Edit Operation in Mini Project

=> Here also we will having two operations
a) Form Launching having old values of specific employee based on empno
b) Edit Operation related to Form Submission

a) Form Launching having old values of specific employee based on empno

(a) Place the following code in service interface and in Service Impl class having logic to get Employee details based on the given employee no.

In service interface

```
public Employee getEmployeeByNo(int eno);
```

In service impl class

```
@Override
    public Employee getEmployeeByNo(int eno) {
        Employee emp=empRepo.findById(eno).orElseThrow(()->new IllegalArgumentException());
        return emp;

    }
```

(b) Place Handler method in Controller class to lanuch edit Employee operation related form page

In Controller class

```
@GetMapping("/emp_edit")
    public String showEditEmployeeFormPage(@RequestParam("no") int no,
                                           @ModelAttribute("emp") Employee emp) {
        //use service
        Employee emp1=empService.getEmployeeByNo(no);
        //copy data
        BeanUtils.copyProperties(emp1, emp);
        // return LVN
        return "update_employee";

    }
```

(c) Design the form page in WEB-INF/pages folder having name update_employee.jsp

```
<%@ page isELIgnored="false"%>
<%@taglib uri="http://www.springframework.org/tags/form" prefix="frm"%>

<h1 style="color:red;text-align:center">Update Employee</h1>

 <frm:form modelAttribute="emp">
  <table align="center" bgcolor="cyan">
  <tr>
    <td> Employee Number: </td>
    <td> <frm:input path="empno" readonly="true"/> </td>
  </tr>
  <tr>
    <td> Employee Name:: </td>
    <td> <frm:input path="ename"/> </td>
  </tr>
   <tr>
    <td> Employee Desg:: </td>
    <td> <frm:input path="job"/> </td>
   </tr>

    <tr>
    <td> Employee salary:: </td>
    <td> <frm:input path="sal"/> </td>
   </tr>

    <tr>
    <td> Deot no </td>
    <td> <frm:input path="deptno"/> </td>
   </tr>

   <tr>
    <td> <input type="submit" value="Update Employee"></td>
    <td> <input type="reset" value="cancel"> </td>
   </tr>

  </table>

 </frm:form>
```

b) Edit Operation related to Form Submission

i) Add the following logic in service interface and service impl class to update record

In service interface

```
public String updateEmployee(Employee emp);
```

In service impl class

```
@Override
    public String updateEmployee(Employee emp) {

        return "Employee is Updated with having id value ::"+empRepo.save(emp).getEmpNo();
    }
```

ii) develop the handler method in controller class to handle the edit employee form page sumbission related request.

```
@PostMapping("/emp_edit")
    public String editEmployee(RedirectAttributes attrs,
                               @ModelAttribute("emp") Employee emp) {
      //user service
      String msg=empService.updateEmployee(emp);
      // add the result message as Flash Attribute
      attrs.addFlashAttribute("resultMsg",msg);
      //redirect the redirect
      return "redirect:emp_report";

    }
```

Delete Operation in the MiniProject

i) Add methods in service interface and service impl class

In service interface

```
public String deleteEmployeeById(int eno);
```

In service impl class

```
@Override
    public String deleteEmployeeById(int eno) {

        empRepo.deleteById(eno);

        return eno+" employee id Employee is deleted";

    }
```

ii) Develope handler method in Controller class to perform delete employee operation

```
@GetMapping("/emp_delete")
    public String deleteEmployee(RedirectAttributes attrs,
                                 @RequestParam int no) {
        //use service
        String msg=empService.deleteEmployeeById(no);
        //keep the result in FlashAttribute
        attrs.addFlashAttribute("resultMsg", msg);
        //redirect the request
        return "redirect:emp_report";
    }
```

note: Here we can take return "forward:report" based handler method chaining becoz the both Source handler method and dest handler method are dealing with same GET mode requests

iii) enable double confirmation from enduser using confirm dialog box of Java script

```
<a href="emp_delete?no=${emp.empno}"
   onclick="return confirm('Do u want to delete the employee')">
                <img src="images/delete.png" width="30px" height="30px"></a>  </td>
```

This return statement sends the confirm dialog box true/false value choosen by enduser to browser .. If true .. browser continues the request submission .. If false the browser s/w cancels the raised request.

When should i use "forward:<path>" and when should i use "redirect: <path>" for Handler method chaining ?

Ans) if the source and dest handler methods are dealing with same mode request then go for "forward:<path>" otherwise go for "redirect:<path>" method

---

Edit Operation

=> Here also we will having two operations

in

Mini Project

a) Form Launching having old values of specific employee based on empno b) Edit Operation related to Form Submission

a) Form Launching having old values of specific employee based on empno

(a) Place the following code in service Interface and in Service Impl class having logic to get Employee details based on the given employee no.

In service Interface

```
public Employee getEmployeeByNo(int eno);
```

In service Impl class

```
@Override
public Employee getEmployeeByNo(int eno) {
Employee emp=empRepo.findById(eno).orElse Throw(()->new IllegalArgumentException()); return emp;
}
```

(b) Place Handler method in Controller class to lanuch edit Employee operation related form page In Controller class

```
@GetMapping("/emp_edit")
public String showEditEmployeeFormPage(@RequestParam("no") int no,
}
//use service
@ModelAttribute("emp") Employee emp) {
Employee emp1=empService.getEmployee ByNo(no);
//copy data
BeanUtils.copyProperties(emp1, emp);
// return LVN
return "update_employee";
```

(c) Design the form page in WEB-INF/pages folder having name

-

update_employee.jsp

```
<%@ page isELIgnored="false" %>
<%@taglib uri="http://www.springframework.org/tags/form" prefix="frm" %>
<h1 style="color:red;text-align:center">Update Employee</h1>
```

```
<frm:form modelAttribute="emp">
<table align="center" bgcolor="cyan">
<tr>
<td> Employee Number:: </td>
<td><frm:input path="empno" readonly="true"/> </td>
</tr>
<tr>
<td> Employee Name::</td>
<td><frm:input path="ename"/> </td>
</tr>
<tr>
<td> Employee Desg::</td>
<td><frm:input path="job"/> </td>
</tr>
<tr>
<td> Employee salary::</td>
<td><frm:input path="sal"/> </td>
</tr>
<tr>
<td> Dept no </td>
<td><frm:input path="deptno"/> </td>
</tr>
<tr>
<td> <input type="submit" value="Update Employee"></td>
<td> <input type="reset" value="cancel"> </td>
</tr>
</table>
</frm:form>
```

b) Edit Operation related to Form Submission

i) Add the following logic in service Interface and service Impl class to update record

In service Interface

```
public String updateEmployee(Employee emp);
```

In service Impl class

```
@Override
public String updateEmployee (Employee emp) {
}
```

```
return "Employee is Updated with having id value ::"+empRepo.save(emp).getEmpNo();
```

ii) develop the handler method in controller class

to handle the edit employee form page sumbission related request.

```
@PostMapping("/emp_edit")
public String editEmployee(RedirectAttributes attrs,
//user service
@ModelAttribute("emp") Employee emp) {
String msg=empService.updateEmployee(emp);
// add the result message as Flash Attribute
attrs.addFlashAttribute("resultMsg",msg);
//redirect the redirect
return "redirect:emp_report";
}
```

Delete Operation in the MiniProject

==========

i) Add methods in service Interface and service Impl class

In service Interface

```
public String deleteEmployee Byld(int eno);
```

In service Impl class

```
@Override
public String deleteEmployee Byld(int eno) {
empRepo.deleteById(eno);
return eno+" employee id Employee is deleted";
}
```

ii) Develope handler method in Controller class to perform delete employee operation

```
@GetMapping("/emp_delete")
public String deleteEmployee(RedirectAttributes attrs,
}
//use service
@RequestParam int no) {
String msg=empService.deleteEmployee Byld(no); //keep the result in FlashAttribute
attrs.addFlashAttribute("resultMsg", msg); //redirect the request
return "redirect:emp_report";
```

note: Here we can take return "forward:report" based handler method chaining becoz the both Source handler method and dest handler method are dealing with same GET mode requests

iii) enable double confirmation from enduser using confirm dialog box of Java script

```
<a href="emp_delete?no=${emp.empno}" onclick="return confirm('Do u want to delete the employee')">
```

<img src="images/delete.png" width="30px" height="30px"> </td>

This return statememt sends the confirm dialog box true/false value choosen by enduser to browser .. if true .. browser continues the request submission.. if false the browser s/w cancels the raised request.

When should i use "forward:<path>" and when should i use "redirect: <path>" for

Handler method chaining?

of

Ans) if the source and dest handler methods are dealing with same mode request then go for "forward:<path>" otherwise go for "redirect:<path>" method