Delivery Agency - V1

Grade settings: Maximum grade: 100
Based on: Delivery Agency - V1
Run: Yes **Evaluate**: Yes
Automatic grade: Yes **Maximum execution time**: 240 s **Maximum memory used**: 1.50 GiB

Delivery Agency

Click here to download the code skeleton

Delivery Agency is a web application to track the delivery details of an order. Along with that they need to search the delivered and pending orders on a particular date. Help them by developing an application

Create a Spring MVC Spring Boot Application for developing a Delivery Agency Application. Design a search page to delivered/pending orders based on date and delivery status

On clicking search button, the application should list the order based on the given search criteria. Then the control has to be redirected to deliveryListPage.jsp page that displays the list of orders in tabular format

Application Work Flow:

- **AgencyController** is the Controller class.
- **Order** is the model class with attributes and its getters and setters.
- **AgencyService** is the Service class that has a method called **searchDeliveryOrders** that takes **Order** as its argument and returns ArrayList<Order>.
- Attribute ArrayList<Order> list; defined in the service class. And the value for this list was populated through **addOrders** method. Donâ€™t change the code in this method.
- Through searchDeliveryOrders method, have to search the **Order** objects based on the given input **delivery date** and **status**. (Iterate the list, get the **Order** objects which have the same date and status entered in UI)

Initially, the searchPage.jsp should be routed via the AgencyController's searchPage method ,that allows user to enter the delivery date and status.

[Note: searchPage method has to be written inside the **AgencyController]**

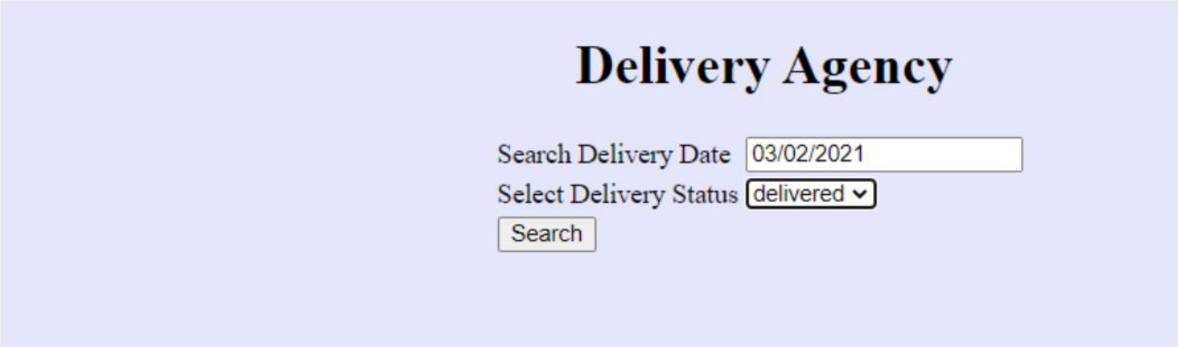
In AgencyService you are provided with a method **addOrders** which populates the list attribute with few Order objects.

On clicking the search button, the **AgencyController's searchDeliveryOrders** method should be called. This method takes three arguments - model attribute named â€œ**orderBean**â€• which holds the form populated **Order** Object, BindingResult and the ModelMap.

- This method should search the Order objects from the list based on the delivery date and status

Screen designs and Expected Output:

searchPage.jsp



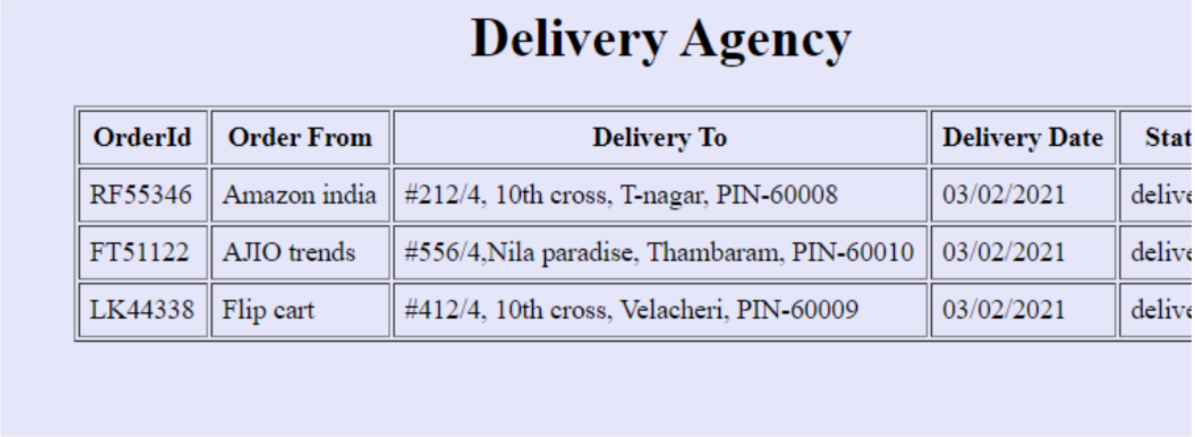
Delivery Agency

Search Delivery Date 03/02/2021

Select Delivery Status delivered

delivery date UI component is mandatory. if its empty display message as **"Delivery Date should be Mandatory"** has to be displayed in the searchPage.jsp. To do this validation use appropriate annotation above **deliveryDate** attribute in **Order** model class.

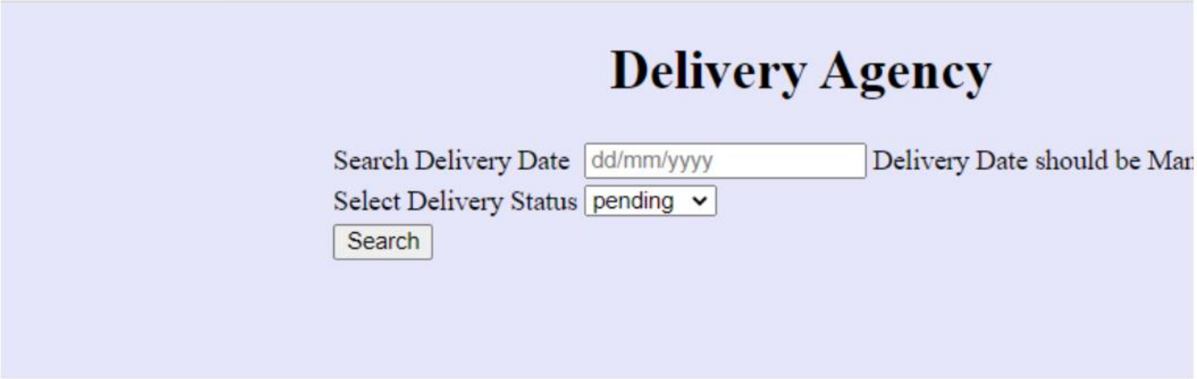
deliveryListPage.jsp



Delivery Agency

OrderId	Order From	Delivery To	Delivery Date	Stat
RF55346	Amazon india	#212/4, 10th cross, T-nagar, PIN-60008	03/02/2021	delive
FT51122	AJIO trends	#556/4,Nila paradise, Thambaram, PIN-60010	03/02/2021	delive
LK44338	Flip cart	#412/4, 10th cross, Velacheri, PIN-60009	03/02/2021	delive

If there is no **Order** objects available based on the search then raise the user-defined **NoOrdersFoundException** with error message **â€œNo orders found for this search criteriaâ€•** Redirect the user to **exceptionPage.jsp** page (display the error message **â€œNo orders found for this search criteria**

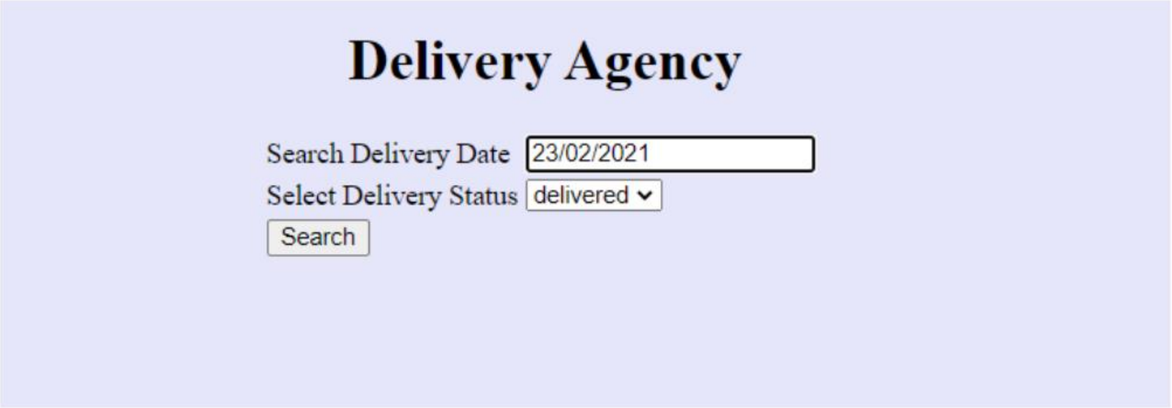


Delivery Agency

Search Delivery Date dd/mm/yyyy Delivery Date should be Man

Select Delivery Status pending

Exception flow:

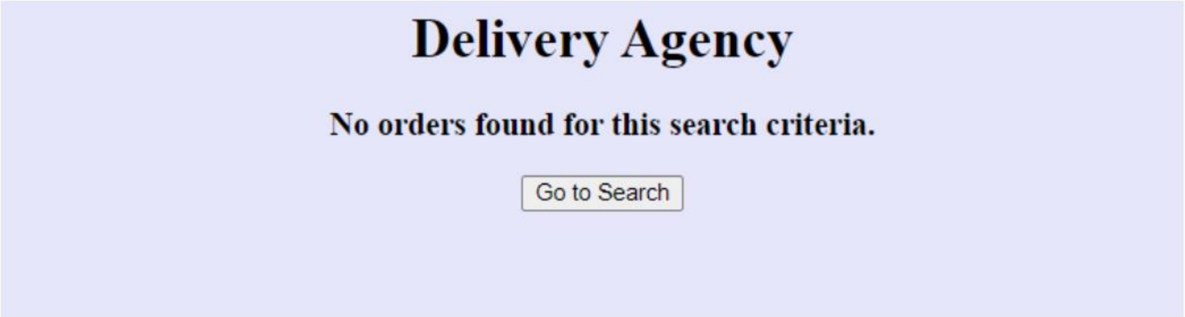


Delivery Agency

Search Delivery Date 23/02/2021

Select Delivery Status delivered

exceptionPage.jsp



Delivery Agency

No orders found for this search criteria.

By clicking the **â€œGo to Searchâ€•** button in **exceptionPage.jsp**, it should redirect to searchPage method (in AgencyController)

Design Constraints:

UI Design Constraints:

searchPage.jsp		
Component	**ID**	**Constraints**
textbox	deliveryDate	Name and id attribute should be deliveryDate, Default placeholder value should be "dd/mm/yyyy"
Drop down	deliveryStatus	Name and id attribute should be deliveryStatus. Populate the values as "delivered" and "pending"
submit	submit	Name and id attribute should be submit

Component Specification

Controller

AgencyController			
AttributeName	**AttributeType**	**Access Specifier**	**Constraints**

AgencyService	service	private	Use annotation to Autowire

AgencyController			
Method Name	**Method Argument name:type**	**Return type**	**RequestMapping URL & Request Method**
searchPage	modelAttribute â€œOrderBeanâ€ • :Order	String	/searchPage & GET
searchDeliveryOrders	modelAttribute â€œorderBeanâ€ • :Order, result:BindingResult, model:ModelMap	String	/orderList & POST

Service:

AgencyService		
Method Name	**Method Argument name:type**	**Return type**
searchDeliveryOrders	orderBean:Order	ArrayList<Order>

Model:

OrderBean	
AttributeName	**AttributeType**
orderId	String
orderFrom	String
deliveryTo	String
deliveryDate	String
deliveryStatus	String

Overall Design Constraints:

- **AgencyController** should be inside the package **com.controller**
- **AgencyService** should be inside the package **com.service**
- **Order** should be in the package **com.model**
- **NoOrdersFoundException** class should in package **com.exception.**
- Use appropriate annotation to configure **AgencyService** as a **Service**
- Use appropriate annotation to configure **AgencyController** as a **Controller**
- AgencyService should be **autowired** inside the **AgencyController**.
- Use appropriate annotation to handle **Exceptions** in **AgencyController** class.
- Use annotations to implement the business Validation as specified in the screenshot [That is when the **deliveryDate** is empty then error message **"Delivery Date should be Mandatory"** should be rendered in the UI.]

- In the pom.xml you are provided with all the dependencies needed for developing the application.
- 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.
- 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 that 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[+]

DeliveryAgency/pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.1.8.RELEASE</version>
                <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>DeliveryAgency</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>DeliveryAgency</name>
    <description>Demo project for Spring Boot</description>
    <properties>
                <java.version>1.8</java.version>
    </properties>

    <dependencies>
                <dependency>
                            <groupId>org.springframework.boot</groupId>
                            <artifactId>spring-boot-starter-web</artifactId>
                </dependency>

                 <dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
    <scope>provided</scope>
  </dependency>

  <dependency>
 <groupId>javax.servlet</groupId>
 <artifactId>servlet-api</artifactId>
 <version>2.5</version>
```

```
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <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>
```

DeliveryAgency/src/main/java/com/controller/AgencyController.java

```java
package com.controller;

import java.util.List;
import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.servlet.ModelAndView;
import com.model.Order;
import com.service.AgencyService;
import com.exception.NoOrdersFoundException;

//use appropriate annotation to configure AgencyController as Controller
@Controller
public class AgencyController {

    //Use appropriate annotation
    @Autowired
    private AgencyService service;

    @RequestMapping(value="/searchPage",method=RequestMethod.GET)
    public String searchPage(@ModelAttribute("orderBean")Order orderBean){
        return "searchPage";
    }


    //invoke the service class - searchDeliveryOrders method.
    @RequestMapping(value="/orderList",method=RequestMethod.POST)
    public String searchDeliveryOrders(@ModelAttribute("orderBean") @Valid Order orderBean,
BindingResult result,
                ModelMap model) throws NoOrdersFoundException {
            //fill the code
            if(result.hasErrors()){
                return "searchPage";
            } else{
                List<Order> ordersFound=service.searchDeliveryOrders(orderBean);
                if(ordersFound.isEmpty()){
                    throw new NoOrdersFoundException("No orders found for this search criteria");
                } else{
                    model.addAttribute("orders", ordersFound);
                    return "deliveryListPage";
                }
            }
    }

    //use appropriate annotation to handle the Exception
    @ExceptionHandler(NoOrdersFoundException.class)
    public ModelAndView exceptionHandler(Exception e) {

            //fill the code
            ModelAndView mav =new ModelAndView();
            mav.addObject("exception",e);
            mav.setViewName("exceptionPage");
            return mav;
    }
}

```

DeliveryAgency/src/main/java/com/example/demo/DeliveryAgencyApplication.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.*"})
public class DeliveryAgencyApplication {

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

}

```

DeliveryAgency/src/main/java/com/exception/NoOrdersFoundException.java

```
package com.exception;

public class NoOrdersFoundException extends Exception{
    private static final long serialVersionUID = 1L;
public NoOrdersFoundException(String msg) {

    //fill the code
        super(msg);
}

}

```

DeliveryAgency/src/main/java/com/model/Order.java

```
package com.model;

import javax.validation.constraints.NotEmpty;
import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters
@Component
public class Order {
    private String orderId;
```

```java
    private String orderFrom;
    private String deliveryTo;
    // fill the code

    @NotEmpty(message="Delivery Date should be Mandatory")
    private String deliveryDate;

    private String deliveryStatus;
    public String getOrderId() {
        return orderId;
    }
    public void setOrderId(String orderId) {
        this.orderId = orderId;
    }
    public String getOrderFrom() {
        return orderFrom;
    }
    public void setOrderFrom(String orderFrom) {
        this.orderFrom = orderFrom;
    }
    public String getDeliveryTo() {
        return deliveryTo;
    }
    public void setDeliveryTo(String deliveryTo) {
        this.deliveryTo = deliveryTo;
    }
    public String getDeliveryDate() {
        return deliveryDate;
    }
    public void setDeliveryDate(String deliveryDate) {
        this.deliveryDate = deliveryDate;
    }

    public String getDeliveryStatus() {
        return deliveryStatus;
    }
    public void setDeliveryStatus(String deliveryStatus) {
        this.deliveryStatus = deliveryStatus;
    }
    public Order(String orderId, String orderFrom, String deliveryTo, String deliveryDate, String deliveryStatus) {
        super();
        this.orderId = orderId;
        this.orderFrom = orderFrom;
        this.deliveryTo = deliveryTo;
        this.deliveryDate = deliveryDate;
        this.deliveryStatus = deliveryStatus;
    }

    public Order()          {}

}

```

DeliveryAgency/src/main/java/com/service/AgencyService.java

```java
package com.service;

import java.util.ArrayList;
import com.model.Order;
import org.springframework.stereotype.Service;

//use appropriate annotation to configure AgencyService as a Service
@Service
```

```
public class AgencyService {
    ArrayList<Order> list = new ArrayList<Order>();

    //search orders and return the ArrayList<Order>
    public ArrayList<Order> searchDeliveryOrders(Order orderBean) {
                //fill the code
                ArrayList<Order> foundOrders = new ArrayList<>();
                for(Order order:list){
                   if(order.getDeliveryDate().equals(orderBean.getDeliveryDate()) && order.getDeliveryStatus().equals(orderBean.getDeliveryStatus())){
                      foundOrders.add(order);
                   }
                }
                return foundOrders;

    }

    public AgencyService() {
                 addOrders();
    }

    public  void addOrders(){                              // don't change this code
                list.add(new Order("DF1234233","AJIO trends","#412, Shipitha cascade, Ganapathy, PIN-641001","10/02/2021","pending"));
                list.add(new Order("FGF7788","Amazon india","#12, shardhalotus, Ram Nagar, PIN-641001", "02/02/2021","pending"));
                list.add(new Order("AF5546","Mynthra","#12, Anandha nilayam,Kuniamuthur, PIN-641001","08/02/2021","pending"));
                list.add(new Order("DF5543","Flip cart","#201, Shipitha cascade, Saravanampatti, PIN-641001","10/02/2021","delivered"));

                list.add(new Order("DX17788","AJIO trends","#34/45, RV Homes, 5th cross, Anssari street, PIN-600001","01/02/2021","pending"));
                list.add(new Order("RF55346","Amazon india","#212/4, 10th cross, T-nagar, PIN-60008","03/02/2021","delivered"));
                list.add(new Order("DF88994","Mynthra","#556/4, 5th cross, Thambaram, PIN-60010","12/02/2021","pending"));
                list.add(new Order("FT51122","AJIO trends","#556/4,Nila paradise, Thambaram, PIN-60010","03/02/2021","delivered"));
                list.add(new Order("LK44338","Flip cart","#412/4, 10th cross, Velacheri, PIN-60009","03/02/2021","delivered"));

                list.add(new Order("JK99008","Amazon india","#45, 12Th main , GM Palya, PIN-560075","03/02/2021","pending"));
                list.add(new Order("MJ99765","Flip cart","#67/2, 3rd cross , Indra nagar, PIN-560077","08/02/2021","delivered"));
                list.add(new Order("NK66754","Mynthra","#302, Sakthi enclave ,1st main , Malleshpalya, PIN-560075","08/02/2021","delivered"));


    }
}


```

DeliveryAgency/src/main/resources/application.properties

```
#Do not edit the key server.port.Port number can be changed
#include other needed configurations
server.port=8088
spring.mvc.view.prefix = /WEB-INF/views/
spring.mvc.view.suffix = .jsp
spring.mvc.static-path-pattern=/resources/**

```

DeliveryAgency/src/main/webapp/WEB-INF/views/deliveryListPage.jsp

```
<%@page isELIgnored="false" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<body bgcolor="lavender">
<h1><center> Delivery Agency </center></h1>
<table>
   <tr>
      <td>OrderId</td>
      <td>Order From</td>
      <td>Delivery To</td>
      <td>Delivery Date</td>
      <td>Status</td>
   </tr>
   <c:forEach items="${orders}" var="order">
      <tr>
         <td>${order.orderId}</td>
         <td>${order.orderFrom}</td>
         <td>${order.deliveryTo}</td>
         <td>${order.deliveryDate}</td>
         <td>${order.deliveryStatus}</td>
      </tr>
    </c:forEach>
</table>
</body>
</html>
```

DeliveryAgency/src/main/webapp/WEB-INF/views/exceptionPage.jsp

```
<%@page isELIgnored="false" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<html>
<body bgcolor="lavender">
<h1><center> Delivery Agency </center></h1>
<center>
    <h3>${exception.message}</h3>
    <a href="/searchPage">
       <button>Go to Search</button>
    </a>
</center>
</body>
</html>
```

DeliveryAgency/src/main/webapp/WEB-INF/views/searchPage.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"%>
<%@ 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>Delivery Agency</center>
</h1>
<center>
  <sf:form action="orderList" name="form" modelAttribute="orderBean"
   method="POST">

   <fieldset>
```

```
            <label for='deliveryDate'>Search Delivery Date</label>
            <sf:input path="deliveryDate" id="deliveryDate" name="deliveryDate"
                placeholder="dd/mm/yyyy" />
            <sf:errors path="deliveryDate" />
            <br/>
            <label for='deliveryStatus'>Select Delivery Status</label>
            <sf:select path="deliveryStatus" id="deliveryStatus" name="deliveryStatus">
                <sf:option label="delivered" value="delivered"/>
                <sf:option label="pending" value="pending"/>
            </sf:select>
        </fieldset>
        <input type="submit" id="submit" value="Search" name="submit" />
        </sf:form>
        </center>
    </body>
    </html>

```

Grade

Reviewed on Wednesday, 12 January 2022, 6:39 AM by Automatic grade

Grade 95.75 / 100

Assessment report

FAILURE TEST CASES-UI

Fail 1 --test5ForSearchPageUIForDeliveryDateValidation::Check for the file searchPage.jsp/Check whether you have provided the components in searchPage.jsp as per the requirement

[-]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: AgencyController
```

[-]Grading and Feedback

```
Feature Test - 18.0 / 18.0(Success)
          * check whether the Controller / Service class with the required annotation is
implemented - 3.0 / 3.0
          * check whether the ExceptionHandler annotation is present above the Method - 6.0
/ 6.0
          * check whether the Service class with the required annotation is implemented -
3.0 / 3.0
          * check whether the model class with the required annotation is implemented - 3.0
/ 3.0

Functional testing - 18.0 / 18.0(Success)
          * check whether the searchDeliveryOrders method in AgencyService class with
pending status - 6.0 / 6.0
          * check the searchDeliveryOrders method in AgencyService class with pending
status  - 6.0 / 6.0
          * check searchDeliveryOrders method in AgencyService class  with delivered status
- 6.0 / 6.0

Spring Testing in the Controller - 28.0 / 28.0(Success)
          * check whether the request is mapped for searchPage and redirected to
searchPage.jsp page - 6.0 / 6.0
          * check whether the request is mapped for orderList and redirected to
deliveryListPage.jsp page - 6.0 / 6.0
          * check Whether exception was thrown properly - 10.0 / 10.0
          * check Delivery date UI component validated properly - 6.0 / 6.0

UI Testing - 28.0 / 31.0(Success)
```

```
         * check whether the searchPage is rendered with proper constraints for Date - 5.0
/ 5.0
         * check whether the searchPage is rendered with proper constraints for Status -
5.0 / 5.0
         * check whether UI from searchPage to deliveryListPage.jsp is redirected and
rendered the required output - 10.0 / 10.0
         * check whether error message is displayed correctly when the list is empty - 8.0
/ 8.0
         * check whether the deliveryDate field validated properly - 0.0 / 3.0

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