Akshansh Kaundal - 23BCS13369 - 622 B

# Experiment 2.4 — Java Applications Using JDBC for Database Connectivity, CRUD Operations, and MVC Architecture

## Part (a): Connecting to MySQL and Fetching Data from a Table

**Aim:** To develop a Java program that connects to a MySQL database and retrieves data from an Employee table using JDBC.

**Procedure:**
1. Create a database named testdb and an Employee table with fields EmpID, Name, and Salary.
2. Insert sample records.
3. Write Java code to connect, execute a SELECT query, and display the data.

```
import java.sql.*;

public class FetchEmployeeData {     public static
void main(String[] args) {         try {
Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/testdb", "root", "password");
            Statement stmt = con.createStatement();
            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"));             }             con.close();
} catch (Exception e) {
            e.printStackTrace();
}
    }
}
```

**Sample Output:**

```
EmpID | Name | Salary
101 | Saksham | 55000.0
102 | Pranav | 60000.0
```

**Result:** Successfully connected to the MySQL database and retrieved employee data.

## Part (b): CRUD Operations on Product Table Using JDBC

**Aim:** To perform Create, Read, Update, and Delete (CRUD) operations on a Product table using JDBC with transaction handling.

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

public class ProductCRUD {     public static
void main(String[] args) {         try {
Class.forName("com.mysql.cj.jdbc.Driver");
```

```
            Connection con = DriverManager.getConnection(
"jdbc:mysql://localhost:3306/testdb", "root", "password");
con.setAutoCommit(false);             Scanner sc = new Scanner(System.in);

            while (true) {                 System.out.println("\n1. Insert\n2. View\n3.
Update\n4. Delete\n5. Exit");                 System.out.print("Enter choice: ");
int choice = sc.nextInt();

                switch (choice) {                     case 1:
PreparedStatement ps = con.prepareStatement(
                            "INSERT INTO Product VALUES (?, ?, ?, ?)");
System.out.print("Enter ID, Name, Price, Quantity: ");
ps.setInt(1, sc.nextInt());                           ps.setString(2,
sc.next());                         ps.setDouble(3, sc.nextDouble());
ps.setInt(4, sc.nextInt());                           ps.executeUpdate();
con.commit();                           System.out.println("Product added
successfully!");                           break;

                    case 2:                           Statement st =
con.createStatement();                          ResultSet rs =
st.executeQuery("SELECT * FROM Product");                             while
(rs.next()) {                             System.out.println(rs.getInt(1) + " |
" +                                       rs.getString(2) + " | " +
rs.getDouble(3) + " | " +
rs.getInt(4));                          }                           break;

                    case 3:                           PreparedStatement
ps2 = con.prepareStatement(
                            "UPDATE Product SET Price=?, Quantity=? WHERE ProductID=?");
System.out.print("Enter new Price, Quantity, and ProductID: ");
ps2.setDouble(1, sc.nextDouble());                                ps2.setInt(2, sc.nextInt());
ps2.setInt(3, sc.nextInt());                           ps2.executeUpdate();
con.commit();                           System.out.println("Product updated
successfully!");                           break;

                    case 4:                           PreparedStatement
ps3 = con.prepareStatement(
                            "DELETE FROM Product WHERE ProductID=?");
System.out.print("Enter ProductID to delete: ");
ps3.setInt(1, sc.nextInt());                           ps3.executeUpdate();
con.commit();                           System.out.println("Product deleted
successfully!");                           break;

                    case 5:
                        con.close();
System.exit(0);
                }
            }         } catch
(Exception e) {
            e.printStackTrace();
}
    }
}
```

**Sample Output:**

```
Enter choice: 1
Enter ID, Name, Price, Quantity: 201 Pen 20.5 50
Product added successfully!
```

**Result:** Successfully implemented CRUD operations with transaction management.

# Part (c): Student Management Application Using JDBC and MVC Architecture

**Aim:** To develop a Student Management Application using JDBC and MVC architecture.

**Model (Student.java):**

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

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

    public int getId() { return id; }     public String
getName() { return name; }     public String
getDepartment() { return department; }     public int
getMarks() { return marks; } }
```

**Controller (StudentDAO.java):**

```
import java.sql.*;

public class StudentDAO {
private Connection con;

    public StudentDAO() throws Exception {
Class.forName("com.mysql.cj.jdbc.Driver");         con =
DriverManager.getConnection(
"jdbc:mysql://localhost:3306/testdb", "root", "password");     }

    public void addStudent(Student s) throws Exception {          PreparedStatement ps =
con.prepareStatement("INSERT INTO Student VALUES (?, ?, ?, ?)");         ps.setInt(1,
s.getId());         ps.setString(2, s.getName());         ps.setString(3, s.getDepartment());
ps.setInt(4, s.getMarks());         ps.executeUpdate();     }

    public void viewStudents() throws Exception {
        Statement st = con.createStatement();         ResultSet rs =
st.executeQuery("SELECT * FROM Student");         while (rs.next()) {
System.out.println(rs.getInt(1) + " | " + rs.getString(2) + " | " +
rs.getString(3) + " | " + rs.getInt(4));         }
    }
}
```

**View (MainApp.java):**

```
import java.util.*;

public class MainApp {     public static void
main(String[] args) {         try {
StudentDAO dao = new StudentDAO();
Scanner sc = new Scanner(System.in);

            while (true) {                 System.out.println("\n1. Add
Student\n2. View Students\n3. Exit");                 System.out.print("Enter
choice: ");                 int ch = sc.nextInt();

                if (ch == 1) {                     System.out.print("Enter ID, Name,
Department, Marks: ");                     Student s = new Student(sc.nextInt(), sc.next(),
sc.next(), sc.nextInt());                     dao.addStudent(s);
System.out.println("Student added successfully!");                 } else if (ch == 2) {
dao.viewStudents();                 } else {
System.out.println("Exiting...");                     break;                 }
            }         } catch
(Exception e) {
```

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

**Sample Output:**

```
Enter choice: 1
Enter ID, Name, Department, Marks: 1 Saksham CSE 90
Student added successfully!
Enter choice: 2
1 | Saksham | CSE | 90
```

**Result:** Successfully implemented MVC-based Student Management System using JDBC.

**Conclusion:** The experiment demonstrated successful database connectivity, CRUD operations, and MVC implementation using JDBC in Java.