CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+
Accredited University

Worksheet No. - 3

Student Name: Devanand Utkarsh
UID: 24MCA20454
Branch: MCA
Section/Group: 6 (B)
Semester: II
Date of Performance: 15-02-2025
Subject Name: AIP LAB
Subject Code: 24CAP652

Aim/Overview of the practical: Create Employee_Management_app and perform CRUD operation to the database.

Objective: The objective of this assignment is to design and develop an Employee Management Application that allows users to perform CRUD (Create, Read, Update, Delete) operations on employee records stored in a database. This project will help in understanding database interactions, backend logic, and frontend integration for managing employee data efficiently.

Input/Apparatus Used: IntelliJ Idea as code editor, MySQL for Database, Hibernate as Database connector.

Procedure/Algorithm/Code:

"Index.jsp"

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
    <title>Employee Management</title>
    <script>
        function fetchEmployees() {
            window.location.href = "hello-servlet";
        }

        function addEmployee(event) {
event.preventDefault();
            console.log("add employee called");
```

```
        let formData = new FormData(document.getElementById("employeeForm"));
fetch("hello-servlet", {

            method: "POST",
            body: new URLSearchParams(formData),
headers: {
                "Content-Type": "application/x-www-form-urlencoded"
            }
        }).then(response => response.text()).then(data => {
alert(data)
        })
    }

    function updateEmployee(event) {
event.preventDefault();
        let formdata = new FormData(event.target);
        fetch("hello-servlet", {
method: "POST",
            body: new URLSearchParams(formdata),
headers: {
                "Content-Type": "application/x-www-form-urlencoded"
            }
        }).then(response => response.text().then(data => alert(data)))
    }

    function deleteEmployee(event) {
event.preventDefault();
        let formdata = new FormData(event.target);
        fetch("hello-servlet", {
method: "POST",
            body: new URLSearchParams(formdata),
headers: {
                "Content-type": "application/x-www-form-urlencoded"
            }
        }).then(response => response.text()).then(data => {
alert(data)
        })
    }
```

```
    </script>
</head>
<body>
<div class="container">
    <h2>Employee Management</h2>
    <form id="employeeForm" onsubmit="addEmployee(event)">
        <input type="text" value="addEmployee" name="action" hidden/>
        <input type="text" name="name" placeholder="Employee Name" required>
        <input type="text" name="department" placeholder="Department" required>
        <input type="number" name="salary" placeholder="Salary" required>
        <button type="submit">Add Employee</button>
    </form>
    <hr>
    <button onclick="fetchEmployees()">Fetch All Employees</button>
<hr>
    <form id="updateForm" onsubmit="updateEmployee(event)">
        <h2>Enter details which you want to update: </h2>
        <input type="text" name="action" value="updateEmployee" hidden>
        <input type="number" name="id" placeholder="Employee id" required>
        <input type="text" name="name" placeholder="Employee Name" required>
        <input type="text" name="department" placeholder="Department" required>
        <input type="number" name="salary" placeholder="Salary" required>
        <button type="submit">Update Employee</button>
    </form>

    <form id="deleteForm" onsubmit="deleteEmployee(event)">
        <h2>Enter the id of the employee which you want to delete: </h2>
        <input type="text" name="action" value="deleteEmployee" hidden>
<input type="number" placeholder="Enter employee id" name="employeeId"
required>
        <button type="submit">Delete Employee</button>
    </form>
</div>
</body>
</html>
```

“Hibernate.cfg.xml”

```
<hibernate-configuration>
```

```
   <session-factory>
      <!-- JDBC Database connection settings -->
<property


name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
      <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
      <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/DBemployee</property>
      <property name="hibernate.connection.username">root</property>
      <property name="hibernate.connection.password">dev&XXXX</property>
      <!-- JDBC connection pool settings -->
      <property name="hibernate.c3p0.min_size">5</property>
      <property name="hibernate.c3p0.max_size">20</property>
      <property name="hibernate.c3p0.timeout">300</property>
      <!-- Enable NHibernate's automatic session context management -->
      <property name="hibernate.current_session_context_class">thread</property>
      <!-- Echo all executed SQL to stdout -->
      <property name="hibernate.show_sql">true</property>
      <!-- Drop and re-create the database schema on startup -->
      <property name="hibernate.hbm2ddl.auto">update</property>
      <!-- Mention annotated class -->
      <mapping
class="com.devanand.employee_management_system.model.Employee"/>    </session-
factory>
</hibernate-configuration>
```

"Employee.java"

```
package com.devanand.kumar.employee_management_system.model;

import javax.persistence.Entity;
 import javax.persistence.Id;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Table;
@Entity
@Table(name = "employees")
```

```
public class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
private int id;


    private String name;
private String department;
private double salary;
public Employee() {}
    public Employee(String name, String department, double salary) {
this.name = name;         this.department = department;
        this.salary = salary;
    }
    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 getDepartment() { return department; }
    public void setDepartment(String department) { this.department = department; }
public double getSalary() { return salary; }
    public void setSalary(double salary) { this.salary = salary; } }
```

“EmployeeService.java”

```
  package com.devanand.employee_management_system.service;

  import com.devanand.employee_management_system.model.Employee;
  import org.hibernate.Session;

import org.hibernate.SessionFactory; import org.hibernate.Transaction;

import org.hibernate.cfg.Configuration;


  import java.util.List;
  public class EmployeeService {
```

```
    private static final SessionFactory factory;
static {
        factory = new
Configuration().configure("hibernate.cfg.xml").addAnnotatedClass(Employee.class).buil
dSessionFactory();
    }

    public static void saveEmployee(Employee employee) {
Session session = factory.getCurrentSession();        Transaction
transaction = session.beginTransaction();
session.persist(employee);
        transaction.commit();
    }

    public static List<Employee> getAllEmployees() {
        Session session = factory.getCurrentSession();
        Transaction transaction = session.beginTransaction();
        List<Employee> employees = session.createQuery("from Employee",
Employee.class).getResultList();
transaction.commit();
        return employees;
    }

    public Employee getEmployee(int employeeId) {
        Session session = factory.getCurrentSession();
        Transaction transaction = session.beginTransaction();
        Employee employee = session.get(Employee.class, employeeId);
transaction.commit();
        return employee;
    }

    public static void updateEmployee(int id, String name, String department, double
salary) {
        Session session = factory.getCurrentSession();


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

```
        Employee employee = session.get(Employee.class, id);
if(employee != null) {            employee.setName(name);
employee.setDepartment(department);
            employee.setSalary(salary);
            session.merge(employee);
        }
        transaction.commit();
    }

    public static void deleteEmployee(int employeeId) {
Session session = factory.getCurrentSession();
        Transaction transaction = session.beginTransaction();
        Employee employee = session.get(Employee.class, employeeId);
session.remove(employee);
        transaction.commit();
    }
}
```

“listEmployee.jsp”

```
<%--
  Created by IntelliJ IDEA.
  User: Devanand
  Date: 12-02-2025
  Time: 09:40 pm
  To change this template use File | Settings | File Templates.
--%>

<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<!DOCTYPE html>
<html>
<head>
    <title>Employee List</title>
</head>
<body>
<h2>Employee List</h2>
```

```
<table border="1">
  <thead>
  <tr>
    <th>ID</th>
    <th>Name</th>
    <th>Department</th>
    <th>Salary</th>
  </tr>
  </thead>
  <tbody>
  <c:forEach var="employee" items="${employees}">
    <tr>
      <td>${employee.id}</td>
      <td>${employee.name}</td>
      <td>${employee.department}</td>
      <td>${employee.salary}</td>
    </tr>
  </c:forEach>
  </tbody>
</table>
</body>
</html>
```

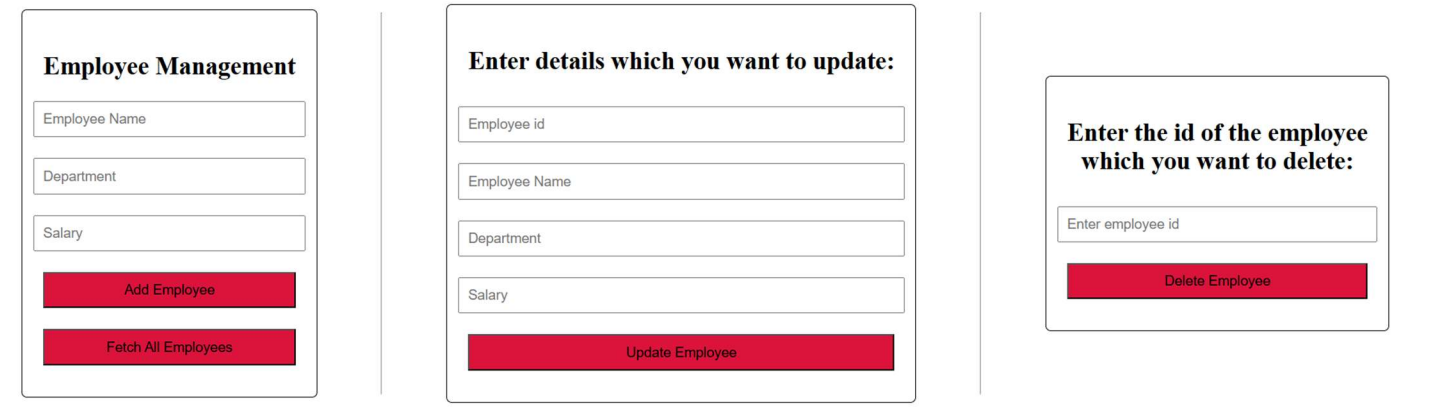
"employeeServlet.java"

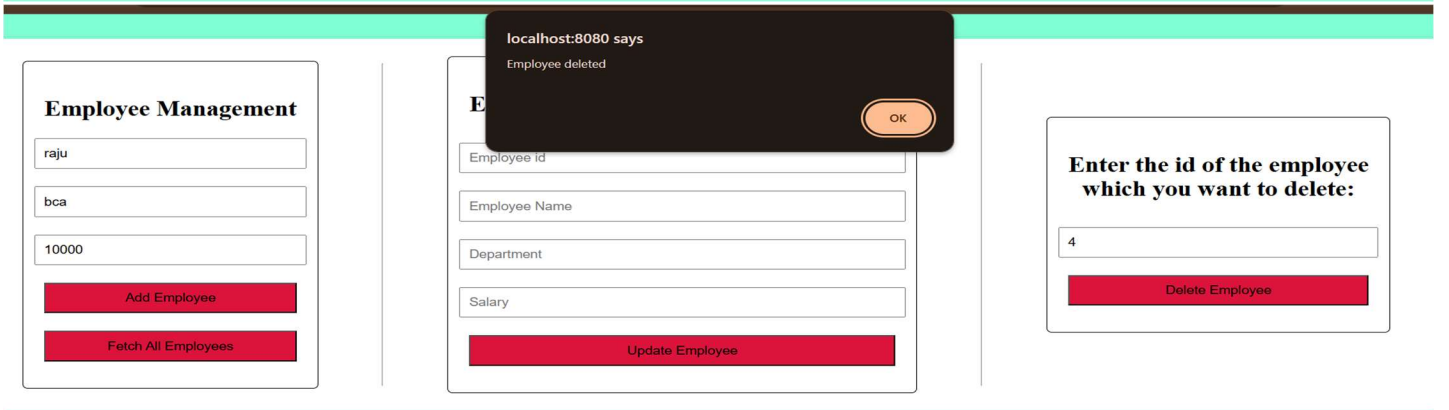
```
package com.employee.Servlet;

import com.employee.model.Employee;
import com.employee.service.EmployeeService;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.util.List;

@WebServlet("/EmployeeServlet")
public class EmployeeServlet extends HttpServlet {
```

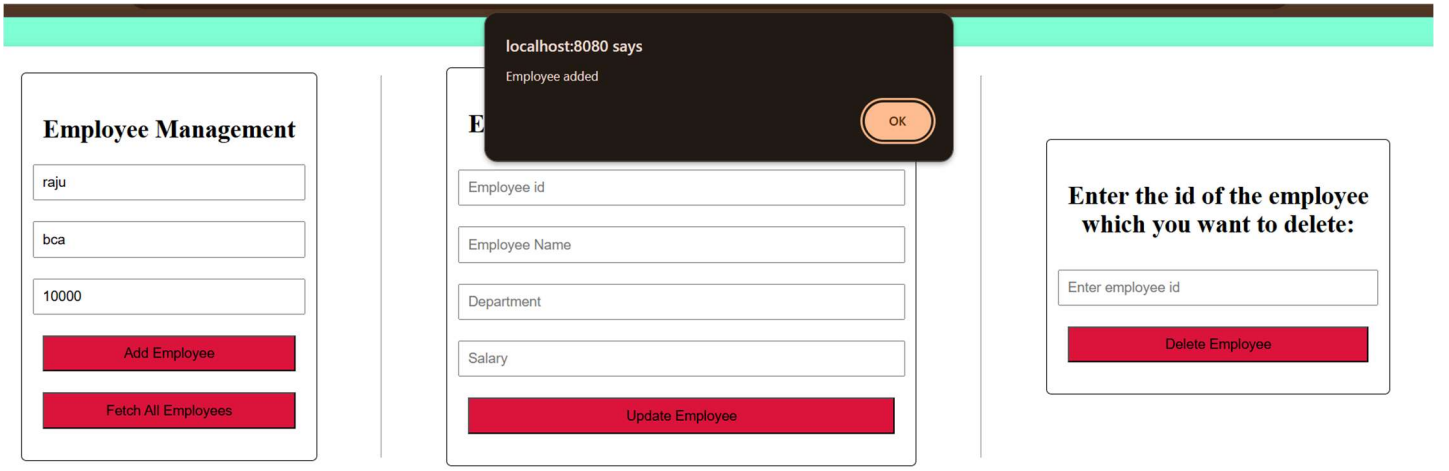
```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet EmployeeServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet EmployeeServlet at " + request.getContextPath() + "</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    List<Employee> employees = EmployeeService.getAllEmployees();
    request.setAttribute("employees", employees);
    request.getRequestDispatcher("/listEmployees.jsp").forward(request, response);
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    String name = request.getParameter("name");
    String department = request.getParameter("department");
    double salary = Double.parseDouble(request.getParameter("salary"));
    System.out.println(name + department + salary);
    Employee employee = new Employee(name, department, salary);
    EmployeeService.saveEmployee(employee);
    response.sendRedirect("EmployeeServlet");
}

@Override
public String getServletInfo() {
```

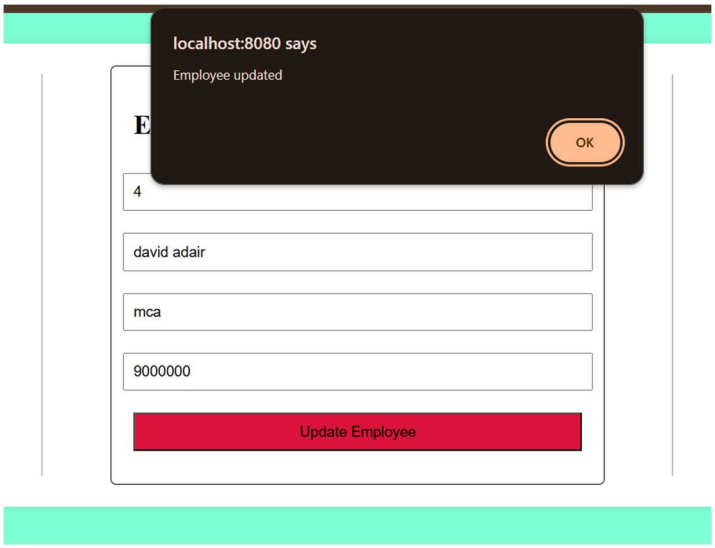
```
        return "Short description";
    }// </editor-fold>

}
```

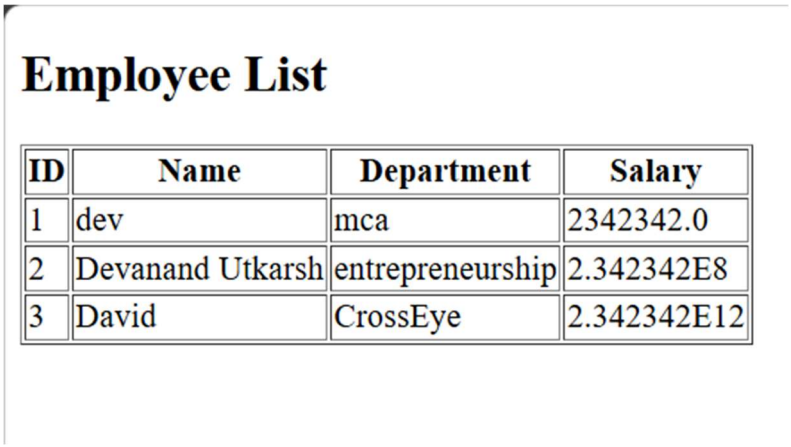
Result/Output :











Employee List

ID	Name	Department	Salary
1	dev	mca	2342342.0
2	Devanand Utkarsh	entrepreneurship	2.342342E8
3	David	CrossEye	2.342342E12

Learning outcomes (What I have learnt):

1. Learned how to implement Create, Read functionalities in a database.
2. Experienced build connectivity with database using hibernate.
3. Understand how to connect backend with frontend to send data and retrive data from database with seamless experience.