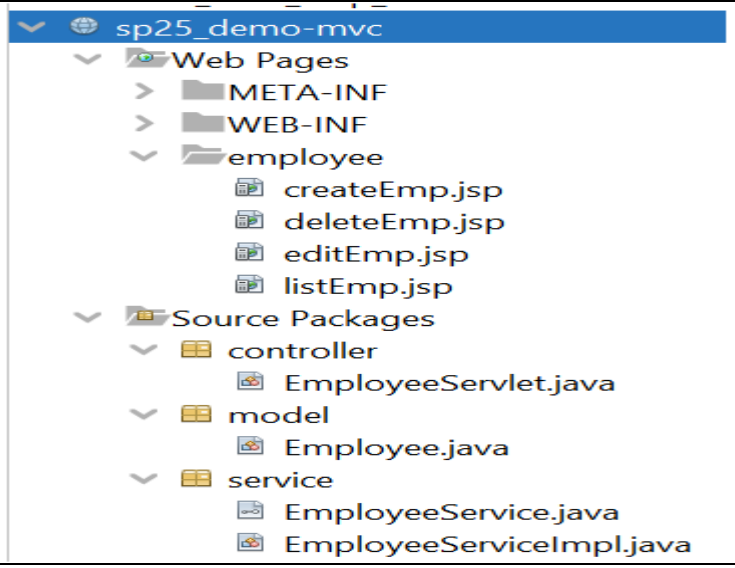
Bài toán	
Bài toánPhát triển một ứng dụng quản lý nhân viên: • Hiển thị danh sách nhân viên • Thêm một nhân viên mới • Sửa đổi thông tin của nhân viên • Xoá một nhân viên • Xem thông tin chi tiết của nhân viên	
Bước 1: Tạo ứng dụng web có tên sp15demo-mvc	

Bước 2: Tạo Employee model

Lớp Employee bao gồm các thuộc tính

id: Id của nhân viên

- name: Tên của nhân viên
- email:Email của nhân viên
- address:Địa chỉ của nhân viên

```java
public class Employee {
    private int id;
    private String name;
    private String email;
    private String address;

    public Employee(int id, String name, String email, String address) {
        this.id = id;
        this.name = name;
        this.email = email;
        this.address = address;
    }

    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 getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getAddress() {
        return address;
    }

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

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

}
```

Bước 3: Tạo interface EmployeeService

Interface EmployeeService là interface định nghĩa các phương thức ở tầng Service để thao tác với đối tượng Employee

Interface EmployeeService bao gồm các phương thức: findAllEmp():Trảvề danh sách tất cả nhân viên

save(): Lưu một nhân viên

findById():Tìm một nhân viên theo Id

update():Cập nhật thông tin của một nhân viên remove():Xoá một nhân viên khỏi danh sách

```java
public interface EmployeeService {

    List<Employee> findAll();
    void save(Employee customer);
    Employee findById(int id);
    void update(int id, Employee customer);
    void remove(int id);
}
```

Bước 4: Tạo class EmployeeServiceImpl

EmployeeServiceImpl là mộ tlớp triển khai đầy đủ các phương thức của interface EmployeeService.

• Trong trường hợp này, chúng ta xem lớp EmployeeServiceImpl Làm đối tượng cung cấp và thao tác dữ liệu. Danh sách nhân viên sẽ được lưu vào trong một đối tượng kiểu Map

DS Nhân viên lưu trong Database sẽ tìm hiểu phần sau.

```java
public class EmployeeServiceImpl implements EmployeeService {
    private static Map<Integer, Employee> employees = new HashMap<>();
    static {
        employees.put(1, new Employee(1, "John Doe", "john@example.com", "New York"));
        employees.put(2, new Employee(2, "Jane Doe", "jane@example.com", "California"));
    }
    @Override
    public List<Employee> findAll() {
        return new ArrayList<>(employees.values());
    }
    @Override
    public void save(Employee emp) {
        employees.put(emp.getId(), emp);
    }
    @Override
    public Employee findById(int id) {
        return employees.get(id);
    }
    @Override
    public void update(int id, Employee emp) {
        employees.put(id, emp);
    }
    @Override
    public void remove(int id) {
        employees.remove(id);
    }
    public static void main(String[] args) {
        for (Integer key : employees.keySet()) {
            System.out.println(key + " " + employees.get(key));
        }
    }
}
```

Bước 5: Tạo EmployeeServlet để xử lý các request.

• Lớp Employee Servlet cómột đối tượng EmployeeServiceImpl dùng để truy xuất đến dữliệu

```
25  @WebServlet(name="EmployeeServlet", urlPatterns={"/employees"})
26  public class EmployeeServlet extends HttpServlet {
        private EmployeeService employeeService = new EmployeeServiceImpl();
28
29      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30      throws ServletException, IOException {...15 lines }
45
46
47      @Override
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
49      throws ServletException, IOException {...20 lines }
69
70
71
72
73      @Override
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
75      throws ServletException, IOException {
76          processRequest(request, response);
77      }
```

Bước 6: Điều hướng việc xử lí các tính năng

Điều hướng việc xử lý các tính năng khác nhau của EmployeeServlet thông quatham số action. • Thamsố action được sử dụng để quy định hành động mà EmployeeServlet xử lý trong từngrequest:

/employees?action=create:Tạo một nhân viên mới /

/employees?action=edit:Cập nhật thông tin nhân viên /

employees?action=delete:Xoá một nhân viên

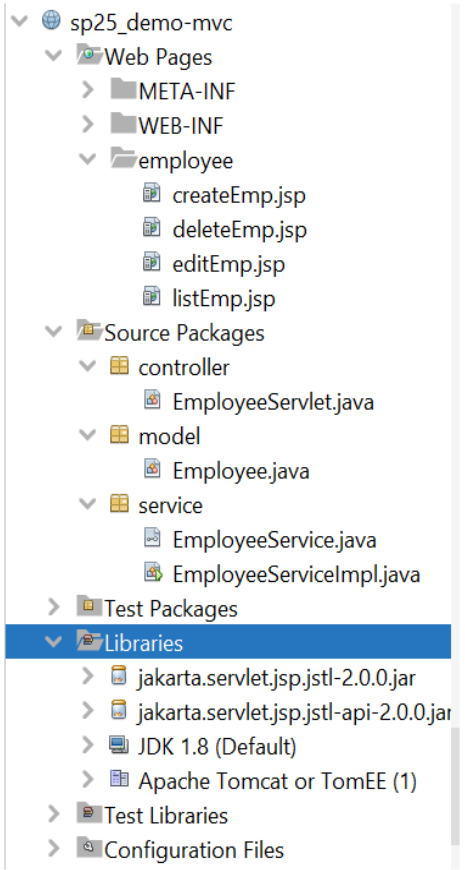
Các giá trị khác hiển thị danh sách nhân viên

```
25  @WebServlet(name="EmployeeServlet", urlPatterns={"/employees"})
26  public class EmployeeServlet extends HttpServlet {
        private EmployeeService employeeService = new EmployeeServiceImpl();
28
29      protected void processRequest(HttpServletRequest request, HttpServletResponse respons
30      throws ServletException, IOException {...15 lines }
45
46
47      @Override
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
49      throws ServletException, IOException {
50
51          String action = request.getParameter("action");
52          if (action == null) {
53              action = "";
54          }
55
56          switch (action) {
57              case "create":
58                  createEmp(request, response);
59                  break;
60              case "edit":
61                  break;
62              case "delete":
63                  break;
64              default:
65                  listEmployees(request, response);
66                  break;
67          }
68      }
```

Bước 7: Viết hàm hiển thị danh sách nhân viên Phương thức listEmployees((request,response)nhận về danhsách nhân viên chuyển sang listEmp/list.jsp để hiển thị

```
34      private void listEmployees(HttpServletRequest request, HttpServletResponse respon
35              throws ServletException, IOException {
36          List<Employee> emplist = employeeService.findAll();
37          request.setAttribute("employee", emplist);
38          RequestDispatcher dispatcher = request.getRequestDispatcher("listEmp.jsp")
39          dispatcher.forward(request, response);
40      }
```

Bước 8: Viết lệnh hiển thị danh sách nhân viên trên file listEmp.jsp
Gọi thư việ JSTL bằng thẻ %@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%




```
7   <%@page contentType="text/html" pageEncoding="UTF-8"%>
8   <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
9   <!DOCTYPE html>
10  <html>
11      <head>
12          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13          <title>JSP Page</title>
14      </head>
15      <body>
16        <h1>Employee List</h1>
17  <table border="1">
18      <tr>
19          <th>ID</th>
20          <th>Name</th>
21          <th>Email</th>
22          <th>Address</th>
23      </tr>
24      <c:forEach var="employee" items="${employees}">
25          <tr>
26              <td>${employee.id}</td>
27              <td>${employee.name}</td>
28              <td>${employee.email}</td>
29              <td>${employee.address}</td>
30          </tr>
31      </c:forEach>
32  </table>
33      </body>
34  </html>
```

Bước 9: Chạy chương trình http://localhost:9999/sp25_demo-mvc/employees JSP Page