```
                    Date : 04/03/2021
                    Spring Boot 9AM
                      Mr. RAGHU
          ---------------------------------------
Code:
https://www.mediafire.com/file/1y6wcxl2e3xsc5r/SpringBoot2WebMvcMySQlCru
d_04032021_9AM_RAGHU.zip/file
                  Mini Project Stages


Stage#1  Register Employee [Done]
Stage#2  Display Data [Done]
Stage#3  Delete By Id [Done]
Stage#4  Edit Page and Update Data [DONE]
Stage#5  UI BootStrap Design
Stege#6  UI-JQuery Validation
Stage#7  AJAX validation
Stage#8  Error Page Handling
Stage#9  Exception Handling
Stage#10 Log Implementation
-------------------------------------------------------
        Stage#4  Edit Page and Update Data


Edit Page :
-> Create One HyperLink using URL-Rewriting 'EDIT'
   at data page
-> On Click HyperLink, read data from DB table using
   id and display at UI using Thymeleaf Form.


--coding---
1. Service Interface
   Optional<Employee> getOneEmployee(Integer id);

2. ServiceImpl

   public Optional<Employee> getOneEmployee(Integer id) {
         return repo.findById(id);
   }

3. Controller
        @GetMapping("/edit")
        public String showEdit(
                      @RequestParam("id")Integer id,
                      Model model
                      )
        {
                String page = null;
                Optional<Employee> opt = service.getOneEmployee(id);

                if(opt.isPresent()) {
                        Employee emp = opt.get();
                        model.addAttribute("employee", emp);
                        page = "EmployeeEdit";
                } else {
                        page = "redirect:all";
                }

                return page;
```

```
        }


4. UI : EmployeeEdit.html
<!DOCTYPE html>
<html xmlns:th="https://www.thymeleaf.org/">
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h2>EMPLOYEE EDIT PAGE</h2>
<form th:action="@{/employee/update}" method="POST"
th:object="${employee}">
<pre>
ID     : <input type="text" th:field="*{empId}" readOnly/>
NAME   : <input type="text" th:field="*{empName}"/>
SALARY : <input type="text" th:field="*{empSal}"/>
DEPT   : <select th:field="*{empDept}">
                        <option value="">-SELECT-</option>
                        <option value="DEV">DEV</option>
                        <option value="QA">QA</option>
                        <option value="BA">BA</option>
                        <option value="MS">MS</option>
                </select>
ADDRESS: <textarea th:field="*{empAddr}"></textarea>
                <input type="submit" value="Update Employee"/>
</pre>
</form>
</body>
</html>


5. EmployeeData.html
<a th:href="@{/employee/edit(id=${ob.empId})}"> EDIT </a>
================================================================
Update Data:
 -> Read Form Data from Edit Page as one Object.
 -> Here Id(PK) is non-editable Input (Readonly)
 -> Object Data read in controller using @ModelAttribute
 -> Perform Update Operation and redirect to Data Page.

--code---
1. Service Interface
  void updateEmployee(Employee e);

2. Service Impl
    public void updateEmployee(Employee e) {
            //---calculations--
            var sal = e.getEmpSal();
            var hra = sal * 12/100;
            var ta = sal * 3/100;

            //set data to model cls obj
            e.setEmpHra(hra);
            e.setEmpTa(ta);

            //save data in db
```

```java
                //this method again returns same object
                // with PK updated value
                e = repo.save(e);
        }
```

3. Controller
```java
        @PostMapping("/update")
        public String updateEmployee(
                        @ModelAttribute Employee employee
                        )
        {
                service.updateEmployee(employee);
                return "redirect:all";
        }
```
--modified controller method--
```java
        @PostMapping("/update")
        public String updateEmployee(
                        @ModelAttribute Employee employee,
                        Model model
                        )
        {
                service.updateEmployee(employee);


                //fetch latest data after delete
                List<Employee> list = service.getAllEmployees();
                model.addAttribute("list", list);

                //create message and display at UI
                model.addAttribute("msg", "Employee
'"+employee.getEmpId()+"' Updated!!");
                // Goto UI Page
                return "EmployeeData";
                //return "redirect:all";
        }
```

==
DRY : Do not Repeate Youself (Dont Write Duplicate code)
Problem:
```java
class A{
 void m1() {
   sysout("1");
   sysout("2");
   sysout("3");
   sysout("4");
 }
 void m2() {
   sysout("1");
   sysout("2");
   sysout("3");
   sysout("5");
 }
}
```
Sol:
```java
class A{
 void m1() {
   m3();
```

```
      sysout("4");
  }
  void m2() {
    m3();
    sysout("5");
  }
  void m3() {
    sysout("1");
    sysout("2");
    sysout("3");
  }
}
```