**Sporty Shoes**

**Source Code:**

**Pom.xml**

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

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

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

<version>2.4.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.mjava</groupId>

<artifactId>SportyShoesOnline</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>SportyShoesOnline</name>

<description>Demo project for Spring Boot</description>

<properties>

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

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

</dependency>

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

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

<version>8.0.16</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

**SportyShoesOnlineApp:**

package com.mjava.SportyShoesOnline;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

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

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

@SpringBootApplication

@ComponentScan("com.mjava")

@EntityScan("com.mjava.model")

@EnableJpaRepositories("com.mjava.dao")

public class SportyShoesOnlineApplication {

public static void main(String[] args) {

SpringApplication.run(SportyShoesOnlineApplication.class, args);

}

}

**OrderDao.java:**

package com.mjava.dao;

import java.sql.Date;

import java.util.List;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import com.mjava.model.OrderedShoeModel;

@Repository

public interface OrdersDao extends CrudRepository<OrderedShoeModel, Integer>{

@Query("from OrderedShoeModel where categeory=:categeoryId and date=:date")

public List<OrderedShoeModel> getRequiredCompleteTransactionsData(int categeoryId,Date date);

}

**ShoesDao.java:**

package com.mjava.dao;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.CrudRepository;

import org.springframework.data.repository.query.Param;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.mjava.model.ShoesDataModel;

import antlr.collections.List;

//@Repository

//public interface ShoesDao extends CrudRepository<ShoesDataModel, Integer> {

@Repository

public interface ShoesDao extends CrudRepository<ShoesDataModel, Integer> {

@Query("from ShoesDataModel where categeory=:categeory")

public Iterable<ShoesDataModel> findByCategory(int categeory);

@Query("from ShoesDataModel where id=:seletedShoeId")

public ShoesDataModel getshoesDataById(int seletedShoeId);

@Modifying

@Transactional

@Query("delete from ShoesDataModel where id=:id")

public int deleteProductwithId(@Param("id") int id);

@Modifying

@Transactional

@Query("update ShoesDataModel set categeory=:categeory, price=:price, name=:name, imagelink=:imagelink where id=:id")

public int updateShoeProduct(@Param("categeory") int categeory, @Param("price") double price,

@Param("name") String name, @Param("imagelink") String imagelink,

@Param("id") int id);

}

**UsersDao.java:**

package com.mjava.dao;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.CrudRepository;

import org.springframework.data.repository.query.Param;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.mjava.model.UserInfoModel;

@Repository

public interface UsersDao extends CrudRepository<UserInfoModel, Integer> {

@Query("from UserInfoModel where firstname=:username and password=:password")

public UserInfoModel isValidAdminUser(String username,String password );

@Query("from UserInfoModel where rollid=:rollid")

public Iterable<UserInfoModel> getUserDatabyRollid(int rollid);

@Query("from UserInfoModel where password=:password")

public UserInfoModel isCorrectPassword\_or\_Not(String password);

@Modifying

@Transactional

@Query("update UserInfoModel set password = :password where rollid=1 and firstname=:username")

public int updateAdminPassword(@Param("password") String password, @Param("username") String username);

}

**OrderesShoeModel.java:**

package com.mjava.model;

import java.sql.Date;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "tbl\_orderinfo")

public class OrderedShoeModel {

@Id

@GeneratedValue

private int orderid;

private int shoeid ;

private int categeory;

private double price;

private int quantity ;

private Date date;

private String name;

private String imagelink;

private double totalprice ;

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public int getOrderid() {

return orderid;

}

public void setOrderid(int orderid) {

this.orderid = orderid;

}

public int getShoeid() {

return shoeid;

}

public void setShoeid(int shoeid) {

this.shoeid = shoeid;

}

public int getCategeory() {

return categeory;

}

public void setCategeory(int categeory) {

this.categeory = categeory;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getImagelink() {

return imagelink;

}

public void setImagelink(String imagelink) {

this.imagelink = imagelink;

}

public double getTotalprice() {

return totalprice;

}

public void setTotalprice(double totalprice) {

this.totalprice = totalprice;

}

public Date getDate() {

return date;

}

public void setDate(Date date) {

this.date = date;

}

@Override

public String toString() {

return "OrderedShoeModel [orderid=" + orderid + ", shoeid=" + shoeid + ", categeory=" + categeory + ", price="

+ price + ", quantity=" + quantity + ", date=" + date + ", name=" + name + ", imagelink=" + imagelink

+ ", totalprice=" + totalprice + "]";

}

public OrderedShoeModel() {

super();

// TODO Auto-generated constructor stub

}

public OrderedShoeModel(int shoeid, int categeory, double price, int quantity, Date date, String name,

String imagelink, double totalprice) {

super();

this.shoeid = shoeid;

this.categeory = categeory;

this.price = price;

this.quantity = quantity;

this.date = date;

this.name = name;

this.imagelink = imagelink;

this.totalprice = totalprice;

}

}

**ShoeDataModel.java:**

package com.mjava.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "tbl\_shoesinfo")

public class ShoesDataModel {

@Id

@GeneratedValue

private int id;

private int categeory;

private double price;

private String name;

private String imagelink;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getCategeory() {

return categeory;

}

public void setCategeory(int categeory) {

this.categeory = categeory;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getImagelink() {

return imagelink;

}

public void setImagelink(String imagelink) {

this.imagelink = imagelink;

}

@Override

public String toString() {

return "ShoesDataModel [categeory=" + categeory + ", price=" + price + ", name=" + name + ", imagelink="

+ imagelink + "]";

}

public ShoesDataModel(int categeory, double price, String name, String imagelink) {

super();

this.categeory = categeory;

this.price = price;

this.name = name;

this.imagelink = imagelink;

}

public ShoesDataModel() {

super();

// TODO Auto-generated constructor stub}}

**UsersInfoModel.java:**

package com.mjava.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="tbl\_userinfo")

public class UserInfoModel {

@Id

@GeneratedValue

private int id;

private String firstname;

private String lastname;

private String password;

private int rollid;

private long phno;

private String email;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getFirstname() {

return firstname;

}

public void setFirstname(String firstname) {

this.firstname = firstname;

}

public String getLastname() {

return lastname;

}

public void setLastname(String lastname) {

this.lastname = lastname;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getRollid() {

return rollid;

}

public void setRollid(int rollid) {

this.rollid = rollid;

}

public long getPhno() {

return phno;

}

public void setPhno(long phno) {

this.phno = phno;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

@Override

public String toString() {

return "UserInoModel [firstname=" + firstname + ", lastname=" + lastname + ", password=" + password

+ ", rollid=" + rollid + ", phno=" + phno + ", email=" + email + "]";

}

public UserInfoModel() {

super();

// TODO Auto-generated constructor stub

}

public UserInfoModel(String firstname, String lastname, String password, int rollid, long phno, String email) {

super();

this.firstname = firstname;

this.lastname = lastname;

this.password = password;

this.rollid = rollid;

this.phno = phno;

this.email = email;

}

}

**OrderService.java:**

package com.mjava.service;

import java.sql.Date;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.mjava.dao.OrdersDao;

import com.mjava.model.OrderedShoeModel;

import com.mjava.model.ShoesDataModel;

@Service

public class OrdersService {

@Autowired

private OrdersDao orderDao;

public void insertBookingInfoService(int shoeid, String name, int categeory,

double price, String imglink, int quantity, double totalprice,Date date )

{

OrderedShoeModel orderedShoeModel =new OrderedShoeModel(shoeid, categeory, price, quantity, date, name, imglink, totalprice);

orderDao.save(orderedShoeModel);

}

public List<OrderedShoeModel> getCompleteTransactionsDataService() {

List<OrderedShoeModel> completeOrdersData= (List)orderDao.findAll();

return completeOrdersData;

}

public List<OrderedShoeModel> getRequiredCompleteTransactionsDataService(int categeoryId,Date date)

{

List<OrderedShoeModel> orderedShoeList= (List)orderDao.getRequiredCompleteTransactionsData(categeoryId, date);

return orderedShoeList;

}

}

**ShoesService.java:**

package com.mjava.service;

import java.sql.Date;

import java.util.List;

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

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import org.springframework.stereotype.Service;

import com.mjava.dao.OrdersDao;

import com.mjava.dao.ShoesDao;

import com.mjava.dao.UsersDao;

import com.mjava.model.OrderedShoeModel;

import com.mjava.model.ShoesDataModel;

import com.mjava.model.UserInfoModel;

@Service

public class ShoesService {

@Autowired

private ShoesDao shoesDao;

@Autowired

private OrdersDao orderDao;

@Autowired

private UsersDao userDao;

public List<ShoesDataModel> getMensShoeData() {

List<ShoesDataModel> mensData= (List)shoesDao.findByCategory(1);

System.out.println(mensData);

return mensData;

}

public List<ShoesDataModel> getWomensShoeData() {

List<ShoesDataModel> womensData= (List)shoesDao.findByCategory(2);

System.out.println(womensData);

return womensData;

}

public List<ShoesDataModel> getKidsShoeData() {

List<ShoesDataModel> kidsData= (List)shoesDao.findByCategory(3);

return kidsData;

}

public List<ShoesDataModel> getShoeDatabyCategeory(int categeoryid) {

List<ShoesDataModel> shoesDatabyCategeory= (List)shoesDao.findByCategory(categeoryid);

return shoesDatabyCategeory;

}

public ShoesDataModel getshoesDataByIdService(int seletedShoeId)

{

ShoesDataModel selectedShoesData= shoesDao.getshoesDataById(seletedShoeId);

System.out.println(selectedShoesData);

return selectedShoesData;

}

public boolean insertNewProductService(int categeory,double price,String name,String imagelink)

{

boolean isInsertedNewProduct = false;

try {

ShoesDataModel s=new ShoesDataModel(categeory,price, name, imagelink);

ShoesDataModel count= shoesDao.save(s);

isInsertedNewProduct = true;

return isInsertedNewProduct;

} catch (Exception e) {

System.out.println("Exception at insertNewProductService(int categeory,double price,String name,String imagelink) " + e.getMessage());

return false;

}

}

public boolean deleteProductwithIdService(int id) {

int count=0;

boolean isdeleted=false;

count= shoesDao.deleteProductwithId(id);

if(count>0)

{

isdeleted=true;

}

return isdeleted;

}

public boolean updateShoeProductService( int categeory, double price,

String name, String imagelink,

int id) {

int count=0;

boolean isUpdated=false;

count= shoesDao.updateShoeProduct(categeory, price, name, imagelink, id);

if(count>0)

{

isUpdated=true;

}

return isUpdated;

}

}

**UsersService.java:**

package com.mjava.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.mjava.dao.OrdersDao;

import com.mjava.dao.UsersDao;

import com.mjava.model.OrderedShoeModel;

import com.mjava.model.ShoesDataModel;

import com.mjava.model.UserInfoModel;

@Service

public class UsersService {

@Autowired

private UsersDao userDao;

public UserInfoModel isValidAdminUserService(String username, String password) {

UserInfoModel userInfoModel = userDao.isValidAdminUser(username, password);

return userInfoModel;

}

public List<UserInfoModel> getUserDataService() {

List<UserInfoModel> usersList= (List)userDao.getUserDatabyRollid(2);

return usersList;

}

public UserInfoModel isCorrectPassword\_or\_NotService(String password) {

UserInfoModel usersList= userDao.isCorrectPassword\_or\_Not(password);

return usersList;

}

public boolean updateAdminPasswordService(UserInfoModel userInfoModel, String currentpassword,

String newpassword, String confirmpassword) {

int count=0;

boolean isUpdated=false;

count= userDao.updateAdminPassword(confirmpassword,userInfoModel.getFirstname());

if(count>0)

{

isUpdated=true;

}

return isUpdated;

}

public void insertUserDataService(String firstname, String lastname, String password, String confirmpassword, int rollid, Long mobileno, String email)

{

UserInfoModel u =new UserInfoModel(firstname, lastname, confirmpassword, rollid, mobileno, email);

userDao.save(u);

}

}

**CommonController.java:**

package com.mjava.controller;

import java.sql.Date;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

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

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

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

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

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

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

import org.springframework.web.servlet.ModelAndView;

import com.mjava.model.OrderedShoeModel;

import com.mjava.model.ShoesDataModel;

import com.mjava.model.UserInfoModel;

import com.mjava.service.OrdersService;

import com.mjava.service.ShoesService;

import com.mjava.service.UsersService;

@Controller

@RequestMapping("/")

public class CommonController {

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

public String homeForm() {

return "index";

}

@Autowired

private ShoesService shoeservice;

@Autowired

private OrdersService orderservice;

@Autowired

private UsersService userservice;

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

public String menShoesForm(ModelMap model) {

List<ShoesDataModel> mensData = shoeservice.getMensShoeData();

model.put("menshoeData", mensData);

return "menShoesForm";

}

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

public String womenMethod(ModelMap model) {

List<ShoesDataModel> womensData = shoeservice.getWomensShoeData();

model.put("womenshoeData", womensData);

return "womenShoesForm";

}

@RequestMapping("/kidshoes")

public String kidMethod(ModelMap model) {

List<ShoesDataModel> kidsData = shoeservice.getKidsShoeData();

model.put("kidshoeData", kidsData);

return "kidsShoesForm";

}

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

public String signinMethod() {

return "signinForm";

}

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

public String sigUpMethod() {

return "signUpForm";

}

@RequestMapping("/selectedShoe")

public String selectedShoeInfotMethod(ModelMap model, @RequestParam int id) {

try {

ShoesDataModel selectedShoesData = shoeservice.getshoesDataByIdService(id);

model.put("selectedShoesData", selectedShoesData);

return "selectedShoeInfoForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.selectedShoeInfotMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "selectedShoeInfoForm";

}

}

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

public String orderInforMethod(ModelMap model, @RequestParam int id, @RequestParam String name,

@RequestParam int categeory, @RequestParam double price, @RequestParam String imglink,

@RequestParam int quantity) {

try {

OrderedShoeModel orderedshoemodel = new OrderedShoeModel();

double totalPrice = price \* quantity;

orderedshoemodel.setShoeid(id);

orderedshoemodel.setName(name);

orderedshoemodel.setCategeory(categeory);

orderedshoemodel.setPrice(price);

orderedshoemodel.setImagelink(imglink);

orderedshoemodel.setQuantity(quantity);

orderedshoemodel.setTotalprice(totalPrice);

model.put("orderedshoemodel", orderedshoemodel);

return "bookShoeForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.orderInforMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "bookShoeForm";

}

}

@GetMapping("/bookandpaymentreturntoHome")

// @RequestMapping(value = "/bookandpaymentreturntoHome", method =

// RequestMethod.GET)

public String bookAndPaymentreturntoHomeMethod(ModelMap model, @RequestParam(name = "id") int shoeid,

@RequestParam String name, @RequestParam int categeory, @RequestParam double price,

@RequestParam String imglink, @RequestParam int quantity, @RequestParam double totalprice) {

try {

long millis = System.currentTimeMillis();

Date date = new java.sql.Date(millis);

orderservice.insertBookingInfoService(shoeid, name, categeory, price, imglink, quantity, totalprice, date);

return "orderplaceForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.bookAndPaymentreturntoHomeMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "orderplaceForm";

}

}

@GetMapping("/adminsignin")

public String adminSigninMethod(ModelMap model, @RequestParam String username, @RequestParam String password) {

int rollid = 0;

try {

UserInfoModel userInfoModel = userservice.isValidAdminUserService(username, password);

if (userInfoModel == null) {

model.put("errorMessage", "Invalid Credentials");

return "signinForm";

}

rollid = userInfoModel.getRollid();

if (rollid == 1) {

List<ShoesDataModel> mensData = shoeservice.getMensShoeData();

model.put("men\_women\_kids\_ShoeDataname", mensData);

model.put("username", username.toUpperCase());

model.put("password", password);

return "adminForm";

} else if (rollid == 2) {

model.put("username", username);

model.put("password", password);

return "index";

} else if (rollid == 0) {

model.put("errorMessage", "Invalid Credentials");

return "signinForm";

}

model.put("errorMessage", "Invalid Credentials");

return "signinForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.signinMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

model.put("errorMessage", "Exception check Log File");

return "signinForm";

}

}

@GetMapping("/getcompletemenshoesforAdmin")

public String getcompletemenshoesforAdmin(ModelMap model) {

try {

List<ShoesDataModel> mensData = shoeservice.getMensShoeData();

model.put("men\_women\_kids\_ShoeDataname", mensData);

return "adminForm";

} catch (Exception e) {

System.out.println("Exception at com.mjava.controllers.CommonController.getcompletemenshoesforAdmin() "

+ e.getMessage());

return "adminForm";

}

}

@GetMapping("/getcompletewomenshoesforAdmin")

public String getcompletewomenshoesforAdmin(ModelMap model) {

try {

List<ShoesDataModel> womensData = shoeservice.getWomensShoeData();

model.put("men\_women\_kids\_ShoeDataname", womensData);

return "adminForm";

} catch (Exception e) {

System.out.println("Exception at com.mjava.controllers.CommonController.getcompletewomenshoesforAdmin() "

+ e.getMessage());

return "adminForm";

}

}

@GetMapping("/getcompletekidshoesforAdmin")

public String getcompletekidshoesforAdmin(ModelMap model) {

try {

List<ShoesDataModel> kidsData = shoeservice.getKidsShoeData();

model.put("men\_women\_kids\_ShoeDataname", kidsData);

return "adminForm";

} catch (Exception e) {

System.out.println("Exception at com.mjava.controllers.CommonController.getcompletekidshoesforAdmin() "

+ e.getMessage());

return "adminForm";

}

}

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

public String changeAdminPasswordForm() {

return "changeAdminPasswordForm";

}

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

public String addNewProductForm() {

return "addNewProductForm";

}

@GetMapping("/customersList")

public String customersListMethod(ModelMap model) {

List<UserInfoModel> usersList = null;

try {

usersList = userservice.getUserDataService();

if (!usersList.isEmpty()) {

model.put("usersList", usersList);

} else {

System.out.println("No Users Found...");

}

return "customersListForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.customersListMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "customersListForm";

}

}

@GetMapping("/reports")

public String reportsMethod(ModelMap model) {

List<OrderedShoeModel> orderedShoeList = null;

try {

orderedShoeList = orderservice.getCompleteTransactionsDataService();

if (orderedShoeList != null) {

double totalSales = 0;

for (OrderedShoeModel osl : orderedShoeList) {

totalSales = totalSales + osl.getTotalprice();

}

model.put("orderedShoeList", orderedShoeList);

model.put("totalSales", totalSales);

}

return "reportsForm";

} catch (Exception e) {

System.out.println("Exception at com.mjava.controllers.CommonController.reportsMethod() " + e.getMessage());

return "reportsForm";

}

}

@GetMapping("/changeadminPassword")

public String changeadminPasswordMethod(ModelMap model, @RequestParam String currentpassword,

@RequestParam String newpassword, @RequestParam String confirmpassword) {

UserInfoModel userInfoModel = null;

boolean isCoreectPassword = false;

try {

userInfoModel = userservice.isCorrectPassword\_or\_NotService(currentpassword);

if (userInfoModel.getRollid() != 0) {

boolean isUpdatedAdminPassword = userservice.updateAdminPasswordService(userInfoModel, currentpassword,

newpassword, confirmpassword);

if (isUpdatedAdminPassword == true) {

System.out.println("Password Updated");

return "signinForm";

} else {

System.out.println("Password Not Updated");

}

}

else {

System.out.println("No Data Found");

}

return "changeAdminPasswordForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.changeadminPasswordMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "changeAdminPasswordForm";

}

}

@GetMapping("/requiredreports")

public String requiredReportsMethod(ModelMap model, @RequestParam String categeory, @RequestParam String date) {

List<OrderedShoeModel> orderedShoeList = null;

try {

int categeoryId = 0;

String categeoryName = categeory;

if (categeoryName.equals("mens"))

categeoryId = 1;

else if (categeoryName.equals("womens"))

categeoryId = 2;

else if (categeoryName.equals("kids"))

categeoryId = 3;

String jspdate = date;

Date sqldate = Date.valueOf(jspdate);

if (categeoryId != 0) {

orderedShoeList = orderservice.getRequiredCompleteTransactionsDataService(categeoryId, sqldate);

if (orderedShoeList != null) {

double totalSales = 0;

for (OrderedShoeModel osl : orderedShoeList) {

totalSales = totalSales + osl.getTotalprice();

}

model.put("orderedShoeList", orderedShoeList);

model.put("totalSales", totalSales);

return "reportsForm";

}

}

return "reportsForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.requiredReportsMethod() " + e.getMessage());

return "reportsForm";

}

}

@RequestMapping("/addnewproduct")

public String addNewProductMethod(ModelMap model, @RequestParam String weartype, @RequestParam String prod\_name,

@RequestParam double prod\_price, @RequestParam String prod\_img) {

ModelAndView mv = null;

try {

int categeroy = 0;

if (weartype.equals("Mens\_Wear")) {

categeroy = 1;

} else if (weartype.equals("Womens\_Wear")) {

categeroy = 2;

} else if (weartype.equals("Kids\_Wear")) {

categeroy = 3;

}

boolean isInsertedNewProduct = false;

isInsertedNewProduct = shoeservice.insertNewProductService(categeroy, prod\_price, prod\_name, prod\_img);

if (isInsertedNewProduct == true) {

if (categeroy == 1) {

List<ShoesDataModel> mensData = shoeservice.getMensShoeData();

model.put("men\_women\_kids\_ShoeDataname", mensData);

return "adminForm";

}

if (categeroy == 2) {

List<ShoesDataModel> womensData = shoeservice.getWomensShoeData();

model.put("men\_women\_kids\_ShoeDataname", womensData);

return "adminForm";

}

if (categeroy == 3) {

List<ShoesDataModel> kidsData = shoeservice.getKidsShoeData();

model.put("men\_women\_kids\_ShoeDataname", kidsData);

return "adminForm";

}

} else {

System.out.println("Product Not Added");

}

return "adminForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.addNewProducttMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "adminForm";

}

}

@GetMapping("/signupnewuser")

public String sigUpNewUserMethod(ModelMap model, @RequestParam String firstname, @RequestParam String lastname,

@RequestParam String password, @RequestParam String confirmpassword, @RequestParam String roll,

@RequestParam Long mobileno, @RequestParam String email) {

ModelAndView mv = null;

boolean isInserted = false;

try {

int rollid = 0;

if (roll.equals("admin")) {

rollid = 1;

} else {

rollid = 2;

}

isInserted = true;

userservice.insertUserDataService(firstname, lastname, password, confirmpassword, rollid, mobileno, email);

if (isInserted == true) {

return "signinForm";

}

if (isInserted == false) {

return "testErrorForm";

}

return "testErrorForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.sigUpNewUserMethod(HttpServletRequest, HttpServletResponse) "

+ e.getMessage());

return "testErrorForm";

}

}

@GetMapping("/deleteProduct")

public String deleteProductMethod(ModelMap model, @RequestParam int id) {

boolean isdeleted = false;

List<ShoesDataModel> shoesDatalist = null;

ShoesDataModel findwhichCategeory = null;

try {

findwhichCategeory = shoeservice.getshoesDataByIdService(id);

int categeory = findwhichCategeory.getCategeory();

isdeleted = shoeservice.deleteProductwithIdService(id);

if (isdeleted == true) {

shoesDatalist = shoeservice.getShoeDatabyCategeory(categeory);

model.put("men\_women\_kids\_ShoeDataname", shoesDatalist);

}

return "adminForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.deleteProductMethod() " + e.getMessage());

return "adminForm";

}

}

@GetMapping("/updateProduct")

public String updateProductMethod(ModelMap model, @RequestParam int id) {

boolean isupdated = false;

ShoesDataModel shoesData = null;

// ShoesDataModel findwhichCategeory = null;

try {

shoesData = shoeservice.getshoesDataByIdService(id);

if (shoesData.getId() > 0) {

model.put("men\_women\_kids\_ShoeDataname", shoesData);

return "updateProductForm";

}

// int categeory = findwhichCategeory.getCategeory();

// isdeleted = shoeservice.deleteProductwithIdService(id);

// if (isdeleted == true) {

// shoesDatalist = shoeservice.getShoeDatabyCategeory(categeory);

//

// model.put("men\_women\_kids\_ShoeDataname", shoesDatalist);

//

// }

//

// return "adminForm";

return "adminForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.deleteProductMethod() " + e.getMessage());

return "adminForm";

}

}

@GetMapping("/updatetoNewproduct")

public String updateProductMethod(ModelMap model, @RequestParam String weartype, @RequestParam String prod\_name,

@RequestParam double prod\_price, @RequestParam String prod\_img, @RequestParam int prod\_id) {

boolean isupdated = false;

ShoesDataModel shoesData = null;

List<ShoesDataModel> shoesDataList = null;

// ShoesDataModel findwhichCategeory = null;

try {

int categeory = 0;

if (weartype.equals("Mens\_Wear")) {

categeory = 1;

} else if (weartype.equals("Womens\_Wear")) {

categeory = 2;

} else if (weartype.equals("Kids\_Wear")) {

categeory = 3;

}

isupdated = shoeservice.updateShoeProductService(categeory, prod\_price, prod\_name, prod\_img, prod\_id);

if (isupdated == true) {

shoesDataList = shoeservice.getShoeDatabyCategeory(categeory);

model.put("men\_women\_kids\_ShoeDataname", shoesDataList);

}

return "adminForm";

} catch (Exception e) {

System.out.println(

"Exception at com.mjava.controllers.CommonController.deleteProductMethod() " + e.getMessage());

return "adminForm";

}

}

}

**GlobalExceptionHandeler.java:**

package com.mjava.exception;

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

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

@ControllerAdvice

public class GlobalExceptionHandeler {

@ExceptionHandler(value=Exception.class)

public String handleAnyException()

{

return "Error";

}

}

**Application.properties:**

logging.level.org.springframework.web: DEBUG

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

spring.mvc.view.suffix=.jsp

server.port=8090

spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:mysql://localhost:3306/sportyshoes\_db

spring.datasource.username=root

spring.datasource.password=12345

**index.jsp:**

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

pageEncoding="ISO-8859-1" errorPage="Error.jsp"%>

<html>

<title>Home</title>

<body>

<%@include file="homeHeader.jsp"%>

<hr>

<a href="menshoes" >

<img src="https://assets.adidas.com/images/h\_840,f\_auto,q\_auto,fl\_lossy,c\_fill,g\_auto/dd9a709ceafa4b69b309a8d400df4d23\_9366/VS\_Pace\_Shoes\_Grey\_B74318\_01\_standard.jpg" style="float: left; width: 30%; margin-right: 1%; margin-bottom: 0.5em;">

</a>

<a href="womenshoes">

<img src="https://assets.ajio.com/medias/sys\_master/root/20211216/aRJZ/61ba5682f997ddf8f152e5e4/-473Wx593H-463506022-pink-MODEL.jpg" style="float: left; width: 30%; margin-right: 1%; margin-bottom: 0.5em;">

</a>

<a href="kidshoes" > <img src="https://assets.ajio.com/cms/AJIO/WEB/11032022-D-KHP-topbanner-allstarssale-5090.gif" style="float: left; width: 30%; margin-right: 1%; margin-bottom: 0.5em;">

</a>

</body>

</html>

**home.jsp:**

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

pageEncoding="ISO-8859-1" errorPage="Error.jsp"%>

<!DOCTYPE html>

<html>

<head>

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

<title>Home</title>

</head>

<body>

<h1 style="text-align:left;float:left;"> <a href="home" style="text-decoration:none;color:black">Sporty Shoes </a> </h1>

<h3 style="text-align:right;float:right;"><a href="signin" style="text-decoration:none;color:black">SignIn</a> | <a href="signup" style="text-decoration:none;color:black">SignUp</a></h3>

<hr style="clear:both;"/>

<b><p> <a href="home" style="text-decoration:none;color:black">Home</a> | <a href="menshoes" style="text-decoration:none ;color:black">Men</a> | <a href="womenshoes" style="text-decoration:none ;color:black">Women</a> | <a href="kidshoes" style="text-decoration:none ;color:black">Kids</a></p></b>

</body>

</html>

**menShoesForm.jsp:**

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

pageEncoding="ISO-8859-1" isELIgnored="false" errorPage="Error.jsp"%>

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

<!DOCTYPE html>

<html>

<head>

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

<title>Mens Shoes</title>

</head>

<body>

<%@include file="homeHeader.jsp" %>

<h2>Men's Shoes</h2>

<table>

<c:forEach items="${menshoeData}" var="fss">

<tr>

<td><span>

<a href="selectedShoe?id=<c:out value='${fss.id}' />">

<img src="${fss.imagelink}" style="float: left; width: 30%; margin-right: 1%; margin-bottom: 0.5em;">

<br><br><label >INR ${fss.price} </label> <br> <label>${fss.name}</label>

</a>

</span>

</td>

</tr>

</c:forEach>

</table>

</body>

</html>

**orderplaceForm.jsp:**

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

pageEncoding="ISO-8859-1" errorPage="Error.jsp"%>

<!DOCTYPE html>

<html>

<head>

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

<title>OrderPlace</title>

</head>

<body>

<h1>Payment Done.Order Placed </h1>

<h2>Thanks for the Shopping..... </h2><br>

<br>

<h2 align="center"> <a href="home" style="text-decoration:none">Back to Home</a> </h2>

</body>

</html>