GAM/IT/2022/F/0064 – S.A.I.I. Subhasingha

Lab Sheet 2: Java JDBC Lab Practical using NetBeans IDE 8.2

1. Set Up MySQL Database

```
CREATE DATABASE employee_db;
USE employee_db;
CREATE TABLE employees (
id INT PRIMARY KEY AUTO_INCREMENT,
name VARCHAR(100),
position VARCHAR(100),
salary DECIMAL(10, 2)
);
```

2. Set Up NetBeans Project

3. Establish JDBC Connection

Create a DatabaseConnection.java class to establish a connection to your database.

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
/**
 *
 * @author USER
 */
public class DatabaseConnection {

    private static final String URL = "jdbc:mysql://localhost:3306/employee_db"; // Database URL
    private static final String USER = "root"; // Your MySQL username
    private static final String PASSWORD = "root"; // Your MySQL password

    public static Connection getConnection() throws SQLException {
        try {
            // Load the JDBC driver
            Class.forName(className:"com.mysql.cj.jdbc.Driver");
            // Return the database connection
            return DriverManager.getConnection(url: URL, user:USER, password: PASSWORD);
        } catch (ClassNotFoundException | SQLException e) {
            System.out.println("Connection failed: " + e.getMessage());
            throw new SQLException(reason:"Failed to establish connection.");
        }
    }

}

```

4. Perform CRUD Operations

1. Create EmployeeDAO.java for CRUD Operations:

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this l
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;

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

/**
 *
 * @author USER
 */
public class EmployeeDAO {

    // Create an employee
    public static void addEmployee(String name, String position, double salary) {
        String sql = "INSERT INTO employees (name, position, salary) VALUES (?, ?, ?)";

        try (Connection conn = DatabaseConnection.getConnection();
             PreparedStatement stmt = conn.prepareStatement(string:sql)) {

            stmt.setString(i: 1, string:name);
            stmt.setString(i: 2, string:position);
            stmt.setDouble(i: 3, d: salary);

            int rowsAffected = stmt.executeUpdate();
            System.out.println("Employee added successfully. Rows affected: " + rowsAffected);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
```

```java

    // Read all employees
    public static List<Employee> getAllEmployees() {
        List<Employee> employees = new ArrayList<>();
        String sql = "SELECT * FROM employees";

        try (Connection conn = DatabaseConnection.getConnection();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(string:sql)) {

            while (rs.next()) {
                Employee employee = new Employee(id: rs.getInt(string:"id"),name: rs.getString(string:"name"),
                        position: rs.getString(string:"position"),salary:rs.getDouble(string:"salary"));
                employees.add(e: employee);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }

        return employees;
    }
```

```java
    // Update an employee's information
    public static void updateEmployee(int id, String name, String position, double salary) {
        String sql = "UPDATE employees SET name = ?, position = ?, salary = ? WHERE id = ?";

        try (Connection conn = DatabaseConnection.getConnection();
             PreparedStatement stmt = conn.prepareStatement(string:sql)) {

            stmt.setString(i: 1, string:name);
            stmt.setString(i: 2, string:position);
            stmt.setDouble(i: 3, d: salary);
            stmt.setInt(i: 4, i1: id);

            int rowsAffected = stmt.executeUpdate();
            System.out.println("Employee updated successfully. Rows affected: " + rowsAffected);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Delete an employee
    public static void deleteEmployee(int id) {
        String sql = "DELETE FROM employees WHERE id = ?";

        try (Connection conn = DatabaseConnection.getConnection();
             PreparedStatement stmt = conn.prepareStatement(string:sql)) {

            stmt.setInt(i: 1, i1: id);
            int rowsAffected = stmt.executeUpdate();
            System.out.println("Employee deleted successfully. Rows affected: " + rowsAffected);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

5. Create Employee.java Class

Create a simple Employee.java POJO (Plain Old Java Object) to represent employee data.

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;
/**
 *
 * @author USER
 */
public class Employee {

    private int id;
    private String name;
    private String position;
    private double salary;

    public Employee(int id, String name, String position, double salary) {
        this.id = id;
        this.name = name;
        this.position = position;
        this.salary = salary;
    }
    // Getters and setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

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

    public String getPosition() { return position; }
    public void setPosition(String position) { this.position = position; }

    public double getSalary() { return salary; }
    public void setSalary(double salary) { this.salary = salary; }

    @Override
    public String toString() {
        return "Employee{id=" + id + ", name='" + name + "', position='" + position + "', salary=" + salary + '}';
    }

}
```

6. Test the Application

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;

import java.util.List;
/**
 *
 * @author USER
 */
public class Main {

    public static void main(String[] args) {
        // Add employees
        EmployeeDAO.addEmployee(name: "Alice Cooper", position: "Developer", salary: 70000);
        EmployeeDAO.addEmployee(name: "Bob Marley", position: "Manager", salary: 80000);

        // Update employee
        EmployeeDAO.updateEmployee(id: 1, name: "John Doe", position: "Senior Software Engineer", salary: 90000);

        // Get all employees
        List<Employee> employees = EmployeeDAO.getAllEmployees();
        employees.forEach(System.out::println);

        // Delete employee
        EmployeeDAO.deleteEmployee(id: 2);
    }
}
```