Source Code:

**package** com.example.MVCSpringSL;

**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;

@SpringBootApplication

@ComponentScan("com")

@EntityScan("com.model")

@EnableJpaRepositories("com.repo")

**public** **class** SportyShoesLaunch {

**public** **static** **void** main(String[] args) {

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

}

}

**package** com.controller;

**import** javax.validation.Valid;

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

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

**import** org.springframework.ui.ModelMap;

**import** org.springframework.validation.BindingResult;

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

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

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

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

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

**import** com.model.Category;

**import** com.model.User;

**import** com.service.CategoryService;

@Controller

@SessionAttributes(names = "user")

@RequestMapping("/sportyshoes")

**public** **class** CategoryController {

@Autowired

**private** CategoryService cs;

@GetMapping("/category")

**public** String category(ModelMap map) {

User user = (User) map.getAttribute("user");

**if**(user.isAdminAcc()) {

map.addAttribute("categories", cs.getAllCategory());

map.addAttribute("category", **new** Category());

**return** "categories";

}

**else** {

map.addAttribute("errorMessage", "Accessible only to admin");

**return** "home";

}

}

@PostMapping("/category")

**public** String addCategory(ModelMap map, @Valid Category cat, BindingResult result) {

**if** (result.hasErrors()) {

**return** "categories";

}

cs.addCategory(cat.getCatName());

map.put("successMessage", "Category " +cat.getCatName() + " registered succesfuly");

**return** "redirect:category";

}

@GetMapping("/update-cat")

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

Category cat = cs.findCat(id);

model.addAttribute("category", cat);

model.addAttribute("categories", cs.getAllCategory());

**return** "categories";

}

@PostMapping("/update-cat")

**public** String updateUser(ModelMap map, @Valid Category cat, BindingResult result) {

**if**(result.hasErrors()) {

**return** "categories";

}

cs.updateCategory(cat);

**return** "redirect:category";

}

@GetMapping("/delete-cat")

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

cs.deleteCategory(id);

**return** "redirect:category";

}

}

@Controller

@SessionAttributes(names = {"user"})

@RequestMapping("/sportyshoes")

**public** **class** HomeController {

@GetMapping("/home")

**public** String home() {

**return** "home";

}

}

@Controller

@SessionAttributes(names = {"user"})

@RequestMapping("/sportyshoes")

**public** **class** LoginController {

@Autowired

**private** UserService us;

@GetMapping("/login")

**public** String login() {

us.createDefaultAdmin();

**return** "login";

}

@PostMapping("/login")

**public** String loginValid(ModelMap map,@RequestParam("uname") String name, @RequestParam("upass") String password) {

**if** (us.loginValid(name, password)) {

List<User> users = (List<User>) us.loadUsers();

**for**(User user : users) {

**if**(user.getName().equals(name)){

**if**(user.getPassword().equals(password)) {

map.put("user", user);

}

}

}

map.put("successMessage", "Welcome " + name + " !!!");

**return** "home";

} **else** {

map.put("errorMessage", "Login credentials invalid");

**return** "login";

}

}

}

package com.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

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

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

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

@Controller

@SessionAttributes(names = {"user"})

@RequestMapping("/sportyshoes")

public class LogoutController {

@GetMapping("/Logout")

public String getPurchasePage(ModelMap map) {

map.clear();

return "redirect:login";

}

}

@Controller

@SessionAttributes(names = "user")

@RequestMapping("/sportyshoes")

**public** **class** ProductController {

@Autowired

**private** ProductService ps;

@Autowired

**private** CategoryService cs;

@GetMapping("/manageProducts")

**public** String getAddProduct(ModelMap map) {

User user = (User) map.getAttribute("user");

**if**(user.isAdminAcc()) {

map.addAttribute("listCategory",cs.getAllCategory());

map.addAttribute("productList",ps.getAllProduct());

**return** "products";

}

**else** {

map.addAttribute("errorMessage", "Accessible only to admin");

**return** "home";

}

}

@PostMapping("/manageProducts")

**public** String addProduct(ModelMap map,@RequestParam("prodStock") **int** stock, @RequestParam("prodName") String name, @RequestParam("prodCategory") **int** catId) {

Category cat = cs.findCat(catId);

**if**(name.equals(**null**)||name.equals("")) {

map.addAttribute("errorMessage", "Name should not be null");

**return** "products";

}

Product prod = **new** Product(name, cat, stock);

map.addAttribute("errorMessage", **null**);

ps.addProdcut(prod.getProdName(), prod.getProdCategory(), prod.getStock());

map.put("successMessage", "Product " +prod.getProdName() + " registered succesfuly");

**return** "redirect:manageProducts";

}

@GetMapping("/update-prod")

**public** String getUpdateProductPage(@RequestParam **int** id, ModelMap map) {

map.addAttribute("listCategory",cs.getAllCategory());

map.addAttribute("productList",ps.getAllProduct());

Product prod = ps.findProd(id);

map.addAttribute("productName", prod.getProdName());

map.addAttribute("selectedCatId",prod.getProdCategory().getCatId());

map.addAttribute("productStock", prod.getStock());

**return** "products";

}

@PostMapping("/update-prod")

**public** String updateProduct(ModelMap map,@RequestParam("id") **int** id,@RequestParam("prodStock") **int** stock, @RequestParam("prodName") String name, @RequestParam("prodCategory") **int** catId) {

Category cat = cs.findCat(catId);

**if**(name.equals(**null**)||name.equals("")) {

map.addAttribute("errorMessage", "Name should not be null");

**return** "products";

}

map.addAttribute("errorMessage", **null**);

map.addAttribute("productName", **null**);

map.addAttribute("selectedCatId",0);

map.addAttribute("productStock", "");

ps.updateProduct(id,name,cat,stock);

**return** "redirect:manageProducts";

}

@GetMapping("/delete-prod")

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

ps.deleteProduct(id);

**return** "redirect:manageProducts";

}

}

@Controller

@SessionAttributes(names = {"user"})

@RequestMapping("/sportyshoes")

**public** **class** PurchaseController {

@Autowired

**private** PurchaseService pr;

@Autowired

**private** ProductService ps;

@Autowired

**private** UserService us;

@Autowired

**private** CategoryService cs;

@InitBinder

**public** **void** initBinder(WebDataBinder binder) {

// Date - dd/MM/yyyy

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

binder.registerCustomEditor(Date.**class**, **new** CustomDateEditor(

dateFormat, **false**));

}

@GetMapping("/purchase")

**public** String getPurchasePage(ModelMap map) {

map.addAttribute("listProduct", ps.getAllProduct());

**return** "purchase";

}

@PostMapping("/purchase")

**public** String addPurchase(ModelMap map,@RequestParam("product")**int** pid, @RequestParam("purchaseCount")**int** count ) {

Product pd = ps.findProd(pid);

User user = (User) map.get("user");

**if**(!ps.checkAvailablitiy(pd,count)) {

map.addAttribute("errorMessage", "Purchase could not be made");

**return** "purchase";

}

pr.addPurchase(user,pd,count);

ps.updateProductStock(pd,count);

map.put("successMessage", "Purchase made succesfuly");

**return** "redirect:purchaseList";

}

@GetMapping("/purchaseList")

**public** String getPurchaseListPage(ModelMap map) {

User user = (User) map.getAttribute("user");

map.addAttribute("listCategory",cs.getAllCategory());

**if**(user.isAdminAcc()) {

map.addAttribute("listUsers",us.loadUsers());

map.addAttribute("purchaseList", pr.getAllPurchase());

}

**else** {

List<User> userList = **new** ArrayList<User>();

userList.add(us.findWithID(user.getId()));

map.addAttribute("listUsers",userList);

map.addAttribute("purchaseList", pr.getUserPurchase(user));

}

**return** "purchaselist";

}

@PostMapping("/purchaseList")

**public** String getPurchaseList(ModelMap map,@RequestParam("username")**int** uid,@RequestParam("productCategory")**int** cid,@RequestParam("purchaseDate")String pdate) {

User user = (User)map.getAttribute("user");

map.addAttribute("listCategory",cs.getAllCategory());

**if**(user.isAdminAcc()) {

map.addAttribute("listUsers",us.loadUsers());

}

**else** {

List<User> userList = **new** ArrayList<User>();

userList.add(us.findWithID(user.getId()));

map.addAttribute("listUsers",userList);

}

map.addAttribute("purchaseList", pr.filterPurchases(uid,cid,pdate, user));

**return** "purchaselist";

}

}

@Controller

@SessionAttributes(names = "user")

@RequestMapping("/sportyshoes")

**public** **class** UserController {

@Autowired

**private** UserService us;

@GetMapping("/register")

**public** String register(ModelMap map) {

User user = (User) map.getAttribute("user");

**if** (user.isAdminAcc()) {

map.addAttribute("user", **new** User());

**return** "registration";

} **else** {

map.addAttribute("errorMessage", "Accessible only to admin");

**return** "home";

}

}

@PostMapping("/register")

**public** String addUser(ModelMap map, @Valid User user, BindingResult result) {

**if** (result.hasErrors()) {

**return** "registration";

}

us.registerUser(user.getName(), user.getPassword(), user.getCity(), user.getEmail(), user.isAdminAcc());

map.put("successMessage", "User " + user.getName() + " registered succesfuly");

**return** "redirect:listUsers";

}

@GetMapping("/listUsers")

**public** String loadAll(ModelMap map) {

map.put("userList", us.loadUsers());

**return** "listUsers";

}

@PostMapping("/listUsers")

**public** String loadRequested(ModelMap map, @RequestParam("uname") String name) {

**if** (name.equals(**null**)) {

map.put("userList", us.loadUsers());

} **else** {

map.put("userList", us.findParticular(name));

}

**return** "listUsers";

}

@GetMapping(value = "/update-user")

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

User user = us.findWithID(id);

model.addAttribute("user", user);

**return** "registration";

}

@PostMapping(value = "/update-user")

**public** String updateUser(ModelMap map, @Valid User user, BindingResult result) {

**if** (result.hasErrors()) {

**return** "registration";

}

us.updateUser(user);

**return** "redirect:listUsers";

}

@GetMapping(value = "/delete-user")

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

us.deleteUser(id);

**return** "redirect:listUsers";

}

@GetMapping(value = "/changepass")

**public** String getChangePass(ModelMap map) {

**return** "changepass";

}

@PostMapping(value = "/changepass")

**public** String changePass(@RequestParam("oldpass") String curPass, @RequestParam("NewPass") String newPass,

@RequestParam("confPass") String confPass, ModelMap model) {

User user = (User) model.getAttribute("user");

**if** (newPass.equals(confPass)) {

**if** (us.changePass(user, curPass, newPass)) {

**return** "redirect:Logout";

} **else** {

model.addAttribute("errorMessage", "Password mismatch");

}

**return** "changepass";

} **else** {

model.addAttribute("errorMessage", "New and confirm password mismatch");

**return** "changepass";

}

}

}

**package** com.model;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.validation.constraints.NotEmpty;

@Entity

**public** **class** Category {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**private** Integer catId;

@NotEmpty

**private** String catName;

**public** Integer getCatId() {

**return** catId;

}

**public** **void** setCatId(Integer catId) {

**this**.catId = catId;

}

**public** String getCatName() {

**return** catName;

}

**public** **void** setCatName(String catName) {

**this**.catName = catName;

}

**public** Category() {

**super**();

}

**public** Category(String catName) {

**super**();

**this**.catName = catName;

}

}

**package** com.model;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

**import** javax.persistence.JoinColumn;

**import** javax.persistence.ManyToOne;

@Entity

**public** **class** Product {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**private** Integer prodId;

**private** String prodName;

**private** **int** stock;

@ManyToOne

@JoinColumn

**private** Category prodCategory;

**public** **int** getStock() {

**return** stock;

}

**public** **void** setStock(**int** stock) {

**this**.stock = stock;

}

**public** Integer getProdId() {

**return** prodId;

}

**public** **void** setProdId(Integer prodId) {

**this**.prodId = prodId;

}

**public** String getProdName() {

**return** prodName;

}

**public** **void** setProdName(String prodName) {

**this**.prodName = prodName;

}

**public** Category getProdCategory() {

**return** prodCategory;

}

**public** **void** setProdCategory(Category prodCategory) {

**this**.prodCategory = prodCategory;

}

**public** Product() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** Product(String prodName, Category prodCategory) {

**super**();

**this**.prodName = prodName;

**this**.prodCategory = prodCategory;

}

**public** Product(String prodName, Category prodCategory, **int** stock) {

**super**();

**this**.prodName = prodName;

**this**.prodCategory = prodCategory;

**this**.stock = stock;

}

@Override

**public** String toString() {

**return** "Product [prodId=" + prodId + ", prodName=" + prodName + ", stock=" + stock + ", prodCategory="

+ prodCategory + "]";

}

}

**package** com.service;

**import** java.util.List;

**import** java.util.Optional;

**import** javax.validation.Valid;

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

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

**import** com.model.Category;

**import** com.repo.CategoryRepo;

@Service

**public** **class** CategoryService {

@Autowired

**private** CategoryRepo catRepo;

**public** **void** addCategory(String catName) {

catRepo.save(**new** Category(catName));

}

**public** List<Category> getAllCategory() {

**return** (List<Category>) catRepo.findAll();

}

**public** **void** updateCategory(@Valid Category cat1) {

Integer id =cat1.getCatId();

Optional<Category> catUp = catRepo.findById(id);

**if**(catUp.isPresent()) {

Category catUpdated = catUp.get();

catUpdated.setCatName(cat1.getCatName());

catRepo.save(catUpdated);

}

}

**public** **void** deleteCategory(**int** id) {

catRepo.deleteById(id);

}

**public** Category findCat(**int** id) {

Category cat = **null**;

Optional<Category> category = catRepo.findById(id);

**if** (category.isPresent()) {

cat = category.get();

}

**return** cat;

}

}