

Name:- Ariwala Premal Vinod

Enrollment number:- 24002401110003

Division:-A

Roll.no:-1

Java Practical Assignment

1. Develop a Java Application that Insert the records of Employee into the MySQL database. After inserting records display all records.

Hint: Create table Employee having field (id, name, address, phoneno, designation).

- Display proper messages
- Use JDBC.

Code :

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

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

```
public class EmployeeJDBCExample {
```

```
public static void main(String[] args) {  
    // Database connection details  
    String url = "jdbc:mysql://localhost:3306/companydb";  
    String user = "root";  
    String password = "";  
    Connection conn = null;  
    PreparedStatement pstmt = null;  
    Statement stmt = null;  
    ResultSet rs = null;  
    Scanner sc = new Scanner(System.in);  
  
    try {  
        Class.forName("com.mysql.cj.jdbc.Driver");  
        System.out.println("MySQL JDBC Driver Registered!");  
        conn = DriverManager.getConnection(url, user, password);
```

```
System.out.println("Connected to the database!");

String insertQuery = "INSERT INTO Employee (id, name, address, phoneno, designation)
VALUES (?, ?, ?, ?, ?)";

pstmt = conn.prepareStatement(insertQuery);

System.out.print("Enter number of employees to insert: ");
int n = sc.nextInt();
sc.nextLine();

for (int i = 1; i <= n; i++) {

    System.out.println("\nEnter details for Employee " + i + ":");

    System.out.print("ID: ");
    int id = sc.nextInt();
    sc.nextLine();

    System.out.print("Name: ");
    String name = sc.nextLine();

    System.out.print("Address: ");
    String address = sc.nextLine();

    System.out.print("Phone No: ");
    String phone = sc.nextLine();

    System.out.print("Designation: ");
    String designation = sc.nextLine();

    pstmt.setInt(1, id);
    pstmt.setString(2, name);
    pstmt.setString(3, address);
```

```

        pstmt.setString(4, phone);

        pstmt.setString(5, designation);


        pstmt.executeUpdate();

        System.out.println("Employee record inserted successfully!");
    }


    System.out.println("\nEmployee Records:");

    stmt = conn.createStatement();

    rs = stmt.executeQuery("SELECT * FROM Employee");


    System.out.printf("%-5s %-20s %-25s %-15s %-15s\n",
        "ID", "Name", "Address", "Phone", "Designation");

    System.out.println("-----");


    while (rs.next()) {

        System.out.printf("%-5d %-20s %-25s %-15s %-15s\n",
            rs.getInt("id"),
            rs.getString("name"),
            rs.getString("address"),
            rs.getString("phoneno"),
            rs.getString("designation"));

    }


    } catch (ClassNotFoundException e) {

        System.out.println("MySQL JDBC Driver not found!");

    } catch (SQLException e) {

        System.out.println("Database error: " + e.getMessage());

    } finally {

        // Close resources

        try { if (rs != null) rs.close(); } catch (Exception ignored) {}

```

```

        try { if (stmt != null) stmt.close(); } catch (Exception ignored) {}
        try { if (pstmt != null) pstmt.close(); } catch (Exception ignored) {}
        try { if (conn != null) conn.close(); } catch (Exception ignored) {}
        sc.close();
    }
}
}

```

Output :

MySQL JDBC Driver Registered!

Connected to the database!

Enter number of employees to insert: 2

Enter details for Employee 1:

ID: 101

Name: John Doe

Address: New York

Phone No: 9876543210

Designation: Manager

Employee record inserted successfully!

Employee record inserted successfully!

Employee Records:

ID	Name	Address	Phone	Designation
101	John Doe	New York	9876543210	Manager

2. Create java application using JDBC that delete the record using employee id from MySQL database.

Hint:

- Use above table.
- Display proper messages.

Code :

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

public class DeleteEmployeeRecord {
    public static void main(String[] args) {

        String url = "jdbc:mysql://localhost:3306/companydb";
        String user = "root";
        String password = "";

        Connection conn = null;
        PreparedStatement pstmt = null;
        Statement stmt = null;
        ResultSet rs = null;
        Scanner sc = new Scanner(System.in);

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");
            System.out.println("MySQL JDBC Driver Registered!");

            conn = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to the database!");

            System.out.print("\nEnter Employee ID to delete: ");
```

```

int empld = sc.nextInt();

String deleteQuery = "DELETE FROM Employee WHERE id = ?";
pstmt = conn.prepareStatement(deleteQuery);
pstmt.setInt(1, empld);

int rowsAffected = pstmt.executeUpdate();
if (rowsAffected > 0) {
    System.out.println("Employee with ID " + empld + " deleted successfully!");
} else {
    System.out.println("No employee found with ID " + empld);
}

System.out.println("\nRemaining Employee Records:");
stmt = conn.createStatement();
rs = stmt.executeQuery("SELECT * FROM Employee");

System.out.printf("%-5s %-20s %-25s %-15s %-15s\n",
    "ID", "Name", "Address", "Phone", "Designation");
System.out.println("-----");

boolean found = false;
while (rs.next()) {
    found = true;
    System.out.printf("%-5d %-20s %-25s %-15s %-15s\n",
        rs.getInt("id"),
        rs.getString("name"),
        rs.getString("address"),
        rs.getString("phoneno"),
        rs.getString("designation"));
}

```



```

        if (!found) {
            System.out.println("No records found in Employee table.");
        }

    } catch (ClassNotFoundException e) {
        System.out.println("MySQL JDBC Driver not found!");
    } catch (SQLException e) {
        System.out.println("Database error: " + e.getMessage());
    } finally {
        try { if (rs != null) rs.close(); } catch (Exception ignored) {}
        try { if (stmt != null) stmt.close(); } catch (Exception ignored) {}
        try { if (pstmt != null) pstmt.close(); } catch (Exception ignored) {}
        try { if (conn != null) conn.close(); } catch (Exception ignored) {}
        sc.close();
    }
}
}

```

Output :

MySQL JDBC Driver Registered!

Connected to the database!

Enter Employee ID to delete: 102

Employee with ID 102 deleted successfully!

Remaining Employee Records:

ID	Name	Address	Phone	Designation

101	John Doe	New York	9876543210	Manager

3. Create java application using JDBC that Update the records of employee using employee id in MySQL database. Hint: • Use above table. • Display proper messages.

Code :

UpdateEmployeeApp.java :-

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

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

```
public class UpdateEmployeeApp {
```

```
    public static void main(String[] args) {
```

```
        String url = "jdbc:mysql://localhost:3306/com";
```

```
        String user = "root";
```

```
        String pass = "";
```

```
        try {
```

```
            Connection con = DriverManager.getConnection(url, user, pass);
```

```
            System.out.println("Connected to MySQL Database.");
```

```
            Scanner sc = new Scanner(System.in);
```

```
            System.out.print("Enter Employee ID to update: ");
```

```
            int id = sc.nextInt();
```

```
            sc.nextLine();
```

```
            System.out.print("Enter new Name: ");
```

```
            String name = sc.nextLine();
```

```
            System.out.print("Enter new Address: ");
```

```
            String address = sc.nextLine();
```

```
System.out.print("Enter new Phone No: ");  
String phone = sc.nextLine();  
System.out.print("Enter new Designation: ");  
String designation = sc.nextLine();
```

```
String query = "UPDATE employee SET name=?, address=?, phoneno=?, designation=?  
WHERE id=?";
```

```
PreparedStatement pst = con.prepareStatement(query);  
pst.setString(1, name);  
pst.setString(2, address);  
pst.setString(3, phone);  
pst.setString(4, designation);  
pst.setInt(5, id);
```

```
int rows = pst.executeUpdate();
```

```
if (rows > 0)
```

```
    System.out.println("Employee record updated successfully!");
```

```
else
```

```
    System.out.println(" No employee found with ID " + id + ".");
```

```
System.out.println("\nUpdated Employee Records:");
```

```
Statement st = con.createStatement();
```

```
ResultSet rs = st.executeQuery("SELECT * FROM employee");
```

```
while (rs.next()) {
```

```
    System.out.println(rs.getInt("id") + " | " +
```

```

        rs.getString("name") + " | " +
        rs.getString("address") + " | " +
        rs.getString("phoneno") + " | " +
        rs.getString("designation"));
    }

    con.close();
    sc.close();
} catch (Exception e) {
    System.out.println( e.getMessage());
}
}
}

```

Output:-

Connected to MySQL Database.

Enter Employee ID to update: 1

Enter new Name: priya

Enter new Address: bcv

Enter new Phone No: s5435

Enter new Designation: bcxc

No employee found with ID 1.

Updated Employee Records:

3 | kdnfd | ewtd | 6577586 | vvcbvc

BUILD SUCCESSFUL (total time: 24 seconds)

4. Develop a Java Networking Application in which client sends message (Hello) to server. Server accept message from client and reply to client (Hi). Hint: • Use TCP . • Create necessary classes for client and server.

Code :

Server.java:-

```
import java.io.*;
```

```
import java.net.*;
```

```
public class Server {
```

```
    public static void main(String[] args) throws Exception {
```

```
        ServerSocket ss = new ServerSocket(5000);
```

```
        System.out.println("Server started, waiting for client...");
```

```
        Socket s = ss.accept();
```

```
        System.out.println("Client connected.");
```

```
        BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
```

```
        PrintWriter pw = new PrintWriter(s.getOutputStream(), true);
```

```
        String msg = br.readLine();
```

```
        System.out.println("Client says: " + msg);
```

```
        pw.println("Hi");
```

```
        System.out.println("Replied: Hi");
```

```
        s.close();
```

```
        ss.close();
```

```
    }
```

```
}
```

Client.java:-

```
import java.io.*;
import java.net.*;

public class Client {

    public static void main(String[] args) throws Exception {

        Socket s = new Socket("localhost", 5000);

        BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
        PrintWriter pw = new PrintWriter(s.getOutputStream(), true);

        pw.println("Hello");
        String reply = br.readLine();

        System.out.println("Server says: " + reply);

        s.close();
    }
}
```

Output:

run:

Server started, waiting for client...

Hi

run:

Server says: Hi

BUILD SUCCESSFUL (total time: 0 seconds)

5. Develop a Java Networking Application in which client sends number to server. Server accept message from client . Find factorial of that number and reply to client. Hint: • Use UDP . • Create necessary classes for client and server.

Code :

UdpSender/ UDPSenderFactorial:-

```
package udpSender;
```

```
import java.net.*;
```

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

```
public class UDPSenderFactorial {
```

```
    public static void main(String[] args) {
```

```
        try {
```

```
            InetAddress ip = InetAddress.getByName("127.0.0.1"); // server IP
```

```
            DatagramSocket ds = new DatagramSocket();
```

```
            Scanner sc = new Scanner(System.in);
```

```
            boolean start = true;
```

```
            while(start) {
```

```
                System.out.print("Enter a number (0 to stop): ");
```

```
                int num = sc.nextInt();
```

```
                if(num == 0) {
```

```
                    start = false;
```

```
                }
```

```
                String msg = String.valueOf(num);
```

```
DatagramPacket dp = new DatagramPacket(msg.getBytes(), msg.length(), ip, 8340);  
ds.send(dp);
```

```
byte[] buffer = new byte[2048];  
DatagramPacket receivePacket = new DatagramPacket(buffer, buffer.length);  
ds.receive(receivePacket);
```

```
String result = new String(receivePacket.getData(), 0, receivePacket.getLength());  
System.out.println("Factorial from server: " + result);  
}
```

```
ds.close();  
System.out.println("Client closed.");  
} catch (Exception e) {  
    System.out.println(e);  
}  
}  
}
```

Udpsender/ UDPReceiverFactorial:-

```
package udpsender;  
import java.net.*;
```

```
public class UDPReceiverFactorial {  
    public static void main(String[] args) {  
        try {  
            DatagramSocket ds = new DatagramSocket(8340);  
            byte[] buffer = new byte[2048];  
  
            System.out.println("Server is running...");
```



```

while(true) {

    DatagramPacket dp = new DatagramPacket(buffer, buffer.length);
    ds.receive(dp);

    String msg = new String(dp.getData(), 0, dp.getLength());
    int num = Integer.parseInt(msg.trim());

    if(num == 0) {
        System.out.println("Server stopping...");
        break;
    }

    long fact = 1;
    for(int i = 1; i <= num; i++) fact *= i;

    String result = String.valueOf(fact);
    DatagramPacket sendPacket = new DatagramPacket(result.getBytes(), result.length(),
dp.getAddress(), dp.getPort());
    ds.send(sendPacket);

    System.out.println("Received: " + num + " | Sent factorial: " + fact);
}

ds.close();
} catch(Exception e) {
    System.out.println(e);
}

```

```
}  
}
```

Output:

run:

Enter a number (0 to stop): 5

120

Server is running...

6. Create a java Web application using Servlet that Insert all the records of Vehicle into the MySQL database and after inserting display all records of vehicle. Hint: Create table Vehicle having field (Vehicle_Id, Model, Company_Name, Owner_name,milage). • Take necessary values from user. • Use necessary Java Beans. • Use sessions.

Vehicle.java

```
package model;
```

```
public class Vehicle {  
    private int vehicleId;  
    private String model;  
    private String companyName;  
    private String ownerName;  
    private int milage;  
  
    public int getVehicleId() { return vehicleId; }  
    public void setVehicleId(int vehicleId) { this.vehicleId = vehicleId; }
```

```

public String getModel() { return model; }

public void setModel(String model) { this.model = model; }


public String getCompanyName() { return companyName; }

public void setCompanyName(String companyName) { this.companyName = companyName; }


public String getOwnerName() { return ownerName; }

public void setOwnerName(String ownerName) { this.ownerName = ownerName; }


public int getMilage() { return milage; }

public void setMilage(int milage) { this.milage = milage; }
}

```

VehicleServlet.java

```

package controller;

```

```

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import model.Vehicle;

```

```

public class VehicleServlet extends HttpServlet {

```

```

    @Override

```

```

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

```

```

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

```

```
// Read form data

int id = Integer.parseInt(request.getParameter("vid"));

String model = request.getParameter("model");

String cname = request.getParameter("cname");

String oname = request.getParameter("oname");

int milage = Integer.parseInt(request.getParameter("milage"));


// Store data in session

HttpSession session = request.getSession();

session.setAttribute("lastVehicle", model);


// JavaBean

Vehicle v = new Vehicle();

v.setVehicleId(id);

v.setModel(model);

v.setCompanyName(cname);

v.setOwnerName(oname);

v.setMilage(milage);


try {

    Class.forName("com.mysql.cj.jdbc.Driver");

    Connection con = DriverManager.getConnection(

        "jdbc:mysql://localhost:3306/test", "root", "");

    PreparedStatement ps = con.prepareStatement(

        "INSERT INTO Vehicle VALUES (?, ?, ?, ?, ?)"

    );

    ps.setInt(1, v.getVehicleId());

    ps.setString(2, v.getModel());

    ps.setString(3, v.getCompanyName());
```

```

ps.setString(4, v.getOwnerName());

ps.setInt(5, v.getMilage());

ps.executeUpdate();

out.println("<h3>Vehicle Inserted Successfully</h3>");

// Retrieve and display all records
Statement st = con.createStatement();
ResultSet rs = st.executeQuery("SELECT * FROM Vehicle");

out.println("<table border='1'>");

out.println("<tr><th>ID</th><th>Model</th><th>Company</th><th>Owner</th><th>Milage</th></tr>");

while(rs.next()) {
    out.println("<tr>");
    out.println("<td>" + rs.getInt(1) + "</td>");
    out.println("<td>" + rs.getString(2) + "</td>");
    out.println("<td>" + rs.getString(3) + "</td>");
    out.println("<td>" + rs.getString(4) + "</td>");
    out.println("<td>" + rs.getInt(5) + "</td>");
    out.println("</tr>");
}

out.println("</table>");

out.println("<p>Last inserted vehicle model (via session): "
    + session.getAttribute("lastVehicle") + "</p>");

} catch (Exception e) {
    out.println(e);
}

```

```
}  
}
```

OUTPUT:

Vehicle Inserted Successfully

```
-----  
| ID   | Model   | Company | Owner | Milage  
-----  
| 101 | Splendor | Hero    | Amit  | 55  
-----
```

Last inserted vehicle model (via session): Splendor

7. Create a java Web application using Servlet that Update and delete I Vehicle using id. And after delete or update display all records of vehicle.

Hint: Create table Vehicle having field (Vehicle_Id, Model, Company_Name, Owner_name,milage). • Take necessary values from user. • Use necessary Java Beans. Display proper messages

Code :

bean/Vehicle.java

```
package bean;
```

```
public class Vehicle {  
    private int vehicleId;  
    private String model;  
    private String companyName;  
    private String ownerName;  
    private double mileage;  
  
    public int getVehicleId() { return vehicleId; }  
    public void setVehicleId(int vehicleId) { this.vehicleId = vehicleId; }
```

```

    public String getModel() { return model; }
    public void setModel(String model) { this.model = model; }
    public String getCompanyName() { return companyName; }
    public void setCompanyName(String companyName) { this.companyName = companyName; }
    public String getOwnerName() { return ownerName; }
    public void setOwnerName(String ownerName) { this.ownerName = ownerName; }
    public double getMileage() { return mileage; }
    public void setMileage(double mileage) { this.mileage = mileage; }
}

```

dao/VehicleDAO.java

```
package dao;
```

```

import java.sql.*;
import java.util.*;
import bean.Vehicle;

```

```
public class VehicleDAO {
```

```

    private static Connection getConnection() throws Exception {
        Class.forName("com.mysql.cj.jdbc.Driver");
        return DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/vehicle_db", "root", "your_password_here");
    }

```

```

    public static List<Vehicle> getAllVehicles() {
        List<Vehicle> list = new ArrayList<>();
        try (Connection con = getConnection();
            PreparedStatement ps = con.prepareStatement("SELECT * FROM Vehicle")) {
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                Vehicle v = new Vehicle();
                v.setVehicleId(rs.getInt(1));
                v.setModel(rs.getString(2));
                v.setCompanyName(rs.getString(3));
                v.setOwnerName(rs.getString(4));
                v.setMileage(rs.getDouble(5));
                list.add(v);
            }
        } catch (Exception e) { e.printStackTrace(); }
        return list;
    }

```

```

    public static int updateVehicle(Vehicle v) {
        int status = 0;
        try (Connection con = getConnection();
            PreparedStatement ps = con.prepareStatement(
                "UPDATE Vehicle SET model=?, company_name=?, owner_name=?, milage=? WHERE

```

```

vehicle_id=?")) {
    ps.setString(1, v.getModel());
    ps.setString(2, v.getCompanyName());
    ps.setString(3, v.getOwnerName());
    ps.setDouble(4, v.getMileage());
    ps.setInt(5, v.getVehicleId());
    status = ps.executeUpdate();
} catch (Exception e) { e.printStackTrace(); }
return status;
}

public static int deleteVehicle(int id) {
    int status = 0;
    try (Connection con = getConnection();
        PreparedStatement ps = con.prepareStatement("DELETE FROM Vehicle WHERE
vehicle_id=?")) {
        ps.setInt(1, id);
        status = ps.executeUpdate();
    } catch (Exception e) { e.printStackTrace(); }
    return status;
}
}

```

servlet/UpdateVehicleServlet.java

```

package servlet;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import bean.Vehicle;
import dao.VehicleDAO;

public class UpdateVehicleServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        int id = Integer.parseInt(request.getParameter("vehicle_id"));
        String model = request.getParameter("model");
        String company = request.getParameter("company_name");
        String owner = request.getParameter("owner_name");
        double mileage = Double.parseDouble(request.getParameter("mileage"));

        Vehicle v = new Vehicle();
        v.setVehicleId(id);
        v.setModel(model);
        v.setCompanyName(company);
        v.setOwnerName(owner);
        v.setMileage(mileage);
    }
}

```



```

        int status = VehicleDAO.updateVehicle(v);

        if (status > 0) {
            request.setAttribute("message", "Vehicle updated successfully!");
        } else {
            request.setAttribute("message", "Error updating vehicle!");
        }

        request.getRequestDispatcher("DisplayVehicleServlet").forward(request, response);
    }
}

```

servlet/DeleteVehicleServlet.java

```

package servlet;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import dao.VehicleDAO;

public class DeleteVehicleServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        int id = Integer.parseInt(request.getParameter("vehicle_id"));
        int status = VehicleDAO.deleteVehicle(id);

        if (status > 0) {
            request.setAttribute("message", "Vehicle deleted successfully!");
        } else {
            request.setAttribute("message", "Vehicle not found!");
        }

        request.getRequestDispatcher("DisplayVehicleServlet").forward(request, response);
    }
}

```

servlet/DisplayVehicleServlet.java

```

package servlet;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.List;
import dao.VehicleDAO;
import bean.Vehicle;

```

```

public class DisplayVehicleServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        List<Vehicle> list = VehicleDAO.getAllVehicles();
        request.setAttribute("list", list);
        request.getRequestDispatcher("displayVehicles.jsp").forward(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }
}

```

index.jsp

```

<h2>Vehicle Management System</h2>
<a href="updateVehicle.jsp">Update Vehicle</a><br>
<a href="deleteVehicle.jsp">Delete Vehicle</a><br>
<a href="DisplayVehicleServlet">Display All Vehicles</a>

```

updateVehicle.jsp

```

<h2>Update Vehicle</h2>
<form action="UpdateVehicleServlet" method="post">
    Vehicle ID: <input type="text" name="vehicle_id"><br>
    Model: <input type="text" name="model"><br>
    Company Name: <input type="text" name="company_name"><br>
    Owner Name: <input type="text" name="owner_name"><br>
    Mileage: <input type="text" name="mileage"><br><br>
    <input type="submit" value="Update">
</form>

```

deleteVehicle.jsp

```

<h2>Delete Vehicle</h2>
<form action="DeleteVehicleServlet" method="post">
    Vehicle ID: <input type="text" name="vehicle_id"><br><br>
    <input type="submit" value="Delete">
</form>

```

displayVehicles.jsp

```

<%@ page import="java.util.*,bean.Vehicle" %>
<h2>All Vehicles</h2>

<%
String message = (String)request.getAttribute("message");
if (message != null) {
%>

```

```

<p style="color:green;"><%= message %></p>
<%
}
List<Vehicle> list = (List<Vehicle>)request.getAttribute("list");
if (list != null && !list.isEmpty()) {
%>
<table border="1">
  <tr>
    <th>ID</th><th>Model</th><th>Company</th><th>Owner</th><th>Mileage</th>
  </tr>
<%
  for (Vehicle v : list) {
%>
    <tr>
      <td><%= v.getId() %></td>
      <td><%= v.getModel() %></td>
      <td><%= v.getCompanyName() %></td>
      <td><%= v.getOwnerName() %></td>
      <td><%= v.getMileage() %></td>
    </tr>
<%
  }
%>
</table>
<%
} else {
%>
<p>No vehicles found.</p>
<%
}
%>
<a href="index.jsp">Back to Home</a>

```

WEB-INF/web.xml

```

<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

  <servlet>
    <servlet-name>UpdateVehicleServlet</servlet-name>
    <servlet-class>servlet.UpdateVehicleServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>UpdateVehicleServlet</servlet-name>
    <url-pattern>/UpdateVehicleServlet</url-pattern>
  </servlet-mapping>

```

```

<servlet>
    <servlet-name>DeleteVehicleServlet</servlet-name>
    <servlet-class>servlet.DeleteVehicleServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>DeleteVehicleServlet</servlet-name>
    <url-pattern>/DeleteVehicleServlet</url-pattern>
</servlet-mapping>

<servlet>
    <servlet-name>DisplayVehicleServlet</servlet-name>
    <servlet-class>servlet.DisplayVehicleServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>DisplayVehicleServlet</servlet-name>
    <url-pattern>/DisplayVehicleServlet</url-pattern>
</servlet-mapping>
</web-app>

```

8. Develop a Java Web Application using JSP that insert all Student records in the MySQL Database. After inserting display all records on web page.

Hint: • Take necessary values using JSP Page from user. • Use Java Beans. • Use cookies to store student information. Create table named Student having field (Rollno,name,address,phoneno,marks)

Code :

Database Table - Student (MySQL)

```

CREATE DATABASE student_db;
CREATE TABLE Student (
    Rollno INT PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    phoneno VARCHAR(15),
    marks DOUBLE
);

```

bean/Student.java

```

package bean;

public class Student {

```

```

private int rollno;
private String name;
private String address;
private String phoneno;
private double marks;

public int getRollno() { return rollno; }
public void setRollno(int rollno) { this.rollno = rollno; }
public String getName() { return name; }
public void setName(String name) { this.name = name; }
public String getAddress() { return address; }
public void setAddress(String address) { this.address = address; }
public String getPhoneno() { return phoneno; }
public void setPhoneno(String phoneno) { this.phoneno = phoneno; }
public double getMarks() { return marks; }
public void setMarks(double marks) { this.marks = marks; }
}

```

dao/StudentDAO.java

```

package dao;

import java.sql.*;
import java.util.*;
import bean.Student;

public class StudentDAO {
    private static Connection getConnection() throws Exception {
        Class.forName("com.mysql.cj.jdbc.Driver");
        return DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/student_db", "root", "your_password_here");
    }

    public static int insertStudent(Student s) {
        int status = 0;
        try (Connection con = getConnection();
            PreparedStatement ps = con.prepareStatement("INSERT INTO Student VALUES (?, ?, ?, ?, ?)"))
        {
            ps.setInt(1, s.getRollno());
            ps.setString(2, s.getName());
            ps.setString(3, s.getAddress());
            ps.setString(4, s.getPhoneno());
            ps.setDouble(5, s.getMarks());
            status = ps.executeUpdate();
        } catch (Exception e) { e.printStackTrace(); }
        return status;
    }

    public static List<Student> getAllStudents() {

```

```

List<Student> list = new ArrayList<>();
try (Connection con = getConnection();
    PreparedStatement ps = con.prepareStatement("SELECT * FROM Student")) {
    ResultSet rs = ps.executeQuery();
    while (rs.next()) {
        Student s = new Student();
        s.setRollno(rs.getInt(1));
        s.setName(rs.getString(2));
        s.setAddress(rs.getString(3));
        s.setPhoneno(rs.getString(4));
        s.setMarks(rs.getDouble(5));
        list.add(s);
    }
} catch (Exception e) { e.printStackTrace(); }
return list;
}
}

```

index.jsp

```

<h2>Student Record Management</h2>
<form action="insertStudent.jsp" method="post">
    Roll No: <input type="text" name="rollno"><br>
    Name: <input type="text" name="name"><br>
    Address: <input type="text" name="address"><br>
    Phone No: <input type="text" name="phoneno"><br>
    Marks: <input type="text" name="marks"><br><br>
    <input type="submit" value="Insert Student">
</form>

```

insertStudent.jsp

```

<%@ page import="bean.Student,dao.StudentDAO,java.util.*" %>
<%
    int rollno = Integer.parseInt(request.getParameter("rollno"));
    String name = request.getParameter("name");
    String address = request.getParameter("address");
    String phoneno = request.getParameter("phoneno");
    double marks = Double.parseDouble(request.getParameter("marks"));

    // Store data in cookies
    Cookie c1 = new Cookie("rollno", String.valueOf(rollno));
    Cookie c2 = new Cookie("name", name);
    Cookie c3 = new Cookie("address", address);
    Cookie c4 = new Cookie("phoneno", phoneno);
    Cookie c5 = new Cookie("marks", String.valueOf(marks));

    response.addCookie(c1);
    response.addCookie(c2);
    response.addCookie(c3);

```

```

response.addCookie(c4);
response.addCookie(c5);

// Insert student record
Student s = new Student();
s.setRollno(rollno);
s.setName(name);
s.setAddress(address);
s.setPhoneno(phoneno);
s.setMarks(marks);

int status = StudentDAO.insertStudent(s);
String message = (status > 0) ? "Student inserted successfully!" : "Error inserting record!";
%>

<h3><%= message %></h3>
<jsp:forward page="displayStudents.jsp"></jsp:forward>

```

displayStudents.jsp

```

<%@ page import="java.util.*,bean.Student,dao.StudentDAO" %>
<h2>All Student Records</h2>
<%
    List<Student> list = StudentDAO.getAllStudents();
    if (list != null && !list.isEmpty()) {
%>
<table border="1">
    <tr>
        <th>Roll No</th><th>Name</th><th>Address</th><th>Phone No</th><th>Marks</th>
    </tr>
<%
    for (Student s : list) {
%>
    <tr>
        <td><%= s.getRollno() %></td>
        <td><%= s.getName() %></td>
        <td><%= s.getAddress() %></td>
        <td><%= s.getPhoneno() %></td>
        <td><%= s.getMarks() %></td>
    </tr>
<%
    }
%>
</table>
<%
    } else {
%>
<p>No student records found.</p>
<%

```

```

    }
%>
<a href="index.jsp">Back to Home</a>

```

WEB-INF/web.xml

```

<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>

```

9. Develop a Java Web Application using JSP update and delete Student records using rollno in the MySQL Database. After Update or delete display all records on web page.

Hint: • Take necessary values using JSP Page from user. • Use Java Beans. • Use cookies to store student information. Create table named Student having field (Rollno,name,address,phoneno,marks)

Code :

Database Table - Studentdb (MySQL)

```

CREATE TABLE student (
    rollno INT PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    phoneno VARCHAR(15),
    marks INT
);

```

Student/student.java (Java Bean)

```

package student;

import java.io.Serializable;

```



```
import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;


@Entity

public class Student implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id

    @GeneratedValue(strategy = GenerationType.AUTO)

    private int id;

    private int rollno;

    private String name;

    private String address;

    private String phoneno;

    private int marks;


    public int getRollno() {

        return rollno;

    }


    public void setRollno(int rollno) {

        this.rollno = rollno;

    }


    public String getName() {

        return name;

    }


    public void setName(String name) {
```

```
    this.name = name;
}
```

```
public String getAddress() {
    return address;
}
```

```
public void setAddress(String address) {
    this.address = address;
}
```

```
public String getPhoneno() {
    return phoneno;
}
```

```
public void setPhoneno(String phoneno) {
    this.phoneno = phoneno;
}
```

```
public int getMarks() {
    return marks;
}
```

```
public void setMarks(int marks) {
    this.marks = marks;
}
```

```
public int getId() {
    return id;
}
```

```
public void setId(int id) {
```

```
    this.id = id;
}
```

```
@Override
public int hashCode() {
    int hash = 0;
    hash += (int) id;
    return hash;
}
```

```
@Override
public boolean equals(Object object) {

    if (!(object instanceof Student)) {
        return false;
    }
    Student other = (Student) object;
    if (this.id != other.id) {
        return false;
    }
    return true;
}
```

```
@Override
public String toString() {
    return "student.Student[ id=" + id + " ]";
}
}
```

Student/DBConnection.java

```
package student;

import java.sql.*;

public class DBConnection {
```

```

public static Connection getConnection() {
    Connection con = null;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        con = DriverManager.getConnection("jdbc:mysql://localhost:3306/studentdb", "root", "");
    } catch (Exception e) {
        e.printStackTrace();
    }
    return con;
}
}

```

student/StudentDAO.java

```

package student;

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

public class StudentDAO {
    public static List<Student> getAllStudents() {
        List<Student> list = new ArrayList<>();
        try {
            Connection con = DBConnection.getConnection();
            PreparedStatement ps = con.prepareStatement("SELECT * FROM student");
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                Student s = new Student();
                s.setRollno(rs.getInt(1));
                s.setName(rs.getString(2));
                s.setAddress(rs.getString(3));
                s.setPhoneno(rs.getString(4));
            }
        }
    }
}

```

```

        s.setMarks(rs.getInt(5));

        list.add(s);
    }

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

return list;
}

public static int updateStudent(Student s) {
    int status = 0;

    try {
        Connection con = DBConnection.getConnection();
        PreparedStatement ps = con.prepareStatement(
            "UPDATE student SET name=?, address=?, phoneno=?, marks=? WHERE rollno=?");
        ps.setString(1, s.getName());
        ps.setString(2, s.getAddress());
        ps.setString(3, s.getPhoneno());
        ps.setInt(4, s.getMarks());
        ps.setInt(5, s.getRollno());
        status = ps.executeUpdate();
        con.close();
    } catch (Exception e) { e.printStackTrace(); }

    return status;
}

public static int deleteStudent(int rollno) {
    int status = 0;

    try {
        Connection con = DBConnection.getConnection();
        PreparedStatement ps = con.prepareStatement("DELETE FROM student WHERE rollno=?");
        ps.setInt(1, rollno);
    }

```

```

        status = ps.executeUpdate();

        con.close();

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

    return status;

}

}

```

student/UpdateDeleteServlet.java

```

package student;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class UpdateDeleteServlet extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {

            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet UpdateDeleteServlet</title>");
            out.println("</head>");
            out.println("<body>");

```

```

        out.println("<h1>Servlet UpdateDeleteServlet at " + request.getContextPath() + "</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}

```

@Override

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
```

```
    throws ServletException, IOException {
```

```
        int rollno = Integer.parseInt(request.getParameter("rollno"));
```

```
        String action = request.getParameter("action");
```

```
        Cookie cookie = new Cookie("rollno", String.valueOf(rollno));
```

```
        response.addCookie(cookie);
```

```
        if ("Update".equals(action)) {
```

```
            Student s = new Student();
```

```
            s.setRollno(rollno);
```

```
            s.setName(request.getParameter("name"));
```

```
            s.setAddress(request.getParameter("address"));
```

```
            s.setPhoneno(request.getParameter("phoneno"));
```

```
            s.setMarks(Integer.parseInt(request.getParameter("marks")));
```

```
            StudentDAO.updateStudent(s);
```

```
        } else if ("Delete".equals(action)) {
```

```
            StudentDAO.deleteStudent(rollno);
```

```
        }
```

```
        response.sendRedirect("showRecords.jsp");
```

```
    }
```

```
}
```

Index.jsp(Input Form)

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
```

```

<!DOCTYPE html>

<html>

  <head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

    <title>JSP Page</title>

  </head>

  <body>

    <form action="UpdateDeleteServlet" method="post" align="center">

Roll No: <input type="text" name="rollno" required><br><br>

Name: <input type="text" name="name"><br><br>

Address: <input type="text" name="address"><br><br>

Phone No: <input type="text" name="phoneno"><br><br>

Marks: <input type="text" name="marks"><br><br>

    <input type="submit" name="action" value="Update">

    <input type="submit" name="action" value="Delete">

  </form>

  </body>

</html>

```

showRecords.jsp (Display all)

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@page import="java.util.*,student.*"%>

<!DOCTYPE html>

<html>

<head><title>All Student Records</title></head>

<body>

<h2 align="center">All Student Records</h2>

<table border="1" align="center" cellpadding="8">

<tr><th>Roll No</th><th>Name</th><th>Address</th><th>Phone No</th><th>Marks</th></tr>

<%

List<Student> list = StudentDAO.getAllStudents();

```



```

for(Student s : list){
%>

<tr>

<td><%=s.getRollno()%></td>

<td><%=s.getName()%></td>

<td><%=s.getAddress()%></td>

<td><%=s.getPhoneno()%></td>

<td><%=s.getMarks()%></td>

</tr>

<% } %>

</table>

<%

Cookie[] ck = request.getCookies();

if (ck != null) {

    for (Cookie c : ck) {

        if (c.getName().equals("rollno")) {

            out.println("<p align='center'>Last modified Roll No: " + c.getValue() + "</p>");

        }

    }

}

%>

</body>

</html>

```

10. Develop a Java Web Application insert records into the MySQL Database and After inserting data show all records on webpage. Hint: Create table Book having field (B_Id,B_name,price,author). • Take necessary values from user. • Create necessary JSP pages and Use Java Beans and Servlet • Use Hibernate framework. • Display proper messages.

Code:

Database Table-bookdb(MySQL)

```
CREATE TABLE book (  
    B_Id INT PRIMARY KEY,  
    B_name VARCHAR(100),  
    price DOUBLE,  
    author VARCHAR(50)  
);
```

model/Book.java(Entity)

```
/*  
    * To change this license header, choose License Headers in Project Properties.  
    * To change this template file, choose Tools | Templates  
    * and open the template in the editor.  
    */
```

```
package model;
```

```
import java.io.Serializable;  
import javax.persistence.Basic;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.Id;  
import javax.persistence.NamedQueries;  
import javax.persistence.NamedQuery;  
import javax.persistence.Table;  
import javax.xml.bind.annotation.XmlRootElement;
```

```
/**  
    *  
    * @author Lenovo  
    */
```

```
@Entity
```

```
@Table(name = "book")
```

```
@XmlRootElement
```

```

@NamedQueries({
    @NamedQuery(name = "Book.findAll", query = "SELECT b FROM Book b")
    , @NamedQuery(name = "Book.findById", query = "SELECT b FROM Book b WHERE b.bld = :bld")
    , @NamedQuery(name = "Book.findByName", query = "SELECT b FROM Book b WHERE b.bname
= :bname")
    , @NamedQuery(name = "Book.findByPrice", query = "SELECT b FROM Book b WHERE b.price =
:price")
    , @NamedQuery(name = "Book.findByAuthor", query = "SELECT b FROM Book b WHERE b.author
= :author"}})

```

```

public class Book implements Serializable {

```

```

    private static final long serialVersionUID = 1L;

```

```

    @Id

```

```

    @Basic(optional = false)

```

```

    @Column(name = "B_Id")

```

```

    private Integer bld;

```

```

    @Column(name = "B_name")

```

```

    private String bname;

```

```

    // @Max(value=?) @Min(value=?)//if you know range of your decimal fields consider using these
    annotations to enforce field validation

```

```

    @Column(name = "price")

```

```

    private Double price;

```

```

    @Column(name = "author")

```

```

    private String author;

```

```

    public Book() {

```

```

    }

```

```

    public Book(Integer bld) {

```

```

        this.bld = bld;

```

```

    }

```

```

    public Integer getBld() {

```

```

        return bld;

```

```

    }

```

```

public void setBld(Integer bld) {
    this.bld = bld;
}

public String getBname() {
    return bname;
}

public void setBname(String bname) {
    this.bname = bname;
}

public Double getPrice() {
    return price;
}

public void setPrice(Double price) {
    this.price = price;
}

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (bld != null ? bld.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not set

```

```

    if (!(object instanceof Book)) {
        return false;
    }

    Book other = (Book) object;

    if ((this.bld == null && other.bld != null) || (this.bld != null && !this.bld.equals(other.bld))) {
        return false;
    }

    return true;
}

@Override

public String toString() {
    return "model.Book[ bld=" + bld + " ]";
}
}

```

hibernate.cfg.xml(Hibernate config file)

```

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

    <session-factory>

        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>

        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/bookdb?zeroDateTimeBehavior=con
vertToNull</property>

        <property name="hibernate.connection.username">root</property>

        <property name="hibernate.connection.password"/>

        <property name="hibernate.hbm2ddl.auto">update</property>

        <property name="hibernate.show_sql">true</property>

        <mapping class="model.Book"/>

    </session-factory>

</hibernate-configuration>

```

util/HibernateUtil.java (Hibernate configuration helper)

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package util;

import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {

    private static final SessionFactory sessionFactory;

    static {
        try {
            // Create the SessionFactory from standard (hibernate.cfg.xml)
            // config file.
            sessionFactory = new Configuration()
                .configure("hibernate.cfg.xml").buildSessionFactory();
        } catch (Throwable ex) {
            // Log the exception.
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

dao/BookDAO.java

```
package dao;
```

```
import org.hibernate.*;
```

```
import org.hibernate.query.Query;
```

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

```
import model.Book;
```

```
import util.HibernateUtil;
```

```
public class BookDAO {
```

```
    public void insertBook(Book book) {
```

```
        Session session = HibernateUtil.getSessionFactory().openSession();
```

```
        Transaction tx = session.beginTransaction();
```

```
        session.save(book);
```

```
        tx.commit();
```

```
        session.close();
```

```
    }
```

```
    public List<Book> getAllBooks() {
```

```
        Session session = HibernateUtil.getSessionFactory().openSession();
```

```
        Query<Book> query = session.createQuery("from Book", Book.class);
```

```
        List<Book> list = query.list();
```

```
        session.close();
```

```
        return list;
```

```
    }
```

```
}
```

```
servlet/BookServlet.java
```

```
package servlet;
```

```
import javax.servlet.*;
```

```
import javax.servlet.http.*;
```

```
import java.io.IOException;
```

```
import dao.BookDAO;
```

```

import model.Book;

import java.util.List;

public class BookServlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        int id = Integer.parseInt(request.getParameter("B_Id"));

        String name = request.getParameter("B_name");

        double price = Double.parseDouble(request.getParameter("price"));

        String author = request.getParameter("author");


        Book book = new Book();

        book.setB_Id(id);

        book.setB_name(name);

        book.setPrice(price);

        book.setAuthor(author);

        BookDAO dao = new BookDAO();

        dao.insertBook(book);

        request.setAttribute("message", "Book inserted successfully!");

        List<Book> books = dao.getAllBooks();

        request.setAttribute("bookList", books);


        RequestDispatcher rd = request.getRequestDispatcher("showBooks.jsp");

        rd.forward(request, response);

    }

}

```

index.jsp

```

<%@ page language="java" contentType="text/html; charset=UTF-8" %>

<html>

<head>

```



```

<title>Add Book</title>

</head>

<body bgcolor="#F5F5F5">

<h2 align="center">Add New Book</h2>

<form action="BookServlet" method="post" align="center">

    Book ID: <input type="text" name="B_Id" required><br><br>

    Book Name: <input type="text" name="B_name" required><br><br>

    Price: <input type="text" name="price" required><br><br>

    Author: <input type="text" name="author" required><br><br>

    <input type="submit" value="Add Book">

</form>

</body>

</html>

```

showBooks.jsp

```

<%@ page import="java.util.*, model.Book" %>

<html>

<head><title>All Books</title></head>

<body>

<h2 align="center">All Book Records</h2>


<%

String msg = (String)request.getAttribute("message");

if (msg != null) {

%>

<p style="color:green;" align="center"><%=msg%></p>

<%

}

List<Book> books = (List<Book>)request.getAttribute("bookList");

if (books != null && !books.isEmpty()) {

%>

```

```
<table border="1" align="center" cellpadding="8">
<tr><th>ID</th><th>Name</th><th>Price</th><th>Author</th></tr>
<%
for(Book b : books){
%>
<tr>
<td><%=b.getB_id()%></td>
<td><%=b.getB_name()%></td>
<td><%=b.getPrice()%></td>
<td><%=b.getAuthor()%></td>
</tr>
<% } %>
</table>
<% } else { %>
<p align="center">No books found.</p>
<% } %>

<p align="center"><a href="index.jsp">Add another book</a></p>
</body>
</html>
```