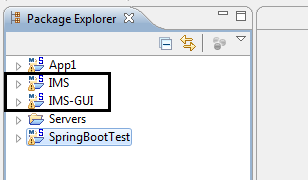
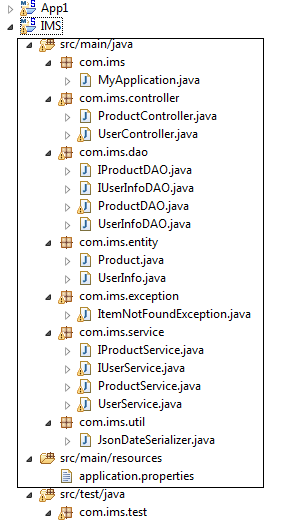
**IMS – Spring Boot Case Study**

It contains 2 projects namely IMS and IMS-GUI



IMS is a RESTful web service project. IMS-GUI is the presentation tier which invokes the restful services in IMS.

**IMS – The RESTful Service Project**

****

**MyApplication.java**

package com.ims;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class MyApplication {

public static void main(String[] args) {

SpringApplication.run(MyApplication.class, args);

}

}

**ProductController.java**

**package** com.ims.controller;

**import** java.util.List;

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

**import** org.springframework.http.HttpHeaders;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.DeleteMapping;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.PutMapping;

**import** org.springframework.web.bind.annotation.RequestBody;

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

**import** org.springframework.web.util.UriComponentsBuilder;

**import** com.ims.entity.Product;

**import** com.ims.exception.ItemNotFoundException;

**import** com.ims.service.IProductService;

@Controller

@RequestMapping("/inventory")

**publicclass**ProductController {

@Autowired

**private** IProductService productService;

@GetMapping("/getallproducts")

**public** ResponseEntity<List<Product>> getAllProducts() {

List<Product> list = productService.getAllProducts();

**returnnew** ResponseEntity<List<Product>>(list, HttpStatus.*OK*);

}

@GetMapping("/insertsampleproducts")

**public** ResponseEntity<Void> insertSampleProducts() {

productService.insertSampleProducts();

**returnnew** ResponseEntity<Void>(HttpStatus.*NO\_CONTENT*);

}

@PutMapping("updateexpiry/{ids}")

**public** ResponseEntity<Void> updateExpiryDate(@PathVariable String ids) {

productService.updateExpiryDate(ids);

**returnnew** ResponseEntity<Void>(HttpStatus.*NO\_CONTENT*);

}

@DeleteMapping("deleteexpiry")

**public** ResponseEntity<Void> deleteExpiredItems() {

productService.deleteExpiredItems();

**returnnew** ResponseEntity<Void>(HttpStatus.*NO\_CONTENT*);

}

@GetMapping("/getallproductssorted")

**public** ResponseEntity<List<Product>> getAllProductsSorted() {

List<Product> list = productService.getAllProductsSorted();

**returnnew** ResponseEntity<List<Product>>(list, HttpStatus.*OK*);

}

@PutMapping("applydiscounts")

**public** ResponseEntity<Void> applyDiscounts() {

productService.applyDiscounts();

**returnnew** ResponseEntity<Void>(HttpStatus.*NO\_CONTENT*);

}

@GetMapping("/getallproductsbydesc/{description}")

**public** ResponseEntity<List<Product>> getAllProductsByDescription(@PathVariable("description") String description) **throws** ItemNotFoundException {

List<Product> list = productService.getAllProductsByDesription(description);

**returnnew** ResponseEntity<List<Product>>(list, HttpStatus.*OK*);

}

@PostMapping("/addproduct")

**public** ResponseEntity<Void> addProduct(@RequestBody Product product, UriComponentsBuilder builder) {

**boolean** flag = productService.addProduct(product);

**if** (flag == **false**) {

**returnnew** ResponseEntity<Void>(HttpStatus.*CONFLICT*);

}

HttpHeaders headers = **new** HttpHeaders();

headers.setLocation(builder.path("/product/{id}").buildAndExpand(product.getProductId()).toUri());

**returnnew** ResponseEntity<Void>(headers, HttpStatus.*CREATED*);

}

}

**UserController.java**

**package** com.ims.controller;

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

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

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

**import** com.ims.entity.UserInfo;

**import** com.ims.service.IUserService;

@Controller

@RequestMapping("user")

**publicclass** UserController {

@Autowired

**private** IUserService userService;

@GetMapping("getactiveuser/{username}")

**public** ResponseEntity<UserInfo> getActiveUser(@PathVariable("username") String userName) {

UserInfo userInfo = userService.getActiveUser(userName);

**returnnew** ResponseEntity<UserInfo>(userInfo, HttpStatus.*OK*);

}

}

**IProductDAO.java**

**package** com.ims.dao;

**import** java.util.List;

**import** com.ims.entity.Product;

**import** com.ims.exception.ItemNotFoundException;

**publicinterface** IProductDAO {

List<Product> getAllProducts();

**void** insertSampleProducts();

**void** updateExpiryDate(String productId);

**void** deleteExpiredItems();

List<Product> getAllProductsSorted();

**void** applyDiscounts();

List<Product> getAllProductsByDesription(String description) **throws** ItemNotFoundException;

**void** addProduct(Product product);

Product getProductById(**int** articleId);

**boolean** productExists(String description, Float weight);

}

**IUserInfoDAO.java**

**package** com.ims.dao;

**import** com.ims.entity.UserInfo;

**publicinterface**IUserInfoDAO {

UserInfo getActiveUser(String userName);

}

**ProductDAO.java**

**package** com.ims.dao;

**import** java.text.ParseException;

**import** java.text.SimpleDateFormat;

**import** java.util.Calendar;

**import** java.util.Date;

**import** java.util.GregorianCalendar;

**import** java.util.List;

**import** javax.persistence.EntityManager;

**import** javax.persistence.PersistenceContext;

**import** javax.persistence.Query;

**import** org.springframework.stereotype.Repository;

**import** org.springframework.transaction.annotation.Transactional;

**import** com.ims.entity.Product;

**import** com.ims.exception.ItemNotFoundException;

@Transactional

@Repository

**publicclass** ProductDAO **implements** IProductDAO {

@PersistenceContext

**private** EntityManager entityManager;

@SuppressWarnings("unchecked")

**public** List<Product> getAllProducts() {

String hql = "FROM Product as pdt ORDER BY pdt.id";

List<Product> productsFromDb = (List<Product>) entityManager.createQuery(hql).getResultList();

**return** productsFromDb;

}

@SuppressWarnings("unchecked")

**publicvoid** insertSampleProducts() {

Date mfgDate = **null**;

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

**try** {

mfgDate = sdf.parse("2017-01-01");

} **catch** (ParseException e) {

e.printStackTrace();

}

entityManager.persist(**new** Product("Cheese", 10.25F, 100.25F,mfgDate,6));

entityManager.flush();

**try** {

mfgDate = sdf.parse("2016-02-02");

} **catch** (ParseException e) {

e.printStackTrace();

}

entityManager.persist(**new** Product("Milk", 11.25F, 101.25F,mfgDate,7));

entityManager.flush();

**try** {

mfgDate = sdf.parse("2017-03-03");

} **catch** (ParseException e) {

e.printStackTrace();

}

entityManager.persist(**new** Product("Wheat", 12.25F, 110.25F,mfgDate,8));

entityManager.flush();

}

@SuppressWarnings("unchecked")

**publicvoid** updateExpiryDate(String productId) {

**int** productIdInt = Integer.*parseInt*(productId);

Product productFromDb = getProductById(productIdInt);

Date expiryDate = productFromDb.getExpiryDate();

**if**(expiryDate == **null**) {

Date mfgDate = productFromDb.getMfgDate();

**int** useBeforeMonths = productFromDb.getUseBeforeMonths();

Calendar mfgCalendar = **new** GregorianCalendar();

mfgCalendar.setTime(mfgDate);

mfgCalendar.add(GregorianCalendar.*MONTH*, useBeforeMonths);

expiryDate = mfgCalendar.getTime();

productFromDb.setExpiryDate(expiryDate);

entityManager.flush();

}

}

@SuppressWarnings("unchecked")

**publicvoid** deleteExpiredItems() {

Date currentDate = **new** Date();

Date expiryDate = **null**;

**int** expiredProductId = 0;

List<Product> products = getAllProducts();

**for**(Product product : products) {

expiryDate = product.getExpiryDate();

**if**(currentDate.after(expiryDate)) {

expiredProductId = product.getProductId();

entityManager.remove(getProductById(expiredProductId));

}

}

}

@SuppressWarnings("unchecked")

**public** List<Product> getAllProductsSorted() {

String hql = "FROM Product as pdt ORDER BY pdt.expiryDate desc";

**return** (List<Product>) entityManager.createQuery(hql).getResultList();

}

@SuppressWarnings("unchecked")

**publicvoid** applyDiscounts() {

Date currentDate = **new** Date();

Date expiryDate = **null**;

Calendar currentCalendar = **new** GregorianCalendar();

currentCalendar.setTime(currentDate);

currentCalendar.add(GregorianCalendar.*MONTH*, 6);

Date dateAfter6Months = currentCalendar.getTime();

Float price = 0F;

Product productFromDb = **null**;

List<Product> products = getAllProducts();

**for**(Product product : products) {

expiryDate = product.getExpiryDate();

**if**(expiryDate.after(currentDate) && expiryDate.before(dateAfter6Months)) {

price = product.getPrice();

productFromDb = getProductById(product.getProductId());

price = price \* .8F;

productFromDb.setPrice(price);

entityManager.flush();

}

}

}

@SuppressWarnings("unchecked")

**public** List<Product> getAllProductsByDesription(String searchParam) **throws** ItemNotFoundException {

String hql = "FROM Product as pdt where pdt.description like :description";

Query query = entityManager.createQuery(hql);

List<Product> productsWithTheDescription = query.setParameter("description", "%" + searchParam + "%").getResultList();

**if**(productsWithTheDescription == **null** || productsWithTheDescription.size() == 0) {

**thrownew** ItemNotFoundException("No Items found!!");

}

**return** productsWithTheDescription;

}

**publicvoid** addProduct(Product product) {

entityManager.persist(product);

}

**public** Product getProductById(**int** productId) {

**return**entityManager.find(Product.**class**, productId);

}

**publicboolean** productExists(String description, Float weight) {

String hql = "FROM Product as pdt WHERE pdt.description = ? and pdt.weight = ?";

**int** count = entityManager.createQuery(hql).setParameter(1, description)

.setParameter(2, weight).getResultList().size();

**return** count > 0 ? **true** : **false**;

}

}

**UserInfoDAO.java**

**package** com.ims.dao;

**import** java.util.List;

**import** javax.persistence.EntityManager;

**import** javax.persistence.PersistenceContext;

**import** org.springframework.stereotype.Repository;

**import** org.springframework.transaction.annotation.Transactional;

**import** com.ims.entity.UserInfo;

@Repository

@Transactional

**publicclass**UserInfoDAO**implements** IUserInfoDAO {

@PersistenceContext

**private** EntityManager entityManager;

**public** UserInfo getActiveUser(String userName) {

UserInfo activeUserInfo = **new** UserInfo();

**short** enabled = 1;

List<?> list = entityManager.createQuery("SELECT u FROM UserInfo u WHERE userName=? and enabled=?")

.setParameter(1, userName).setParameter(2, enabled).getResultList();

**if**(!list.isEmpty()) {

activeUserInfo = (UserInfo)list.get(0);

}

**return** activeUserInfo;

}

}

**Product.java**

**package** com.ims.entity;

**import** java.io.Serializable;

**import** java.util.Date;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** com.fasterxml.jackson.databind.annotation.JsonSerialize;

**import** com.ims.util.JsonDateSerializer;

@Entity

@Table(name="products")

**publicclass**Product**implements** Serializable {

**privatestaticfinallong***serialVersionUID* = 1L;

@Id

@GeneratedValue(strategy=GenerationType.*AUTO*)

@Column(name="id")

**privateint**productId;

@Column(name="description")

**private** String description;

@Column(name="weight")

**private** Float weight;

@Column(name="price")

**private** Float price;

@Column(name="mfg\_date")

**private** Date mfgDate;

@Column(name="use\_before\_months")

**privateint**useBeforeMonths;

@Column(name="expiry\_date")

**private** Date expiryDate;

**public** Product() {

}

**public** Product(String description,Float weight,Float price,Date mfgDate,**int** useBeforeMonths) {

**this**.description = description;

**this**.weight = weight;

**this**.price = price;

**this**.mfgDate = mfgDate;

**this**.useBeforeMonths = useBeforeMonths;

}

**publicint** getProductId() {

**return**productId;

}

**publicvoid** setProductId(**int** productId) {

**this**.productId = productId;

}

**public** String getDescription() {

**return**description;

}

**publicvoid** setDescription(String description) {

**this**.description = description;

}

**public** Float getWeight() {

**return**weight;

}

**publicvoid** setWeight(Float weight) {

**this**.weight = weight;

}

**public** Float getPrice() {

**return**price;

}

**publicvoid** setPrice(Float price) {

**this**.price = price;

}

@JsonSerialize(using=JsonDateSerializer.**class**)

**public** Date getMfgDate() {

**return**mfgDate;

}

**publicvoid** setMfgDate(Date mfgDate) {

**this**.mfgDate = mfgDate;

}

**publicint** getUseBeforeMonths() {

**return**useBeforeMonths;

}

**publicvoid** setUseBeforeMonths(**int** useBeforeMonths) {

**this**.useBeforeMonths = useBeforeMonths;

}

@JsonSerialize(using=JsonDateSerializer.**class**)

**public** Date getExpiryDate() {

**return**expiryDate;

}

**publicvoid** setExpiryDate(Date expiryDate) {

**this**.expiryDate = expiryDate;

}

}

**UserInfo.java**

**package** com.ims.entity;

**import** java.io.Serializable;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

@Entity

@Table(name="users")

**publicclass**UserInfo**implements** Serializable {

**privatestaticfinallong***serialVersionUID* = 1L;

@Id

@Column(name="username")

**private** String userName;

@Column(name="password")

**private** String password;

@Column(name="role")

**private** String role;

@Column(name="full\_name")

**private** String fullName;

@Column(name="country")

**private** String country;

@Column(name="enabled")

**privateshort**enabled;

**public** String getUserName() {

**return**userName;

}

**publicvoid** setUserName(String userName) {

**this**.userName = userName;

}

**public** String getPassword() {

**return**password;

}

**publicvoid** setPassword(String password) {

**this**.password = password;

}

**public** String getRole() {

**return**role;

}

**publicvoid** setRole(String role) {

**this**.role = role;

}

**public** String getFullName() {

**return**fullName;

}

**publicvoid** setFullName(String fullName) {

**this**.fullName = fullName;

}

**public** String getCountry() {

**return**country;

}

**publicvoid** setCountry(String country) {

**this**.country = country;

}

**publicshort** getEnabled() {

**return**enabled;

}

**publicvoid** setEnabled(**short** enabled) {

**this**.enabled = enabled;

}

}

**ItemNotFoundException.java**

**package** com.ims.exception;

**publicclass** ItemNotFoundException **extends** Exception {

**privatestaticfinallong***serialVersionUID* = 1L;

**public** ItemNotFoundException(String message) {

**super**(message);

}

}

**IproductService.java**

**package** com.ims.service;

**import** java.util.List;

**import** com.ims.entity.Product;

**import** com.ims.exception.ItemNotFoundException;

**publicinterface**IProductService {

List<Product> getAllProducts();

**void** insertSampleProducts();

**void** updateExpiryDate(String ids);

**void** deleteExpiredItems();

List<Product> getAllProductsSorted();

**void** applyDiscounts();

List<Product> getAllProductsByDesription(String description) **throws** ItemNotFoundException;

**boolean** addProduct(Product product);

Product getArticleById(**int** productId);

}

**IuserService.java**

**package** com.ims.service;

**import** com.ims.entity.UserInfo;

**publicinterface**IUserService {

UserInfo getActiveUser(String userName);

}

**ProductService.java**

**package** com.ims.service;

**import** java.util.List;

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

**import** org.springframework.stereotype.Service;

**import** com.ims.dao.IProductDAO;

**import** com.ims.entity.Product;

**import** com.ims.exception.ItemNotFoundException;

@Service

**publicclass**ProductService**implements** IProductService {

@Autowired

**private** IProductDAO productDAO;

**public** List<Product> getAllProducts(){

**return**productDAO.getAllProducts();

}

**publicvoid**insertSampleProducts(){

productDAO.insertSampleProducts();

}

**publicvoid** updateExpiryDate(String ids) {

String[] productIds = ids.split("-");

String productId = **null**;

Product updatedProduct = **null**;

**for**(**int** i=0; i<productIds.length;i++) {

productId = productIds[i];

productDAO.updateExpiryDate(productId);

}

}

**publicvoid** deleteExpiredItems() {

productDAO.deleteExpiredItems();

}

**public** List<Product> getAllProductsSorted(){

**return**productDAO.getAllProductsSorted();

}

**publicvoid** applyDiscounts() {

productDAO.applyDiscounts();

}

**public** List<Product> getAllProductsByDesription(String description) **throws** ItemNotFoundException {

**return**productDAO.getAllProductsByDesription(description);

}

**publicboolean** addProduct(Product product) {

**if** (productDAO.productExists(product.getDescription(), product.getWeight())) {

**returnfalse**;

}

**else** {

productDAO.addProduct(product);

**returntrue**;

}

}

**public** Product getArticleById(**int** articleId) {

Product obj = productDAO.getProductById(articleId);

**return** obj;

}

}

**UserService.java**

**package** com.ims.service;

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

**import** org.springframework.stereotype.Service;

**import** com.ims.dao.IUserInfoDAO;

**import** com.ims.entity.UserInfo;

@Service

**publicclass**UserService**implements** IUserService {

@Autowired

**private** IUserInfoDAO userDAO;

**public** UserInfo getActiveUser(String userName){

**return**userDAO.getActiveUser(userName);

}

}

**JsonDateSerializer.java**

**package** com.ims.util;

**import** java.io.IOException;

**import** java.text.SimpleDateFormat;

**import** java.util.Date;

**import** org.springframework.stereotype.Component;

**import** com.fasterxml.jackson.core.JsonGenerator;

**import** com.fasterxml.jackson.core.JsonProcessingException;

**import** com.fasterxml.jackson.databind.JsonSerializer;

**import** com.fasterxml.jackson.databind.SerializerProvider;

@Component

**publicclass**JsonDateSerializer**extends** JsonSerializer<Date>{

**privatestaticfinal** SimpleDateFormat *dateFormat* = **new** SimpleDateFormat("yyyy-MM-dd");

@Override

**publicvoid** serialize(Date date, JsonGenerator gen, SerializerProvider provider)

**throws** IOException, JsonProcessingException {

String formattedDate = *dateFormat*.format(date);

gen.writeString(formattedDate);

}

}

**application.properties**

#spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ims

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.tomcat.max-wait=20000

spring.datasource.tomcat.max-active=50

spring.datasource.tomcat.max-idle=20

spring.datasource.tomcat.min-idle=15

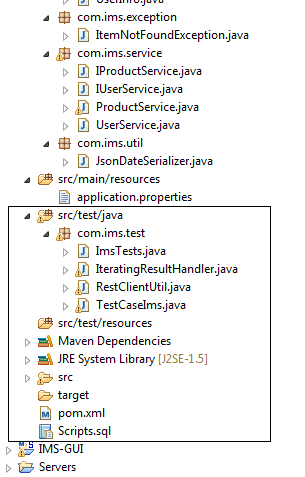
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect

spring.jpa.properties.hibernate.id.new\_generator\_mappings = false

spring.jpa.properties.hibernate.format\_sql = true

logging.level.org.hibernate.SQL=DEBUG

logging.level.org.hibernate.type.descriptor.sql.BasicBinder=TRACE



**ImsTests.java**

**package** com.ims.test;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** org.springframework.test.context.junit4.SpringRunner;

@RunWith(SpringRunner.**class**)

@SpringBootTest

**publicclass**ImsTests {

@Test

**publicvoid** contextLoads() {

}

}

**IteratingResultHandler.java**

**package** com.ims.test;

**import** org.springframework.stereotype.Component;

**import** org.springframework.test.web.servlet.MvcResult;

**import** org.springframework.test.web.servlet.ResultHandler;

@Component

**publicclass** IteratingResultHandler **implements** ResultHandler{

**publicvoid** handle(MvcResult result) **throws** Exception {

System.*out*.println("Response: " +result.getResponse().getContentAsString());

}

}

**RestClientUtil.java** – It is not developed.

**TestCaseIms.java**

**importstatic** org.springframework.test.web.servlet.request.MockMvcRequestBuilders.*get*;

**import** org.junit.Before;

**import** org.junit.Test;

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

**import** org.springframework.test.web.servlet.MockMvc;

**import** org.springframework.test.web.servlet.ResultHandler;

**import** org.springframework.test.web.servlet.setup.MockMvcBuilders;

**import** org.springframework.web.context.WebApplicationContext;

**publicclass**TestCaseIms**extends** ImsTests {

@Autowired

**private** WebApplicationContext webApplicationContext;

**private** MockMvc mockMvc;

@Autowired

**private** ResultHandler resultHandler;

@Before

**publicvoid** setup() {

mockMvc = MockMvcBuilders.*webAppContextSetup*(webApplicationContext).build();

}

@Test

**publicvoid** testGetAllProducts() **throws** Exception {

mockMvc.perform(*get*("/inventory/getallproducts")).andDo(resultHandler);

}

}

**pom.xml**

<?xmlversion=*"1.0"*encoding=*"UTF-8"*?>

<projectxmlns=*"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.concretepage</groupId>

<artifactId>spring-boot-demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-demo</name>

<description>Spring Boot Demo Project</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.2.RELEASE</version>

</parent>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<optional>true</optional>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**Scripts.sql**

**CREATE TABLE** `products` (

`id` **INTNOTNULL** AUTO\_INCREMENT ,

`description` **VARCHAR**(50) **NULL** ,

`weight` **DECIMAL**(5,2) **NULL** ,

`price` **DECIMAL**(5,2) **NULL** ,

`mfg\_date` **DATENULL** ,

`use\_before\_months` **INTNULL** ,

`expiry\_date` **DATENULL** ,

**PRIMARY KEY** (`id`) );

**CREATETABLE** users (

username **varchar**(50) **primary key**,

password **varchar**(100),

role **varchar**(50),

full\_name **varchar**(50),

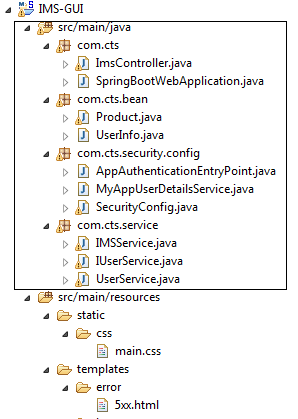
country **varchar**(50),

enabled **integer**);

**INSERTINTO** users **VALUES** ("james", "pwd123","ROLE\_USER","James Bond","UK",1);

**INSERTINTO** users **VALUES** ("daniel", "pwd124","ROLE\_ADMIN","Daniel Craig","UK",1);

**IMS-GUI – The Web Tier Project (Using Mustache)**

****

**ImsController.java**

**package** com.cts;

**import** java.util.List;

**import** java.util.Map;

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

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

**import** org.springframework.security.access.annotation.Secured;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.GetMapping;

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

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.PostMapping;

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

**import** com.cts.bean.Product;

**import** com.cts.service.IMSService;

@Controller

**publicclass**ImsController {

// inject via application.properties

@Value("${app.welcome.message}")

**private** String MESSAGE = "";

@Value("${app.welcome.title}")

**private** String TITLE = "";

@Autowired

**private** IMSService imsService;

@GetMapping("/")

**public** String welcome(Map<String, Object> model) {

model.put("title", TITLE);

model.put("message", MESSAGE);

**return**"welcome";

}

@GetMapping("/403")

**public** String accessDenied(Map<String, Object> model) {

model.put("title", TITLE);

model.put("message", "You are not authorised to view this page");

**return**"accessDenied";

}

@Secured ({"ROLE\_ADMIN", "ROLE\_USER"})

@GetMapping("/display/getallproducts")

**public** String displayAllProducts(Map<String, Object> model) {

List<Product> products = imsService.getAllProducts();

model.put("title", TITLE);

model.put("message", "Products in inventory");

model.put("products", products);

**return**"products";

}

@GetMapping("/display/calculateexpiry/{ids}")

**public** String calculateExpiryAndDisplay(Map<String, Object> model, @PathVariable("ids") String ids) {

imsService.calculateExpiry(ids);

List<Product> products = imsService.getAllProducts();

model.put("title", TITLE);

model.put("message", "Products in inventory");

model.put("products", products);

**return**"products";

}

@GetMapping("/display/deleteexpiry")

**public** String deleteExpiredProductsAndDisplay(Map<String, Object> model) {

imsService.deleteExpiredProducts();

List<Product> products = imsService.getAllProducts();

model.put("title", TITLE);

model.put("message", "Products in inventory");

model.put("products", products);

**return**"products";

}

@GetMapping("/display/applydiscounts")

**public** String applyDiscountsAndDisplay(Map<String, Object> model) {

imsService.applyDiscounts();

List<Product> products = imsService.getAllProducts();

model.put("title", TITLE);

model.put("message", "Products in inventory");

model.put("products", products);

**return**"products";

}

@Secured ({"ROLE\_ADMIN"})

@GetMapping("/display/addproductpage")

**public** String displayAddProductScreen(Map<String, Object> model) {

model.put("title", TITLE);

model.put("message", "Add Product");

**return**"product";

}

@PostMapping("/display/addproduct")

**public** String displayAddProductScreen(@ModelAttribute("productForm") Product product, Map<String, Object> model) {

imsService.addProduct(product);

List<Product> products = imsService.getAllProducts();

model.put("title", TITLE);

model.put("message", "Products in inventory");

model.put("products", products);

**return**"products";

}

@GetMapping("/display/searchproductpage")

**public** String displaySearchProductScreen(Map<String, Object> model) {

model.put("title", TITLE);

model.put("message", "Search Product");

**return**"search";

}

@GetMapping("/display/searchproducts/{searchparam}")

**public** String searchProductsAndDisplay(Map<String, Object> model, @PathVariable("searchparam") String searchParam) {

List<Product> products = imsService.searchProductsAndDisplay(searchParam);

model.put("title", TITLE);

model.put("message", "Products that contains the word \""+searchParam+"\"");

model.put("searchSortResult", "true");

model.put("products", products);

**return**"products";

}

@GetMapping("/display/sortproducts")

**public** String sortProductsByExpiryAndDisplay(Map<String, Object> model) {

List<Product> products = imsService.sortProductsByExpiryAndDisplay();

model.put("title", TITLE);

model.put("message", "Products in descending order of Expiry Date");

model.put("searchSortResult", "true");

model.put("products", products);

**return**"products";

}

// test 5xx errors

@RequestMapping("/5xx")

**public** String ServiceUnavailable() {

**thrownew** RuntimeException("ABC");

}

}

**SpringBootWebApplication.java**

**package** com.cts;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**publicclass**SpringBootWebApplication {

**publicstaticvoid** main(String[] args) **throws** Exception {

SpringApplication.*run*(SpringBootWebApplication.**class**, args);

}

}

**Product.java**

**package** com.cts.bean;

**import** java.io.Serializable;

**publicclass** Product **implements** Serializable {

**privatestaticfinallong***serialVersionUID* = 1L;

**privateint**productId;

**private** String description;

**private** Float weight;

**private** Float price;

**private** String mfgDate;

**privateint**useBeforeMonths;

**private** String expiryDate;

**public** Product() {

}

**publicint** getProductId() {

**return**productId;

}

**publicvoid** setProductId(**int** productId) {

**this**.productId = productId;

}

**public** String getDescription() {

**return**description;

}

**publicvoid** setDescription(String description) {

**this**.description = description;

}

**public** Float getWeight() {

**return**weight;

}

**publicvoid** setWeight(Float weight) {

**this**.weight = weight;

}

**public** Float getPrice() {

**return**price;

}

**publicvoid** setPrice(Float price) {

**this**.price = price;

}

**public** String getMfgDate() {

**return**mfgDate;

}

**publicvoid** setMfgDate(String mfgDate) {

**this**.mfgDate = mfgDate;

}

**publicint** getUseBeforeMonths() {

**return**useBeforeMonths;

}

**publicvoid** setUseBeforeMonths(**int** useBeforeMonths) {

**this**.useBeforeMonths = useBeforeMonths;

}

**public** String getExpiryDate() {

**if**(expiryDate == **null**) {

expiryDate = "";

}

**return**expiryDate;

}

**publicvoid** setExpiryDate(String expiryDate) {

**this**.expiryDate = expiryDate;

}

}

**UserInfo.java**

**package** com.cts.bean;

**import** java.io.Serializable;

**publicclass**UserInfo**implements** Serializable {

**privatestaticfinallong***serialVersionUID* = 1L;

**private** String userName;

**private** String password;

**private** String role;

**private** String fullName;

**private** String country;

**privateshort**enabled;

**public** String getUserName() {

**return**userName;

}

**publicvoid** setUserName(String userName) {

**this**.userName = userName;

}

**public** String getPassword() {

**return**password;

}

**publicvoid** setPassword(String password) {

**this**.password = password;

}

**public** String getRole() {

**return**role;

}

**publicvoid** setRole(String role) {

**this**.role = role;

}

**public** String getFullName() {

**return**fullName;

}

**publicvoid** setFullName(String fullName) {

**this**.fullName = fullName;

}

**public** String getCountry() {

**return**country;

}

**publicvoid** setCountry(String country) {

**this**.country = country;

}

**publicshort** getEnabled() {

**return**enabled;

}

**publicvoid** setEnabled(**short** enabled) {

**this**.enabled = enabled;

}

}

**AppAuthenticationEntryPoint.java**

**package** com.cts.security.config;

**import** java.io.IOException;

**import** javax.servlet.ServletException;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** org.springframework.security.core.AuthenticationException;

**import** org.springframework.security.web.authentication.www.BasicAuthenticationEntryPoint;

**import** org.springframework.stereotype.Component;

@Component

**publicclass**AppAuthenticationEntryPoint**extends** BasicAuthenticationEntryPoint {

@Override

**publicvoid** commence(HttpServletRequest request, HttpServletResponse response,

AuthenticationException authException) **throws** IOException, ServletException {

response.addHeader("WWW-Authenticate", "Basic realm=\"" + getRealmName() + "\"");

response.sendError(HttpServletResponse.*SC\_UNAUTHORIZED*, authException.getMessage());

}

@Override

**publicvoid** afterPropertiesSet() **throws** Exception {

setRealmName("MY APP REALM");

}

}

**MyAppUserDetailsService.java**

**package** com.cts.security.config;

**import** java.util.Arrays;

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

**import** org.springframework.security.core.GrantedAuthority;

**import** org.springframework.security.core.authority.SimpleGrantedAuthority;

**import** org.springframework.security.core.userdetails.User;

**import** org.springframework.security.core.userdetails.UserDetails;

**import** org.springframework.security.core.userdetails.UserDetailsService;

**import** org.springframework.security.core.userdetails.UsernameNotFoundException;

**import** org.springframework.stereotype.Service;

**import** com.cts.bean.UserInfo;

**import** com.cts.service.IUserService;

@Service

**publicclass**MyAppUserDetailsService**implements** UserDetailsService {

@Autowired

**private** IUserService userService;

**public** UserDetails loadUserByUsername(String userName)

**throws** UsernameNotFoundException {

UserInfo activeUserInfo = userService.getActiveUser(userName);

System.*out*.println("activeUserInfo ="+activeUserInfo);

GrantedAuthority authority = **new** SimpleGrantedAuthority(activeUserInfo.getRole());

UserDetails userDetails = (UserDetails)**new** User(activeUserInfo.getUserName(),activeUserInfo.getPassword(), Arrays.*asList*(authority));

**return** userDetails;

}

}

**SecurityConfig.java**

**package** com.cts.security.config;

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

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

**import** org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;

**import** org.springframework.security.config.annotation.web.builders.HttpSecurity;

**import** org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

**import** org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

**import** org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration

@EnableWebSecurity

@EnableGlobalMethodSecurity(securedEnabled=**true**)

**publicclass** SecurityConfig **extends** WebSecurityConfigurerAdapter {

@Autowired

**private** MyAppUserDetailsService myAppUserDetailsService;

/\*@Autowired

String userDetails;\*/

@Autowired

**private** AppAuthenticationEntryPoint appAuthenticationEntryPoint;

@Override

**protectedvoid** configure(HttpSecurity http) **throws** Exception {

http.csrf().disable()

.authorizeRequests()

.antMatchers("/display/addproductpage").hasRole("ADMIN")

.antMatchers("/display/getallproducts").hasAnyRole("ADMIN","USER")

.and().httpBasic().realmName("MY APP REALM")

.authenticationEntryPoint(appAuthenticationEntryPoint);//.exceptionHandling()

http.exceptionHandling().accessDeniedPage("/403");

}

@Autowired

**publicvoid** configureGlobal(AuthenticationManagerBuilder auth) **throws** Exception {

BCryptPasswordEncoder passwordEncoder = **new** BCryptPasswordEncoder();

auth.userDetailsService(myAppUserDetailsService);

//auth.userDetailsService(myAppUserDetailsService).passwordEncoder(passwordEncoder);

//auth.inMemoryAuthentication().withUser("daniel").password("pwd124").roles("ADMIN");

}

}

**IMSService.java**

**package** com.cts.service;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.springframework.http.HttpEntity;

**import** org.springframework.http.HttpHeaders;

**import** org.springframework.http.HttpMethod;

**import** org.springframework.http.MediaType;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Service;

**import** org.springframework.web.client.RestTemplate;

**import** com.cts.bean.Product;

@Service

**publicclass**IMSService {

**public** List<Product> getAllProducts() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/getallproducts";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Product[]> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, Product[].**class**, 1);

Product[] products = responseEntity.getBody();

**return** Arrays.*asList*(products);

}

**publicvoid** calculateExpiry(String ids) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/updateexpiry/"+ids;

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*PUT*, requestEntity, Void.**class**, 1);

}

**publicvoid** deleteExpiredProducts() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/deleteexpiry/";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*DELETE*, requestEntity, Void.**class**, 1);

}

**publicvoid** applyDiscounts() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/applydiscounts";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*PUT*, requestEntity, Void.**class**, 1);

}

**publicvoid** addProduct(Product product) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/addproduct";

HttpEntity<Product> requestEntity = **new** HttpEntity<Product>(product,headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*POST*, requestEntity, Void.**class**, 1);

}

**public** List<Product> searchProductsAndDisplay(String searchParam) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/getallproductsbydesc/"+searchParam;

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Product[]> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, Product[].**class**, 1);

Product[] products = responseEntity.getBody();

**return** Arrays.*asList*(products);

}

**public** List<Product> sortProductsByExpiryAndDisplay() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/getallproductssorted";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Product[]> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, Product[].**class**, 1);

Product[] products = responseEntity.getBody();

**return** Arrays.*asList*(products);

}

}

**IuserService.java**

**package** com.cts.service;

**import** org.springframework.stereotype.Service;

**import** com.cts.bean.UserInfo;

@Service

**publicinterface**IUserService {

**public** UserInfo getActiveUser(String userName);

}

**UserService.java**

**package** com.cts.service;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.springframework.http.HttpEntity;

**import** org.springframework.http.HttpHeaders;

**import** org.springframework.http.HttpMethod;

**import** org.springframework.http.MediaType;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Service;

**import** org.springframework.web.client.RestTemplate;

**import** com.cts.bean.Product;

**import** com.cts.bean.UserInfo;

@Service

**publicclass**UserService**implements** IUserService {

**public** UserInfo getActiveUser(String userName) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/user/getactiveuser/"+userName;

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<UserInfo> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, UserInfo.**class**, 1);

UserInfo userInfo = responseEntity.getBody();

**return** userInfo;

}

**publicvoid** calculateExpiry(String ids) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/updateexpiry/"+ids;

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*PUT*, requestEntity, Void.**class**, 1);

}

**publicvoid** deleteExpiredProducts() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/deleteexpiry/";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*DELETE*, requestEntity, Void.**class**, 1);

}

**publicvoid** applyDiscounts() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/applydiscounts";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*PUT*, requestEntity, Void.**class**, 1);

}

**publicvoid** addProduct(Product product) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/addproduct";

HttpEntity<Product> requestEntity = **new** HttpEntity<Product>(product,headers);

ResponseEntity<Void>responseEntity = restTemplate.exchange(url, HttpMethod.*POST*, requestEntity, Void.**class**, 1);

}

**public** List<Product> searchProductsAndDisplay(String searchParam) {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/getallproductsbydesc/"+searchParam;

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

ResponseEntity<Product[]> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, Product[].**class**, 1);

Product[] products = responseEntity.getBody();

**return** Arrays.*asList*(products);

}

**public** List<Product> sortProductsByExpiryAndDisplay() {

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.*APPLICATION\_JSON*);

RestTemplate restTemplate = **new** RestTemplate();

String url = "http://localhost:8084/inventory/getallproductssorted";

HttpEntity<String> requestEntity = **new** HttpEntity<String>(headers);

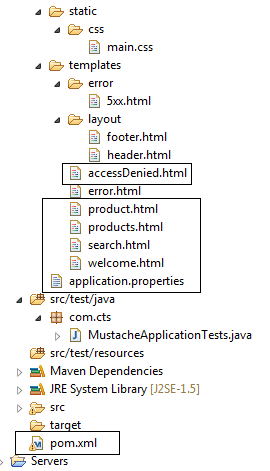
ResponseEntity<Product[]> responseEntity = restTemplate.exchange(url, HttpMethod.*GET*, requestEntity, Product[].**class**, 1);

Product[] products = responseEntity.getBody();

**return** Arrays.*asList*(products);

}

}



**accessDenied.html**

{{>layout/header}}

<body>

<navclass=*"navbar navbar-inverse"*>

<divclass=*"container"*>

<divclass=*"navbar-header"*>

<aclass=*"navbar-brand"*href=*"#"*>Access Denied</a>

</div>

</div>

</nav>

<divclass=*"container"*align=*"center"*>

<divclass=*"starter-template"*>

<h3>Inventory Management System</h3>

<h1>

{{message}}

</h1>

</div>

</div>

<!-- /.container -->

<formaction=*"/display/getallproducts"*>

<divalign=*"center"*>

<inputtype=*"submit"*value=*"Go To Products"*/>

</div>

</form>

<scripttype=*"text/javascript"*src=*"webjars/bootstrap/3.3.7/js/bootstrap.min.js"*></script>

</body>

{{>layout/footer}}

**product.html**

{{>layout/header}}

<script>

**function** submitForm(buttonName) {

**if**(buttonName == "addProduct") {

document.productForm.action = "/display/addproduct";

document.productForm.submit();

}

**if**(buttonName == "gotoProducts") {

document.productForm.method = "get";

document.productForm.action = "/display/getallproducts";

document.productForm.submit();

}

}

</script>

<body>

<navclass=*"navbar navbar-inverse"*>

<divclass=*"container"*>

<divclass=*"navbar-header"*>

<aclass=*"navbar-brand"*href=*"#"*>Inventory Management System</a>

</div>

</div>

</nav>

<formname=*"productForm"*method=*"post"*>

<divclass=*"container"*>

<divclass=*"starter-template"*>

<h1align=*"center"*>{{message}}</h1>

<tablealign=*"center"*>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*"descriptionId"*>Product Description</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;\*</td>

<td>

<inputtype=*"text"*name=*"description"*id=*"descriptionId"*/>

</td>

</tr>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*""*>Weight</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;\*</td>

<td>

<inputtype=*"text"*name=*"weight"*id=*"weightId"*/>

</td>

</tr>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*""*>Price</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;\*</td>

<td>

<inputtype=*"text"*name=*"price"*id=*"priceId"*/>

</td>

</tr>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*""*>Manufacturing Date(yyyy-MM-dd)</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;\*</td>

<td>

<inputtype=*"text"*name=*"mfgDate"*id=*"mfgDateId"*/>

</td>

</tr>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*""*>Use Before Months</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;\*</td>

<td>

<inputtype=*"text"*name=*"useBeforeMonths"*id=*"useBeforeMonthsId"*/>

</td>

</tr>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*""*>Expiry Date(yyyy-MM-dd)</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</td>

<td>

<inputtype=*"text"*name=*"expiryDate"*id=*"expiryDateId"*/>

</td>

</tr>

</table>

</div>

</div>

<br>

<divalign=*"center"*>

<inputtype=*"submit"*value=*"Go To Products"*onclick="submitForm('gotoProducts')"/>

<inputtype=*"submit"*value=*"Add Product"*onclick="submitForm('addProduct')"/>

</div>

</form>

<scripttype=*"text/javascript"*src=*"webjars/bootstrap/3.3.7/js/bootstrap.min.js"*></script>

</body>

{{>layout/footer}}

**products.html**

{{>layout/header}}

<script>

**function** submitForm(buttonName) {

**if**(buttonName == "back") {

document.productsForm.action = "/";

document.productsForm.submit();

}

**if**(buttonName == "calculateExpiry") {

**var** ids = "";

**for**(**var** index = 0;index < document.productsForm.products.length; index++) {

**if**(document.productsForm.products[index].checked == **true**) {

**if**(ids != "") {

ids = ids + "-" + document.productsForm.products[index].value;

}

**else** {

ids = document.productsForm.products[index].value;

}

}

}

document.productsForm.action = "/display/calculateexpiry/"+ids;

**if**(ids != "") {

document.productsForm.submit();

}

**else** {

**return**;

}

}

**if**(buttonName == "deleteExpiry") {

document.productsForm.action = "/display/deleteexpiry/";

document.productsForm.submit();

}

**if**(buttonName == "applyDiscounts") {

document.productsForm.action = "/display/applydiscounts/";

document.productsForm.submit();

}

**if**(buttonName == "sortProducts") {

document.productsForm.action = "/display/sortproducts/";

document.productsForm.submit();

}

**if**(buttonName == "addProductPage") {

document.productsForm.action = "/display/addproductpage/";

document.productsForm.submit();

}

**if**(buttonName == "searchProductPage") {

document.productsForm.action = "/display/searchproductpage/";

document.productsForm.submit();

}

**if**(buttonName == "gotoProducts") {

document.productsForm.action = "/display/getallproducts";

document.productsForm.submit();

}

}

</script>

<body>

<navclass=*"navbar navbar-inverse"*>

<divclass=*"container"*>

<divclass=*"navbar-header"*>

<aclass=*"navbar-brand"*href=*"#"*>Inventory Management System</a>

</div>

</div>

</nav>

<formname=*"productsForm"*>

<divclass=*"container"*>

<divclass=*"starter-template"*>

<h1align=*"center"*>{{message}}</h1>

<tableborder=*"1"*align=*"center"*>

<tralign=*"center"*>

<th>Select</th>

<th>Product Id</th>

<th>Description</th>

<th>Weight</th>

<th>Price</th>

<th>Mfg Date(yyyy-MM-dd)</th>

<th>Use Before Months</th>

<th>Expiry Date</th>

</tr>

{{#products}}

<tralign=*"center"*>

<td>

<inputtype=*"checkbox"*name=*"products"*value=*"{{productId}}"*/>

</td>

<td>{{productId}}</td>

<td>{{description}}</td>

<td>{{weight}}</td>

<td>{{price}}</td>

<td>{{mfgDate}}</td>

<td>{{useBeforeMonths}}</td>

<td>{{expiryDate}}</td>

</tr>

{{/products}}

</table>

</div>

</div>

<!-- /.container -->

<br>

<divalign=*"center"*>

{{#searchSortResult}}

<inputtype=*"submit"*value=*"Go To Products"*onclick="submitForm('gotoProducts')"/>

{{/searchSortResult}}

{{^searchSortResult}}

<inputtype=*"submit"*value=*"Back"*onclick="submitForm('back')"/>

<inputtype=*"button"*value=*"Calculate Expiry"*onclick="submitForm('calculateExpiry')"/>

<inputtype=*"submit"*value=*"Delete Expired Products"*onclick="submitForm('deleteExpiry')"/>

<inputtype=*"submit"*value=*"Apply Discounts"*onclick="submitForm('applyDiscounts')"/>

<inputtype=*"submit"*value=*"Sort Products by Expiry"*onclick="submitForm('sortProducts')"/>

<inputtype=*"submit"*value=*"Add Product"*onclick="submitForm('addProductPage')"/>

<inputtype=*"submit"*value=*"Search Product"*onclick="submitForm('searchProductPage')"/>

{{/searchSortResult}}

</div>

</form>

<scripttype=*"text/javascript"*src=*"webjars/bootstrap/3.3.7/js/bootstrap.min.js"*></script>

</body>

{{>layout/footer}}

**search.html**

{{>layout/header}}

<script>

**function** submitForm(buttonName) {

**if**(buttonName == "searchProduct") {

**var** searchParam = document.getElementById("searchParamId").value;

document.productForm.action = "/display/searchproducts/"+searchParam;

document.productForm.submit();

}

}

</script>

<body>

<navclass=*"navbar navbar-inverse"*>

<divclass=*"container"*>

<divclass=*"navbar-header"*>

<aclass=*"navbar-brand"*href=*"#"*>Inventory Management System</a>

</div>

</div>

</nav>

<formname=*"productForm"*method=*"get"*>

<divclass=*"container"*>

<divclass=*"starter-template"*>

<h1align=*"center"*>{{message}}</h1>

<tablealign=*"center"*>

<tralign=*"center"*>

<tdalign=*"right"*>

<labelfor=*"searchParamId"*>Search Parameter:</label>

</td>

<td>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</td>

<td>

<inputtype=*"text"*name=*"searchParam"*id=*"searchParamId"*/>

</td>

</tr>

</table>

</div>

</div>

<br>

<divalign=*"center"*>

<inputtype=*"submit"*value=*"Search Product"*onclick="submitForm('searchProduct')"/>

</div>

</form>

<scripttype=*"text/javascript"*src=*"webjars/bootstrap/3.3.7/js/bootstrap.min.js"*></script>

</body>

{{>layout/footer}}

**welcome.html**

{{>layout/header}}

<body>

<navclass=*"navbar navbar-inverse"*>

<divclass=*"container"*>

<divclass=*"navbar-header"*>

<aclass=*"navbar-brand"*href=*"#"*>Spring Boot</a>

</div>

</div>

</nav>

<divclass=*"container"*align=*"center"*>

<divclass=*"starter-template"*>

<h1>Inventory Management System</h1>

<h2>

{{message}}

</h2>

</div>

</div>

<!-- /.container -->

<formaction=*"/display/getallproducts"*>

<divalign=*"center"*>

<inputtype=*"submit"*value=*"Go To Products"*/>

</div>

</form>

<scripttype=*"text/javascript"*src=*"webjars/bootstrap/3.3.7/js/bootstrap.min.js"*></script>

</body>

{{>layout/footer}}

**application.properties**

app.welcome.message: Hello,WelcometoIMS!!!

app.welcome.title: InventoryManagementSystem

**pom.xml**

<?xmlversion=*"1.0"*encoding=*"UTF-8"*?>

<projectxmlns=*"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>

<artifactId>spring-boot-web-mustache</artifactId>

<packaging>jar</packaging>

<name>Spring Boot Web Mustache Example</name>

<description>Spring Boot Web Mustache Example</description>

<version>1.0</version>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.2.RELEASE</version>

</parent>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-mustache</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<!-- hot swapping, live reload -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-core</artifactId>

<version>4.0.1.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-crypto</artifactId>

<version>4.0.1.RELEASE</version>

</dependency>

<!-- Optional, for bootstrap -->

<dependency>

<groupId>org.webjars</groupId>

<artifactId>bootstrap</artifactId>

<version>3.3.7</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Package as an executable jar/war -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>