Flooring Cost Estimator - V1

Grade settings: Maximum grade: 100
Based on: Spring MVC GLOBAL METADATA_Report
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 240 s **Maximum memory used**: 1.50 GiB **Maximum execution file size**: 1.50 GiB

Flooring Cost Estimator

One of the leading builders in Tamilnadu approaches you to create a separate online application for calculating flooring cost based on tiles brand (Kajaria/Bajaj/Nitco/OrientBell/Johnson) and the total square feet value chosen by the customer.

Create a Spring MVC Spring Boot Application for developing a Flooring Cost Estimator Application. Design a Flooring Cost Estimator Page to choose an appropriate tiles brand (Kajaria/Bajaj/Nitco/OrientBell/Johnson) and enter the total square feet.

On clicking CalculateCost button, the application should calculate the Flooring cost based on the tilesBrand chosen for the entered total square feet value. The customer has to then be redirected to result.jsp page that displays the message "**Approximate flooring cost for the given Square feet is Rs.<<flooringCost>>"**

Application Work Flow:

- **ContractController** is the Controller class.
- **FlooringContract** is the model class with three attributes tilesBrandName, costPerSqFt and totalSqFt along with its getters and setters.
- **ContractService** is the Service class which has a method called calculateFlooringCost that takes FlooringContract as its argument and returns double.

1. Calculate the flooring cost depending on the tiles rate per square feet and average flooring cost for the totalSqFt and tiles brand chosen by the customer.
2. This method should set the **costPerSqFt** instance variable based on tiles value given in the below table.
3. It should be returned as the double.

tilesBrandName	costPerSqFt
Kajaria	180.00
Bajaj	175.00
Nitco	160.00
OrientBell	155.00
Johnson	200.00

Initially, the customer should be routed via the ContractController's **estimatorPage** method to **estimatorpage.jsp** that allows customer to choose the tilesBrandName and enter the totalSqFt.

[Note: estimatorPage method has to be written inside the ConstructionController]

A method in the **ContractController** known as buildState should be annotated with the ModelAttribute "**brandNameList"**. This method should populate the tiles brand name **(Kajaria/Bajaj/Nitco/OrientBell/Johnson)** the Map as key-value pair. Both key and value be the same tilesBrandName. (Eg: Key- **Kajaria**, Value- **Kajaria**) and then return the Map. This should be then used to autopopulate the tilesBrandName in the **estimatorpage.jsp**

[Note: buildState method should be written inside the ContractController]

On clicking the Calculate button, the ContractController's **calculateFlooringCost** method should be called. This method takes three arguments - model attribute named "**contract"** which holds the form populated **FlooringContract** Object, BindingResult and the ModelMap.

- This method should calculate the **flooringCost** by invoking the **calculateFlooringCost** method of the **ContractService**.
- **flooringCost** has to be calculated as shown in the below example,
- **Example:**

If customer choose tilesBrand as "Johnson" and total square feet value as 1300

tilesCost= costPerSqFt * totalSqFt = 200.00 * 1300 = 260000.00

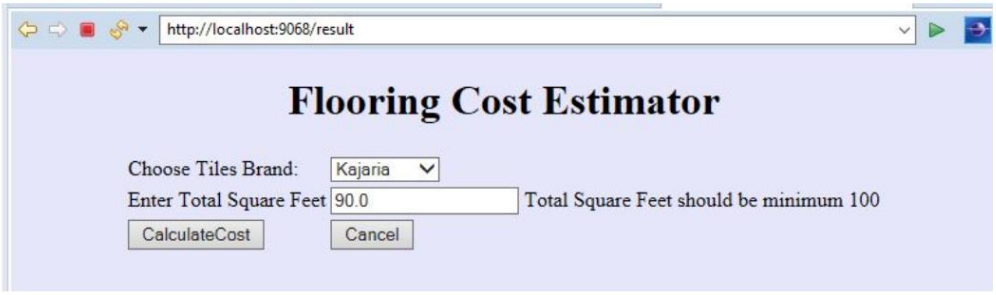
flooringCost = tilesCost + avgFlooringCost = 260000.00 + 94500.00 = 354500.00

Note: avgFlooringCost is given in the below table

totalSqFt	avgFlooringCost
>=100 & <=400	42000.00
>400 & <=700	73500.00
>700 & <=1400	94500.00
>1400	147000.00

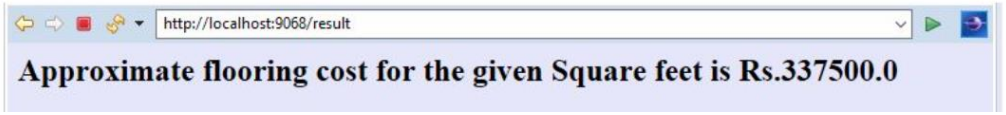
- If the totalSqFt entered by the customer is less than 100 then an error message "**Total Square Feet should be minimum 100**" has to be displayed.

estimatorpage.jsp



Redirect the customer to result.jsp page with a message "**Approximate flooring cost for the given Square feet is Rs.<<flooringCost>>**"

result.jsp



Design Constraints:

UI Design Constraints:

estimatorpage.jsp		
Component	**ID**	**Constraints**
Select	tilesBrandName	Should be auto populated using the model attribute written above the **buildState** method inside the **ContractController**. Do not hard code the values
Textbox	totalSqFt	Minimum value should be 100
submit	submit	-

Note:

- In **result.jsp**, the Result has to be rendered in the **<h2>** tag

Component Specification

Controller

ContractController

AttributeName	AttributeType	Access Specifier	Constraints
service	ContractService	private	Use annotation to Autowire

ContractController			
Method Name	**Method Argument name: type**	**Return type**	**RequestMapping URL & Request Method**
estimatorPage	modelAttribute "contract" FlooringContract	String	/estimatorPage & GET
calculateFlooringCost	modelAttribute "contract": FlooringContract, result:BindingResult, map:ModelMap	String	/result & POST
buildState		Map <String,String>	Should be annotated with ModelAttribute with name "**brandNameList"**

Service

ContractService		
Method Name	**Method Argument name: type**	**Return type**
calculateFlooringCost	flooringContract : FlooringContract	double

Model

FlooringContract	
AttributeName	**AttributeType**
tilesBrandName	String
totalSqFt	double
costPerSqFt	double

Overall Design Constraints:

- **ContractController** should be inside the package **com.controller**
- **ContractService** should be inside the package **com.service**
- **FlooringContract** should be in the package **com.model**
- Use appropriate annotation to configure ContractService as a Service
- Use appropriate annotation to configure ContractController as a Controller
- ContractService should be autowired inside the ContractController.
- Use annotations to implement the business Validation as specified in the screen shot **[That is, when the totalSqFt is less than 100 then error message "Total Square Feet should be minimum 100" should be rendered in the UI]**

- Do not change the property name given in the application.properties files, you can change the value and you can include additional property if needed.
- In the pom.xml you are provided with all the dependencies needed for developing the application.
- You will not be evaluated based on the UI design (layout, color, formatting, etc.). You are free to have a basic UI with all the required UI components (input fields, buttons, labels, etc.). Expected components with the id alone should be designed as per the requirement.
- Adhere to the design specifications mentioned in the case study.
- Do not change or delete the class/method names or return types which are provided to you as a part of the base code skeleton.
- Please make sure that your code does not have any compilation errors while submitting your case study solution.

Automatic evaluation[+]

FlooringCostEstimator/pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4         <modelVersion>4.0.0</modelVersion>
5         <parent>
6                 <groupId>org.springframework.boot</groupId>
7                 <artifactId>spring-boot-starter-parent</artifactId>
8                 <version>2.1.8.RELEASE</version>
9                 <relativePath/> <!-- lookup parent from repository -->
10        </parent>
11        <groupId>com.controller</groupId>
12        <artifactId>FlooringCostEstimator</artifactId>
13        <version>0.0.1-SNAPSHOT</version>
14        <name>FlooringCostEstimator</name>
15        <description>Demo project for Spring Boot</description>
16
17        <properties>
18                <java.version>1.8</java.version>
19        </properties>
20
21        <dependencies>
22                <dependency>
23                        <groupId>org.springframework.boot</groupId>
24                        <artifactId>spring-boot-starter-web</artifactId>
25                </dependency>
26
27                 <dependency>
28    <groupId>org.apache.tomcat.embed</groupId>
29    <artifactId>tomcat-embed-jasper</artifactId>
30    <scope>provided</scope>
31   </dependency>
32   <dependency>
33  <groupId>javax.servlet</groupId>
34  <artifactId>servlet-api</artifactId>
35  <version>2.5</version>
36  <scope>provided</scope>
37 </dependency>
38 <dependency>
39  <groupId>javax.servlet.jsp</groupId>
40  <artifactId>jsp-api</artifactId>
41  <version>2.1</version>
```

```
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>taglibs</groupId>
  <artifactId>standard</artifactId>
  <version>1.1.2</version>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
<!-- Spring test dependencies -->
        <dependency>
                                <groupId>org.mockito</groupId>
                                <artifactId>mockito-all</artifactId>
                                <version>1.10.19</version>
                                <scope>test</scope>
                        </dependency>

                        <!-- <dependency>
                                <groupId>org.seleniumhq.selenium</groupId>
                                <artifactId>selenium-java</artifactId>
                                <version>2.53.0</version>
                        </dependency>-->

                        <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>htmlunit-driver</artifactId>
    <version>2.26</version>
</dependency>

        <!-- https://mvnrepository.com/artifact/org.w3c.css/sac -->
<dependency>
    <groupId>org.w3c.css</groupId>
    <artifactId>sac</artifactId>
    <version>1.3</version>
</dependency>


<dependency>
                                <groupId>org.springframework</groupId>
                                <artifactId>spring-test</artifactId>
                                 <version>5.1.7.RELEASE</version>  <!-- 4.0.5 -->
                                <scope>test</scope>
                        </dependency>
<!-- End of spring test dependencies -->



                        <dependency>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-starter-test</artifactId>
                                <scope>test</scope>
                                <exclusions>
                                        <exclusion>
                                                <groupId>org.junit.vintage</groupId>
                                                <artifactId>junit-vintage-engine</artifactId>
                                        </exclusion>
                                </exclusions>
                        </dependency>
        </dependencies>

        <build>
                        <plugins>
                                <plugin>
```

```
                                        <groupId>org.springframework.boot</groupId>
                                        <artifactId>spring-boot-maven-plugin</artifactId>
                                </plugin>
                        </plugins>
        </build>

</project>


```

FlooringCostEstimator/src/main/java/com/controller/ContractController.java

```
package com.controller;

import java.util.HashMap;
import java.util.Map;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.model.FlooringContract;
import com.service.ContractService;

@Controller
public class ContractController {

        @Autowired
        private ContractService service;

        @RequestMapping(value = "/estimatorPage", method = RequestMethod.GET)
        public String estimatorPage(@ModelAttribute("contract") FlooringContract contract)
        {
                        return "estimatorpage";
        }

        @ModelAttribute("brandNameList")
        public  Map<String, String> buildState(){

                        Map<String, String> serviceMap = new HashMap<String, String>();
                        serviceMap.put("Kajaria", "Kajaria");
                        serviceMap.put("Bajaj", "Bajaj");
                        serviceMap.put("Nitco", "Nitco");
                        serviceMap.put("OrientBell", "OrientBell");
                        serviceMap.put("Johnson", "Johnson");
                        return serviceMap;
        }

        @RequestMapping(value = "/result", method = RequestMethod.POST)
        public String calculateFlooringCost(@Valid @ModelAttribute("contract") FlooringContract contract,
                                BindingResult result,ModelMap map)
        {
                        if(result.hasErrors()) {
                                return "estimatorpage";
        }
                        else
                        {
                        double cost=service.calculateFlooringCost(contract);
                        map.addAttribute("cost",cost);
                        return "result";

                        }
```

```
	}
}
```

FlooringCostEstimator/src/main/java/com/example/demo/FlooringCostEstimatorApplication.java

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan({"com.controller","com.model","com.service"})
public class FlooringCostEstimatorApplication {

	public static void main(String[] args) {
		SpringApplication.run(FlooringCostEstimatorApplication.class, args);
	}

}
```

FlooringCostEstimator/src/main/java/com/model/FlooringContract.java

```
package com.model;

import javax.validation.constraints.Min;

import org.springframework.stereotype.Component;

@Component
public class FlooringContract {

	private String tilesBrandName;
	@Min(value=100, message="Total Square Feet should be minimum 100")
	private double totalSqFt;
	private double costPerSqFt;

	public String getTilesBrandName() {
		return tilesBrandName;
	}
	public void setTilesBrandName(String tilesBrandName) {
		this.tilesBrandName = tilesBrandName;
	}
	public double getTotalSqFt() {
		return totalSqFt;
	}
	public void setTotalSqFt(double totalSqFt) {
		this.totalSqFt = totalSqFt;
	}
	public double getCostPerSqFt() {
		return costPerSqFt;
	}
	public void setCostPerSqFt(double costPerSqFt) {
		this.costPerSqFt = costPerSqFt;
	}



}

```

FlooringCostEstimator/src/main/java/com/service/ContractService.java

```java
package com.service;

import org.springframework.stereotype.Service;

import com.model.FlooringContract;

@Service
public class ContractService {

	public double calculateFlooringCost(FlooringContract obj)
	{
		double tilesCost,flooringCost,avgCost;
		if(obj.getTotalSqFt() >=100 && obj.getTotalSqFt() <=400)
		{
			avgCost=42000.0;
		}
		else if(obj.getTotalSqFt() >400 && obj.getTotalSqFt() <=700)
		{
			avgCost=73500.0;
		}
		else if(obj.getTotalSqFt() >700 && obj.getTotalSqFt() <=1400)
		{
			avgCost=94500.0;
		}
		else
		{
			avgCost=147000.00;
		}
		if(obj.getTilesBrandName().equalsIgnoreCase("Kajaria"))
		{
			obj.setCostPerSqFt(180.0);
			tilesCost=obj.getCostPerSqFt() * obj.getTotalSqFt();
			flooringCost=tilesCost+avgCost;
		}
		else if(obj.getTilesBrandName().equalsIgnoreCase("Bajaj"))
		{
			obj.setCostPerSqFt(175.0);
			tilesCost=obj.getCostPerSqFt() * obj.getTotalSqFt();
			flooringCost=tilesCost+avgCost;
		}
		else if(obj.getTilesBrandName().equalsIgnoreCase("Nitco"))
		{
			obj.setCostPerSqFt(160.0);
			tilesCost=obj.getCostPerSqFt() * obj.getTotalSqFt();
			flooringCost=tilesCost+avgCost;
		}
		else if(obj.getTilesBrandName().equalsIgnoreCase("OrientBell"))
		{
			obj.setCostPerSqFt(155.0);
			tilesCost=obj.getCostPerSqFt() * obj.getTotalSqFt();
			flooringCost=tilesCost+avgCost;
		}
		else
		{
			obj.setCostPerSqFt(200.0);
			tilesCost=obj.getCostPerSqFt() * obj.getTotalSqFt();
			flooringCost=tilesCost+avgCost;
		}
		return flooringCost;
	}

}

```

FlooringCostEstimator/src/main/resources/application.properties

```
server.port=9068
spring.mvc.view.prefix = /WEB-INF/views/
spring.mvc.view.suffix = .jsp
spring.mvc.static-path-pattern=/resources/**
```

FlooringCostEstimator/src/main/webapp/WEB-INF/views/estimatorpage.jsp

```
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>
<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form" %>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" isELIgnored="false"%>


<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<body style="background-color:lavender">
<h1><center> Flooring Cost Estimator </center></h1>
<form:form method="post" action="result" modelAttribute="contract">

<table style="margin: 0px auto; margin-left: auto; margin-right:auto">

                                    <tr>
                                                <td>Choose Tiles Brand:</td>
                                                <td>
                                                 <form:select path="tilesBrandName" id="tilesBrandName">
                                                                        <form:options items="${brandNameList}"/>
                                                            </form:select>

                                                </td>
                                    </tr>

                                    <tr>
                                    <td>Enter Total Square Feet</td><td><form:input path="totalSqFt"
id="totalSqFt"/></td><td><form:errors path="totalSqFt"/></tr>
                                    <tr>

                                                <td><input type="submit" value="CalculateCost" id="submit"
/></td>
                                                <td><input type="reset" value="Cancel"/></td>
                                    </tr>

                        </table>
</form:form>


</body>
</html>

```

FlooringCostEstimator/src/main/webapp/WEB-INF/views/result.jsp

```
<%@page isELIgnored="false" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<body bgcolor="lavender">
<h2>Approximate flooring cost for the given Square feet is Rs.${cost}</h2>
</body>
</html>
```

Grade

Reviewed on Thursday, 27 January 2022, 3:25 PM by Automatic grade
Grade 98.75 / 100

Assessment report
[-]SOURCE CODE ANALYZER REPORT

Error Msg: A method should have only one exit point, and that should be the last statement in the method
Variable Name:
Class Name: ContractController

Assessment Completed Successfully
[-]Grading and Feedback

Feature Test - 15.00 / 15.00(Success)
*   * check whether Service is configured in Controller class with the required Annotation - 4.50 / 4.50*
*   * check whether the Controller class with the required annotation is implemented - 3.50 / 3.50*
*   * check whether the Service class with the required annotation is implemented - 3.50 / 3.50*
*   * check whether the model class with the required annotation is implemented - 3.50 / 3.50*

Functional testing - 25.00 / 25.00(Success)
*   * check whether the flooring cost is calculated as per the requirement for KajariaTiles - 5.00 / 5.00*
*   * check whether the flooring cost is calculated as per the requirement for BajajTiles - 5.00 / 5.00*
*   * check whether the flooring cost is calculated as per the requirement for NitcoTiles - 5.00 / 5.00*
*   * check whether the flooring cost is calculated as per the requirement for OrientBellTiles - 5.00 / 5.00*
*   * check whether the flooring cost is calculated as per the requirement for JohnsonTiles - 5.00 / 5.00*

Spring Testing in the Controller - 25.00 / 25.00(Success)
*   * check whether the request is mapped for estimatorpage and redirected to estimatorpage.jsp - 5.00 / 5.00*
*   * check whether the model attribute is specified above the buildState method that holds the Map containing the tilesBrandName - 5.00 / 5.00*
*   * check whether the model attribute with the name brandNameList is specified above the buildState method that holds the Map containing the tilesBrandName - 5.00 / 5.00*
*   * check whether layering is followed that is whether flooring cost is calculated by invoking the service method inside the Controller - 5.00 / 5.00*
*   * check whether the validations are working correctly - 5.00 / 5.00*

UI Testing - 30.00 / 30.00(Success)
*   * check whether UI from estimatorpage to result page is redirected after calculating the flooring cost for KajariaTiles - 5.00 / 5.00*
*   * check whether UI from estimatorpage to result page is redirected after calculating the flooring cost for BajajTiles - 5.00 / 5.00*
*   * check whether UI from estimatorpage to result page is redirected after calculating the flooring cost for NitcoTiles - 5.00 / 5.00*
*   * check whether UI from estimatorpage to result page is redirected after calculating the flooring cost for OrientBellTiles - 5.00 / 5.00*
*   * check whether UI from estimatorpage to result page is redirected after calculating the flooring cost for JohnsonTiles - 5.00 / 5.00*
*   * check whether totalSqFt is rendered in estimatorpage as per the requirement - 2.00 / 2.00*
*   * check whether tilesBrandName is rendered in estimatorpage as per the requirement - 3.00 / 3.00*

Comments and best practices/standards - 3.75 / 5.0(Partial)