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";

}

}

}