName("com.mysql.cj.jdbc.Driver");
        System.out.println("Connected to the database!");
        boolean exit = false;

        while (!exit) {
            System.out.println("\n=== Product CRUD Operations ===");
            System.out.println("1. Create Product");
            System.out.println("2. Read Products");
            System.out.println("3. Update Product");
            System.out.println("4. Delete Product");
            System.out.println("5. Exit");
            System.out.print("Choose an option: ");

            int choice = scanner.nextInt();
            scanner.nextLine();
```

```
switch (choice) {
case 1 -> createProduct(conn, scanner);
case 2 -> readProducts(conn);
case 3 -> updateProduct(conn, scanner);
case 4 -> deleteProduct(conn, scanner);
case 5 -> exit = true;
default -> System.out.println("Invalid option. Try again.");
 }
 }
 }
catch (ClassNotFoundException e) {
 System.out.println("MySQL Driver not found: " + e.getMessage());
 }
 catch (SQLException e) {
  System.out.println("SQL Error: " + e.getMessage());
 }
  scanner.close();
  }
  private static void createProduct(Connection conn, Scanner scanner) throws SQLException {
  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 query = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)";
  try (PreparedStatement pstmt = conn.prepareStatement(query)) {
   conn.setAutoCommit(false);
   pstmt.setString(1, name);
   pstmt.setDouble(2, price);
   pstmt.setInt(3, quantity);
   int rows = pstmt.executeUpdate(); conn.commit();

            System.out.println(rows + " product(s) inserted successfully!");
        } catch (SQLException e) {
          conn.rollback();
          System.out.println("Transaction rolled back due to error: " + e.getMessage());
        } finally {
          conn.setAutoCommit(true);
        }
    }
```

```
    private static void readProducts(Connection conn) throws SQLException {        String query =
"SELECT * FROM Product";

        try (Statement stmt = conn.createStatement();             ResultSet
rs = stmt.executeQuery(query)) {

            System.out.println("\nProduct Records:");
            System.out.println("-------------------------------------------------");
            System.out.printf("%-10s %-20s %-10s %-10s%n", "ProductID", "ProductName", "Price", "Quantity");
            System.out.println("-------------------------------------------------");

            while (rs.next()) {
                int id = rs.getInt("ProductID");
                String name = rs.getString("ProductName");
double price = rs.getDouble("Price");                int quantity =
rs.getInt("Quantity");

                System.out.printf("%-10d %-20s %-10.2f %-10d%n", id, name, price, quantity);
            }
        }
    }


    private static void updateProduct(Connection conn, Scanner scanner) throws SQLException {
        System.out.print("Enter product ID to update: ");        int id =
scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter new name: ");
        String name = scanner.nextLine();
              System.out.print("Enter new price: ");
         double price = scanner.nextDouble();
         System.out.print("Enter new quantity: ");        int
quantity = scanner.nextInt();

String query = "UPDATE Product SET ProductName = ?, Price = ?, Quantity = ? WHERE ProductID = ?";

        try (PreparedStatement pstmt = conn.prepareStatement(query)) {
conn.setAutoCommit(false);

            pstmt.setString(1, name);
pstmt.setDouble(2, price);
pstmt.setInt(3, quantity);
```

```
pstmt.setInt(4, id);

            int rows = pstmt.executeUpdate();            conn.commit();

            System.out.println(rows + " product(s) updated successfully!");

        } catch (SQLException e) {
          conn.rollback();
           System.out.println("Transaction rolled back due to error: " + e.getMessage());
        }
          finally {
           conn.setAutoCommit(true);
        }
    }

    private static void deleteProduct(Connection conn, Scanner scanner) throws SQLException {
        System.out.print("Enter product ID to delete: ");
       int id = scanner.nextInt();

        String query = "DELETE FROM Product WHERE ProductID = ?";

        try (PreparedStatement pstmt = conn.prepareStatement(query)) {
          conn.setAutoCommit(false);

          pstmt.setInt(1, id);
          int rows = pstmt.executeUpdate();            conn.commit();

             System.out.println(rows + " product(s) deleted successfully!");
        }
          catch (SQLException e) {
          conn.rollback();
           System.out.println("Transaction rolled back due to error: " + e.getMessage());
        }
          finally {
           conn.setAutoCommit(true);
        }
    }
}
```

7.2.4 Output:

```
(base) PS C:\Users\virat\OneDrive\Desktop\java exp7> java -cp ".;lib/mysql-connector-j-9.2.0.jar
" ProductCRUD
>>
Connected to the database!

=== Product CRUD Operations ===
1. Create Product
2. Read Products
3. Update Product
4. Delete Product
5. Exit
Choose an option: 2

Product Records:
------------------------------------------------
ProductID  ProductName          Price      Quantity
------------------------------------------------
1          Laptop               75000.00   10
2          Mobile Phone         30000.00   25
3          Tablet               20000.00   15
4          Headphones           5000.00    50
5          Smartwatch           12000.00   30
6          Camera               45000.00   12
```

7.2.5 Learning Outcomes:

1. Understanding JDBC Integration**:**Gained practical experience in integrating JDBC with a Java application for database connectivity.
2. MVC Architecture Implementation**:**Learned how to implement the ModelView-Controller (MVC) architecture in Java for better code organization and separation of concerns.
3. Database CRUD Operations:Acquired the ability to perform CRUD operations (Create, Read, Update, Delete) using SQL queries in Java applications.
4. Transaction Handling**:**Understood the importance of transaction handling for maintaining data integrity during database operations.