* **src/main/java**

1. coms.SportyShoes :

**SportyShoes.java**

package coms.SportyShoes;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SportyShoes {

public static void main(String[] args) {

SpringApplication.run(SportyShoes.class, args);

System.out.println("Server Started....");

}

}

1. coms.SportyShoes.controller :

**AdminController.java**

package coms.SportyShoes.controller;

import java.util.List;

import java.util.stream.Collectors;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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.PostMapping;

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

import org.springframework.web.servlet.ModelAndView;

import coms.SportyShoes.model.Product;

import coms.SportyShoes.model.ProductCategory;

import coms.SportyShoes.model.PurchaseItem;

import coms.SportyShoes.model.User;

import coms.SportyShoes.repository.AdminRepository;

import coms.SportyShoes.repository.UserRepository;

import coms.SportyShoes.service.ProductCategoryService;

import coms.SportyShoes.service.ProductService;

import coms.SportyShoes.service.PurchaseItemService;

import coms.SportyShoes.service.UserService;

@Controller

public class AdminController {

@Autowired

private UserService userService;

@Autowired

UserRepository userRepository;

@Autowired

AdminRepository adminRepository;

@Autowired

ProductCategoryService productCategoryService;

@Autowired

ProductService productService;

@Autowired

PurchaseItemService purchaseItemService;

@GetMapping("/admin")

public String adminHome() {

return "adminDetails";

}

@GetMapping(path = "/admin/userList")

public ModelAndView listOfRegisteredUsers() {

List<User> customerList = userService.fetchUserList().stream()

.filter(user -> user.getDiscriminatorValue().equals("Customer")).collect(Collectors.toList());

ModelAndView modelAndView = new ModelAndView("registeredUserList");

modelAndView.addObject("userList", customerList);

return modelAndView;

}

@PostMapping(path = "/admin/users/search")

public ModelAndView searchUsersList(ModelMap map, javax.servlet.http.HttpServletRequest request,

@RequestParam(value = "searchId", required = false) String searchId) {

/\*

\* ResponseEntity<List<Todo>> responseEntity = restTemplate.exchange(BASE\_URL,

\* HttpMethod.GET,null,new ParameterizedTypeReference<List<Todo>>() { });

\*/

List<User> userList = userRepository.findByfirstNameContainingIgnoreCase(searchId).stream()

.filter(user -> user.getDiscriminatorValue().equals("Customer")).collect(Collectors.toList());

ModelAndView modelAndView = new ModelAndView("registeredUserList");

modelAndView.addObject("userList", userList);

return modelAndView;

}

@GetMapping("/admin/categories")

public String getCategory(Model model) {

model.addAttribute("categories", productCategoryService.fetchProductCategoryList());

return "productCategoryList";

}

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

public String addCategory(Model model) {

return "addProductCategory";

}

@PostMapping("/categories/add")

public String addCategories(@ModelAttribute("productCategory") ProductCategory productCategory,

HttpServletRequest request) throws ServletException {

productCategoryService.saveCategory(productCategory);

return "adminDetails";

}

@GetMapping(path = "/admin/category/Edit")

public ModelAndView showEditCategoryPage(@RequestParam("categoryId") Long theId, Model theModel) {

ProductCategory productCategory = productCategoryService.getCategoryById(theId);

theModel.addAttribute("productCategory", productCategory);

return new ModelAndView("editProductCategory");

}

@GetMapping(path = "/admin/category/Delete")

public ModelAndView deleteCategory(@RequestParam("categoryId") Long theId) {

productCategoryService.deleteCategory(theId);

return new ModelAndView("redirect:/admin/categories");

}

@PostMapping(path = "/category/update/save")

public ModelAndView updateCategory(@ModelAttribute("productCategory") ProductCategory category) {

ModelAndView modelAndView = new ModelAndView("redirect:/admin/categories");

productCategoryService.updateCategory(category);

return modelAndView;

}

@GetMapping("/admin/products")

public String getProductList(Model model) {

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

return "productList";

}

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

public String addProduct(Model model) {

model.addAttribute("categories", productCategoryService.fetchProductCategoryList());

return "addProduct";

}

@PostMapping("admin/product/add")

public ModelAndView updateProduct(@ModelAttribute("product") Product product, HttpServletRequest request)

throws ServletException {

productService.saveProduct(product);

ModelAndView modelAndView = new ModelAndView("redirect:/admin/products");

return modelAndView;

}

@GetMapping(path = "/admin/product/Edit")

public ModelAndView showEditProductPage(@RequestParam("productId") Long theId, Model theModel) {

Product product = productService.getProductById(theId);

theModel.addAttribute("categories", productCategoryService.fetchProductCategoryList());

theModel.addAttribute("product", product);

return new ModelAndView("editProduct");

}

@PostMapping(path = "/admin/product/Edit/save")

public ModelAndView updateProduct(@ModelAttribute("product") Product product) {

ModelAndView modelAndView = new ModelAndView("redirect:/admin/products");

productService.updateProduct(product);

return modelAndView;

}

@GetMapping(path = "/admin/product/Delete")

public ModelAndView deleteProduct(@RequestParam("productId") Long theId) {

System.out.println(theId);

productService.deleteProduct(theId);

return new ModelAndView("redirect:/admin/products");

}

@GetMapping(path = "/admin/purchaseReport")

public ModelAndView purchaseReport() {

List<PurchaseItem> purchaseItemList = purchaseItemService.getAllPurchaseItemList();

ModelAndView modelAndView = new ModelAndView("purchaseReport");

modelAndView.addObject("purchaseItemList", purchaseItemList);

return modelAndView;

}

}

**CartController.java**

package coms.SportyShoes.controller;

import java.math.BigDecimal;

import java.math.RoundingMode;

import java.time.LocalDate;

import java.time.ZoneId;

import java.util.ArrayList;

import java.util.List;

import java.util.Objects;

import javax.servlet.http.HttpSession;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.ui.ModelMap;

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.RequestParam;

import coms.SportyShoes.model.CartItem;

import coms.SportyShoes.model.Product;

import coms.SportyShoes.model.PurchaseItem;

import coms.SportyShoes.model.PurchaseOrder;

import coms.SportyShoes.model.User;

import coms.SportyShoes.service.ProductService;

import coms.SportyShoes.service.PurchaseItemService;

import coms.SportyShoes.service.PurchaseOrderService;

import coms.SportyShoes.service.UserService;

@Controller

public class CartController {

@Autowired

ProductService productService;

@Autowired

UserService userService;

@Autowired

PurchaseOrderService purchaseOrderService;

@Autowired

PurchaseItemService purchaseItemService;

private int cartCount;

@SuppressWarnings("unchecked")

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

public String cartAddItem(@PathVariable Long productId, javax.servlet.http.HttpServletRequest request) {

HttpSession session = request.getSession();

// if cart is already in session then retrieve it else create a new cart list

// and

// save it to session

List<CartItem> cartItems = new ArrayList<CartItem>();

if (session.getAttribute("cart\_items") != null)

cartItems = (List<CartItem>) session.getAttribute("cart\_items");

if (isItemInCart(cartItems, productId)) {

System.out.println("Item already in cart - quantity increased by 1");

} else {

Product product = productService.getProductById(productId);

CartItem item = new CartItem();

item.setProductId(productId);

item.setQuantity(1);

item.setRate(product.getPrice());

BigDecimal dprice = item.getRate().multiply(new BigDecimal(item.getQuantity()));

item.setPrice(dprice);

item.setName(product.getProductName());

item.setImage\_Url(product.getImage\_Url());

cartItems.add(item);

}

BigDecimal totalValue = getCartValue(cartItems);

session.setAttribute("cartCount", cartCount);

session.setAttribute("cartValue", totalValue);

session.setAttribute("cart\_items", cartItems);

return "redirect:/cart";

}

@SuppressWarnings("unchecked")

@GetMapping("/cart/deleteItem")

public String cartDeleteItem(ModelMap map, javax.servlet.http.HttpServletRequest request,

@RequestParam(value = "productId", required = true) Long productId) {

HttpSession session = request.getSession();

List<CartItem> cartItems = new ArrayList<CartItem>();

if (session.getAttribute("cart\_items") != null)

cartItems = (List<CartItem>) session.getAttribute("cart\_items");

for (CartItem item : cartItems) {

if (item.getProductId() == productId) {

if (item.getQuantity() == 1) {

cartItems.remove(item);

BigDecimal totalValue = getCartValue(cartItems);

session.setAttribute("cartValue", totalValue);

System.out.println("###Total " + totalValue);

break;

} else {

item.setQuantity(item.getQuantity() - 1);

BigDecimal totalValue = getCartValue(cartItems);

session.setAttribute("cartValue", totalValue);

System.out.println("###Total " + totalValue);

break;

}

}

}

session.setAttribute("cart\_items", cartItems);

return "redirect:/cart";

}

@SuppressWarnings("unchecked")

@GetMapping("/cart")

public String viewCart(ModelMap map, javax.servlet.http.HttpServletRequest request) {

// check if user is logged in

HttpSession session = request.getSession();

String useremailId = (String) session.getAttribute("userEmailId");

if (Objects.nonNull(useremailId)) {

session.removeAttribute("errMessage");

}

List<CartItem> cartItems = new ArrayList<CartItem>();

if (session.getAttribute("cart\_items") != null)

cartItems = (List<CartItem>) session.getAttribute("cart\_items");

// get total of all cart items

BigDecimal totalValue = getCartValue(cartItems);

System.out.println("###Total " + totalValue);

map.addAttribute("cartValue", totalValue);

map.addAttribute("cartItems", cartItems);

session.setAttribute("cartValue", totalValue);

session.setAttribute("cartItems", cartItems);

return "cart";

}

@GetMapping("/checkout")

public String checkout(Model model, javax.servlet.http.HttpServletRequest request) {

HttpSession session = request.getSession();

String useremailId = (String) session.getAttribute("userEmailId");

if (Objects.isNull(useremailId)) {

session.setAttribute("errMessage", "You need Login before checkout!!");

return "redirect:/cart";

}

BigDecimal cartValue = (BigDecimal) session.getAttribute("cartValue");

System.out.println(cartValue);

if (cartValue != null) {

model.addAttribute("cartTotal", session.getAttribute("cartValue"));

BigDecimal shipping = (BigDecimal) ((BigDecimal) session.getAttribute("cartValue"))

.multiply(new BigDecimal(0.10)).setScale(2, RoundingMode.CEILING);

model.addAttribute("shipping", shipping);

BigDecimal total = shipping.add(cartValue).setScale(2, RoundingMode.CEILING);

model.addAttribute("total", total);

} else {

model.addAttribute("cartTotal", 0);

model.addAttribute("shipping", 0);

model.addAttribute("total", 0);

}

return "checkout";

}

@PostMapping("/payNow")

public String completePurchase(ModelMap map, javax.servlet.http.HttpServletRequest request) {

HttpSession session = request.getSession();

String useremailId = (String) session.getAttribute("userEmailId");

List<CartItem> cartItems = new ArrayList<CartItem>();

if (session.getAttribute("cart\_items") != null)

cartItems = (List<CartItem>) session.getAttribute("cart\_items");

BigDecimal totalValue = getCartValue(cartItems);

User user = userService.getUser(useremailId);

PurchaseOrder purchaseOrder = new PurchaseOrder();

purchaseOrder.setUser(user);

purchaseOrder.setDate(java.sql.Date.valueOf(LocalDate.now(ZoneId.systemDefault())));

purchaseOrder.setTotal(totalValue);

purchaseOrder = purchaseOrderService.savePurchaseOrder(purchaseOrder);

for (CartItem item : cartItems) {

PurchaseItem pItem = new PurchaseItem();

pItem.setPurchaseOrder(purchaseOrder);

pItem.setPoduct(productService.getProductById(item.getProductId()));

pItem.setUser(user);

pItem.setRate(item.getRate());

pItem.setQuantity(item.getQuantity());

pItem.setPrice(item.getPrice());

purchaseItemService.savePurchaseItem(pItem);

}

map.addAttribute("cartValue", 0);

map.addAttribute("cartItems", null);

map.addAttribute("orderNo", purchaseOrder.getID());

session.setAttribute("cartValue", 0);

session.setAttribute("cartItems", null);

cartCount = 0;

session.setAttribute("cartCount", cartCount);

return "confirm";

}

/\*\*

\* Check if an item is already in the cart

\*

\* @param list

\* @param item

\* @return

\*/

private boolean isItemInCart(List<CartItem> list, long item) {

boolean retVal = false;

for (CartItem thisItem : list) {

if (item == thisItem.getProductId()) {

thisItem.setQuantity(thisItem.getQuantity() + 1);

retVal = true;

break;

}

}

return retVal;

}

/\*\*

\* Get total value of items in cart

\*

\* @param list

\* @return

\*/

private BigDecimal getCartValue(List<CartItem> list) {

BigDecimal total = new BigDecimal(0.0);

cartCount = 0;

for (CartItem item : list) {

cartCount = cartCount + item.getQuantity();

BigDecimal dprice = item.getRate().multiply(new BigDecimal(item.getQuantity()));

total = total.add(dprice);

}

return total;

}

}

**HomeController.java**

package coms.SportyShoes.controller;

import java.util.Objects;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

@Controller

public class HomeController {

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

public String home(Model model,javax.servlet.http.HttpServletRequest request ) {

HttpSession session = request.getSession();

if(Objects.isNull(session.getAttribute("cartCount")))

session.setAttribute("cartCount", 0);

return "homePage";

}

}

**LoginController.java**

package coms.SportyShoes.controller;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

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.PathVariable;

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

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

import org.springframework.web.servlet.ModelAndView;

import coms.SportyShoes.model.User;

import coms.SportyShoes.service.UserService;

import coms.SportyShoes.validatator.EmailValidator;

@Controller

public class LoginController {

@Autowired

private UserService userService;

String errMessage;

@PostMapping("/login")

public String loginAction(ModelMap map, javax.servlet.http.HttpServletRequest request,

@RequestParam(value="emailId", required=true) String emailId,

@RequestParam(value="password", required=true) String password)

{

if(EmailValidator.isValid(emailId)) {

User userDB = userService.getUser(emailId,password);

if(userDB!=null) {

System.out.println("############# " + userDB.getFirstName());

String userRole;

if( userDB.getDiscriminatorValue().equals("Admin")) {

userRole = "Admin";

map.addAttribute(userRole);

HttpSession session = request.getSession();

session.setAttribute("userEmailId", emailId);

session.setAttribute("userName", userDB.getFirstName());

return "redirect:/admin";

}

else

{

userRole ="Customer";

map.addAttribute(userRole);

HttpSession session = request.getSession();

session.setAttribute("userEmailId", emailId);

session.setAttribute("userName", userDB.getFirstName());

return "redirect:/homePage";

}

}else {

errMessage = "Invalid Credentials";

map.addAttribute("errorMessage", errMessage);

return "redirect:/login";

}

}else {

errMessage ="Invalid email id";

map.addAttribute("errorMessage", errMessage);

System.out.println("################### "+ errMessage);

return "redirect:/login";

}

}

@GetMapping("/logout")

public String logout(javax.servlet.http.HttpServletRequest request) {

HttpSession session = request.getSession();

if(session !=null) {

session.invalidate();

return "redirect:/homePage";

}

return "redirect:/homePage";

}

@GetMapping("/login/changePassword/{emailID}")

public String changePassword(@PathVariable String emailID,javax.servlet.http.HttpServletRequest request ) {

request.setAttribute("emailIdVal", emailID);

return "changePassword";

}

@GetMapping("/login/changePassword")

public ModelAndView changePassword() {

ModelAndView modelAndView = new ModelAndView("changePassword");

//modelAndView.addObject("message", "test data");

modelAndView.addObject("emailIdVal", "dev@123"); return modelAndView;

}

@PostMapping("/login/changePassword")

public String saveChangePassword(@ModelAttribute("user") User user , HttpServletRequest request) {

User user1 = userService.getUser(user.getEmailId());

System.out.println("##"+ user.getEmailId() );

user1.setPassword(user.getPassword());

userService.updateUser(user1);

request.setAttribute("changePasswordMsg", "Password changed");

return "homePage";

}

@GetMapping("/register")

public String registerGet() {

return "register";

}

@PostMapping("/register/save")

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

userService.saveUser(user);

return "homePage";

}

}

**ProductController.java**

package coms.SportyShoes.controller;

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

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

import coms.SportyShoes.service.\*;

@RestController

public class ProductController {

@Autowired

private ProductService productService;

}

**ShopController.java**

package coms.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 coms.SportyShoes.repository.ProductCategoryRepository;

import coms.SportyShoes.repository.ProductRepository;

import coms.SportyShoes.service.ProductCategoryService;

import coms.SportyShoes.service.ProductService;

@Controller

public class ShopController {

@Autowired

ProductCategoryService productCategoryService;

@Autowired

ProductService productService;

@Autowired

ProductRepository productRepository;

@Autowired

ProductCategoryRepository productCategoryRepository;

@GetMapping("/shop")

public String shop(Model model) {

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

return "shop";

}

@GetMapping({ "/shop/Gender/{gender}", "/shop/Kids/{gender}" })

public String shopByMen(@PathVariable String gender, Model model) {

model.addAttribute("products", productRepository.findAllProductsByTargetGroup(gender));

return "shop";

}

@GetMapping("/shop/brand/{brandName}")

public String shopByBrand(@PathVariable String brandName, Model model) {

model.addAttribute("products", productRepository.findAllProductsByBrand(brandName));

return "shop";

}

@GetMapping("/shop/style/{styleVal}")

public String shopByStyle(@PathVariable String styleVal, Model model) {

model.addAttribute("products", productRepository.findAllProductsByStyle(styleVal));

return "shop";

}

@GetMapping("/shop/viewProduct/{productCode}")

public String viewProduct(@PathVariable Long productCode, Model model) {

System.out.println("## View Product ->Product Code " + productCode);

model.addAttribute("product", productService.getProductById(productCode));

return "viewProduct";

}

}

**UserController.java**

package coms.SportyShoes.controller;

import java.net.URI;

import java.util.List;

import javax.validation.Valid;

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

import org.springframework.http.ResponseEntity;

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

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

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

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

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.support.ServletUriComponentsBuilder;

import coms.SportyShoes.model.User;

import coms.SportyShoes.service.UserService;

@RestController

public class UserController {

@Autowired

private UserService userService;

// fetchAll users

// Http method : get

// uri : /users

@GetMapping("/users")

public List<User> getAllUsers() {

return userService.fetchUserList();

}

@RequestMapping("/user-list")

public List<User> userList(Model model) {

List<User> users = userService.fetchUserList();

model.addAttribute("users", users);

return users;

}

@GetMapping(path = "/users/{emailId}")

public User getUser(@PathVariable String emailId) {

User user = userService.getUser(emailId);

if (user == null) {

throw new RuntimeException("id null");// user not foundexception

}

return user;

}

@PostMapping(path = "/users") // Request body --> json data

public ResponseEntity<User> createUser(@RequestBody User user) {

// System.out.println("username"+ username);

System.out.println(user.getFirstName());

User SaveUser = userService.saveUser(user);

// response header : location

// status code :201

URI location = ServletUriComponentsBuilder.fromCurrentRequest().path("/{id}")

.buildAndExpand(SaveUser.getEmailId()).toUri();

return ResponseEntity.created(location).build();

}

// PUT

@PutMapping(value = "createUser")

public User createUser2(@RequestBody User userDataInput) {

return userService.saveUser(userDataInput);

}

// update

@PutMapping(value = "/users/{emailId}")

public void updateUser(@PathVariable String emailId, @RequestBody User userDataInput) {

userService.updateUser(userDataInput);

}

@GetMapping("/goToViewPage")

public ModelAndView passParametersWithModelAndView() {

ModelAndView modelAndView = new ModelAndView("UserList");

modelAndView.addObject("message", "test data");

return modelAndView;

}

@GetMapping("/login")

public ModelAndView login() {

ModelAndView modelAndView = new ModelAndView("login");

modelAndView.addObject("message", "test data");

return modelAndView;

}

}

1. coms.SportyShoes.model:

**Admin.java**

package coms.SportyShoes.model;

import javax.persistence.DiscriminatorValue;

import javax.persistence.Entity;

@Entity

@DiscriminatorValue("Admin")

public class Admin extends User{

public Admin() {

super();

}

}

**CartItem.java**

**package** coms.SportyShoes.model;

**import** java.math.BigDecimal;

/\*pojo class\*/

**public** **class** CartItem {

**private** Long productId;

**private** String name;

**private** BigDecimal rate; //per item

**private** **int** quantity;

**private** BigDecimal price; //total

**private** String image\_Url;

**public** Long getProductId() {**return** **this**.productId; }

**public** String getName() { **return** **this**.name;}

**public** BigDecimal getPrice() { **return** **this**.price;}

**public** BigDecimal getRate() { **return** **this**.rate;}

**public** **int** getQuantity() { **return** **this**.quantity;}

**public** **void** setProductId(Long id) { **this**.productId= id;}

**public** **void** setName(String value) { **this**.name = value;}

**public** **void** setPrice(BigDecimal value) { **this**.price = value;}

**public** **void** setRate(BigDecimal value) { **this**.rate = value;}

**public** **void** setQuantity(**int** value) { **this**.quantity = value;}

**public** String getImage\_Url() {

**return** image\_Url;

}

**public** **void** setImage\_Url(String image\_Url) {

**this**.image\_Url = image\_Url;

}

}

**Product.java**

package coms.SportyShoes.model;

import java.math.BigDecimal;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import lombok.Getter;

import lombok.NoArgsConstructor;

import lombok.Setter;

import lombok.ToString;

@Entity

@NoArgsConstructor

@Setter

@Getter

@ToString

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long productCode;

private String productName;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "productCategory\_fk")

private ProductCategory productCategory;

private int size;

private BigDecimal price;

private String image\_Url;

public Long getProductCode() {

return productCode;

}

public void setProductCode(Long productCode) {

this.productCode = productCode;

}

public String getProductName() {

return productName;

}

public void setProductName(String productName) {

this.productName = productName;

}

public ProductCategory getProductCategory() {

return productCategory;

}

public void setProductCategory(ProductCategory productCategory) {

this.productCategory = productCategory;

}

public int getSize() {

return size;

}

public void setSize(int size) {

this.size = size;

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

}

public String getImage\_Url() {

return image\_Url;

}

public void setImage\_Url(String image\_Url) {

this.image\_Url = image\_Url;

}

@Override

public String toString() {

return "Product [productCode=" + productCode + ", productName=" + productName + ", productCategory="

+ productCategory + ", size=" + size + ", price=" + price + ", image\_Url=" + image\_Url + "]";

}

public Product() {

}

public Product(Long productCode, String productName, ProductCategory productCategory, int size, BigDecimal price,

String image\_Url) {

super();

this.productCode = productCode;

this.productName = productName;

this.productCategory = productCategory;

this.size = size;

this.price = price;

this.image\_Url = image\_Url;

}

}

**ProductCategory.java**

package coms.SportyShoes.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class ProductCategory {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String brandName;

private String style; // wear hiking sneaker biking basketball

private String targetGroup;//gender M F kids- boys girl

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getBrandName() {

return brandName;

}

public void setBrandName(String brandName) {

this.brandName = brandName;

}

public String getStyle() {

return style;

}

public void setStyle(String style) {

this.style = style;

}

public String getTargetGroup() {

return targetGroup;

}

public void setTargetGroup(String targetGroup) {

this.targetGroup = targetGroup;

}

@Override

public String toString() {

return "ProductCategory [id=" + id + ", brandName=" + brandName + ", style=" + style + ", targetGroup="

+ targetGroup + "]";

}

public ProductCategory(Long id, String brandName, String style, String targetGroup) {

super();

this.id = id;

this.brandName = brandName;

this.style = style;

this.targetGroup = targetGroup;

}

public ProductCategory() {

}

}

**PurchaseItem.java**

package coms.SportyShoes.model;

import java.math.BigDecimal;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import javax.persistence.Table;

@Entity

@Table

public class PurchaseItem {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long ID;

@ManyToOne

@JoinColumn(name = "purchaseOrder\_fk")

private PurchaseOrder purchaseOrder;

@ManyToOne

@JoinColumn(name = "product\_fk")

private Product poduct;

@ManyToOne

@JoinColumn(name = "user\_fk")

private User user;

private BigDecimal rate;

private int quantity;

private BigDecimal price;

public PurchaseItem(Long iD, Product poduct, User user, BigDecimal rate, int quantity, BigDecimal price) {

super();

ID = iD;

this.poduct = poduct;

this.user = user;

this.rate = rate;

this.quantity = quantity;

this.price = price;

}

public PurchaseItem() {

}

public Long getID() {

return ID;

}

public void setID(Long iD) {

ID = iD;

}

public Product getPoduct() {

return poduct;

}

public void setPoduct(Product poduct) {

this.poduct = poduct;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

public BigDecimal getRate() {

return rate;

}

public void setRate(BigDecimal rate) {

this.rate = rate;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

}

public PurchaseOrder getPurchaseOrder() {

return purchaseOrder;

}

public void setPurchaseOrder(PurchaseOrder purchaseOrder) {

this.purchaseOrder = purchaseOrder;

}

@Override

public String toString() {

return "PurchaseItem [ID=" + ID + ", poduct=" + poduct + ", user=" + user + ", rate=" + rate + ", quantity="

+ quantity + ", price=" + price + "]";

}

}

**PurchaseOrder.java**

package coms.SportyShoes.model;

import java.math.BigDecimal;

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.JoinColumn;

import javax.persistence.ManyToOne;

import javax.persistence.Temporal;

import javax.persistence.TemporalType;

@Entity

public class PurchaseOrder {

@Id

@GeneratedValue(strategy = GenerationType.SEQUENCE)

@Column(name = "ID")

private Long ID;

@ManyToOne

@JoinColumn(name = "user\_fk")

private User user;

@Temporal(TemporalType.DATE)

private Date date;

@Column(name = "gross\_total")

private BigDecimal total;

public PurchaseOrder(Long iD, User user, Date date, BigDecimal total) {

super();

ID = iD;

this.user = user;

this.date = date;

this.total = total;

}

public Long getID() {

return ID;

}

public void setID(Long iD) {

ID = iD;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

public Date getDate() {

return date;

}

public void setDate(Date date) {

this.date = date;

}

public BigDecimal getTotal() {

return total;

}

public void setTotal(BigDecimal total) {

this.total = total;

}

@Override

public String toString() {

return "PurchaseOrder [ID=" + ID + ", user=" + user + ", date=" + date + ", total=" + total + "]";

}

public PurchaseOrder() {

}

}

**User.java**

package coms.SportyShoes.model;

import java.util.Date;

import java.util.List;

import javax.persistence.Column;

import javax.persistence.DiscriminatorColumn;

import javax.persistence.DiscriminatorType;

import javax.persistence.DiscriminatorValue;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Inheritance;

import javax.persistence.InheritanceType;

import javax.persistence.Table;

import javax.persistence.Temporal;

import javax.persistence.TemporalType;

import javax.persistence.Transient;

import lombok.Getter;

import lombok.NoArgsConstructor;

import lombok.Setter;

import lombok.ToString;

@Entity

@Table(name = "USER")

@Inheritance(strategy = InheritanceType.SINGLE\_TABLE)

@DiscriminatorColumn(discriminatorType = DiscriminatorType.STRING, name = "User\_Role")

@DiscriminatorValue("Customer")

/\*

\* @Setter

\*

\* @Getter

\*

\* @NoArgsConstructor

\*/

public class User {

public User() {

}

@Id

private String emailId;

@Column(length = 20)

private String password;

@Column

private String firstName;

@Column

private String lastName;

@Column

private String country;

@Column

private String gender;

@Temporal(TemporalType.DATE)

private Date dateOfBirth;

@Override

public String toString() {

return "User [emailId=" + emailId + ", password=" + password + ", firstName=" + firstName + ", lastName="

+ lastName + ", country=" + country + ", gender=" + gender + ", dateOfBirth=" + dateOfBirth + "]";

}

public User(String emailId, String password, String firstName, String lastName, String country, String gender,

Date dateOfBirth) {

super();

this.emailId = emailId;

this.password = password;

this.firstName = firstName;

this.lastName = lastName;

this.country = country;

this.gender = gender;

this.dateOfBirth = dateOfBirth;

}

public String getEmailId() {

return emailId;

}

public void setEmailId(String emailId) {

this.emailId = emailId;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

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 getCountry() {

return country;

}

public void setCountry(String country) {

this.country = country;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public Date getDateOfBirth() {

return dateOfBirth;

}

public void setDateOfBirth(Date dateOfBirth) {

this.dateOfBirth = dateOfBirth;

}

@Transient

public String getDiscriminatorValue() {

return this.getClass().getAnnotation(DiscriminatorValue.class).value();

}

/\*

\* @OneToMany private List<Purchase> listOfPurchases; // cartID

\*/}

1. Coms.SportyShoes.repository:

**AdminRepository.java**

package coms.SportyShoes.repository;

import java.util.List;

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

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

import coms.SportyShoes.model.Admin;

public interface AdminRepository extends JpaRepository<Admin, String>{

@Query("from Admin")

public List<Admin> getUsers();

}

**ProductCategoryRepository.java**

package coms.SportyShoes.repository;

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

import org.springframework.stereotype.Repository;

import coms.SportyShoes.model.ProductCategory;

@Repository

public interface ProductCategoryRepository extends JpaRepository<ProductCategory,Long >{

}

**ProductRepository.java**

package coms.SportyShoes.repository;

import java.util.List;

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

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

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

import org.springframework.stereotype.Repository;

import coms.SportyShoes.model.Product;

import coms.SportyShoes.model.ProductCategory;

@Repository

public interface ProductRepository extends JpaRepository<Product, Long> {

List<Product>findByProductCategory(ProductCategory productCategory);

@Query(value = "SELECT \* FROM product JOIN product\_category ON product.product\_category\_fk=product\_category.id AND product\_category.brand\_name =:brand\_name",nativeQuery = true)

List<Product> findAllProductsByBrand(@Param("brand\_name") String brand\_name);

@Query(value = "SELECT \* FROM product JOIN product\_category ON product.product\_category\_fk=product\_category.id AND product\_category.style =:style",nativeQuery = true)

List<Product> findAllProductsByStyle(@Param("style") String style);

@Query(value = "SELECT \* FROM product JOIN product\_category ON product.product\_category\_fk=product\_category.id AND product\_category.target\_group =:target\_group", nativeQuery = true)

List<Product> findAllProductsByTargetGroup(@Param("target\_group") String target\_group);

}

**PurchaseItemRepository.java**

package coms.SportyShoes.repository;

import java.util.List;

import org.springframework.data.domain.Sort;

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

import org.springframework.stereotype.Repository;

import coms.SportyShoes.model.PurchaseItem;

import coms.SportyShoes.model.User;

@Repository

public interface PurchaseItemRepository extends JpaRepository<PurchaseItem, Long>{

List<PurchaseItem> findByUser(User user, Sort sort);

}

**PurchaseOrderRepository.java**

package coms.SportyShoes.repository;

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

import coms.SportyShoes.model.PurchaseOrder;

public interface PurchaseOrderRepository extends JpaRepository<PurchaseOrder, Long> {

}

**UserRepository.java**

package coms.SportyShoes.repository;

import java.util.List;

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

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

import org.springframework.stereotype.Repository;

import coms.SportyShoes.model.User;

@Repository

public interface UserRepository extends JpaRepository<User, String>{

User findByEmailIdAndPassword(String emailId, String password);

List<User> findByfirstNameContaining(String firstName);

List<User> findByfirstNameContainingIgnoreCase(String firstName);

@Query("from User")

public List<User> getUsers();

}

1. Coms.SportyShoes.service:

**UserService.java**

package coms.SportyShoes.service;

import java.util.List;

import coms.SportyShoes.model.User;

public interface UserService {

public List<User> fetchUserList();

public User saveUser(User user);

public void deleteByEmailId(String emailId);

public User getUser(String emailId);

public void updateUser(User user);

public User getUser(String emailId, String password);

}

**UserServiceImpl.java**

package coms.SportyShoes.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import coms.SportyShoes.model.User;

import coms.SportyShoes.repository.UserRepository;

@Service(value = "userService")//service interface first letter is small

public class UserServiceImpl implements UserService{

@Autowired

private UserRepository userRepository;

@Override

public List<User> fetchUserList() {

System.out.println(userRepository.findAll().toString());

return (List<User>) userRepository.findAll();

}

@Override

public User saveUser(User user) {

System.out.println("Save User");

return userRepository.save(user);

}

@Override

public void deleteByEmailId(String emailId) {

userRepository.deleteById(emailId);

}

@Override

public User getUser(String emailId) {

return userRepository.findById(emailId).get();

}

@Override

public User getUser(String emailId, String password) {

return userRepository.findByEmailIdAndPassword(emailId, password);

}

@Override

public void updateUser(User user) {

System.out.println(user.getEmailId() + user.getFirstName());

if(userRepository.findById(user.getEmailId()).get() != null)

userRepository.save(user);

System.out.println(user.getEmailId() + user.getFirstName());

}

}

**ProductCategoryService.java**

package coms.SportyShoes.service;

import java.util.List;

import coms.SportyShoes.model.ProductCategory;

public interface ProductCategoryService {

public ProductCategory getCategoryById(Long id);

public void updateCategory(ProductCategory category);

public ProductCategory saveCategory(ProductCategory category);

public void deleteCategory(Long id) ;

public List<ProductCategory> fetchProductCategoryList();

}

**ProductCategoryServiceImpl.java**

package coms.SportyShoes.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import coms.SportyShoes.model.ProductCategory;

import coms.SportyShoes.repository.ProductCategoryRepository;

@Service(value = "productCategoryService")

public class ProductCategoryServiceImpl implements ProductCategoryService {

@Autowired

ProductCategoryRepository productCategoryRepository;

@Override

public ProductCategory getCategoryById(Long id) {

// TODO Auto-generated method stub

return productCategoryRepository.findById(id).get();

}

@Override

public void updateCategory(ProductCategory category) {

productCategoryRepository.save(category);

}

@Override

public void deleteCategory(Long id) {

productCategoryRepository.deleteById(id);

}

@Override

public ProductCategory saveCategory(ProductCategory category) {

return productCategoryRepository.save(category);

}

@Override

public List<ProductCategory> fetchProductCategoryList() {

return productCategoryRepository.findAll();

}

}

**ProductyService.java**

package coms.SportyShoes.service;

import java.util.List;

import coms.SportyShoes.model.Product;

import coms.SportyShoes.model.ProductCategory;

public interface ProductService {

public Product getProductById(Long productCode);

public Product getProductByProductCategory(ProductCategory productCategory);

public void updateProduct(Product product);

public Product saveProduct(Product product);

public void deleteProduct(Long productCode);

public List<Product> getAllProducts();

}

**ProductServiceImpl.java**

package coms.SportyShoes.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import coms.SportyShoes.model.Product;

import coms.SportyShoes.model.ProductCategory;

import coms.SportyShoes.repository.ProductRepository;

@Service(value = "productService")

public class ProductServiceImpl implements ProductService{

@Autowired

private ProductRepository productRepository;

@Override

public Product getProductById(Long productCode) {

return productRepository.findById(productCode).get();

}

@Override

public void updateProduct(Product product) {

if(productRepository.findById(product.getProductCode()).get() != null) {// add conditions if not found

productRepository.save(product);

}

}

@Override

public void deleteProduct(Long productCode) {

productRepository.deleteById(productCode);

}

@Override

public List<Product> getAllProducts() {

return productRepository.findAll();

}

@Override

public Product getProductByProductCategory(ProductCategory productCategory) {

// TODO Auto-generated method stub

//productRepository.fin

return null;

}

@Override

public Product saveProduct(Product product) {

return productRepository.save(product);

}

}

**PurchaseItemService.java**

package coms.SportyShoes.service;

import java.util.List;

import coms.SportyShoes.model.PurchaseItem;

import coms.SportyShoes.model.PurchaseOrder;

import coms.SportyShoes.model.User;

public interface PurchaseItemService {

public PurchaseItem getPurchaseItemById(Long id);

public List<PurchaseItem> getAllItemsByPurchaseOrder(PurchaseOrder order);

public List<PurchaseItem> getAllPurchaseItemByUserId(User userId);

public PurchaseItem savePurchaseItem(PurchaseItem purchaseItem);

public List<PurchaseItem> getAllPurchaseItemList();

}

**PurchaseItremSrviceImpl.java**

package coms.SportyShoes.service;

import java.util.List;

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

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import coms.SportyShoes.model.PurchaseItem;

import coms.SportyShoes.model.PurchaseOrder;

import coms.SportyShoes.model.User;

import coms.SportyShoes.repository.PurchaseItemRepository;

@Service(value = "purchaseItemService")

public class PurchaseItemServiceImpl implements PurchaseItemService {

@Autowired

PurchaseItemRepository purchaseItemRepository;

@Override

public PurchaseItem getPurchaseItemById(Long id) {

// TODO Auto-generated method stub

return null;

}

@Override

public List<PurchaseItem> getAllItemsByPurchaseOrder(PurchaseOrder order) {

// TODO Auto-generated method stub

return null;

}

@Override

public List<PurchaseItem> getAllPurchaseItemByUserId(User userId) {

// TODO Auto-generated method stub

return null;

}

@Override

public PurchaseItem savePurchaseItem(PurchaseItem purchaseItem) {

purchaseItemRepository.save(purchaseItem);

return purchaseItem;

}

@Override

public List<PurchaseItem> getAllPurchaseItemList() {

return purchaseItemRepository.findAll(Sort.by(Sort.Direction.ASC, "purchaseOrder\_ID"));

}

}

**PurchaseOrderService.java**

package coms.SportyShoes.service;

import java.util.List;

import coms.SportyShoes.model.PurchaseOrder;

public interface PurchaseOrderService {

public PurchaseOrder getPurchaseById(long id);

public List<PurchaseOrder> getAllPurchaseOrders();

public List<PurchaseOrder> getAllPurchaseOrderByUserId(Long userId);

public PurchaseOrder savePurchaseOrder(PurchaseOrder purchaseOrder);

}

**PurchaseOrderServiceImpl.java**

package coms.SportyShoes.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import coms.SportyShoes.model.PurchaseOrder;

import coms.SportyShoes.repository.PurchaseOrderRepository;

@Service(value = "purchaseOrderService")

public class PurchaseOrderServiceImpl implements PurchaseOrderService {

@Autowired

PurchaseOrderRepository purchaseOrderRepository;

@Override

public PurchaseOrder getPurchaseById(long id) {

// TODO Auto-generated method stub

return null;

}

@Override

public List<PurchaseOrder> getAllPurchaseOrders() {

// TODO Auto-generated method stub

return null;

}

@Override

public List<PurchaseOrder> getAllPurchaseOrderByUserId(Long userId) {

// TODO Auto-generated method stub

return null;

}

@Override

public PurchaseOrder savePurchaseOrder(PurchaseOrder purchaseOrder) {

purchaseOrderRepository.save(purchaseOrder);

return purchaseOrder;

}

}

1. Coms.SportyShoes.Validatator:

**EmailValidator.java**

package coms.SportyShoes.validatator;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class EmailValidator{

private static Pattern pattern;

private static Matcher matcher;

private static final String EMAIL\_PATTERN =

"^[\_A-Za-z0-9-]+(\\.[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$";

public EmailValidator(){

pattern = Pattern.compile(EMAIL\_PATTERN);

}

/\*\*

\* Validate with regular expression

\* @param email for validation

\* @return true valid email, false invalid email

\*/

public static boolean isValid(final String email){

pattern = Pattern.compile(EMAIL\_PATTERN);

matcher = pattern.matcher(email);

return matcher.matches();

}

}

**Application.properties**

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

spring.datasource.username=root

spring.datasource.password=Nilesh@1991

spring.sql.init.mode=always

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

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

spring.jpa.defer-datasource-initialization=true

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update

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

spring.mvc.view.suffix=.jsp

**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>2.7.6</version>

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

</parent>

<groupId>coms</groupId>

<artifactId>SportyShoes</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>SportyShoes</name>

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

<properties>

<java.version>11</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-validation</artifactId>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet.jsp.jstl</groupId>

<artifactId>javax.servlet.jsp.jstl-api</artifactId>

<version>1.2.1</version>

</dependency>

<dependency>

<groupId>org.glassfish.web</groupId>

<artifactId>javax.servlet.jsp.jstl</artifactId>

<version>1.2.1</version>

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>