**LOAN ELIGIBILTY**

------------- loan eligibility .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>

<head>

<title>Loan Bazaar : Loan Eligibility Check</title>

</head>

<body style="background-color: rgb(200, 250, 300)">

<!-- ADD YOUR CODE HERE -->

<center>

<h3 style="color: blue">Loan Bazaar - Find the Best Loan For

You</h3>

<form:form modelAttribute="customerDetails"

action="eligibilityCheck"

method="post" id="loanEligibilityForm">

<table>

<tr>

<td>Name</td>

<td><form:input type="text" path="name" id="name"

/></td>

<td><form:errors path="name"

cssStyle="color:red"></form:errors></td>

</tr>

<tr>

<td>Gender</td>

<td><form:radiobuttons path="gender"

items="${genderList}" /></td>

<td><form:errors path="gender"

cssStyle="color:red"></form:errors>

</td>

</tr>

<tr>

<td>Email</td>

<td><form:input type="text" path="email"

id="email" /></td>

<td><form:errors path="email"

cssStyle="color:red" ></form:errors></td>

</tr>

<tr>

<td>Customer City</td>

<td><form:select path="customerCity"

items="${cityList}"

id="customerCity"/></td>

<td><form:errors path="customerCity"

cssStyle="color:red"></form:errors></td>

</tr>

<tr>

<td>Employment Type</td>

<td><form:select path="employmentType"

items="${employeeList}"

id="employmentType" /></td>

<td><form:errors path="employmentType"

cssStyle="color:red"></form:errors></td>

</tr>

<tr>

<td>Monthly Income in INR</td>

<td><form:input type="text" path="monthlyIncome"

id="monthlyIncome" /></td>

<td><form:errors path="monthlyIncome"

cssStyle="color:red"></form:errors></td>

</tr>

<tr>

<td>Desired Loan Amount in INR</td>

<td><form:input type="text"

path="desiredLoanAmount"

id="desiredLoanAmount" /></td>

<td><form:errors path="desiredLoanAmount"

cssStyle="color:red"></form:errors></td>

</tr>

</table>

<button type="submit" id="submit">Submit</button>

</form:form>

</center>

</body>

</html>

------------------ Results.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>

<head>

<title>Loan Bazaar : Loan Eligibility Check Results</title>

</head>

<body style="background-color: rgb(200, 250, 300)">

<!-- ADD YOUR CODE HERE -->

<h3>${msg}</h3>

<form:form id="results">

<c:if test="${not empty data}">

<table id="resultsTable" border="1">

<tr>

<td>Bank Name</td>

<td>Loan Product Name</td>

<td>Max Eligible Loan Amount in Rupees</td>

<td>Tenure</td>

<td>Interest</td>

<td>EMI in Rupees</td>

</tr>

<c:forEach items="${data}" var="a">

<tr>

<td>${a.bankName}</td>

<td>${a.loanProductName}</td>

<td>${a.maxLoanAmount}</td>

<td>${a.tenure}</td>

<td>${a.interest}</td>

<td>${a.monthlyInstallment}</td>

</tr>

</c:forEach>

</table>

</c:if>

</form:form>

</html>

-------------------- Error.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>

<head>

<title>Loan Bazaar : Loan Eligibility Errors</title>

</head>

<body style="background-color: rgb(200, 250, 300)">

<!-- ADD YOUR CODE HERE -->

<form:form id="error">

<h3>Unable to retrieve loan information. Below are the error details:</h3>

<h3>Response Code: ${code }</h3>

<h3>Error Message: ${message }</h3>

<h3>Error Occurred on: ${curtime }</h3>

</form:form>

</body>

</html>

------------------- Loan EligibiltyController.java

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class LoanEligibilityController is control the views and model objects

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\* ADD REQUEST MAPPING URI AND RETURN TYPE AS PER DESIGN DOCUMENT

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.controller;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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.cts.loanbazaar.loaneligibility.exception.ApplicationException;

import com.cts.loanbazaar.loaneligibility.model.CustomerDetails;

import com.cts.loanbazaar.loaneligibility.model.LoanProduct;

import com.cts.loanbazaar.loaneligibility.service.LoanEligibilityService;

/\*\*

\* Loan Eligibility Controller

\*

\*/

@Controller

public class LoanEligibilityController {

@Autowired

LoanEligibilityService loanEligibilityService;

/\*\*

\* @param model

\* @return String

\*/

@RequestMapping(value = "/home", method = RequestMethod.GET)

public String showHomePage(Model model) {

CustomerDetails customerDetails = new CustomerDetails();

model.addAttribute("customerDetails", customerDetails);

return "loanEligibility";

}

/\*\*

\* @param model

\* @param request

\* @param response

\* @param customerDetails

\* @param result

\* @return String

\* @throws ApplicationException

\*/

@RequestMapping(value = "/eligibilityCheck", method = RequestMethod.POST)

public String getLoanProducts(Model model, HttpServletRequest request,

HttpServletResponse response,

@Valid CustomerDetails customerDetails, BindingResult result) throws

ApplicationException {

if (!result.hasErrors()) {

List<LoanProduct> data = new ArrayList<LoanProduct>();

data = loanEligibilityService.checkEligibleLoanProducts(customerDetails);

if (data.isEmpty()) {

model.addAttribute("msg", "Sorry, no loan products matching your

profile.");

} else {

model.addAttribute("data", data);

model.addAttribute("msg", "Congratulations. You are Eligible for the

below Loan Products:");

}

}

return (result.hasErrors()?"loanEligibility":"results");

}

/\*\*

\* @return List<String>

\*/

@ModelAttribute("cityList")

public List<String> getCities() {

List<String> cityList = new ArrayList<String>();

cityList.add("");

cityList.add("Chennai");

cityList.add("Mumbai");

cityList.add("Bangalore");

cityList.add("Delhi");

// cityList.add("Pune");

cityList.add("Kolkatta");

// cityList.add("Delhi");

// cityList.add("Bangalore");

return cityList;

}

/\*\*

\* @return List<String>

\*/

@ModelAttribute("employeeList")

public List<String> getEmploymentTypes() {

List<String> employeeList = new ArrayList<String>();

employeeList.add("");

employeeList.add("Salaried");

employeeList.add("Self-Employed");

employeeList.add("Contractual Employment");

employeeList.add("Student");

employeeList.add("Pensioner");

return employeeList;

}

/\*\*

\* @return List<String>

\*/

@ModelAttribute("genderList")

public List<String> getGenderOptions() {

List<String> genderList = new ArrayList<String>();

genderList.add("Male");

genderList.add("Female");

return genderList;

}

}

---------------------customer details.java

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class CustomerDetails is the value object that must be binded to loanEligibility view

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.model;

import javax.validation.constraints.Size;

import javax.validation.constraints.Email;

import javax.validation.constraints.NotEmpty;

import javax.validation.constraints.NotNull;

public class CustomerDetails {

/// DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS

@NotNull(message = "must not be null")

@Size(min = 4, max = 30, message = "size must be between 4 and 30")

public String name;

@NotNull(message = "must not be null")

public String gender;

@NotEmpty(message = "must not be empty")

@Email(message = "must be a well-formed email address")

public String email;

@NotNull(message = "must not be null")

public Double monthlyIncome;

@NotEmpty(message = "must not be empty")

public String customerCity;

@NotEmpty(message = "must not be empty")

public String employmentType;

@NotNull(message = "must not be null")

public Double desiredLoanAmount;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public Double getMonthlyIncome() {

return monthlyIncome;

}

public void setMonthlyIncome(Double monthlyIncome) {

this.monthlyIncome = monthlyIncome;

}

public String getCustomerCity() {

return customerCity;

}

public void setCustomerCity(String customerCity) {

this.customerCity = customerCity;

}

public String getEmploymentType() {

return employmentType;

}

public void setEmploymentType(String employmentType) {

this.employmentType = employmentType;

}

public Double getDesiredLoanAmount() {

return desiredLoanAmount;

}

public void setDesiredLoanAmount(Double desiredLoanAmount) {

this.desiredLoanAmount = desiredLoanAmount;

}

}

------------------ Error response .java

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class ErrorResponse is the value object that must be binded to error view

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.model;

/\*\*

\*

\*/

public class ErrorResponse {

/// DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS

private String errorMessage;

private String requestedURI;

public String getErrorMessage() {

return errorMessage;

}

public void setErrorMessage(String errorMessage) {

this.errorMessage = errorMessage;

}

public String getRequestedURI() {

return requestedURI;

}

public void setRequestedURI(String requestedURI) {

this.requestedURI = requestedURI;

}

}

----------- Loan product .java

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class ErrorResponse is the value object that must be binded to results view

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.model;

public class LoanProduct {

///DO NOT CHANGE THE VARIABLE NAMES OR DATA TYPES OR ACCESS SPECIFIERS

public String bankName;

public String loanProductName;

public Double maxLoanAmount;

public Integer tenure;

public Double interest;

public Double monthlyInstallment;

/\*\*

\* @return the bankName

\*/

public String getBankName() {

return bankName;

}

/\*\*

\* @param bankName the bankName to set

\*/

public void setBankName(String bankName) {

this.bankName = bankName;

}

/\*\*

\* @return the loanProductName

\*/

public String getLoanProductName() {

return loanProductName;

}

/\*\*

\* @param loanProductName the loanProductName to set

\*/

public void setLoanProductName(String loanProductName) {

this.loanProductName = loanProductName;

}

/\*\*

\* @return the maxLoanAmount

\*/

public Double getMaxLoanAmount() {

return maxLoanAmount;

}

/\*\*

\* @param maxLoanAmount the maxLoanAmount to set

\*/

public void setMaxLoanAmount(Double maxLoanAmount) {

this.maxLoanAmount = maxLoanAmount;

}

/\*\*

\* @return the tenure

\*/

public Integer getTenure() {

return tenure;

}

/\*\*

\* @param tenure the tenure to set

\*/

public void setTenure(Integer tenure) {

this.tenure = tenure;

}

/\*\*

\* @return the interest

\*/

public Double getInterest() {

return interest;

}

/\*\*

\* @param interest the interest to set

\*/

public void setInterest(Double interest) {

this.interest = interest;

}

/\*\*

\* @return the monthlyInstallment

\*/

public Double getMonthlyInstallment() {

return monthlyInstallment;

}

/\*\*

\* @param monthlyInstallment the monthlyInstallment to set

\*/

public void setMonthlyInstallment(Double monthlyInstallment) {

this.monthlyInstallment = monthlyInstallment;

}

}

------------- service

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class LoanEligibilityService is to build the eligible loan products for the customer employment

type

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.service;

import java.util.ArrayList;

import java.util.List;

import org.springframework.stereotype.Service;

import com.cts.loanbazaar.loaneligibility.exception.ApplicationException;

import com.cts.loanbazaar.loaneligibility.model.CustomerDetails;

import com.cts.loanbazaar.loaneligibility.model.LoanProduct;

@Service

public class LoanEligibilityService {

/\*\*

\* @param details

\* @return List<LoanProduct>

\* @throws ApplicationException

\*/

public List<LoanProduct> checkEligibleLoanProducts(CustomerDetails details) throws

ApplicationException {

//TODO add your code here

if(details.getDesiredLoanAmount()> (12\*details.getMonthlyIncome())) {

throw new ApplicationException("Customer Not Eligible for the Loan");

}

List<LoanProduct> data = new ArrayList<LoanProduct>();

if(details.getEmploymentType().length()>=4

&&details.getEmploymentType().length()<=30) {

String employmentType = details.getEmploymentType();

List<LoanProduct> product = new ArrayList<LoanProduct>();

product = buildLoanProducts();

for(int i=0;i<product.size();i++) {

if(product.get(i).getLoanProductName().contains(employmentType)) {

data.add(product.get(i));

}

}

}

return data; //TODO CHANGE THIS RETURN TYPE

}

/\*\*

\* Use the below method to test the appropriate loan products based on employment type

of the customer

\* DO NOT CHANGE THE VALUES OF THE LOAN PRODUCT DETAILS

\* @return List<LoanProduct>

\*/

private List<LoanProduct> buildLoanProducts() {

List<LoanProduct> products = new ArrayList<LoanProduct>();

LoanProduct product1 = new LoanProduct();

product1.setBankName("MNQ Bank");

product1.setLoanProductName("Pensioner Pre-Approved Personal Loan");

product1.setMaxLoanAmount(500000.00);

product1.setMonthlyInstallment(7000.00);

product1.setTenure(24);

product1.setInterest(16.40);

products.add(product1);

LoanProduct product2 = new LoanProduct();

product2.setBankName("PMT Bank");

product2.setLoanProductName("Student Pre-Approved Education Loan");

product2.setMaxLoanAmount(1200000.00);

product2.setMonthlyInstallment(11000.00);

product2.setTenure(48);

product2.setInterest(12.40);

products.add(product2);

LoanProduct product3 = new LoanProduct();

product3.setBankName("MNQ Bank");

product3.setLoanProductName("Pre-Approved Personal Loan for Salaried");

product3.setMaxLoanAmount(1000000.00);

product3.setMonthlyInstallment(9000.00);

product3.setTenure(36);

product3.setInterest(15.40);

products.add(product3);

LoanProduct product4 = new LoanProduct();

product4.setBankName("MNQ Bank");

product4.setLoanProductName("Pre-Approved Personal Loan for Salaried");

product4.setMaxLoanAmount(700000.00);

product4.setMonthlyInstallment(8000.00);

product4.setTenure(24);

product4.setInterest(15.20);

products.add(product4);

LoanProduct product5 = new LoanProduct();

product5.setBankName("MNQ Bank");

product5.setLoanProductName("Pre-Approved Personal Loan for Self-Employed");

product5.setMaxLoanAmount(2500000.00);

product5.setMonthlyInstallment(34000.00);

product5.setTenure(720);

product5.setInterest(11.40);

products.add(product5);

return products;

}

}

----------------- ApplicationException

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class ApplicationException is a user defined exception for the proposed system

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\* ADD CONSTRUCTORS AS NEEDED

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.exception;

/\*\*

\*

\*/

public class ApplicationException extends Exception {

private static final long serialVersionUID = -9079454849611061074L;

public ApplicationException(String message) {

super(message);

}

}

-------------------- ExceptionHandlerControllerAdvice

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class ExceptionHandlerControllerAdvice is used to handle different exceptions raised by

Controller

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.exception;

import java.io.IOException;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.Date;

import javax.servlet.http.HttpServletRequest;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ControllerAdvice;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.bind.annotation.ResponseStatus;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.http.HttpStatus;

import com.cts.loanbazaar.loaneligibility.model.ErrorResponse;

@ControllerAdvice

public class ExceptionHandlerControllerAdvice {

@ExceptionHandler(ApplicationException.class)

@ResponseStatus(HttpStatus.INTERNAL\_SERVER\_ERROR)

@ResponseBody

public ModelAndView handleResourceNotFound(final ApplicationException exception, final

HttpServletRequest request,

final Model model) {

// TODO add your code here

int statusCode = 0;

ErrorResponse err = new ErrorResponse();

ModelAndView mav = new ModelAndView();

err.setErrorMessage(exception.getMessage());

err.setRequestedURI("http://localhost:8085/" + request.getRequestURI());

try {

URL url = new URL(err.getRequestedURI());

HttpURLConnection http = (HttpURLConnection) url.openConnection();

statusCode = http.getResponseCode();

mav.addObject("code", statusCode);

mav.addObject("curtime", new Date());

mav.addObject("message", err.getErrorMessage());

mav.setViewName("error");

} catch (IOException e) {

System.out.println(e);

}

return mav; // TODO change the return type here

}

}

-------------------- application

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This class LoanEligibilityServiceApplication is the starter class for Spring Boot

\*

\* DO NOT CHANGE THE CLASS NAME, PUBLIC METHODS, SIGNATURE OF METHODS, EXCEPTION

CLAUSES, RETURN TYPES

\* YOU CAN ADD ANY NUMBER OF PRIVATE METHODS TO MODULARIZE THE CODE

\* DO NOT SUBMIT THE CODE WITH COMPILATION ERRORS

\* CHANGE THE RETURN TYPE FROM NULL OF THE METHODS ONCE YOU BUILT THE LOGIC

\* DO NOT ADD ANY ADDITIONAL EXCEPTIONS IN THE THROWS CLAUSE OF THE METHOD. IF NEED

BE,

\* YOU CAN CATCH THEM AND THROW ONLY THE APPLICATION SPECIFIC EXCEPTION AS PER

EXCEPTION CLAUSE

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

package com.cts.loanbazaar.loaneligibility.main;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan(basePackages =

{"com.cts.loanbazaar.loaneligibility.controller","com.cts.loanbazaar.loaneligibility.model","com.cts.l

oanbazaar.loaneligibility.service","com.cts.loanbazaar.loaneligibility.exception"})

public class LoanEligibilityServiceApplication {

public static void main(String[] args) {

SpringApplication.run(LoanEligibilityServiceApplication.class, args);

}

}

------------ app.pro

#DO NOT CHANGE THE BELOW VALUES

server.port=8085

logging.level.org.springframework.web=DEBUG

#ADD YOUR CODE BELOW

spring.mvc.view.prefix = /WEB-INF/jsp/

spring.mvc.view.suffix = .jsp

**CAR**

package com.cts.carstore;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

import com.cts.carstore.skeletonvalidator.SkeletonValidator;

@SpringBootApplication

@ComponentScan("com.cts.\*")

public class CarStoreApplication {

public static void main(String[] args) {

SpringApplication.run(CarStoreApplication.class, args);

new SkeletonValidator();

}

}

package com.cts.carstore.controller;

import java.util.ArrayList;

import java.util.List;

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.validation.Validator;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import com.cts.carstore.exception.ApplicationException;

import com.cts.carstore.model.CarSearch;

import com.cts.carstore.service.CarStoreService;

@Controller

public class CarStoreController {

@Autowired

private CarStoreService service;

@Autowired

private Validator validator;

public CarStoreController() {

System.out.println("in default constr of controller");

}

@Autowired

public CarStoreController(CarStoreService service, Validator validator) {

this.service=service;

this.validator=validator;

}

@RequestMapping(value = "/", method = RequestMethod.GET)

public String showCarSearchForm(@ModelAttribute("carSearch") CarSearch carSearch) {

//Add code here..

return "carSearch"; //TODO, modify return value

}

@RequestMapping(value = "/getCarSearchResultPage", method = RequestMethod.POST)

public String getCarSearchResultForm(@ModelAttribute("carSearch") CarSearch carSearch,

ModelMap map,

BindingResult result)throws ApplicationException {

//Add code here..

// Redirecting to the same form page if invalid

if (result.hasErrors()) {

return "carSearch";

}

else {

return "carSearchResult"; //TODO, modify return value

}

}

@ModelAttribute("cityList")

public List<String> populateCities() {

List<String> cities = new ArrayList<String>();

cities.add("Chennai");

cities.add("Mumbai");

cities.add("Delhi");

cities.add("Bangalore");

cities.add("Pune");

cities.add("Kolkatta");

return cities;

}

@ModelAttribute("brandList")

public List<String> populateBrands() {

List<String> brands = new ArrayList<String>();

brands.add("Maruti Suzuki");

brands.add("Honda");

brands.add("Mahindra");

brands.add("Toyota");

brands.add("Hundai");

return brands;

}

@ModelAttribute("budgetList")

public List<String> populateBudget() {

List<String> budgetUpto = new ArrayList<String>();

budgetUpto.add("Below 3 Lakh");

budgetUpto.add("3 Lakh");

budgetUpto.add("5 Lakh");

budgetUpto.add("7 Lakh");

budgetUpto.add("10 Lakh");

return budgetUpto;

}

@ModelAttribute("fuelTypeList")

public List<String> populateFuelTypes() {

List<String> fuelTypes = new ArrayList<String>();

fuelTypes.add("Petrol");

fuelTypes.add("Diesel");

fuelTypes.add("CNG");

return fuelTypes;

}

}

package com.cts.carstore.exception;

public class ApplicationException extends Exception {

private static final long serialVersionUID = -9079454849611061074L;

public String errorMessage;

}

package com.cts.carstore.exception;

import java.util.Date;

import javax.servlet.http.HttpServletRequest;

import org.springframework.http.HttpStatus;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ControllerAdvice;

import org.springframework.web.bind.annotation.ResponseStatus;

import org.springframework.web.servlet.ModelAndView;

@ControllerAdvice

public class ExceptionHandlerControllerAdvice {

@ResponseStatus(value = HttpStatus.INTERNAL\_SERVER\_ERROR)

public ModelAndView handleResourceNotFound(final ApplicationException exception,

final HttpServletRequest request, final Model model) {

//Add code here..

String message="Low Budget- No car available below 3 lakh price";

Date date= new Date();

//add attribute error and timestamp

return null; //TODO, change return value

}

}

package com.cts.carstore.model;

public class Car {

private String brandName;

private String modelName;

private double price; //price in lakh

private String fuelType;

private double mileage;

private int seatingCapacity;

public Car() {

// TODO Auto-generated constructor stub

}

public Car(String brandName, String modelName, double price, String

fuelType,double mileage, int seatingCapacity) {

super();

this.brandName = brandName;

this.modelName = modelName;

this.price = price;

this.fuelType = fuelType;

this.mileage = mileage;

this.seatingCapacity = seatingCapacity;

}

public String getBrandName() {

return brandName;

}

public void setBrandName(String brandName) {

this.brandName = brandName;

}

public String getModelName() {

return modelName;

}

public void setModelName(String modelName) {

this.modelName = modelName;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public String getFuelType() {

return fuelType;

}

public void setFuelType(String fuelType) {

this.fuelType = fuelType;

}

public int getSeatingCapacity() {

return seatingCapacity;

}

public void setSeatingCapacity(int seatingCapacity) {

this.seatingCapacity = seatingCapacity;

}

public double getMileage() {

return mileage;

}

public void setMileage(double mileage) {

this.mileage = mileage;

}

}

package com.cts.carstore.model;

import javax.validation.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Pattern;

public class CarSearch {

// Use validation annotations as per the requirement

@NotNull(message = "Customer name is required")

private String customerName;

@NotBlank(message = "Gender is required")

private String gender;

@NotBlank(message = "Mobile number is required")

@Pattern(regexp="([7,8,9]{1}[0-9]{9})",message="Mobile number should be 10

digit starting with 7,8,9")

private String mobileNumber;

private String customerCity;

private String brand;

private String budgetUpto;

private String fuelType;

public CarSearch() {

// TODO Auto-generated constructor stub

}

public String getCustomerName() {

return customerName;

}

public void setCustomerName(String customerName) {

this.customerName = customerName;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getMobileNumber() {

return mobileNumber;

}

public void setMobileNumber(String mobileNumber) {

this.mobileNumber = mobileNumber;

}

public String getCustomerCity() {

return customerCity;

}

public void setCustomerCity(String customerCity) {

this.customerCity = customerCity;

}

public String getBrand() {

return brand;

}

public void setBrand(String brand) {

this.brand = brand;

}

public String getBudgetUpto() {

return budgetUpto;

}

public void setBudgetUpto(String budgetUpto) {

this.budgetUpto = budgetUpto;

}

public String getFuelType() {

return fuelType;

}

public void setFuelType(String fuelType) {

this.fuelType = fuelType;

}

}

package com.cts.carstore.model;

public class ErrorResponse {

private String errorMessage;

private String requestedURI;

//add code here

}

package com.cts.carstore.service;

import java.util.ArrayList;

import java.util.List;

import org.springframework.stereotype.Service;

import com.cts.carstore.exception.ApplicationException;

import com.cts.carstore.model.Car;

import com.cts.carstore.model.CarSearch;

@Service

public class CarStoreService {

public List<Car> getCarSearchResult(CarSearch carSearch) throws ApplicationException {

//Add code here..

List<Car> l=buildCars();

List<Car> r=new ArrayList<>();

for(Car i:l) {

if(i.getBrandName().equals(carSearch.getBrand()) &&

i.getFuelType().equals(carSearch.getFuelType()) &&

i.getPrice()<=Double.parseDouble(carSearch.getBudgetUpto().substring(0, 1))) {

r.add(i);

}

}

return r; //TODO, modify this return value

}

// DO NOT CHANGE THIS METHOD

//DO NOT CHANGE CODE WITHIN METHOD

private List<Car> buildCars() {

List<Car> cars = new ArrayList<Car>();

// brand,modelName,price,fuelType,seatingCapacity

Car car1 = new Car("Maruti Suzuki", "Swift", 5.20, "Petrol", 21.21, 5);

Car car2 = new Car("Maruti Suzuki", "Alto", 3, "Petrol", 22.05, 5);

Car car3 = new Car("Maruti Suzuki", "Ertiga LXi", 7.5, "Petrol", 19.01, 7);

Car car4 = new Car("Maruti Suzuki", "Ertiga VXi", 8.9, "CNG", 26.2, 7);

Car car5 = new Car("Maruti Suzuki", "WagonR", 4.5, "Petrol", 20.52, 5);

Car car6 = new Car("Maruti Suzuki", "WagonR LXi", 5.20, "CNG", 32.52, 5);

Car car7 = new Car("Honda", "Amaze", 6.20, "Petrol", 18.6, 5);

Car car8 = new Car("Honda", "Amaze1.2E MT", 7.6, "Diesel", 24.7, 5);

Car car9 = new Car("Honda", "Amaze1.5E MT", 8, "Diesel", 25.2, 5);

Car car10 = new Car("Honda", "City", 9.30, "Petrol", 17.4, 5);

Car car11 = new Car("Mahindra", "KUV100", 5.6, "Petrol", 15, 6);

Car car12 = new Car("Toyota", "Yaris", 8.8, "Petrol", 18.1, 5);

Car car13 = new Car("Hundai", "Aura", 7.30, "CNG", 20.5, 5);

Car car14 = new Car("Hundai", "Creta", 9.9, "Petrol", 21.5, 5);

cars.add(car1);

cars.add(car2);

cars.add(car3);

cars.add(car4);

cars.add(car5);

cars.add(car6);

cars.add(car7);

cars.add(car8);

cars.add(car9);

cars.add(car10);

cars.add(car11);

cars.add(car12);

cars.add(car13);

cars.add(car14);

return cars;

}

}

package com.cts.carstore.skeletonvalidator;

import java.lang.reflect.Method;

import java.util.logging.Level;

import java.util.logging.Logger;

public class SkeletonValidator {

public SkeletonValidator() {

validateClassName("com.cts.carstore.model.Car");

validateClassName("com.cts.carstore.model.CarSearch");

validateClassName("com.cts.carstore.model.ErrorResponse");

validateClassName("com.cts.carstore.service.CarStoreService");

validateClassName("com.cts.carstore.controller.CarStoreController");

validateClassName("com.cts.carstore.exception.ApplicationException");

validateClassName("com.cts.carstore.exception.ExceptionHandlerControllerAdvice");

validateMethodSignature("getCarSearchResult:java.util.List",

"com.cts.carstore.service.CarStoreService");

validateMethodSignature("showCarSearchForm:java.lang.String,populateCities:java.util.List,

populateBrands:java.util.List,populateBudget:java.util.List,populateFuelTypes:java.util.List",

"com.cts.carstore.controller.CarStoreController");

}

private static final Logger LOG = Logger.getLogger("SkeletonValidator");

protected final boolean validateClassName(String className) {

boolean iscorrect = false;

try {

Class.forName(className);

iscorrect = true;

LOG.info("Class Name " + className + " is correct");

} catch (ClassNotFoundException e) {

LOG.log(Level.SEVERE, "You have changed either the " + "class

name/package. Use the correct package "

+ "and class name as provided in the skeleton");

} catch (Exception e) {

LOG.log(Level.SEVERE,

"There is an error in validating the " + "Class Name. Please

manually verify that the "

+ "Class name is same as skeleton before

uploading");

}

return iscorrect;

}

protected final void validateMethodSignature(String methodWithExcptn, String className) {

Class cls = null;

try {

String[] actualmethods = methodWithExcptn.split(",");

boolean errorFlag = false;

String[] methodSignature;

String methodName = null;

String returnType = null;

for (String singleMethod : actualmethods) {

boolean foundMethod = false;

methodSignature = singleMethod.split(":");

methodName = methodSignature[0];

returnType = methodSignature[1];

cls = Class.forName(className);

Method[] methods = cls.getMethods();

for (Method findMethod : methods) {

if (methodName.equals(findMethod.getName())) {

foundMethod = true;

if

(!(findMethod.getReturnType().getName().equals(returnType))) {

errorFlag = true;

LOG.log(Level.SEVERE, " You have changed

the " + "return type in '" + methodName

+ "' method. Please stick to

the " + "skeleton provided");

} else {

LOG.info("Method signature of " +

methodName + " is valid");

}

}

}

if (!foundMethod) {

errorFlag = true;

LOG.log(Level.SEVERE, " Unable to find the given public

method " + methodName

+ ". Do not change the " + "given public

method name. " + "Verify it with the skeleton");

}

}

if (!errorFlag) {

LOG.info("Method signature is valid");

}

} catch (Exception e) {

LOG.log(Level.SEVERE,

" There is an error in validating the " + "method structure.

Please manually verify that the "

+ "Method signature is same as the skeleton

before uploading");

}

}

}

server.port=3030

spring.mvc.view.prefix=/WEB-INF/views/

spring.mvc.view.suffix=.jsp

spring.mvc.static-class-path=/resources/\*\*

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<br>

<br>

<!-- Add code here.. -->

<h1 id="heading">Windsor Car Showroom</h1>

<table>

<center></center><caption style="font-weight: bolder;">Search

Cars</caption></center>

<sf:form action="getCarSearchResultPage" modelAttribute="carSearch"

name="form">

<tr>

<td>Customer Name:</td>

<td><sf:input path="customerName" id="customerName"

name="customerName"/></td>

<td><sf:errors path="customerName" /></td>

</tr>

<tr>

<td>Mobile Number:</td>

<td><sf:input path="mobileNumber" id="mobileNumber"

/></td>

<td><sf:errors path="mobileNumber" /></td>

</tr>

<tr>

<td>Gender:</td>

<td>

<sf:radiobutton path="gender" value="male"/>

<label class='radiolabel'>Male</label>

<sf:radiobutton path="gender" value="female" />

<label class='radiolabel'>Female</label> </td>

</tr>

<tr>

<tr>

<td>Customer City:</td>

<td><sf:select path="gender" id="gender"

items="${cityList}" /></td>

</tr>

<tr>

<tr>

<td>Brand:</td>

<td><sf:select path="gender" id="gender"

items="${brandList}" /></td>

</tr>

<tr>

<tr>

<td>Fuel/Transmission:</td>

<td><sf:select path="gender" id="gender"

items="${fuelTypeList}" /></td>

</tr>

<tr>

<tr>

<tr>

<td>Budget Upto:</td>

<td><sf:select path="gender" id="gender"

items="${budgetList}" /></td>

</tr>

<td><input type="submit" value="CarSearch"

name="CarSearch" /></td>

<td><input type="reset" value="Clear" /></td>

</tr>

</sf:form>

</table>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags"%>

<%@ taglib prefix="sf" uri="http://www.springframework.org/tags/form"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<br>

<br>

<!--Add code here -->

<br>

<br>

<h3>Here are cars matching your search criteria:</h3>

<table border="1">

<tr><td>Brand Name</td><td>Model Name</td><td>Price in

lakh</td><td>Fuel/Transmission</td><td>Mileage</td><td>Seating Capacity</td></tr>

<c:forEach var="i" items="${list}">

<tr><td><c:out value="${i.getBrandName()}"/></td>

<td><c:out value="${i.getModelName()}"/></td>

<td><c:out value="${i.getPrice()}"/></td>

<td><c:out value="${i.getFuelType()}"/></td>

<td><c:out value="${i.getMileage()}"/></td>

<td><c:out value="${i.getSeatingCapacity()}"/></td>

</tr>

</c:forEach>

</table>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1" isELIgnored="false" isErrorPage="true"%>

<%@ 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"%>

<!DOCTYPE HTML>

<html>

<head>

<title>Car Search Errors</title>

</head>

<body>

<!--Add code here.. -->

<form:form id="error">

<h3>Unable to retrieve loan information. Below are the error details:</h3>

<h3>Response Code: ${code }</h3>

<h3>Error Message: ${message }</h3>

<h3>Error Occurred on: ${curtime }</h3>

</form:form>

</body>

</html>

**TMS**

package com;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan("com.\*")

public class TmsApplication {

/\*\*

\* Starting point of the application

\*

\* @param args Arguments passed to the application

\*/

public static void main(String[] args) {

SpringApplication.run(TmsApplication.class, args);

}

}

package com.controller;

import java.util.Locale;

import org.springframework.context.MessageSource;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.support.ReloadableResourceBundleMessageSource;

import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;

import org.springframework.web.servlet.LocaleResolver;

import org.springframework.web.servlet.config.annotation.InterceptorRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

import org.springframework.web.servlet.i18n.LocaleChangeInterceptor;

import org.springframework.web.servlet.i18n.SessionLocaleResolver;

@Configuration

public class InternationalizationConfig extends WebMvcConfigurerAdapter {

/\*\*

\* Set default Locale

\*

\* @return A bean of LocalResolver

\*/

@Bean

public LocaleResolver localeResolver() {

SessionLocaleResolver slr = new SessionLocaleResolver();

slr.setDefaultLocale(Locale.US);

return slr;

}

/\*\*

\* Set path variable name for changing language

\*

\* @return A bean of LocaleChangeInterceptor

\*/

@Bean

public LocaleChangeInterceptor localeChangeInterceptor() {

LocaleChangeInterceptor lci = new LocaleChangeInterceptor();

lci.setParamName("language");

return lci;

}

/\*\*

\* Add interceptor into the registry

\*/

@Override

public void addInterceptors(InterceptorRegistry registry) {

registry.addInterceptor(localeChangeInterceptor());

}

/\*\*

\* Set base name for messages.properties files Set default encoding to UTF-8

\*

\* @return A bean of MessageSource

\*/

@Bean

public MessageSource messageSource() {

ReloadableResourceBundleMessageSource rrbms = new

ReloadableResourceBundleMessageSource();

rrbms.setBasename("classpath:messages");

rrbms.setDefaultEncoding("UTF-8");

return rrbms;

}

/\*\*

\* Set validation message source

\*

\* @return A bean of LocalValidatorFactoryBean

\*/

@Bean

public LocalValidatorFactoryBean localValidatorFactoryBean() {

LocalValidatorFactoryBean lvfb = new LocalValidatorFactoryBean();

lvfb.setValidationMessageSource(messageSource());

return lvfb;

}

}

package com.controller;

import java.util.Arrays;

import java.util.List;

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.UserClaim;

import com.service.TaxService;

@Controller

public class TaxController {

@Autowired

public TaxService taxService;

/\*\*

\* Display taxclaim.jsp page when a get request is pushed on url

\* /getTaxClaimFormPage

\*

\* @param userClaim Is the UserClaim component

\* @return taxclaim as a jsp page

\* @see UserClaim

\*/

@RequestMapping(value = "/getTaxClaimFormPage", method = RequestMethod.GET)

public String claimPage(@ModelAttribute("userClaim") UserClaim userClaim) {

return "taxclaim";

}

/\*\*

\* Return result.jsp age when validation is successful Otherwise return back to

\* taxclaim page with error message

\*

\* @param userClaim UserClaim component

\* @param result BindingResult which validate the user input

\* @param map ModelMap to put attribute which will be forwarded to next

\* page

\* @return "result.jsp" page if the validation is successful otherwise

\* "taxclaim.jsp" with error included

\*/

@RequestMapping(value = "/calculateTax", method = RequestMethod.GET)

public String calculateTax(@Valid @ModelAttribute("userClaim") UserClaim userClaim,

BindingResult result,

ModelMap map) {

if (result.hasErrors()) {

return "taxclaim";

}

double amount = taxService.calculateTax(userClaim);

map.addAttribute("amount", amount);

return "result";

}

/\*\*

\* Populate <form:select /> tag in the taxclaim.jsp page

\*

\* @return List of expenses

\*/

@ModelAttribute("expenseList")

public List<String> populateExpense() {

return Arrays.asList("MedicalExpense", "TravelExpense", "FoodExpense");

}

}

package com.model;

import javax.validation.constraints.NotBlank;

import javax.validation.constraints.PositiveOrZero;

import javax.validation.constraints.Size;

import org.springframework.stereotype.Component;

@Component

public class UserClaim {

private String expenseType;

@PositiveOrZero(message = "{error.expenseAmount.negative}")

private double expenseAmt;

@NotBlank(message = "{error.employeeId}")

@Size(min = 5, message = "{error.employeeId.size}")

private String employeeId;

public String getExpenseType() {

return expenseType;

}

public void setExpenseType(String expenseType) {

this.expenseType = expenseType;

}

public double getExpenseAmt() {

return expenseAmt;

}

public void setExpenseAmt(double expenseAmt) {

this.expenseAmt = expenseAmt;

}

public String getEmployeeId() {

return employeeId;

}

public void setEmployeeId(String employeeId) {

this.employeeId = employeeId;

}

}

package com.service;

import org.springframework.stereotype.Service;

import com.model.UserClaim;

@Service

public interface TaxService {

/\*\*

\* Calculate Tax

\*

\* @param userClaim UserClaim bean

\* @return Calculated tax

\*/

public double calculateTax(UserClaim userClaim);

}

package com.service;

import org.springframework.stereotype.Service;

import com.model.UserClaim;

@Service

public class TaxServiceImpl implements TaxService {

/\*\*

\* Calculate the tax according to the srs

\*

\* @param userClaim UserClaim component to get the values

\* @return Calculated tax

\*/

@Override

public double calculateTax(UserClaim userClaim) {

String e = userClaim.getExpenseType();

double a = userClaim.getExpenseAmt();

double t = 0.0;

if (e.startsWith("M")) {

if (a <= 1000) {

t = 15.0;

} else if (a > 1000 && a <= 10000) {

t = 20.0;

} else if (a > 10000) {

t = 25.0;

}

} else if (e.startsWith("T")) {

if (a <= 1000) {

t = 10.0;

} else if (a > 1000 && a <= 10000) {

t = 15.0;

} else if (a > 10000) {

t = 20.0;

}

} else if (e.startsWith("F")) {

if (a <= 1000) {

t = 5.0;

} else if (a > 1000 && a <= 10000) {

t = 10.0;

} else if (a > 10000) {

t = 15.0;

}

}

return a \* (t / 100.0);

}

}

server.port=9095

spring.mvc.view.prefix=/WEB-INF/jsp/

spring.mvc.view.suffix=.jsp

spring.mvc.static-class-path=/resources/\*\*

---------------de.pro

label.employeeId=Employee ID in German

label.expenseType=Expense Type in German

label.expenseAmount=Expense Amount in German

error.employeeId=Employee ID cannot be empty in German

error.employeeId.size=Employee ID should be at least 5 characters in German

error.expenseAmount=Expense Amount cannot be empty in German

error.expenseAmount.numeric=Expense amount should be numeric only in German

error.expenseAmount.negative=Expense amount should not be a negative number in

German

------ fr.pro

label.employeeId=Employee ID in French

label.expenseType=Expense Type in French

label.expenseAmount=Expense Amount in French

error.employeeId=Employee ID cannot be empty in French

error.employeeId.size=Employee ID should be at least 5 characters in French

error.expenseAmount=Expense Amount cannot be empty in French

error.expenseAmount.numeric=Expense amount should be numeric only in French

error.expenseAmount.negative=Expense amount should not be a negative number in

French

------------- msg.pro

label.employeeId=Employee ID in English

label.expenseType=Expense Type in English

label.expenseAmount=Expense Amount in English

error.employeeId=Employee ID cannot be empty in English

error.employeeId.size=Employee ID should be at least 5 characters in English

error.expenseAmount=Expense Amount cannot be empty in English

error.expenseAmount.numeric=Expense amount should be numeric only in English

error.expenseAmount.negative=Expense amount should not be a negative number in

English

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<h2>The tax claim for ${ userClaim.expenseType } with expense amount

${ userClaim.expenseAmt } is ${ amount }</h2>

</body>

</html>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1" isELIgnored="false"%>

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags"%>

<%@ taglib prefix="form" 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"%>

<!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>Tax: Tax Claim</center>

</h1>

<a href="/getTaxClaimFormPage?language=en">English</a>|

<a href="/getTaxClaimFormPage?language=de">German</a>|

<a href="/getTaxClaimFormPage?language=fr">French</a>

</align>

<form:form action="/calculateTax" method="get" modelAttribute="userClaim">

<table>

<tr>

<td id="id1">

<spring:message code="label.employeeId" />

</td>

<td id="id2">

<form:input path="employeeId" id="employeeId" />

</td>

<td id="id3">

<form:errors path="employeeId" />

</td>

</tr>

<tr>

<td id="id4">

<spring:message code="label.expenseType" />

</td>

<td id="id5">

<form:select path="expenseType" items="${

expenseList }" id="expenseType" />

</td>

<td id="id6"></td>

</tr>

<tr>

<td id="id7">

<spring:message code="label.expenseAmount" />

</td>

<td id="id8">

<form:input path="expenseAmt" id="expenseAmount"

/>

</td>

<td id=id9>

<form:errors path="expenseAmt" />

</td>

</tr>

<tr>

<td><input type="Submit" name="submit" value="Calculate

Claim" /></td>

<td></td>

</tr>

<tr>

<td><input type="reset" name="reset" value="Clear"

/></td>

<td></td>

</tr>

</table>

</form:form>

</body>

</html>

**PINK FITNESS**

**FitnessBO:**

package com.spring.bo;

import com.spring.model.FitnessFranchise;

public class FitnessBO {

public double calculateNetProfit(FitnessFranchise franchise) {

//// Calculating netProfit based on factors

double expense = franchise.getTotalIncome() - (franchise.getTrainerSalary()

+ franchise.getCommonExpenses().get("Rent") + franchise.getCommonExpenses().get("HouseKeeping")

+ franchise.getCommonExpenses().get("ElectricityCharges")

+ franchise.getCommonExpenses().get("Maintenence"));

return expense;

}

}

**No ProfitException**

package com.spring.exception;

public class NoProfitException extends Exception {

public NoProfitException(String msg) {

//fill the code

super(msg);

}

}

**Driver:**

package com.spring.main;

import java.util.\*;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.spring.exception.NoProfitException;

import com.spring.model.FitnessFranchise;

import com.spring.service.FitnessService;

public class Driver {

public static void main(String[] args) {

// TODO Auto-generated method stub

// fill the code

Scanner sc = new Scanner(System.in);

ApplicationContext applicationContext = new ClassPathXmlApplicationContext("beans.xml");

System.out.println("Pink Fitness Franchise 1 details");

System.out.println("Enter the total location:");

String location = sc.next();

System.out.println("Enter the total income:");

double totalIncome = sc.nextDouble();

System.out.println("Enter total expense amount for trainer salary:");

double trainerSalary = sc.nextDouble();

FitnessFranchise fitnessFranchise1 = (FitnessFranchise) applicationContext.getBean("fitnessFranchise");

fitnessFranchise1.setLocation(location);

fitnessFranchise1.setTotalIncome(totalIncome);

fitnessFranchise1.setTrainerSalary(trainerSalary);

System.out.println("Pink Fitness Franchise 2 details");

System.out.println("Enter the total location:");

String location2 = sc.next();

System.out.println("Enter the total income:");

double totalIncome2 = sc.nextDouble();

System.out.println("Enter total expense amount for trainer salary:");

double trainerSalary2 = sc.nextDouble();

FitnessFranchise fitnessFranchise2 = (FitnessFranchise) applicationContext.getBean("fitnessFranchise");

fitnessFranchise2.setLocation(location2);

fitnessFranchise2.setTotalIncome(totalIncome2);

fitnessFranchise2.setTrainerSalary(trainerSalary2);

FitnessService fitnessService = (FitnessService) applicationContext.getBean("fitnessService");

try {

fitnessService.calculateNetProfit(fitnessFranchise1);

System.out.println("Pink Fitness at " + fitnessFranchise1.getLocation() + " franchise Amount is RS:"

+ fitnessFranchise1.getFranchiseAmount());

} catch (NoProfitException e) {

System.out.println(e.getMessage());

}

try {

fitnessService.calculateNetProfit(fitnessFranchise2);

System.out.println("Pink Fitness at " + fitnessFranchise2.getLocation() + " franchise Amount is RS:"

+ fitnessFranchise2.getFranchiseAmount());

} catch (NoProfitException e) {

System.out.println(e.getMessage());

}

}

}

**FitnessFranchise:**

package com.spring.model;

import java.util.Map;

// implement the FitnessFranchisor interface

public class FitnessFranchise implements FitnessFranchisor {

private String location;

private double totalIncome;

private double trainerSalary;

private double franchiseAmount;

private double percentage;

private Map<String, Double> commonExpenses;

public FitnessFranchise(double percentage, Map<String, Double> commonExpenses) {

super();

this.percentage = percentage;

this.commonExpenses = commonExpenses;

}

public void setPercentage(double percentage) {

this.percentage = percentage;

}

public double getPercentage() {

return percentage;

}

public double getFranchiseAmount() {

return franchiseAmount;

}

public Map<String, Double> getCommonExpenses() {

return commonExpenses;

}

public void setCommonExpenses(Map<String, Double> commonExpenses) {

this.commonExpenses = commonExpenses;

}

public void setFranchiseAmount(double franchiseAmount) {

this.franchiseAmount = franchiseAmount;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

public double getTotalIncome() {

return totalIncome;

}

public void setTotalIncome(double totalIncome) {

this.totalIncome = totalIncome;

}

public double getTrainerSalary() {

return trainerSalary;

}

public void setTrainerSalary(double trainerSalary) {

this.trainerSalary = trainerSalary;

}

public void calculateFranchiseAmount(double netProfit) {

//Calculating Franchise Amount

this.franchiseAmount=netProfit\*(percentage)/100;

}

}

**FitnessFranchiser:**

package com.spring.model;

public interface FitnessFranchisor {

public void calculateFranchiseAmount(double netProfit);

}

**FitnessService:**

package com.spring.service;

import com.spring.bo.FitnessBO;

import com.spring.exception.NoProfitException;

import com.spring.model.FitnessFranchise;

public class FitnessService {

private FitnessBO fitnessBO;

public FitnessBO getFitnessBO() {

return fitnessBO;

}

public void setFitnessBO(FitnessBO fitnessBO) {

this.fitnessBO = fitnessBO;

}

public void calculateNetProfit(FitnessFranchise franchise) throws NoProfitException {

//fill the code

double netProfit=fitnessBO.calculateNetProfit(franchise);

if(netProfit<0)

{

throw new NoProfitException("No profit to calculate franchise amount");

}

else

{

franchise.calculateFranchiseAmount(netProfit);

}

}

public FitnessService(FitnessBO fitnessBO) {

super();

this.fitnessBO = fitnessBO;

}

}

**Beans.xml:**

<?xml version="1.0"?>

[<beans xsi:schemaLocation="**http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.2.xsd http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.2.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.2.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-3.2.xsd**" xmlns:task="**http://www.springframework.org/schema/task**" xmlns:tx="**http://www.springframework.org/schema/tx**" xmlns:jee="**http://www.springframework.org/schema/jee**" xmlns:context="**http://www.springframework.org/schema/context**" xmlns:aop="**http://www.springframework.org/schema/aop**" xmlns:p="**http://www.springframework.org/schema/p**" xmlns:xsi="**http://www.w3.org/2001/XMLSchema-instance**" xmlns="**http://www.springframework.org/schema/beans**">](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\src\main\resources\beans.xml)<context:property-placeholder location="**classpath:FranchisePercent.properties**"/>

<!-- fill the code -->

[<bean scope="**prototype**" class="**com.spring.model.FitnessFranchise**" name="**fitnessFranchise**">](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\src\main\resources\beans.xml)<constructor-arg name="**percentage**" value="**${percentage}**"/>[<constructor-arg name="**commonExpenses**"><map>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\src\main\resources\beans.xml)<entry value="**12000.0**" key="**Rent**"/><entry value="**4000.0**" key="**HouseKeeping**"/><entry value="**2000.0**" key="**ElectricityCharges**"/><entry value="**3000.0**" key="**Maintenence**"/></map></constructor-arg></bean>[<bean scope="**prototype**" class="**com.spring.service.FitnessService**" name="**fitnessService**">](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\src\main\resources\beans.xml)<constructor-arg ref="**fitnessBO**"/></bean><bean scope="**prototype**" class="**com.spring.bo.FitnessBO**" name="**fitnessBO**"/></beans>

**POM.xml**

<?xml version="1.0"?>

[<project xsi:schemaLocation="**http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd**" xmlns:xsi="**http://www.w3.org/2001/XMLSchema-instance**" xmlns="**http://maven.apache.org/POM/4.0.0**">](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<modelVersion>4.0.0</modelVersion><groupId>PinkFitness</groupId><artifactId>PinkFitness</artifactId><version>0.0.1-SNAPSHOT</version>[<dependencies><dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>org.springframework</groupId><artifactId>spring-context</artifactId><version>4.3.10.RELEASE</version></dependency>[<dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>junit</groupId><artifactId>junit</artifactId><version>4.12</version><scope>test</scope></dependency>[<dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>xmlunit</groupId><artifactId>xmlunit</artifactId><version>1.6</version><scope>test</scope></dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

[<dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>com.github.stefanbirkner</groupId><artifactId>system-rules</artifactId><version>1.16.0</version><scope>test</scope></dependency>[<dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>com.sun.xml.bind</groupId><artifactId>jaxb-impl</artifactId><version>2.1.13</version></dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

[<dependency>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>javax.xml.bind</groupId><artifactId>jaxb-api</artifactId><version>2.1</version></dependency></dependencies>[<build><plugins><plugin>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<groupId>org.apache.maven.plugins</groupId><artifactId>maven-surefire-plugin</artifactId><version>3.0.0-M1</version>[<configuration>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<testFailureIgnore>true</testFailureIgnore></configuration></plugin></plugins></build>[<properties>](file:///C:\Users\dell\Downloads\5_6161222816239715837\PinkFitness\pom.xml)<maven.compiler.source>1.8</maven.compiler.source><maven.compiler.target>1.8</maven.compiler.target></properties></project>

**ADVERTISEMENT BOOKING**

**CostPerSecond.properties**

costPerSecond=200

**AdvertisementBO.java**

package com.spring.bo;

import org.springframework.context.annotation.ComponentScan;

import com.spring.model.Advertisement;

//use appropriate annotation to make this class as component class

@ComponentScan("com.spring")

public class AdvertisementBO {

public double bookAdvertisement(Advertisement advertisement) {

double result=0.0;

// fill the code

return result;

}

}

**ApplicationConfig.java**

package com.spring.config;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

// Use appropriate annotation

@Configuration

@ComponentScan("com.spring")

public class ApplicationConfig {

}

**InvalidDurationException.java**

package com.spring.exception;

public class InvalidDurationException extends Exception {

public InvalidDurationException(String msg){

// fill the code

System.out.println("Invalid Duration");

}

}

**Driver.java**

package com.spring.main;

import java.util.Scanner;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.spring.exception.InvalidDurationException;

import com.spring.service.AdvertisementService;

public class Driver {

public static void main(String[] args){

// fill the code

ApplicationContext ctx = new ClassPathXmlApplicationContext("ApplicationConfig.java");

AdvertisementService advertisementService = ctx.getBean(AdvertisementService.class);

Scanner sc = new Scanner("System.in");

System.out.println("Enter the Advertisement Id:");

int advertisementId= sc.nextInt();

System.out.println("Enter the Advertisement Type:");

String advertisementType= sc.next();

System.out.println("Enter the Advertisement Duration(in seconds):");

int duration = sc.nextInt();

System.out.println("Enter number of Days/Week to telecast:");

int daysPerWeek = sc.nextInt();

System.out.println("Enter preferred time slot");

String timeSlot = sc.next();

System.out.println("Enter the television name:");

String channelName = sc.next();

try {

System.out.println("Total cost to telecast the advertisement is RS:" +advertisementService.bookAdvertisement(advertisementId, advertisementType, duration, daysPerWeek, timeSlot, channelName));

} catch (InvalidDurationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**Advertisement.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class Advertisement {

private int advertisementId;

private String advertisementType;

private int duration;

private int daysPerWeek;

private String timeSlot;

private TelevisionChannel televisionChannel;

// fill the code

@Autowired

public Advertisement(TelevisionChannel televisionChannel) {

super();

this.televisionChannel = televisionChannel;

}

public int getAdvertisementId() {

return advertisementId;

}

public void setAdvertisementId(int advertisementId) {

this.advertisementId = advertisementId;

}

public String getTimeSlot() {

return timeSlot;

}

public void setTimeSlot(String timeSlot) {

this.timeSlot = timeSlot;

}

public String getAdvertisementType() {

return advertisementType;

}

public void setAdvertisementType(String advertisementType) {

this.advertisementType = advertisementType;

}

public int getDuration() {

return duration;

}

public void setDuration(int duration) {

this.duration = duration;

}

public int getDaysPerWeek() {

return daysPerWeek;

}

public void setDaysPerWeek(int daysPerWeek) {

this.daysPerWeek = daysPerWeek;

}

public TelevisionChannel getTelevisionChannel() {

return televisionChannel;

}

public void setTelevisionChannel(TelevisionChannel televisionChannel) {

this.televisionChannel = televisionChannel;

}

}

**TelevisionChannel.java**

package com.spring.model;

import java.util.Map;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component()

public class TelevisionChannel {

private String channelName;

public String getChannelName() {

return channelName;

}

public void setChannelName(String channelName) {

this.channelName = channelName;

}

// fill the code

@Value("${costPerSecond.properties}")

private double costPerSecond;

// fill the code

@Value("#{${maps: {6pm-7pm : '250' , 7pm-8pm :'500' , 8pm-9pm :'750' , 9pm-10pm :'1000'}}}")

private Map<String,Double> primeRate;

public double getCostPerSecond() {

return costPerSecond;

}

public void setCostPerSecond(double costPerSecond) {

this.costPerSecond = costPerSecond;

}

public Map<String, Double> getPrimeRate() {

return primeRate;

}

public void setPrimeRate(Map<String, Double> primeRate) {

this.primeRate = primeRate;

}

}

**AdvertisementService.java**

package com.spring.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.ComponentScan;

import com.spring.bo.AdvertisementBO;

import com.spring.exception.InvalidDurationException;

//Use appropriate annotation

@ComponentScan("com.spring")

public class AdvertisementService {

private AdvertisementBO advertisementBO;

//Use appropriate annotation

@Autowired

public AdvertisementService(AdvertisementBO advertisementBO) {

super();

this.advertisementBO = advertisementBO;

}

public AdvertisementBO getAdvertisementBO() {

return advertisementBO;

}

public void setAdvertisementBO(AdvertisementBO advertisementBO) {

this.advertisementBO = advertisementBO;

}

public double bookAdvertisement(int advertisementId, String advertisementType, int duration, int daysPerWeek, String timeSlot,String channelName) throws InvalidDurationException {

//double result=0.0;

//fill the code

if(duration<5 || duration>90) {

throw new InvalidDurationException("Invalid Duration");

} else {

double TotalCost = (duration \* 200 \* daysPerWeek) ;

return TotalCost;

}

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>AdvertisementBooking</groupId>

<artifactId>AdvertisementBooking</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>AdvertisementBooking</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>

**FURNTURN OUTLET**

**ProductBO.java**

package com.spring.bo;

import java.util.Map;

import org.springframework.stereotype.Component;

import com.spring.model.Product;

@Component

public class ProductBO {

public double calculateBill(Product furObj,String woodType) {

double amount=0;

Map<String,Integer> map= furObj.getShopDetails().getDiscountDetails();

if(map.containsKey(woodType)) {

int d=map.get(woodType);

amount= furObj.getMrpValue()-(furObj.getMrpValue()\*d)/100.0;

}

return amount;

}

}

**ApplicationConfig.java**

package com.spring.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.stereotype.Component;

import org.springframework.context.annotation.ComponentScan;

import com.spring.bo.ProductBO;

import com.spring.model.Product;

import com.spring.model.Shop;

import com.spring.service.ProductService;

@Configuration

@Component

@ComponentScan

public class ApplicationConfig {

@Bean

public Shop shop() {

return new Shop();

}

@Bean

public Product product() {

return new Product(shop());

}

@Bean

public ProductBO productBO() {

return new ProductBO();

}

@Bean

public ProductService productService() {

return new ProductService(productBO());

}

}

**InvalidProductIdException.java**

package com.spring.exception;

import org.springframework.stereotype.Component;

@Component

public class InvalidProductIdException extends Exception {

public InvalidProductIdException(String msg) {

// fill the code

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.Scanner;

import org.springframework.stereotype.Component;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidProductIdException;

import com.spring.model.Product;

import com.spring.service.ProductService;

@Component

public class Driver {

public static void main(String[] args) {

// fill the code

Scanner sc=new Scanner(System.in);

System.out.println("Enter the product id:");

String productId= sc.nextLine();

System.out.println("Enter the product name:");

String productName=sc.nextLine();

System.out.println("Enter the mrp value:");

double mrp= sc.nextDouble();

sc.nextLine();

System.out.println("Enter the dimension details:");

String dimension= sc.nextLine();

System.out.println("Enter the wood type:");

String woodType= sc.nextLine();

ConfigurableApplicationContext context=new AnnotationConfigApplicationContext(ApplicationConfig.class);

ProductService product=(ProductService)context.getBean(ProductService.class);

try {

double d =product.calculateBill(productId, productName, mrp, dimension, woodType);

System.out.println("Amount to be paid is :"+ d);

} catch (InvalidProductIdException e) {

System.out.println(e.getMessage());

}

}

}

**Product.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class Product {

private String productId;

private String productName;

private double mrpValue;

private Shop shopDetails;

private String dimension;

@Autowired

public Product(Shop shopDetails) {

super();

this.shopDetails = shopDetails;

}

public String getDimension() {

return dimension;

}

public void setDimension(String dimension) {

this.dimension = dimension;

}

public String getProductId() {

return productId;

}

public void setProductId(String productId) {

this.productId = productId;

}

public String getProductName() {

return productName;

}

public void setProductName(String productName) {

this.productName = productName;

}

public double getMrpValue() {

return mrpValue;

}

public void setMrpValue(double mrpValue) {

this.mrpValue = mrpValue;

}

public Shop getShopDetails() {

return shopDetails;

}

public void setShopDetails(Shop shopDetails) {

this.shopDetails = shopDetails;

}

}

**Shop.java**

package com.spring.model;

import java.util.Map;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

@Component

public class Shop {

@Value("FurnTurn Bang")

private String shopName;

@Value("Bangalore")

private String shopLocation;

@Value("#{${discountDetails : {Teak: '3',Sheesham: '5',RubberWood: '2',Wenge: '8',Pine: '4'}}}")

private Map<String,Integer> discountDetails;

public String getShopName() {

return shopName;

}

public void setShopName(String shopName) {

this.shopName = shopName;

}

public String getShopLocation() {

return shopLocation;

}

public void setShopLocation(String shopLocation) {

this.shopLocation = shopLocation;

}

public Map<String, Integer> getDiscountDetails() {

return discountDetails;

}

public void setDiscountDetails(Map<String, Integer> discountDetails) {

this.discountDetails = discountDetails;

}

}

**ProductService.java**

package com.spring.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.spring.bo.ProductBO;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidProductIdException;

import org.springframework.stereotype.Component;

import com.spring.model.Product;

@Component

public class ProductService {

private ProductBO productBOObj;

@Autowired

public ProductService(ProductBO productBOObj) {

super();

this.productBOObj = productBOObj;

}

public ProductBO getProductBOObj() {

return productBOObj;

}

public void setProductBOObj(ProductBO productBOObj) {

this.productBOObj = productBOObj;

}

public double calculateBill(String productId,String productName,double mrpValue,String dimension, String woodType) throws InvalidProductIdException {

double amount=0;

// fill the code

if(productId.matches("[0-9]+")) {

ConfigurableApplicationContext context=new AnnotationConfigApplicationContext(ApplicationConfig.class);

Product product=(Product)context.getBean(Product.class);

product.setDimension(dimension);

product.setMrpValue(mrpValue);

product.setProductId(productId);

product.setProductName(productName);

amount = productBOObj.calculateBill(product, woodType);

}

else

{

throw new InvalidProductIdException("Invalid Product ID");

}

return amount;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>FurnTurnOutlet</groupId>

<artifactId>FurnTurnOutlet</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

</dependencies>

</project>

**GO-GO PARCEL**

**CourierBO.java**

package com.spring.bo;

import com.spring.model.Courier; //importing the packages and subpackages required

import java.util.Map;

public class CourierBO {

public double calculateCourierCharge(Courier cObj,String city) {

double courierCharge=0.0;

//fill the code

courierCharge = cObj.getWeight() \* cObj.getChargePerKg();

Map<String,Float> data = cObj.getServiceCharge().getLocationServiceCharge();

if(data.containsKey(city)){

courierCharge += data.get(city); //setting as counter

}

return courierCharge;

}

}

**InvalidParcelWeightException.java**

package com.spring.exception;

public class InvalidParcelWeightException extends Exception {

public InvalidParcelWeightException(String msg) {

super(msg); //argumentizing the string to superclass

//fill the code

}

}

**Driver.java**

package com.spring.main;

import com.spring.exception.InvalidParcelWeightException; //importing packages and subpackages required

import com.spring.service.CourierService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import java.util.Scanner;

public class Driver {

public static void main(String[] args) {

//fill the code

Scanner sc=new Scanner(System.in);

System.out.println("Enter the courier ID:");

int courierId = Integer.parseInt(sc.nextLine());

System.out.println("Enter the total weight of parcel:");

int weight = Integer.parseInt(sc.nextLine());

System.out.println("Enter the city:");

String city = sc.nextLine();

ApplicationContext context=new ClassPathXmlApplicationContext("beans.xml");

CourierService courierService=(CourierService)context.getBean("courierService");

//catching the unexpected exceptions using try catch block

try{

System.out.println("Total Courier Charge: " + courierService.calculateCourierCharge(courierId,weight,city));

}

catch(InvalidParcelWeightException e){

System.out.println(e.getMessage());

}

}

}

**Courier.java**

package com.spring.model;

public class Courier {

private int courierId;

private int weight;

private float chargePerKg;

private ServiceChargeInfo serviceCharge;

public ServiceChargeInfo getServiceCharge() {

return serviceCharge;

}

public void setServiceCharge(ServiceChargeInfo serviceCharge) {

this.serviceCharge = serviceCharge;

}

public int getCourierId() {

return courierId;

}

public void setCourierId(int courierId) {

this.courierId = courierId;

}

public int getWeight() {

return weight;

}

public void setWeight(int weight) {

this.weight = weight;

}

public float getChargePerKg() {

return chargePerKg;

}

public void setChargePerKg(float chargePerKg) {

this.chargePerKg = chargePerKg;

}

}

**ServiceChargeInfo.java**

package com.spring.model;

import java.util.Map;

public class ServiceChargeInfo {

private Map<String,Float> locationServiceCharge;

public Map<String, Float> getLocationServiceCharge() {

return locationServiceCharge;

}

public void setLocationServiceCharge(Map<String, Float> locationServiceCharge) {

this.locationServiceCharge = locationServiceCharge;

}

}

**CourierService.java**

package com.spring.service;

import com.spring.bo.CourierBO; //importing packages and subpackages required

import com.spring.exception.InvalidParcelWeightException;

import com.spring.model.Courier;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class CourierService {

private CourierBO cBoObj;

public CourierBO getcBoObj() {

return cBoObj;

}

public void setcBoObj(CourierBO cBoObj) {

this.cBoObj = cBoObj;

}

public double calculateCourierCharge(int courierId,int weight,String city)throws InvalidParcelWeightException {

double courierCharge=0.0;

//fill your code

if(weight>0 && weight<1000)

{

ApplicationContext context=new ClassPathXmlApplicationContext("beans.xml");

Courier cObj=(Courier)context.getBean("courier");

cObj.setCourierId(courierId);

cObj.setWeight(weight);

courierCharge = cBoObj.calculateCourierCharge(cObj, city);

}

else

{

throw new InvalidParcelWeightException("Invalid Parcel Weight");

}

return courierCharge;

}

}

**Beans.xml**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"

xmlns:aop="http://www.springframework.org/schema/aop" xmlns:context="http://www.springframework.org/schema/context"

xmlns:jee="http://www.springframework.org/schema/jee" xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:task="http://www.springframework.org/schema/task"

xsi:schemaLocation="http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.2.xsd http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.2.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.2.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-3.2.xsd">

<context:property-placeholder location="classpath:charges.properties" />

<bean id="courierBoObj" class="com.spring.bo.CourierBO" />

<bean id="courierService" class="com.spring.service.CourierService">

<property name="cBoObj" ref="courierBoObj" />

</bean>

<bean id="courier" class="com.spring.model.Courier">

<property name="courierId" value="123"/>

<property name="weight" value="45"/>

<property name="chargePerKg" value="${chargePerKg}"></property>

<property name="serviceCharge">

<bean class="com.spring.model.ServiceChargeInfo">

<property name="locationServiceCharge">

<map>

<entry key="Coimbatore" value="200.0"/>

<entry key="Chennai" value="300.0"/>

<entry key="Madurai" value="150.0"/>

</map>

</property>

</bean>

</property>

</bean>

</beans>

**Charges.properties**

chargePerKg=20.0

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>GoGoParcelService</groupId>

<artifactId>GoGoParcelService</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

</dependencies>

</project>

**LEXA KIWI SHOWROOM**

**Automobile.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class Automobile {

private String brandName;

private String brandModel;

private Showroom showroomDetails;

private String carColour;

private String carFuelType;

@Autowired

public Automobile(Showroom showroomDetails) {

super();

this.showroomDetails = showroomDetails;

}

public String getBrandName() {

return brandName;

}

public void setBrandName(String brandName) {

this.brandName = brandName;

}

public String getBrandModel() {

return brandModel;

}

public void setBrandModel(String brandModel) {

this.brandModel = brandModel;

}

public Showroom getShowroomDetails() {

return showroomDetails;

}

public void setShowroomDetails(Showroom showroomDetails) {

this.showroomDetails = showroomDetails;

}

public String getCarColour() {

return carColour;

}

public void setCarColour(String carColour) {

this.carColour = carColour;

}

public String getCarFuelType() {

return carFuelType;

}

public void setCarFuelType(String carFuelType) {

this.carFuelType = carFuelType;

}

}

**Showroom.java**

package com.spring.model;

import java.util.Map;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

@Component

public class Showroom {

@Value("Lexa Kiwi")

private String showroomName;

@Value("Melbourne")

private String showroomLocation;

@Value("#($(carPriceDetails : (Climber: '429000',Dusture: '857000',"

+ "Triber: '1503000',Elantra: '910000',Sonata: '3750000')))")

private Map<String,Double> carPriceDetails;

public String getShowroomName() {

return showroomName;

}

public void setShowroomName(String showroomName) {

this.showroomName = showroomName;

}

public String getShowroomLocation() {

return showroomLocation;

}

public void setShowroomLocation(String showroomLocation) {

this.showroomLocation = showroomLocation;

}

public Map<String, Double> getCarPriceDetails() {

return carPriceDetails;

}

public void setCarPriceDetails(Map<String, Double> carPriceDetails) {

this.carPriceDetails = carPriceDetails;

}

}

**AutomobileService.java**

package com.spring.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.bo.AutomobileBO;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidFuelTypeException;

import com.spring.model.Automobile;

@Component

public class AutomobileService {

private AutomobileBO automobileBOObj;

@Autowired

public AutomobileService(AutomobileBO automobileBOObj) {

super();

this.automobileBOObj = automobileBOObj;

}

public AutomobileBO getAutomobileBOObj() {

return automobileBOObj;

}

public void setAutomobileBOObj(AutomobileBO automobileBOObj) {

this.automobileBOObj = automobileBOObj;

}

public double calculateOnRoadPrice(String brandName,String brandModel,String carColour,String carFuelType,String carName)

throws InvalidFuelTypeException {

double amount=0;

// fill the code

String s1 = "Petrol";

String s2 = "Diesel";

if(carFuelType.equals(s1) || carFuelType.equals(s2)) {

@SuppressWarnings("resource")

ConfigurableApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

Automobile am =(Automobile)context.getBean(Automobile.class);

am.setBrandName(brandName);

am.setBrandModel(brandModel);

am.setCarColour(carColour);

am.setCarFuelType(carFuelType);

amount = automobileBOObj.calculateOnRoadPrice(am, carName);

}

else {

throw new InvalidFuelTypeException("Fuel Type is Not Valid");

}

return amount;

}

}

**AutomobileBO.java**

package com.spring.bo;

import java.util.Map;

import org.springframework.stereotype.Component;

import com.spring.model.Automobile;

@Component

public class AutomobileBO {

public double calculateOnRoadPrice(Automobile carObj,String carName) {

double amount=0;

// fill the code

Map<String, Double> map = carObj.getShowroomDetails().getCarPriceDetails();

if(map.containsKey(carName)) {

double d = map.get(carName);

amount = d + 38200 + ((d\*3)/100);

}

return amount;

}

}

**ApplicationConfig.java**

package com.spring.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.stereotype.Component;

import com.spring.bo.AutomobileBO;

import com.spring.model.Automobile;

import com.spring.model.Showroom;

import com.spring.service.AutomobileService;

// fill the code

@ComponentScan

@Configuration

@Component

public class ApplicationConfig {

@Bean

public Showroom showroom() {

return new Showroom();

}

@Bean

public Automobile automobile() {

return new Automobile(showroom());

}

@Bean

public AutomobileBO automobileBO() {

return new AutomobileBO();

}

public AutomobileService automobileService() {

return new AutomobileService(automobileBO());

}

}

**InvalidFuelTypeException.java**

package com.spring.exception;

public class InvalidFuelTypeException extends Exception {

public InvalidFuelTypeException(String msg) {

// fill the code

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.Scanner;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidFuelTypeException;

import com.spring.service.AutomobileService;

@Component

public class Driver {

public static void main(String[] args) {

Scanner in=new Scanner(System.in);

// fill the code

System.out.println("Enter the brand Name");

String brandName = in.nextLine();

System.out.println("Enter the brand Model");

String brandModel = in.nextLine();

System.out.println("Enter the preferred car colour");

String carColour = in.nextLine();

System.out.println("Enter the preferred fuel type");

String carFuelType = in.nextLine();

System.out.println("Enter the car Name");

String carName = in.nextLine();

@SuppressWarnings("resource")

ConfigurableApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

AutomobileService as = (AutomobileService) context.getBean(AutomobileService.class);

try {

double d = as.calculateOnRoadPrice(brandName, brandModel, carColour, carFuelType, carName);

System.out.println("Estimated on road price for the preferred car is: "+d);

}

catch(InvalidFuelTypeException e) {

System.out.println(e.getMessage());

}

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>LexaKiwiShowroom</groupId>

<artifactId>LexaKiwiShowroom</artifactId>

<version>0.0.1-SNAPSHOT</version>

<properties>

<maven.compiler.target>1.8</maven.compiler.target>

<maven.compiler.source>1.8</maven.compiler.source>

<testing.version>6.14.3</testing.version>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.springframwork/spring.context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.2.RELEASE</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugin</groupId>

<artifactId>maven-surefire-report-plugin</artifactId>

<version>3.0.0-M4</version>

</plugin>

</plugins>

</build>

</project>

**STATIONARY BILL**

**Application.properties**

#Don't change property names, you can change value

notebook.CostPerQuantity=100

pen.CostPerQuantity=50

pencil.CostPerQuantity=10

notebook.quantity=7

pen.quantity=10

pencil.quantity=1

**Spring.xml**

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context.xsd">

<context:property-placeholder

location="classpath:application.properties" />

<bean id="notebook" class="com.cts.stationarybill.vo.Item" >

<property name="itemName" value="Notebook"/>

<property name="itemCostPerQuantity" value="${notebook.CostPerQuantity}"/>

<property name="quantity" value="${notebook.quantity}"/>

</bean>

<bean id="pen" class="com.cts.stationarybill.vo.Item" >

<property name="itemName" value="Pen"/>

<property name="itemCostPerQuantity" value="${pen.CostPerQuantity}"/>

<property name="quantity" value="${pen.quantity}"/>

</bean>

<bean id="pencil" class="com.cts.stationarybill.vo.Item" >

<property name="itemName" value="Pencil"/>

<property name="itemCostPerQuantity" value="${pencil.CostPerQuantity}"/>

<property name="quantity" value="${pencil.quantity}"/>

</bean>

<bean id="myCart" class="com.cts.stationarybill.vo.MyCart">

<property name="items">

<list>

<ref bean="notebook"/>

<ref bean="pen"/>

<ref bean="pencil"/>

</list>

</property>

</bean>

<bean id="cartBo" class="com.cts.stationarybill.bo.CartBO">

<constructor-arg ref="myCart"/>

</bean>

<bean id="cartService" class="com.cts.stationarybill.service.CartService">

<property name="cartBo" ref="cartBo" />

</bean>

</beans>

**CartBO.java**

package com.cts.stationarybill.bo;

import com.cts.stationarybill.vo.Item;

import com.cts.stationarybill.vo.MyCart;

import java.util.\*;

public class CartBO {

private MyCart cart;

public CartBO() {

// TODO Auto-generated constructor stub

}

public CartBO(MyCart cart) {

super();

this.cart = cart;

}

public MyCart getCart() {

return cart;

}

public void setCart(MyCart cart) {

this.cart = cart;

}

public double calculateBillAfterDiscount(double totalOrderBill) {

List<Item> items= cart.getItems();

double nbprice=0.0;

double penprice=0.0;

for(Item i : items)

{

if(i.getItemName().equals("Notebook"))

nbprice=i.getItemCostPerQuantity();

if(i.getItemName().equals("Pen"))

penprice=i.getItemCostPerQuantity();

}

int notebooks= countNoteBook();

int pens= countPen();

double disc=0.0;

if(totalOrderBill>500)

{

if(notebooks>=3)

disc+=(notebooks\*0.1\*nbprice);

if(pens>=10)

disc+=(pens\*0.2\*penprice);

}

return (totalOrderBill-disc); //TODO, change this value

}

public int countNoteBook() {

List<Item> items= cart.getItems();

int count = 0;

for(Item i : items)

{

if(i.getItemName().equals("Notebook"))

{

count=i.getQuantity();

break;

}

}

return count; //TODO, change this value

}

public int countPen() {

List<Item> items= cart.getItems();

int count = 0;

for(Item i : items)

{

if(i.getItemName().equals("Pen"))

{

count=i.getQuantity();

break;

}

}

return count; //TODO, change this value

}

}

**InvalidCostPerQuantityException.java**

package com.cts.stationarybill.exception;

public class InvalidCostPerQuantityException extends Exception{

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

public InvalidCostPerQuantityException(String message) {

super(message);

}

}

**Main.java**

package com.cts.stationarybill.main;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.cts.stationarybill.exception.InvalidCostPerQuantityException;

import com.cts.stationarybill.service.CartService;

import com.cts.stationarybill.skeletonvalidator.SkeletonValidator;

public class Main

{

public static void main( String[] args )

{

SkeletonValidator validator = new SkeletonValidator();

@SuppressWarnings("resource")

ApplicationContext ctx=new ClassPathXmlApplicationContext("spring.xml");

CartService service=(CartService)ctx.getBean("cartService");

try

{

double totalBill=service.calculateOrderTotalBill();

System.out.println("Your total bill is = "+totalBill);

}

catch(InvalidCostPerQuantityException e)

{

System.out.println(e.getMessage());

}

}

}

**CartService.java**

package com.cts.stationarybill.service;

import java.util.List;

import com.cts.stationarybill.bo.CartBO;

import com.cts.stationarybill.exception.InvalidCostPerQuantityException;

import com.cts.stationarybill.vo.Item;

public class CartService {

private CartBO cartBo;

public CartService() {

// TODO Auto-generated constructor stub

}

public CartService(CartBO cartBo) {

super();

this.cartBo = cartBo;

}

public CartBO getCartBo() {

return cartBo;

}

public void setCartBo(CartBO cartBo) {

this.cartBo = cartBo;

}

/\*\*

\* Method to calculate total bill

\*

\* @param MyCart

\* @return totalOrderBill

\*/

public double calculateOrderTotalBill() throws InvalidCostPerQuantityException {

double totalOrderBill = 0;

List<Item> items = cartBo.getCart().getItems();

if(validateCostPerQuantity())

{

for(Item i:items)

totalOrderBill+=i.getItemCostPerQuantity()\*i.getQuantity();

totalOrderBill=cartBo.calculateBillAfterDiscount(totalOrderBill);

return totalOrderBill;

}

else

{

throw new InvalidCostPerQuantityException("Cost per quantity must be positive non zero value");

}

}

/\*\*

\* Method to validate cost per quantity and throws exception if validation failes

\*

\* @exception InvalidCostPerQuantityException

\*/

public boolean validateCostPerQuantity() throws InvalidCostPerQuantityException{

List<Item> items = cartBo.getCart().getItems();

for(Item i:items)

{

if(i.getItemCostPerQuantity()<1)

throw new InvalidCostPerQuantityException("Cost per quantity must be positive non zero value");

}

return true;

}

}

**SkeletonValidator.java**

package com.cts.stationarybill.skeletonvalidator;

import java.lang.reflect.Method;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* @author t-aarti3

\* This class is used to verify if the Code Skeleton is intact and not

\* modified by participants thereby ensuring smooth auto evaluation

\* \*/

public class SkeletonValidator {

public SkeletonValidator() {

validateClassName("com.cts.stationarybill.service.CartService");

validateClassName("com.cts.stationarybill.vo.Item");

validateClassName("com.cts.stationarybill.vo.MyCart");

validateClassName("com.cts.stationarybill.bo.CartBO");

validateMethodSignature(

"calculateOrderTotalBill:double,validateCostperQuantity:boolean",

"com.cts.stationarybill.service.CartService");

validateMethodSignature(

"calculateBillAfterDiscount:double,countNoteBook:int,countPen:int",

"com.cts.stationarybill.bo.CartBO");

}

private static final Logger LOG = Logger.getLogger("SkeletonValidator");

protected final boolean validateClassName(String className) {

boolean iscorrect = false;

try {

Class.forName(className);

iscorrect = true;

LOG.info("Class Name " + className + " is correct");

} catch (ClassNotFoundException e) {

LOG.log(Level.SEVERE, "You have changed either the " + "class name/package. Use the correct package "

+ "and class name as provided in the skeleton");

} catch (Exception e) {

LOG.log(Level.SEVERE,

"There is an error in validating the " + "Class Name. Please manually verify that the "

+ "Class name is same as skeleton before uploading");

}

return iscorrect;

}

protected final void validateMethodSignature(String methodWithExcptn, String className) {

Class cls = null;

try {

String[] actualmethods = methodWithExcptn.split(",");

boolean errorFlag = false;

String[] methodSignature;

String methodName = null;

String returnType = null;

for (String singleMethod : actualmethods) {

boolean foundMethod = false;

methodSignature = singleMethod.split(":");

methodName = methodSignature[0];

returnType = methodSignature[1];

cls = Class.forName(className);

Method[] methods = cls.getMethods();

for (Method findMethod : methods) {

if (methodName.equals(findMethod.getName())) {

foundMethod = true;

if (!(findMethod.getReturnType().getName().equals(returnType))) {

errorFlag = true;

LOG.log(Level.SEVERE, " You have changed the " + "return type in '" + methodName

+ "' method. Please stick to the " + "skeleton provided");

} else {

LOG.info("Method signature of " + methodName + " is valid");

}

}

}

if (!foundMethod) {

errorFlag = true;

LOG.log(Level.SEVERE, " Unable to find the given public method " + methodName

+ ". Do not change the " + "given public method name. " + "Verify it with the skeleton");

}

}

if (!errorFlag) {

LOG.info("Method signature is valid");

}

} catch (Exception e) {

LOG.log(Level.SEVERE,

" There is an error in validating the " + "method structure. Please manually verify that the "

+ "Method signature is same as the skeleton before uploading");

}

}

}

**Item.java**

package com.cts.stationarybill.vo;

public class Item {

// member variables

private String itemName;

private double itemCostPerQuantity;

private int quantity;

public Item() {

// TODO Auto-generated constructor stub

}

// Parameterized Constructor

public Item(String itemName, double itemCostPerQuantity, int quantity) {

super();

this.itemName = itemName;

this.itemCostPerQuantity = itemCostPerQuantity;

this.quantity = quantity;

}

// getter and setter methods

/\*\*

\* @return the itemName

\*/

public String getItemName() {

return itemName;

}

/\*\*

\* @param itemName

\* the itemName to set

\*/

public void setItemName(String itemName) {

this.itemName = itemName;

}

/\*\*

\* @return the itemCostPerQuantity

\*/

public double getItemCostPerQuantity() {

return itemCostPerQuantity;

}

/\*\*

\* @param itemCostPerQuantity

\* the itemCostPerQuantity to set

\*/

public void setItemCostPerQuantity(double itemCostPerQuantity) {

this.itemCostPerQuantity = itemCostPerQuantity;

}

/\*\*

\* @return the quantity

\*/

public int getQuantity() {

return quantity;

}

/\*\*

\* @param quantity

\* the quantity to set

\*/

public void setQuantity(int quantity) {

this.quantity = quantity;

}

@Override

public String toString() {

return "Item [itemName=" + itemName + ", itemCostPerQuantity=" + itemCostPerQuantity + ", quantity="

+ quantity + "]";

}

}

**MyCart.java**

package com.cts.stationarybill.vo;

import java.util.List;

public class MyCart {

private List<Item> items;

public MyCart() {

// TODO Auto-generated constructor stub

}

public MyCart(List<Item> items) {

super();

this.items = items;

}

public List<Item> getItems() {

return items;

}

public void setItems(List<Item> items) {

this.items = items;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cts.stationerybill</groupId>

<artifactId>RoughStationeryBill</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>StationeryBill</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

<junit.jupiter.version>5.4.0</junit.jupiter.version>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.0.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-engine -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>${junit.jupiter.version}</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>${junit.jupiter.version}</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.platform/junit-platform-launcher -->

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-launcher</artifactId>

<version>1.5.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-commons</artifactId>

<version>1.5.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.1</version>

</plugin>

</plugins>

</build>

<reporting>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-report-plugin</artifactId>

<version>2.19.1</version>

</plugin>

</plugins>

</reporting>

</project>

**COLLEGE ELECTION**

**totalVotes.properties**

totalVotes=300

**Application.java**

package com.spring.config;

import java.util.ArrayList;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.annotation.PropertySource;

import com.spring.model.Candidate;

//Use appropriate annotation

@Configuration

@ComponentScan(basePackages = "com.spring")

@PropertySource("totalVotes.properties")

public class ApplicationConfig {

@Bean

public ArrayList<Candidate> getVoteList() {

ArrayList<Candidate> voteList = new ArrayList<Candidate>();

voteList.add(new Candidate("Rahul", "RA102021", "BBA", 110));

voteList.add(new Candidate("Pavithra", "RA222021", "BSC", 97));

voteList.add(new Candidate("Jerom", "RA332021", "BBA", 78));

return voteList;

}

}

**ElectionDAO.java**

package com.spring.dao;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.spring.model.Candidate;

//Use appropriate annotation

@Component

public class ElectionDAO {

@Autowired

private ArrayList<Candidate> voteList = new ArrayList<Candidate>();

public Candidate candidateWithMaximumVote() {

int max = 0;

Candidate withMaxVote = null;

for (Candidate candidate : voteList) {

if (candidate.getNumberOfVotes() > max) {

withMaxVote = candidate;

max = candidate.getNumberOfVotes();

}

}

return withMaxVote;

}

public int getTotalVotes() {

return Candidate.getTotalVotes();

}

public int unpolledVotes() {

return getTotalVotes() - totalpolledVotes();

}

public int totalpolledVotes() {

int total = 0;

for (Candidate candidate : voteList) {

total += candidate.getNumberOfVotes();

}

return total;

}

public ElectionDAO() {

}

public ArrayList<Candidate> getVoteList() {

return voteList;

}

public void setVoteList(ArrayList<Candidate> voteList) {

this.voteList = voteList;

}

}

**Driver.java**

package com.spring.main;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.spring.config.ApplicationConfig;

import com.spring.model.Candidate;

import com.spring.service.ElectionService;

public class Driver {

public static void main(String[] args) {

ApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

ElectionService service = context.getBean(ElectionService.class);

System.out.println("College Election Report");

System.out.println("Total Number of Votes:" + service.getTotalVotes());

System.out.println("Total Number of polled Votes:" + service.totalpolledVotes());

System.out.println("Total Number of unpolled Votes:" + service.unpolledVotes());

for (Candidate candidate : service.getVoteList()) {

System.out.println(candidate);

}

Candidate winner = service.candidateWithMaximumVote();

System.out.println("Election won by " + winner.getName() + " with " + winner.getNumberOfVotes() + " votes");

}

}

**Candidate.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

//POJO class

//Use appropriate annotation

@Component

public class Candidate {

private String name;

private String candidateId;

private String department;

private int numberOfVotes;

private static int totalVotes;

public Candidate(String name, String candidateId, String department, int numberOfVotes) {

super();

this.name = name;

this.candidateId = candidateId;

this.department = department;

this.numberOfVotes = numberOfVotes;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCandidateId() {

return candidateId;

}

public void setCandidateId(String candidateId) {

this.candidateId = candidateId;

}

public static int getTotalVotes() {

return totalVotes;

}

@Value("${totalVotes}")

public void setTotalVotes(int totalVotes) {

Candidate.totalVotes = totalVotes;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

public int getNumberOfVotes() {

return numberOfVotes;

}

public void setNumberOfVotes(int numberOfVotes) {

this.numberOfVotes = numberOfVotes;

}

public Candidate() {

}

@Override

public String toString() {

return "Candidate " + this.name + " got " + this.numberOfVotes + " votes";

}

}

**ElectionService.java**

package com.spring.service;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.spring.dao.ElectionDAO;

import com.spring.model.Candidate;

//Use appropriate annotation

@Component

public class ElectionService {

private ElectionDAO electionDAO;

@Autowired

public ElectionService(ElectionDAO electionDAO) {

super();

this.electionDAO = electionDAO;

}

public Candidate candidateWithMaximumVote() {

return electionDAO.candidateWithMaximumVote();

}

public int unpolledVotes() {

return electionDAO.unpolledVotes();

}

public int totalpolledVotes() {

return electionDAO.totalpolledVotes();

}

public int getTotalVotes() {

return electionDAO.getTotalVotes();

}

public ArrayList<Candidate> getVoteList() {

return electionDAO.getVoteList();

}

public void setElectionDAO(ElectionDAO electionDAO) {

this.electionDAO = electionDAO;

}

public ElectionDAO getElectionDAO() {

return electionDAO;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>CollegeElection</groupId>

<artifactId>CollegeElection</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>

**EASY RECHARGE**

**Beans.xml**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"

xmlns:aop="http://www.springframework.org/schema/aop" xmlns:context="http://www.springframework.org/schema/context"

xmlns:jee="http://www.springframework.org/schema/jee" xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:task="http://www.springframework.org/schema/task"

xsi:schemaLocation="http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.2.xsd http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.2.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.2.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-3.2.xsd">

<context:property-placeholder location="classpath:cashBack.properties" />

<!-- fill the code -->

<bean id="recharge" class="com.spring.model.Recharge">

<property name="rechargePlan">

<bean id="rechargeplanobj" class="com.spring.model.RechargePlan">

<property name="rechargePlans">

<map>

<entry key="199" value="1.5GB data,28 days validity"/>

<entry key="399" value="2GB data,58 days validity"/>

<entry key="550" value="2.5GB data,85 days validity"/>

<entry key="51" value="6GB data,6 days validity"/>

<entry key="750" value="4GB data,90 days validity"/>

</map>

</property>

</bean>

</property>

<property name="cashBackPercent" value="${cashBackPercent}" />

</bean>

<bean id="rechargeBoObj" class="com.spring.bo.RechargeBO"/>

<bean id="rechargeservice" class="com.spring.service.RechargeService">

<property name="rechargeBO" ref="rechargeBoObj"/>

</bean>

</beans>

**cashBack.properties**

cashBackPercent=5

**RechargeBO.java**

package com.spring.bo;

import com.spring.model.Recharge;

import com.spring.model.RechargePlan;

import java.util.\*;

public class RechargeBO {

public double processRecharge(Recharge recharge){

double amount=0;

// fill the code

double rechargeamount=recharge.getRechargeAmount();

RechargePlan rechargeplanobj=recharge.getRechargePlan();

Map<Double, String> rechargeplans=rechargeplanobj.getRechargePlans();

boolean flag=false;

for(Double findamount:rechargeplans.keySet())

{

if(findamount==rechargeamount)

{

flag=true;

break;

}

}

if(flag==true)

{

amount=(rechargeamount\*recharge.getCashBackPercent())/100;

}

return amount;

}

}

**InvalidRechargePackException.java**

package com.spring.exception;

public class InvalidRechargePackException extends Exception {

public InvalidRechargePackException(String msg) {

//fill the code

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.\*;

import com.spring.service.RechargeService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.spring.exception.InvalidRechargePackException;

public class Driver {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

System.out.println("Enter the Recharge Id:");

String rechargeId=sc.next();

System.out.println("Enter the Operator:");

String operator=sc.next();

System.out.println("Enter the Customer PhoneNumber:");

long phonenumber=sc.nextLong();

System.out.println("Enter the Recharge Amount:");

double rechargeamount=sc.nextDouble();

ApplicationContext ctx=new ClassPathXmlApplicationContext("beans.xml");

RechargeService rechargeserviceobj=(RechargeService) ctx.getBean("rechargeservice");

double amount=0;

try

{

amount=rechargeserviceobj.processRecharge(rechargeId,operator,phonenumber,rechargeamount);

}

catch(InvalidRechargePackException exc)

{

System.out.println(exc.getMessage());

}

if(amount!=0)

{

System.out.println("Recharge of "+rechargeamount+" is successful. You got a cash back of RS:"+amount);

}

}

}

**Recharge.java**

package com.spring.model;

//pojo class with required attributes,getters and setters

public class Recharge {

private String rechargeId;

private long phoneNumber;

private double rechargeAmount;

private RechargePlan rechargePlan;

private int cashBackPercent;

public int getCashBackPercent() {

return cashBackPercent;

}

public void setCashBackPercent(int cashBackPercent) {

this.cashBackPercent = cashBackPercent;

}

public String getRechargeId() {

return rechargeId;

}

public void setRechargeId(String rechargeId) {

this.rechargeId = rechargeId;

}

public RechargePlan getRechargePlan() {

return rechargePlan;

}

public void setRechargePlan(RechargePlan rechargePlan) {

this.rechargePlan = rechargePlan;

}

public long getPhoneNumber() {

return phoneNumber;

}

public void setPhoneNumber(long phoneNumber) {

this.phoneNumber = phoneNumber;

}

public double getRechargeAmount() {

return rechargeAmount;

}

public void setRechargeAmount(double rechargeAmount) {

this.rechargeAmount = rechargeAmount;

}

}

**RechargePlan.java**

package com.spring.model;

import java.util.Map;

//pojo class with required attributes,getters and setters

public class RechargePlan {

private String operator;

private Map<Double, String> rechargePlans;

public Map<Double, String> getRechargePlans() {

return rechargePlans;

}

public void setRechargePlans(Map<Double, String> rechargePlans) {

this.rechargePlans = rechargePlans;

}

public String getOperator() {

return operator;

}

public void setOperator(String operator) {

this.operator = operator;

}

}

**RechargeService.java**

package com.spring.service;

import com.spring.bo.RechargeBO;

import com.spring.model.Recharge;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.spring.exception.InvalidRechargePackException;

public class RechargeService {

private RechargeBO rechargeBO;

public RechargeBO getRechargeBO() {

return rechargeBO;

}

public void setRechargeBO(RechargeBO rechargeBO) {

this.rechargeBO = rechargeBO;

}

public double processRecharge(String rechargeId, String operator,long phoneNumber,double rechargeAmount) throws InvalidRechargePackException{

double amount=0;

ApplicationContext ctx=new ClassPathXmlApplicationContext("beans.xml");

Recharge rechargeobj=(Recharge) ctx.getBean("recharge");

rechargeobj.setRechargeId(rechargeId);

rechargeobj.setPhoneNumber(phoneNumber);

rechargeobj.setRechargeAmount(rechargeAmount);

amount=rechargeBO.processRecharge(rechargeobj);

if(amount==0)

{

throw new InvalidRechargePackException("Recharge Pack not available");

}

return amount;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>EasyRecharge</groupId>

<artifactId>EasyRecharge</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>EasyRecharge</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>

**HOSTEL MANAGEMENT**

**feesDiscountDetails.properties**

feesDiscount.map={'O':'50', 'D':'40', 'A':'30', 'B':'20'}

**StudentBO.java**

package com.spring.bo;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.config.ApplicationConfig;

import com.spring.model.FeesDiscountDetails;

import com.spring.model.Student;

//use appropriate annotation to make this class as component class

@Component

public class StudentBO {

public float calculateFee(Student obj) {

ApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

FeesDiscountDetails fd = context.getBean(FeesDiscountDetails.class);

float fee = obj.getHostel().getRent();

if (fd.getFeesDiscount().keySet().contains(Character.toString(obj.getGrade())))

fee -= (fee \* fd.getFeesDiscount().get(Character.toString(obj.getGrade())) / 100);

return fee;

}

}

**ApplicationConfig.java**

package com.spring.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.annotation.PropertySource;

import com.spring.bo.StudentBO;

import com.spring.model.FeesDiscountDetails;

import com.spring.model.Hostel;

import com.spring.model.Student;

import com.spring.service.StudentService;

@Configuration

@PropertySource("feesDiscountDetails.properties")

@ComponentScan

public class ApplicationConfig {

@Bean

public FeesDiscountDetails feesDiscountDetails() {

FeesDiscountDetails feesDiscountDetails = new FeesDiscountDetails();

return feesDiscountDetails;

}

@Bean

public Hostel hostel() {

Hostel hostel = new Hostel();

return hostel;

}

@Bean

public Student student() {

Student student = new Student(hostel());

return student;

}

@Bean

public StudentBO studentBO() {

StudentBO studentBO = new StudentBO();

return studentBO;

}

@Bean

public StudentService studentService() {

StudentService studentService = new StudentService(studentBO());

return studentService;

}

}

**InvalidGradeException.java**

package com.spring.exception;

public class InvalidGradeException extends Exception {

public InvalidGradeException(String msg) {

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.Scanner;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidGradeException;

import com.spring.service.StudentService;

public class Driver {

public static void main(String[] args) {

ApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

StudentService studentService = context.getBean(StudentService.class);

Scanner sc = new Scanner(System.in);

System.out.println("Enter the student name:");

String name = sc.nextLine();

System.out.println("Enter the admission number:");

String admissionNo = sc.nextLine();

System.out.println("Enter the grade:");

char grade = sc.nextLine().charAt(0);

System.out.println("Enter the hostel details");

System.out.println("Enter the hostel name:");

String hostelName = sc.nextLine();

System.out.println("Enter the room rent:");

float rent = sc.nextFloat();

try {

System.out.println(

"Hostel fee is Rs:" + studentService.calculateFee(name, admissionNo, grade, hostelName, rent));

} catch (InvalidGradeException e) {

System.out.println(e.getMessage());

}

}

}

**FeesDiscountDetails.java**

package com.spring.model;

import java.util.Map;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class FeesDiscountDetails {

@Value("#{${feesDiscount.map}}")

private Map<String, Integer> feesDiscount;

public Map<String, Integer> getFeesDiscount() {

return feesDiscount;

}

public void setFeesDiscount(Map<String, Integer> feesDiscount) {

this.feesDiscount = feesDiscount;

}

@Override

public String toString() {

return "FeesDiscountDetails [feesDiscount=" + feesDiscount + "]";

}

}

**Hostel.java**

package com.spring.model;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class Hostel {

private String hostelName;

private float rent;

public String getHostelName() {

return hostelName;

}

public void setHostelName(String hostelName) {

this.hostelName = hostelName;

}

public float getRent() {

return rent;

}

public void setRent(float rent) {

this.rent = rent;

}

@Override

public String toString() {

return "Hostel [hostelName=" + hostelName + ", rent=" + rent + "]";

}

}

**Student.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class Student {

private String name;

private String admissionNo;

private char grade;

private Hostel hostel;

@Autowired

public Student(Hostel hostel) {

super();

this.hostel = hostel;

}

public Hostel getHostel() {

return hostel;

}

public void setHostel(Hostel hostel) {

this.hostel = hostel;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAdmissionNo() {

return admissionNo;

}

public void setAdmissionNo(String admissionNo) {

this.admissionNo = admissionNo;

}

public char getGrade() {

return grade;

}

public void setGrade(char grade) {

this.grade = grade;

}

@Override

public String toString() {

return "Student [name=" + name + ", admissionNo=" + admissionNo + ", grade=" + grade + ", hostel=" + hostel

+ "]";

}

}

**StudentService.java**

package com.spring.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.bo.StudentBO;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidGradeException;

import com.spring.model.Student;

//use appropriate annotation to make this class as component class

@Component

public class StudentService {

public StudentBO getStudentBOObj() {

return studentBOObj;

}

public void setStudentBOObj(StudentBO studentBOObj) {

this.studentBOObj = studentBOObj;

}

private StudentBO studentBOObj;

// fill the code

@Autowired

public StudentService(StudentBO studentBOObj) {

super();

this.studentBOObj = studentBOObj;

}

public float calculateFee(String name, String admissionNo, char grade, String hostelName, float rent)

throws InvalidGradeException {

String validChars = "ODABCF";

if (!validChars.contains(Character.toString(grade))) {

throw new InvalidGradeException("Invalid Grade");

}

ApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);

Student student = context.getBean(Student.class);

student.setName(name);

student.setAdmissionNo(admissionNo);

student.setGrade(grade);

student.getHostel().setHostelName(hostelName);

student.getHostel().setRent(rent);

return this.studentBOObj.calculateFee(student);

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>HostelManagement</groupId>

<artifactId>HostelManagement</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>

**NAKSHATRA FASHION**

**couponCode.properties**

couponCodes.map={'MAX':'40', 'EPIC':'30', 'GRAB':'60', 'MARK':'55'}

**ShipmentBO.java**

package com.spring.bo;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.config.ApplicationConfig;

import com.spring.model.CouponUtility;

import com.spring.model.Shipment;

//use appropriate annotation to make this class as component class

@Component

public class ShipmentBO {

public double processDiscount(Shipment shipmentObj) {

double amount = 0;

// fill the code

ApplicationContext ctx = new AnnotationConfigApplicationContext(ApplicationConfig.class);

CouponUtility util = ctx.getBean(CouponUtility.class);

int couponCode = util.getCouponCodes().get(shipmentObj.getCouponCode());

double price = shipmentObj.getItem().getPrice();

amount = price - (price \* couponCode) / 100.0;

return amount;

}

}

**ApplicationConfig.java**

package com.spring.config;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.annotation.PropertySource;

//Use appropriate annotation

@Configuration

@ComponentScan("com.spring.\*")

@PropertySource("couponCode.properties")

public class ApplicationConfig {

}

**InvalidCouponCodeException.java**

package com.spring.exception;

public class InvalidCouponCodeException extends Exception {

public InvalidCouponCodeException(String msg) {

// fill the code

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.Scanner;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidCouponCodeException;

import com.spring.service.ShipmentService;

public class Driver {

public static void main(String[] args) {

// fill the code

ApplicationContext ctx = new AnnotationConfigApplicationContext(ApplicationConfig.class);

Scanner sc = new Scanner(System.in);

System.out.println("Enter the Item Id:");

String itemId = sc.nextLine();

System.out.println("Enter the Item Type:");

String itemType = sc.nextLine();

System.out.println("Enter the Price:");

double price = Double.parseDouble(sc.nextLine());

System.out.println("Enter the Shipment Details");

System.out.println("Enter the Shipment Id:");

String shipId = sc.nextLine();

System.out.println("Enter the Shipment Date:");

String shipDate = sc.nextLine();

System.out.println("Enter the Shipment Address:");

String shipAddress = sc.nextLine();

System.out.println("Enter the Coupon code:");

String couponCode = sc.nextLine();

try {

ShipmentService service = ctx.getBean(ShipmentService.class);

double amount = service.processDiscount(shipId, shipDate, itemId, itemType, price, shipAddress, couponCode);

System.out.print("Discounted Amount: " + amount);

} catch (InvalidCouponCodeException e) {

System.out.println(e.getMessage());

}

}

}

**CouponUtility.java**

package com.spring.model;

import java.util.Map;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class CouponUtility {

public Map<String, Integer> getCouponCodes() {

return couponCodes;

}

public void setCouponCodes(Map<String, Integer> couponCodes) {

this.couponCodes = couponCodes;

}

// Fill the code

@Value(value = "#{${couponCodes.map}}")

private Map<String, Integer> couponCodes;

}

**Item.java**

package com.spring.model;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class Item {

private String itemId;

private String itemType;

private double price;

public String getItemId() {

return itemId;

}

public void setItemId(String itemId) {

this.itemId = itemId;

}

public String getItemType() {

return itemType;

}

public void setItemType(String itemType) {

this.itemType = itemType;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

}

**Shipment.java**

package com.spring.model;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

//pojo class with required attributes,getters and setters

//use appropriate annotation to make this class as component class

@Component

public class Shipment {

private String shipmentId;

private String shipmentDate;

private Item item;

private String deliveryAddress;

private String couponCode;

public String getCouponCode() {

return couponCode;

}

public void setCouponCode(String couponCode) {

this.couponCode = couponCode;

}

public String getDeliveryAddress() {

return deliveryAddress;

}

public void setDeliveryAddress(String deliveryAddress) {

this.deliveryAddress = deliveryAddress;

}

// fill the code

@Autowired

public Shipment(Item item) {

super();

this.item = item;

}

public String getShipmentId() {

return shipmentId;

}

public void setShipmentId(String shipmentId) {

this.shipmentId = shipmentId;

}

public String getShipmentDate() {

return shipmentDate;

}

public void setShipmentDate(String shipmentDate) {

this.shipmentDate = shipmentDate;

}

public Item getItem() {

return item;

}

public void setItem(Item item) {

this.item = item;

}

}

**ShipmentService.java**

package com.spring.service;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.stereotype.Component;

import com.spring.bo.ShipmentBO;

import com.spring.config.ApplicationConfig;

import com.spring.exception.InvalidCouponCodeException;

import com.spring.model.CouponUtility;

import com.spring.model.Shipment;

//use appropriate annotation to make this class as component class

@Component

public class ShipmentService {

private ShipmentBO shipmentBOObj;

// fill the code

@Autowired

public ShipmentService(ShipmentBO shipmentBOObj) {

super();

this.shipmentBOObj = shipmentBOObj;

}

public ShipmentBO getShipmentBOObj() {

return shipmentBOObj;

}

public void setShipmentBOObj(ShipmentBO shipmentBOObj) {

this.shipmentBOObj = shipmentBOObj;

}

public double processDiscount(String shipmentId, String shipmentDate, String itemId, String itemType, double price,

String address, String couponCode) throws InvalidCouponCodeException {

double amount = 0;

// fill the code

ApplicationContext ctx = new AnnotationConfigApplicationContext(ApplicationConfig.class);

CouponUtility util = ctx.getBean(CouponUtility.class);

if (!util.getCouponCodes().containsKey(couponCode))

throw new InvalidCouponCodeException("Invalid Coupon Code");

Shipment spmt = ctx.getBean(Shipment.class);

ShipmentBO bo = ctx.getBean(ShipmentBO.class);

spmt.setCouponCode(couponCode);

spmt.setDeliveryAddress(address);

spmt.setShipmentDate(shipmentDate);

spmt.setShipmentId(shipmentId);

spmt.getItem().setItemId(itemId);

spmt.getItem().setItemType(itemType);

spmt.getItem().setPrice(price);

amount = bo.processDiscount(spmt);

return amount;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>NakshatraFashion</groupId>

<artifactId>NakshatraFashion</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>

**PINK FTINESS FRANCHISE**

**franchisePercent.properties**

percentage=12.5

**Beans.xml**

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"

xmlns:aop="http://www.springframework.org/schema/aop" xmlns:context="http://www.springframework.org/schema/context"

xmlns:jee="http://www.springframework.org/schema/jee" xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:task="http://www.springframework.org/schema/task"

xsi:schemaLocation="http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.2.xsd http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.2.xsd http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.2.xsd http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.2.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.2.xsd http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-3.2.xsd">

<context:property-placeholder location="classpath:FranchisePercent.properties" />

<!-- fill the code -->

<bean name="fitnessFranchise" class="com.spring.model.FitnessFranchise" scope="prototype">

<constructor-arg name="percentage" value="${percentage}" />

<constructor-arg name="commonExpenses">

<map>

<entry key="Rent" value="12000.0"/>

<entry key="HouseKeeping" value="4000.0"/>

<entry key="ElectricityCharges" value="2000.0"/>

<entry key="Maintenence" value="3000.0"/>

</map>

</constructor-arg>

</bean>

<bean name="fitnessService" class="com.spring.service.FitnessService" scope="prototype" >

<constructor-arg ref="fitnessBO"/>

</bean>

<bean name="fitnessBO" class="com.spring.bo.FitnessBO" scope="prototype"/>

</beans>

**FitnessBO.java**

package com.spring.bo;

import com.spring.model.FitnessFranchise;

public class FitnessBO {

public double calculateNetProfit(FitnessFranchise franchise) {

//// Calculating netProfit based on factors

double expense = franchise.getTotalIncome() - (franchise.getTrainerSalary()

+ franchise.getCommonExpenses().get("Rent") + franchise.getCommonExpenses().get("HouseKeeping")

+ franchise.getCommonExpenses().get("ElectricityCharges")

+ franchise.getCommonExpenses().get("Maintenence"));

return expense;

}

}

**NoProfitException.java**

package com.spring.exception;

public class NoProfitException extends Exception {

public NoProfitException(String msg) {

//fill the code

super(msg);

}

}

**Driver.java**

package com.spring.main;

import java.util.\*;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.spring.exception.NoProfitException;

import com.spring.model.FitnessFranchise;

import com.spring.service.FitnessService;

public class Driver {

public static void main(String[] args) {

// TODO Auto-generated method stub

// fill the code

Scanner sc = new Scanner(System.in);

ApplicationContext applicationContext = new ClassPathXmlApplicationContext("beans.xml");

System.out.println("Pink Fitness Franchise 1 details");

System.out.println("Enter the total location:");

String location = sc.next();

System.out.println("Enter the total income:");

double totalIncome = sc.nextDouble();

System.out.println("Enter total expense amount for trainer salary:");

double trainerSalary = sc.nextDouble();

FitnessFranchise fitnessFranchise1 = (FitnessFranchise) applicationContext.getBean("fitnessFranchise");

fitnessFranchise1.setLocation(location);

fitnessFranchise1.setTotalIncome(totalIncome);

fitnessFranchise1.setTrainerSalary(trainerSalary);

System.out.println("Pink Fitness Franchise 2 details");

System.out.println("Enter the total location:");

String location2 = sc.next();

System.out.println("Enter the total income:");

double totalIncome2 = sc.nextDouble();

System.out.println("Enter total expense amount for trainer salary:");

double trainerSalary2 = sc.nextDouble();

FitnessFranchise fitnessFranchise2 = (FitnessFranchise) applicationContext.getBean("fitnessFranchise");

fitnessFranchise2.setLocation(location2);

fitnessFranchise2.setTotalIncome(totalIncome2);

fitnessFranchise2.setTrainerSalary(trainerSalary2);

FitnessService fitnessService = (FitnessService) applicationContext.getBean("fitnessService");

try {

fitnessService.calculateNetProfit(fitnessFranchise1);

System.out.println("Pink Fitness at " + fitnessFranchise1.getLocation() + " franchise Amount is RS:"

+ fitnessFranchise1.getFranchiseAmount());

} catch (NoProfitException e) {

System.out.println(e.getMessage());

}

try {

fitnessService.calculateNetProfit(fitnessFranchise2);

System.out.println("Pink Fitness at " + fitnessFranchise2.getLocation() + " franchise Amount is RS:"

+ fitnessFranchise2.getFranchiseAmount());

} catch (NoProfitException e) {

System.out.println(e.getMessage());

}

}

}

**FitnessFranchise.java**

package com.spring.model;

import java.util.Map;

// implement the FitnessFranchisor interface

public class FitnessFranchise implements FitnessFranchisor {

private String location;

private double totalIncome;

private double trainerSalary;

private double franchiseAmount;

private double percentage;

private Map<String, Double> commonExpenses;

public FitnessFranchise(double percentage, Map<String, Double> commonExpenses) {

super();

this.percentage = percentage;

this.commonExpenses = commonExpenses;

}

public void setPercentage(double percentage) {

this.percentage = percentage;

}

public double getPercentage() {

return percentage;

}

public double getFranchiseAmount() {

return franchiseAmount;

}

public Map<String, Double> getCommonExpenses() {

return commonExpenses;

}

public void setCommonExpenses(Map<String, Double> commonExpenses) {

this.commonExpenses = commonExpenses;

}

public void setFranchiseAmount(double franchiseAmount) {

this.franchiseAmount = franchiseAmount;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

public double getTotalIncome() {

return totalIncome;

}

public void setTotalIncome(double totalIncome) {

this.totalIncome = totalIncome;

}

public double getTrainerSalary() {

return trainerSalary;

}

public void setTrainerSalary(double trainerSalary) {

this.trainerSalary = trainerSalary;

}

public void calculateFranchiseAmount(double netProfit) {

//Calculating Franchise Amount

this.franchiseAmount=netProfit\*(percentage)/100;

}

}

**FitnessFranchisor.java**

package com.spring.model;

public interface FitnessFranchisor {

public void calculateFranchiseAmount(double netProfit);

}

**FitnessService.java**

package com.spring.service;

import com.spring.bo.FitnessBO;

import com.spring.exception.NoProfitException;

import com.spring.model.FitnessFranchise;

public class FitnessService {

private FitnessBO fitnessBO;

public FitnessBO getFitnessBO() {

return fitnessBO;

}

public void setFitnessBO(FitnessBO fitnessBO) {

this.fitnessBO = fitnessBO;

}

public void calculateNetProfit(FitnessFranchise franchise) throws NoProfitException {

//fill the code

double netProfit=fitnessBO.calculateNetProfit(franchise);

if(netProfit<0)

{

throw new NoProfitException("No profit to calculate franchise amount");

}

else

{

franchise.calculateFranchiseAmount(netProfit);

}

}

public FitnessService(FitnessBO fitnessBO) {

super();

this.fitnessBO = fitnessBO;

}

}

**POM.XML**

<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>PinkFitness</groupId>

<artifactId>PinkFitness</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>xmlunit</groupId>

<artifactId>xmlunit</artifactId>

<version>1.6</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/com.github.stefanbirkner/system-rules -->

<dependency>

<groupId>com.github.stefanbirkner</groupId>

<artifactId>system-rules</artifactId>

<version>1.16.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.xml.bind</groupId>

<artifactId>jaxb-impl</artifactId>

<version>2.1.13</version>

</dependency>

<!-- Thanks for using https://jar-download.com -->

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->

<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

<version>2.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

</plugins>

</build>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

</project>