**package com.project.sportyshoes;**

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SportyshoesApplication {

public static void main(String[] args) {

SpringApplication.run(SportyshoesApplication.class, args);

}

}

**package com.project.sportyshoes.configuration;**

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

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

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

import org.springframework.security.config.annotation.web.builders.WebSecurity;

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;

import org.springframework.security.web.util.matcher.AntPathRequestMatcher;

import com.project.sportyshoes.service.CustomUserDetailService;

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Autowired

CustomUserDetailService customUserDetailService;

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.authorizeRequests()

.antMatchers("/" , "/shop/\*\*" , "/forgotpassword" , "/register" , "/h2-console/\*\*" ).permitAll()

.antMatchers("/admin/\*\*").hasRole("ADMIN")

.anyRequest()

.authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll()

.failureUrl("/login?error=true")

.defaultSuccessUrl("/")

.usernameParameter("email")

.passwordParameter("password")

.and()

.logout()

.logoutRequestMatcher(new AntPathRequestMatcher("/logout"))

.logoutSuccessUrl("/login")

.invalidateHttpSession(true)

.deleteCookies("JSESSIONID")

.and()

.exceptionHandling()

.and()

.csrf()

.disable();

http.headers().frameOptions().disable();

}

@Bean

public BCryptPasswordEncoder bCryptPasswordEncoder() {

return new BCryptPasswordEncoder();

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(customUserDetailService);

}

@Override

public void configure(WebSecurity web) throws Exception {

web.ignoring().antMatchers("/resources/\*\*" , "/static/\*\*" , "/images/\*\*" , "/productImages/\*\*" , "/css/\*\*" , "/js/\*\*")

;

}

**package com.project.sportyshoes.controller;**

**Admin Controller**

package com.project.sportyshoes.controller;

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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.RequestParam;

import org.springframework.web.multipart.MultipartFile;

import com.project.sportyshoes.dto.ProductDTO;

import com.project.sportyshoes.global.PurchaseReport;

import com.project.sportyshoes.model.Category;

import com.project.sportyshoes.model.Product;

import com.project.sportyshoes.model.Purchase;

import com.project.sportyshoes.model.User;

import com.project.sportyshoes.repository.UserRepository;

import com.project.sportyshoes.service.CategoryService;

import com.project.sportyshoes.service.ProductService;

import com.project.sportyshoes.service.PurchaseService;

@Controller

public class AdminController {

public static String uploadDir = System.getProperty("user.dir") + "/src/main/resources/static/productImages";

@Autowired

CategoryService categoryService;

@Autowired

ProductService productService;

@Autowired

PurchaseService purchaseService;

@Autowired

UserRepository userRepository;

@GetMapping("/admin")

public String adminHome() {

return "adminHome";

}

@GetMapping("/admin/categories")

public String getCat(Model model) {

model.addAttribute("categories" , categoryService.getAllCategories());

return "categories";

}

@GetMapping("/admin/categories/add")

public String getCatAdd(Model model) {

model.addAttribute("category" , new Category());

return "categoriesAdd";

}

@PostMapping("/admin/categories/add")

public String postCatAdd(@ModelAttribute("category") Category category) {

categoryService.addCategory(category);

return "redirect:/admin/categories";

}

@GetMapping("/admin/categories/delete/{id}")

public String deleteCat(@PathVariable int id) {

categoryService.removeCategoryById(id);

return "redirect:/admin/categories";

}

@GetMapping("/admin/categories/update/{id}")

public String updateCat(@PathVariable int id , Model model) {

Optional<Category> category = categoryService.getCatById(id);

if(category.isPresent()) {

model.addAttribute("category" , category.get());

return "categoriesAdd";

}else {

return "404";

}

}

// Product Section

@GetMapping("/admin/products")

public String showProducts(Model model) {

model.addAttribute("products" , productService.getAllProducts());

return "products";

}

@GetMapping("/admin/products/add")

public String addProduct(Model model) {

model.addAttribute("productDTO" , new ProductDTO());

model.addAttribute("categories" , categoryService.getAllCategories());

return "productsAdd";

}

@PostMapping("/admin/products/add")

public String productAddPost(@ModelAttribute ("productDTO") ProductDTO productDTO ,

@RequestParam("productImage") MultipartFile file,

@RequestParam("imgName") String imgName) throws IOException {

Product product = new Product();

product.setId(productDTO.getId());

product.setName(productDTO.getName());

product.setCategory(categoryService.getCatById(productDTO.getCategoryId()).get());

product.setPrice(productDTO.getPrice());

product.setWeight(productDTO.getWeight());

product.setDescription(productDTO.getDescription());

String imageUUID;

if (!file.isEmpty()) {

imageUUID = file.getOriginalFilename();

Path fileNameAndPath = Paths.get(uploadDir, imageUUID);

Files.write(fileNameAndPath, file.getBytes());

}else {

imageUUID = imgName;

}

product.setImageName(imageUUID);

productService.addProduct(product);

return "redirect:/admin/products";

}

@GetMapping("/admin/product/delete/{id}")

public String deleteProd(@PathVariable long id) {

productService.removeProductById(id);

return "redirect:/admin/products";

}

@GetMapping("/admin/product/update/{id}")

public String updateProd(@PathVariable long id , Model model) {

Product product = productService.getProductById(id).get();

ProductDTO productDTO = new ProductDTO();

productDTO.setId(product.getId());

productDTO.setName(product.getName());

productDTO.setCategoryId(product.getCategory().getId());

productDTO.setPrice(product.getPrice());

productDTO.setWeight(product.getWeight());

productDTO.setDescription(product.getDescription());

productDTO.setImageName(product.getImageName());

model.addAttribute("productDTO" , productDTO);

model.addAttribute("categories" , categoryService.getAllCategories());

return "productsAdd";

}

//Purchase Report

@GetMapping("/admin/purchaseReport")

public String purchaseReport(Model model) {

List<Purchase>purchaseList = purchaseService.getAllPurchases();

List<PurchaseReport>purchaseReportList = new ArrayList<PurchaseReport>();

for(Purchase pur : purchaseList) {

Product product = new Product();

User user = new User();

Category category = new Category();

PurchaseReport purchaseReport = new PurchaseReport();

Long productId = pur.getProductId();

int userId = pur.getUserId();

product = productService.getProductById(productId).get();

user = userRepository.findById(userId).get();

category = product.getCategory();

purchaseReport.setEmail(user.getEmail());

purchaseReport.setName(user.getFirstName());

purchaseReport.setProductId(product.getId());

purchaseReport.setProductName(product.getName());

purchaseReport.setPrice(product.getPrice());

purchaseReport.setDate(pur.getOrderDate().toString());

purchaseReport.setCategory(category.getName());

//System.out.println(purchaseReport.getEmail() + " " + purchaseReport.getProductId() + " " + purchaseReport.getDate());

purchaseReportList.add(purchaseReport);

}

model.addAttribute("puchaseList" , purchaseReportList);

return "purchaseReport";

}

@GetMapping("/admin/users")

public String listUsers(Model model) {

List<User>userList = userRepository.findAll();

model.addAttribute("userlist" , userList);

return "userList";

}

}

**Cart Controller**

package com.project.sportyshoes.controller;

import java.time.LocalDate;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

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

import org.springframework.security.core.Authentication;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

import com.project.sportyshoes.global.GlobalData;

import com.project.sportyshoes.model.Product;

import com.project.sportyshoes.model.Purchase;

import com.project.sportyshoes.model.User;

import com.project.sportyshoes.repository.PurchaseRepository;

import com.project.sportyshoes.repository.UserRepository;

import com.project.sportyshoes.service.ProductService;

@Controller

public class CartController {

@Autowired

ProductService productService;

@Autowired

UserRepository userRepository;

@Autowired

PurchaseRepository purchaseRepository;

@GetMapping("/addToCart/{id}")

public String addToCart(@PathVariable int id) {

GlobalData.cart.add(productService.getProductById(id).get());

return "redirect:/shop";

}

@GetMapping("/cart")

public String cartGet(Model model) {

model.addAttribute("cartCount" , GlobalData.cart.size());

model.addAttribute("total" , GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());

model.addAttribute("cart" , GlobalData.cart);

return "cart";

}

@GetMapping("/cart/removeItem/{index}")

public String cartItemRemove(@PathVariable int index) {

GlobalData.cart.remove(index);

return "redirect:/cart";

}

@GetMapping("/checkout")

public String checkout(Model model) {

model.addAttribute("total" , GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());

return "checkout";

}

@PostMapping("/payNow")

public String orderConfirmation(Model model) {

model.addAttribute("total" , GlobalData.cart.stream().mapToDouble(Product::getPrice).sum());

Authentication auth = SecurityContextHolder.getContext().getAuthentication();

String currentPrincipalName = auth.getName();

List<Purchase>purchaseList = new ArrayList<Purchase>();

//System.out.println(currentPrincipalName);

User user = userRepository.findUserByEmail(currentPrincipalName).get();

for(Product product: GlobalData.cart) {

Purchase purchase = new Purchase();

//System.out.println(product.getId() + " " + product.getName());

purchase.setProductId(product.getId());

purchase.setUserId(user.getId());

purchase.setOrderDate(LocalDate.now());

purchaseList.add(purchase);

}

int n = 10000 + new Random().nextInt(90000);

model.addAttribute("Reciept" , n);

//System.out.println(purchaseList.toString());

model.addAttribute("products" , GlobalData.cart);

purchaseRepository.saveAll(purchaseList);

return "orderPlaced";

}

}

Home controller

package com.project.sportyshoes.controller;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

import com.project.sportyshoes.global.GlobalData;

import com.project.sportyshoes.service.CategoryService;

import com.project.sportyshoes.service.ProductService;

@Controller

public class HomeController {

@Autowired

CategoryService categoryService;

@Autowired

ProductService productService;

@GetMapping({"/" , "/home"})

public String home(Model model) {

model.addAttribute("cartCount" , GlobalData.cart.size());

return "index";

}

@GetMapping("/shop")

public String shop(Model model) {

model.addAttribute("cartCount" , GlobalData.cart.size());

model.addAttribute("categories" , categoryService.getAllCategories());

model.addAttribute("products" , productService.getAllProducts());

return "shop";

}

@GetMapping("/shop/category/{id}")

public String shopByCategory(Model model , @PathVariable int id) {

model.addAttribute("cartCount" , GlobalData.cart.size());

model.addAttribute("categories" , categoryService.getAllCategories());

model.addAttribute("products" , productService.getAllProductsByCategoryId(id));

return "shop";

}

@GetMapping("/shop/viewproduct/{id}")

public String viewProduct(Model model , @PathVariable long id) {

model.addAttribute("product" , productService.getProductById(id).get());

model.addAttribute("cartCount" , GlobalData.cart.size());

return "viewProduct";

}

**Login Controller**

package com.project.sportyshoes.controller;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

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

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

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.PostMapping;

import com.project.sportyshoes.global.GlobalData;

import com.project.sportyshoes.model.Role;

import com.project.sportyshoes.model.User;

import com.project.sportyshoes.repository.RoleRepository;

import com.project.sportyshoes.repository.UserRepository;

@Controller

public class LoginController {

@Autowired

private BCryptPasswordEncoder bCryptPasswordEncoder;

@Autowired

UserRepository userRepository;

@Autowired

RoleRepository roleRepository;

@GetMapping("/login")

public String login() {

GlobalData.cart.clear();

return "login";

}

@GetMapping("/register")

public String registerGet() {

return "register";

}

@PostMapping("/register")

public String registerPost(@ModelAttribute("user") User user , HttpServletRequest request) throws ServletException {

String passoword = user.getPassword();

user.setPassword(bCryptPasswordEncoder.encode(passoword));

List<Role>roles = new ArrayList<Role>();

roles.add(roleRepository.findById(2).get());

user.setRoles(roles);

userRepository.save(user);

request.login(user.getEmail(), passoword);

return "redirect:/";

}

}

}

**package** com.project.sportyshoes.dto;

**public** **class** ProductDTO {

**private** Long id;

**private** String name;

**private** **int** categoryId;

**private** **double** price;

**private** **double** weight;

**private** String description;

**private** String imageName;

**public** Long getId() {

**return** id;

}

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getCategoryId() {

**return** categoryId;

}

**public** **void** setCategoryId(**int** categoryId) {

**this**.categoryId = categoryId;

}

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

**public** **double** getWeight() {

**return** weight;

}

**public** **void** setWeight(**double** weight) {

**this**.weight = weight;

}

**public** String getDescription() {

**return** description;

}

**public** **void** setDescription(String description) {

**this**.description = description;

}

**public** String getImageName() {

**return** imageName;

}

**public** **void** setImageName(String imageName) {

**this**.imageName = imageName;

}

}

**package** com.project.sportyshoes.global;

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

@Component

**public** **class** PurchaseReport {

**private** String email;

**private** String name;

**private** Long productId;

**private** String productName;

**private** **double** price;

**private** String date;

**private** String category;

**public** String getCategory() {

**return** category;

}

**public** **void** setCategory(String category) {

**this**.category = category;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** Long getProductId() {

**return** productId;

}

**public** **void** setProductId(Long productId) {

**this**.productId = productId;

}

**public** String getProductName() {

**return** productName;

}

**public** **void** setProductName(String productName) {

**this**.productName = productName;

}

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

**public** String getDate() {

**return** date;

}

**public** **void** setDate(String date) {

**this**.date = date;

}

}

package com.project.sportyshoes.service;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.project.sportyshoes.model.Category;

import com.project.sportyshoes.repository.CategoryRepository;

@Service

public class CategoryService {

@Autowired

CategoryRepository categoryRepository;

public List<Category> getAllCategories(){

return categoryRepository.findAll();

}

public void addCategory(Category category) {

categoryRepository.save(category);

}

public void removeCategoryById(int id) {

categoryRepository.deleteById(id);

}

public Optional<Category> getCatById(int id) {

return categoryRepository.findById(id);

}

}

package com.project.sportyshoes.service;

import java.util.Optional;

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

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.project.sportyshoes.model.CustomUserDetail;

import com.project.sportyshoes.model.User;

import com.project.sportyshoes.repository.UserRepository;

@Service

public class CustomUserDetailService implements UserDetailsService {

@Autowired

UserRepository userRepository;

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

Optional<User> user = userRepository.findUserByEmail(username);

user.orElseThrow(() -> new UsernameNotFoundException("User Not Found"));

return user.map(CustomUserDetail::new).get();

}

}

package com.project.sportyshoes.service;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.project.sportyshoes.model.Product;

import com.project.sportyshoes.repository.ProductRepository;

@Service

public class ProductService {

@Autowired

ProductRepository productRepository;

public List<Product> getAllProducts(){

return productRepository.findAll();

}

public void addProduct(Product product) {

productRepository.save(product);

}

public void removeProductById(long id) {

productRepository.deleteById(id);

}

public Optional<Product> getProductById(long id){

return productRepository.findById(id);

}

public List<Product> getAllProductsByCategoryId(int id){

return productRepository.findAllByCategory\_Id(id);

}

}