**Controller Code :**

**CartController.java**

**package** com.foodbox.controller;  
  
**import** com.fsd.foodbox.service.CartService;  
//import org.slf4j.Logger;  
//import org.slf4j.LoggerFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.bind.annotation.RequestParam;  
**import** org.springframework.web.servlet.ModelAndView;  
  
@Controller  
@RequestMapping({"/cart"})  
**public** **class** CartController {  
 //private static final Logger logger = LoggerFactory.getLogger(com.foodbox.controller.CartController.class);  
   
 @Autowired  
 **private** CartService cartService;  
   
 @RequestMapping({"/show"})  
 **public** ModelAndView showCart(@RequestParam(name = "result", required = **false**) String result) {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "Shopping Cart");  
 mv.addObject("userClickShowCart", Boolean.valueOf(**true**));  
 **if** (result != **null**) {  
 String str;  
 **switch** ((str = result).hashCode()) {  
 **case** -665462704:  
 **if** (!str.equals("unavailable"))  
 **break**;   
 mv.addObject("message", "Product quantity is not available!");  
 **break**;  
 **case** -615513399:  
 **if** (!str.equals("modified"))  
 **break**;   
 mv.addObject("message", "One or more items inside cart has been modified!");  
 **break**;  
 **case** -234430277:  
 **if** (!str.equals("updated"))  
 **break**;   
 mv.addObject("message", "Cart has been updated successfully!");  
 **this**.cartService.validateCartLine();  
 **break**;  
 **case** 92659968:  
 **if** (!str.equals("added"))  
 **break**;   
 mv.addObject("message", "Product has been successfully added inside cart!");  
 **this**.cartService.validateCartLine();  
 **break**;  
 **case** 844740128:  
 **if** (!str.equals("maximum"))  
 **break**;   
 mv.addObject("message", "Maximum limit for the item has been reached!");  
 **break**;  
 **case** 1550463001:  
 **if** (!str.equals("deleted"))  
 **break**;   
 mv.addObject("message", "CartLine has been successfully removed!");  
 **break**;  
 }   
 } **else** {  
 String response = **this**.cartService.validateCartLine();  
 **if** (response.equals("result=modified"))  
 mv.addObject("message", "One or more items inside cart has been modified!");   
 }   
 mv.addObject("cartLines", **this**.cartService.getCartLines());  
 **return** mv;  
 }  
   
 @RequestMapping({"/{cartLineId}/update"})  
 **public** String udpateCartLine(@PathVariable **int** cartLineId, @RequestParam **int** count) {  
 String response = **this**.cartService.manageCartLine(cartLineId, count);  
 **return** "redirect:/cart/show?" + response;  
 }  
   
 @RequestMapping({"/add/{productId}/product"})  
 **public** String addCartLine(@PathVariable **int** productId) {  
 String response = **this**.cartService.addCartLine(productId);  
 **return** "redirect:/cart/show?" + response;  
 }  
   
 @RequestMapping({"/{cartLineId}/remove"})  
 **public** String removeCartLine(@PathVariable **int** cartLineId) {  
 String response = **this**.cartService.removeCartLine(cartLineId);  
 **return** "redirect:/cart/show?" + response;  
 }  
   
 @RequestMapping({"/validate"})  
 **public** String validateCart() {  
 String response = **this**.cartService.validateCartLine();  
 **if** (!response.equals("result=success"))  
 **return** "redirect:/cart/show?" + response;   
 **return** "redirect:/cart/checkout";  
 }  
}

GlobalController.java

**package** com.foodbox.controller;  
  
**import** javax.servlet.http.HttpSession;  
**import** com.fsd.foodbox.model.UserModel;  
**import** com.fsd.foodboxbackend.dao.UserDAO;  
**import** com.fsd.foodboxbackend.dto.User;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.security.core.Authentication;  
**import** org.springframework.security.core.context.SecurityContextHolder;  
**import** org.springframework.web.bind.annotation.ControllerAdvice;  
**import** org.springframework.web.bind.annotation.ModelAttribute;  
  
@ControllerAdvice  
**public** **class** GlobalController {  
 @Autowired  
 **private** UserDAO userDAO;  
   
 @Autowired  
 **private** HttpSession session;  
   
 **private** UserModel userModel = **null**;  
   
 **private** User user = **null**;  
   
 @ModelAttribute("userModel")  
 **public** UserModel getUserModel() {  
 **if** (**this**.session.getAttribute("userModel") == **null**) {  
 Authentication authentication = SecurityContextHolder.getContext().getAuthentication();  
 **if** (!authentication.getPrincipal().equals("anonymousUser")) {  
 **this**.user = **this**.userDAO.getByEmail(authentication.getName());  
 **if** (**this**.user != **null**) {  
 **this**.userModel = **new** UserModel();  
 **this**.userModel.setId(**this**.user.getId());  
 **this**.userModel.setFullName(String.valueOf(**this**.user.getFirstName()) + " " + **this**.user.getLastName());  
 **this**.userModel.setRole(**this**.user.getRole());  
 **if** (**this**.user.getRole().equals("USER"))  
 **this**.userModel.setCart(**this**.user.getCart());   
 **this**.session.setAttribute("userModel", **this**.userModel);  
 **return** **this**.userModel;  
 }   
 }   
 }   
 **return** (UserModel)**this**.session.getAttribute("userModel");  
 }  
}

**JsonDataController.java**

**package** com.foodbox.controller;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dto.Product;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.bind.annotation.ResponseBody;  
  
@Controller  
@RequestMapping({"/json/data"})  
**public** **class** JsonDataController {  
 @Autowired  
 **private** ProductDAO productDAO;  
   
 @RequestMapping({"/admin/all/products"})  
 @ResponseBody  
 **public** List<Product> getAllProductsList() {  
 **return** **this**.productDAO.list();  
 }  
   
 @RequestMapping({"/all/products"})  
 @ResponseBody  
 **public** List<Product> getAllProducts() {  
 **return** **this**.productDAO.listActiveProducts();  
 }  
   
 @RequestMapping({"/cuisine/{id}/products"})  
 @ResponseBody  
 **public** List<Product> getProductsByCuisine(@PathVariable **int** id) {  
 **return** **this**.productDAO.listActiveProductsByCuisine(id);  
 }  
   
 @RequestMapping({"/mv/products"})  
 @ResponseBody  
 **public** List<Product> getMostViewedProducts() {  
 **return** **this**.productDAO.getProductsByParam("views", 5);  
 }  
   
 @RequestMapping({"/mp/products"})  
 @ResponseBody  
 **public** List<Product> getMostPurchasedProducts() {  
 **return** **this**.productDAO.getProductsByParam("purchases", 5);  
 }  
}

ManagementController.java

**package** com.foodbox.controller;  
  
**import** java.util.List;  
**import** javax.servlet.http.HttpServletRequest;  
**import** javax.validation.Valid;  
**import** com.fsd.foodbox.util.FileUtil;  
**import** com.fsd.foodbox.validator.ProductValidator;  
**import** com.fsd.foodboxbackend.dao.CuisineDAO;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dto.Cuisine;  
**import** com.fsd.foodboxbackend.dto.Product;  
//import org.slf4j.Logger;  
//import org.slf4j.LoggerFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.ui.Model;  
**import** org.springframework.validation.BindingResult;  
**import** org.springframework.validation.Errors;  
**import** org.springframework.web.bind.annotation.ModelAttribute;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**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.ResponseBody;  
**import** org.springframework.web.servlet.ModelAndView;  
  
@Controller  
@RequestMapping({"/manage"})  
**public** **class** ManagementController {  
 //private static final Logger logger = LoggerFactory.getLogger(com.foodbox.controller.ManagementController.class);  
   
 @Autowired  
 **private** ProductDAO productDAO;  
   
 @Autowired  
 **private** CuisineDAO cuisineDAO;  
   
 @RequestMapping({"/product"})  
 **public** ModelAndView manageProduct(@RequestParam(name = "success", required = **false**) String success) {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "Product Management");  
 mv.addObject("userClickManageProduct", Boolean.valueOf(**true**));  
 Product nProduct = **new** Product();  
 nProduct.setSupplierId(1);  
 nProduct.setActive(**true**);  
 mv.addObject("product", nProduct);  
 **if** (success != **null**)  
 **if** (success.equals("product")) {  
 mv.addObject("message", "Product submitted successfully!");  
 } **else** **if** (success.equals("cuisine")) {  
 mv.addObject("message", "Cuisine submitted successfully!");  
 }   
 **return** mv;  
 }  
   
 @RequestMapping({"/{id}/product"})  
 **public** ModelAndView manageProductEdit(@PathVariable **int** id) {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "Product Management");  
 mv.addObject("userClickManageProduct", Boolean.valueOf(**true**));  
 mv.addObject("product", **this**.productDAO.get(id));  
 **return** mv;  
 }  
   
 @RequestMapping(value = {"/product"}, method = {RequestMethod.POST})  
 **public** String managePostProduct(@Valid @ModelAttribute("product") Product mProduct, BindingResult results, Model model, HttpServletRequest request) {  
 **if** (mProduct.getId() == 0) {  
 (**new** ProductValidator()).validate(mProduct, (Errors)results);  
 } **else** **if** (!mProduct.getFile().getOriginalFilename().equals("")) {  
 (**new** ProductValidator()).validate(mProduct, (Errors)results);  
 }   
 **if** (results.hasErrors()) {  
 model.addAttribute("message", "Validation fails for adding the product!");  
 model.addAttribute("userClickManageProduct", Boolean.valueOf(**true**));  
 **return** "page";  
 }   
 **if** (mProduct.getId() == 0) {  
 **this**.productDAO.add(mProduct);  
 } **else** {  
 **this**.productDAO.update(mProduct);  
 }   
 **if** (!mProduct.getFile().getOriginalFilename().equals(""))  
 FileUtil.uploadFile(request, mProduct.getFile(), mProduct.getCode());   
 **return** "redirect:/manage/product?success=product";  
 }  
   
 @RequestMapping(value = {"/product/{id}/activation"}, method = {RequestMethod.GET})  
 @ResponseBody  
 **public** String managePostProductActivation(@PathVariable **int** id) {  
 Product product = **this**.productDAO.get(id);  
 **boolean** isActive = product.isActive();  
 product.setActive(!isActive);  
 **this**.productDAO.update(product);  
 **return** isActive ? "Product Dectivated Successfully!" : "Product Activated Successfully";  
 }  
   
 @RequestMapping(value = {"/cuisine"}, method = {RequestMethod.POST})  
 **public** String managePostCuisine(@ModelAttribute("cuisine") Cuisine mCuisine, HttpServletRequest request) {  
 **this**.cuisineDAO.add(mCuisine);  
 **return** "redirect:" + request.getHeader("Referer") + "?success=cuisine";  
 }  
   
 @ModelAttribute("categories")  
 **public** List<Cuisine> modelCategories() {  
 **return** **this**.cuisineDAO.list();  
 }  
   
 @ModelAttribute("cuisine")  
 **public** Cuisine modelCuisine() {  
 **return** **new** Cuisine();  
 }  
}

PageController.java

**package** com.foodbox.controller;  
  
**import** javax.servlet.http.HttpServletRequest;  
**import** javax.servlet.http.HttpServletResponse;  
**import** com.fsd.foodbox.exception.ProductNotFoundException;  
**import** com.fsd.foodboxbackend.dao.CuisineDAO;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dto.Cuisine;  
**import** com.fsd.foodboxbackend.dto.Product;  
//import org.slf4j.Logger;  
//import org.slf4j.LoggerFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.security.core.Authentication;  
**import** org.springframework.security.core.context.SecurityContextHolder;  
**import** org.springframework.security.web.authentication.logout.SecurityContextLogoutHandler;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.bind.annotation.RequestParam;  
**import** org.springframework.web.servlet.ModelAndView;  
  
@Controller  
**public** **class** PageController {  
 //private static final Logger logger = LoggerFactory.getLogger(com.foodbox.controller.PageController.class);  
   
 @Autowired  
 **private** CuisineDAO cuisineDAO;  
   
 @Autowired  
 **private** ProductDAO productDAO;  
   
 @RequestMapping({"/", "/home", "/index"})  
 **public** ModelAndView index(@RequestParam(name = "logout", required = **false**) String logout) {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "Home");  
 // logger.info("Inside PageController index method - INFO");  
 // logger.debug("Inside PageController index method - DEBUG");  
 mv.addObject("categories", **this**.cuisineDAO.list());  
 **if** (logout != **null**)  
 mv.addObject("message", "You have successfully logged out!");   
 mv.addObject("userClickHome", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/about"})  
 **public** ModelAndView about() {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "About Us");  
 mv.addObject("userClickAbout", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/contact"})  
 **public** ModelAndView contact() {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "Contact Us");  
 mv.addObject("userClickContact", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/show/all/products"})  
 **public** ModelAndView showAllProducts() {  
 ModelAndView mv = **new** ModelAndView("page");  
 mv.addObject("title", "All Food Items");  
 mv.addObject("categories", **this**.cuisineDAO.list());  
 mv.addObject("userClickAllProducts", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/show/cuisine/{id}/products"})  
 **public** ModelAndView showCuisineProducts(@PathVariable("id") **int** id) {  
 ModelAndView mv = **new** ModelAndView("page");  
 Cuisine cuisine = **null**;  
 cuisine = **this**.cuisineDAO.get(id);  
 mv.addObject("title", cuisine.getName());  
 mv.addObject("categories", **this**.cuisineDAO.list());  
 mv.addObject("cuisine", cuisine);  
 mv.addObject("userClickCuisineProducts", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/show/{id}/product"})  
 **public** ModelAndView showSingleProduct(@PathVariable **int** id) **throws** ProductNotFoundException {  
 ModelAndView mv = **new** ModelAndView("page");  
 Product product = **this**.productDAO.get(id);  
 **if** (product == **null**)  
 **throw** **new** ProductNotFoundException();   
 product.setViews(product.getViews() + 1);  
 **this**.productDAO.update(product);  
 mv.addObject("title", product.getName());  
 mv.addObject("product", product);  
 mv.addObject("userClickShowProduct", Boolean.valueOf(**true**));  
 **return** mv;  
 }  
   
 @RequestMapping({"/membership"})  
 **public** ModelAndView register() {  
 ModelAndView mv = **new** ModelAndView("page");  
 //logger.info("Page Controller membership called!");  
 **return** mv;  
 }  
   
 @RequestMapping({"/login"})  
 **public** ModelAndView login(@RequestParam(name = "error", required = **false**) String error, @RequestParam(name = "logout", required = **false**) String logout) {  
 ModelAndView mv = **new** ModelAndView("login");  
 mv.addObject("title", "Login");  
 **if** (error != **null**)  
 mv.addObject("message", "Username and Password is invalid!");   
 **if** (logout != **null**)  
 mv.addObject("logout", "You have logged out successfully!");   
 **return** mv;  
 }  
   
 @RequestMapping({"/logout"})  
 **public** String logout(HttpServletRequest request, HttpServletResponse response) {  
 Authentication auth = SecurityContextHolder.getContext().getAuthentication();  
 **if** (auth != **null**)  
 (**new** SecurityContextLogoutHandler()).logout(request, response, auth);   
 **return** "redirect:/login?logout";  
 }  
   
 @RequestMapping({"/access-denied"})  
 **public** ModelAndView accessDenied() {  
 ModelAndView mv = **new** ModelAndView("error");  
 mv.addObject("errorTitle", "Aha! Caught You.");  
 mv.addObject("errorDescription", "You are not authorized to view this page!");  
 mv.addObject("title", "403 Access Denied");  
 **return** mv;  
 }  
}

Handler:-

**package** com.fsd.foodbox.handler;  
  
  
**import** java.util.ArrayList;  
**import** java.util.Date;  
**import** java.util.List;  
**import** javax.servlet.http.HttpSession;  
**import** com.fsd.foodbox.model.CheckoutModel;  
**import** com.fsd.foodbox.model.UserModel;  
**import** com.fsd.foodboxbackend.dao.CartLineDAO;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dao.UserDAO;  
**import** com.fsd.foodboxbackend.dto.Address;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.CartLine;  
**import** com.fsd.foodboxbackend.dto.OrderDetail;  
**import** com.fsd.foodboxbackend.dto.OrderItem;  
**import** com.fsd.foodboxbackend.dto.Product;  
**import** com.fsd.foodboxbackend.dto.User;  
//import org.slf4j.Logger;  
//import org.slf4j.LoggerFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Component;  
  
@Component  
**public** **class** CheckoutHandler {  
// private static final Logger logger = LoggerFactory.getLogger(com.fsd.foodbox.handler.CheckoutHandler.class);  
   
 @Autowired  
 **private** UserDAO userDAO;  
   
 @Autowired  
 **private** ProductDAO productDAO;  
   
 @Autowired  
 **private** CartLineDAO cartLineDAO;  
   
 @Autowired  
 **private** HttpSession session;  
   
 **public** CheckoutModel init(String name) **throws** Exception {  
 User user = **this**.userDAO.getByEmail(name);  
 CheckoutModel checkoutModel = **null**;  
 **if** (user != **null**) {  
 checkoutModel = **new** CheckoutModel();  
 checkoutModel.setUser(user);  
 checkoutModel.setCart(user.getCart());  
 **double** checkoutTotal = 0.0D;  
 List<CartLine> cartLines = **this**.cartLineDAO.listAvailable(user.getCart().getId());  
 **if** (cartLines.size() == 0)  
 **throw** **new** Exception("There are no products available for checkout now!");   
 **for** (CartLine cartLine : cartLines)  
 checkoutTotal += cartLine.getTotal();   
 checkoutModel.setCartLines(cartLines);  
 checkoutModel.setCheckoutTotal(checkoutTotal);  
 }   
 **return** checkoutModel;  
 }  
   
 **public** List<Address> getShippingAddresses(CheckoutModel model) {  
 List<Address> addresses = **this**.userDAO.listShippingAddresses(model.getUser().getId());  
 **if** (addresses.size() == 0)  
 addresses = **new** ArrayList<>();   
 addresses.add(addresses.size(), **this**.userDAO.getBillingAddress(model.getUser().getId()));  
 **return** addresses;  
 }  
   
 **public** String saveAddressSelection(CheckoutModel checkoutModel, **int** shippingId) {  
 String transitionValue = "success";  
 Address shipping = **this**.userDAO.getAddress(shippingId);  
 checkoutModel.setShipping(shipping);  
 **return** transitionValue;  
 }  
   
 **public** String saveAddress(CheckoutModel checkoutModel, Address shipping) {  
 String transitionValue = "success";  
 shipping.setUserId(checkoutModel.getUser().getId());  
 shipping.setShipping(**true**);  
 **this**.userDAO.addAddress(shipping);  
 checkoutModel.setShipping(shipping);  
 **return** transitionValue;  
 }  
   
 **public** String saveOrder(CheckoutModel checkoutModel) {  
 String transitionValue = "success";  
 OrderDetail orderDetail = **new** OrderDetail();  
 orderDetail.setUser(checkoutModel.getUser());  
 orderDetail.setShipping(checkoutModel.getShipping());  
 Address billing = **this**.userDAO.getBillingAddress(checkoutModel.getUser().getId());  
 orderDetail.setBilling(billing);  
 List<CartLine> cartLines = checkoutModel.getCartLines();  
 OrderItem orderItem = **null**;  
 **double** orderTotal = 0.0D;  
 **int** orderCount = 0;  
 Product product = **null**;  
 **for** (CartLine cartLine : cartLines) {  
 orderItem = **new** OrderItem();  
 orderItem.setBuyingPrice(cartLine.getBuyingPrice());  
 orderItem.setProduct(cartLine.getProduct());  
 orderItem.setProductCount(cartLine.getProductCount());  
 orderItem.setTotal(cartLine.getTotal());  
 orderItem.setOrderDetail(orderDetail);  
 orderDetail.getOrderItems().add(orderItem);  
 orderTotal += orderItem.getTotal();  
 orderCount++;  
 product = cartLine.getProduct();  
 product.setQuantity(product.getQuantity() - cartLine.getProductCount());  
 product.setPurchases(product.getPurchases() + cartLine.getProductCount());  
 **this**.productDAO.update(product);  
 **this**.cartLineDAO.remove(cartLine);  
 }   
 orderDetail.setOrderTotal(orderTotal);  
 orderDetail.setOrderCount(orderCount);  
 orderDetail.setOrderDate(**new** Date());  
 **this**.cartLineDAO.addOrderDetail(orderDetail);  
 checkoutModel.setOrderDetail(orderDetail);  
 Cart cart = checkoutModel.getCart();  
 cart.setGrandTotal(cart.getGrandTotal() - orderTotal);  
 cart.setCartLines(cart.getCartLines() - orderCount);  
 **this**.cartLineDAO.updateCart(cart);  
 UserModel userModel = (UserModel)**this**.session.getAttribute("userModel");  
 **if** (userModel != **null**)  
 userModel.setCart(cart);   
 **return** transitionValue;  
 }  
   
 **public** OrderDetail getOrderDetail(CheckoutModel checkoutModel) {  
 **return** checkoutModel.getOrderDetail();  
 }  
}

Register :

**package** com.fsd.foodbox.handler;  
  
**import** com.fsd.foodbox.model.RegisterModel;  
**import** com.fsd.foodboxbackend.dao.UserDAO;  
**import** com.fsd.foodboxbackend.dto.Address;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.User;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.binding.message.MessageBuilder;  
**import** org.springframework.binding.message.MessageContext;  
**import** org.springframework.security.crypto.password.PasswordEncoder;  
**import** org.springframework.stereotype.Component;  
  
@Component  
**public** **class** RegisterHandler {  
 @Autowired  
 **private** PasswordEncoder passwordEncoder;  
   
 @Autowired  
 **private** UserDAO userDAO;  
   
 **public** RegisterModel init() {  
 **return** **new** RegisterModel();  
 }  
   
 **public** **void** addUser(RegisterModel registerModel, User user) {  
 registerModel.setUser(user);  
 }  
   
 **public** **void** addBilling(RegisterModel registerModel, Address billing) {  
 registerModel.setBilling(billing);  
 }  
   
 **public** String validateUser(User user, MessageContext error) {  
 String transitionValue = "success";  
 **if** (!user.getPassword().equals(user.getConfirmPassword())) {  
 error.addMessage((**new** MessageBuilder()).error().source(  
 "confirmPassword").defaultText("Password does not match confirm password!").build());  
 transitionValue = "failure";  
 }   
 **if** (**this**.userDAO.getByEmail(user.getEmail()) != **null**) {  
 error.addMessage((**new** MessageBuilder()).error().source(  
 "email").defaultText("Email address is already taken!").build());  
 transitionValue = "failure";  
 }   
 **return** transitionValue;  
 }  
   
 **public** String saveAll(RegisterModel registerModel) {  
 String transitionValue = "success";  
 User user = registerModel.getUser();  
 **if** (user.getRole().equals("USER")) {  
 Cart cart = **new** Cart();  
 cart.setUser(user);  
 user.setCart(cart);  
 }   
 user.setPassword(**this**.passwordEncoder.encode(user.getPassword()));  
 **this**.userDAO.add(user);  
 Address billing = registerModel.getBilling();  
 billing.setUserId(user.getId());  
 billing.setBilling(**true**);  
 **this**.userDAO.addAddress(billing);  
 **return** transitionValue;  
 }  
}

**MODEL**

**package** com.fsd.foodbox.model;  
  
**import** java.io.Serializable;  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dto.Address;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.CartLine;  
**import** com.fsd.foodboxbackend.dto.OrderDetail;  
**import** com.fsd.foodboxbackend.dto.User;  
  
**public** **class** CheckoutModel **implements** Serializable {  
 **private** **static** **final** **long** serialVersionUID = 1L;  
   
 **private** User user;  
   
 **private** Address shipping;  
   
 **private** Cart cart;  
   
 **private** List<CartLine> cartLines;  
   
 **private** OrderDetail orderDetail;  
   
 **private** **double** checkoutTotal;  
   
 **public** OrderDetail getOrderDetail() {  
 **return** **this**.orderDetail;  
 }  
   
 **public** **void** setOrderDetail(OrderDetail orderDetail) {  
 **this**.orderDetail = orderDetail;  
 }  
   
 **public** Cart getCart() {  
 **return** **this**.cart;  
 }  
   
 **public** **void** setCart(Cart cart) {  
 **this**.cart = cart;  
 }  
   
 **public** **double** getCheckoutTotal() {  
 **return** **this**.checkoutTotal;  
 }  
   
 **public** **void** setCheckoutTotal(**double** checkoutTotal) {  
 **this**.checkoutTotal = checkoutTotal;  
 }  
   
 **public** List<CartLine> getCartLines() {  
 **return** **this**.cartLines;  
 }  
   
 **public** **void** setCartLines(List<CartLine> cartLines) {  
 **this**.cartLines = cartLines;  
 }  
   
 **public** User getUser() {  
 **return** **this**.user;  
 }  
   
 **public** **void** setUser(User user) {  
 **this**.user = user;  
 }  
   
 **public** Address getShipping() {  
 **return** **this**.shipping;  
 }  
   
 **public** **void** setShipping(Address shipping) {  
 **this**.shipping = shipping;  
 }  
}

**package** com.fsd.foodbox.model;  
  
  
**import** java.io.Serializable;  
**import** com.fsd.foodboxbackend.dto.Cart;  
  
**public** **class** UserModel **implements** Serializable {  
 **private** **static** **final** **long** serialVersionUID = 1L;  
   
 **private** **int** id;  
   
 **private** String fullName;  
   
 **private** String role;  
   
 **private** Cart cart;  
   
 **public** String getRole() {  
 **return** **this**.role;  
 }  
   
 **public** **void** setRole(String role) {  
 **this**.role = role;  
 }  
   
 **public** **int** getId() {  
 **return** **this**.id;  
 }  
   
 **public** **void** setId(**int** id) {  
 **this**.id = id;  
 }  
   
 **public** String getFullName() {  
 **return** **this**.fullName;  
 }  
   
 **public** **void** setFullName(String fullName) {  
 **this**.fullName = fullName;  
 }  
   
 **public** Cart getCart() {  
 **return** **this**.cart;  
 }  
   
 **public** **void** setCart(Cart cart) {  
 **this**.cart = cart;  
 }  
}

**SERVICE**

**package** com.fsd.foodbox.service;  
  
  
**import** java.util.List;  
**import** javax.servlet.http.HttpSession;  
**import** com.fsd.foodbox.model.UserModel;  
**import** com.fsd.foodboxbackend.dao.CartLineDAO;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.CartLine;  
**import** com.fsd.foodboxbackend.dto.Product;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Service;  
  
@Service("cartService")  
**public** **class** CartService {  
 @Autowired  
 **private** CartLineDAO cartLineDAO;  
   
 @Autowired  
 **private** ProductDAO productDAO;  
   
 @Autowired  
 **private** HttpSession session;  
   
 **public** List<CartLine> getCartLines() {  
 **return** **this**.cartLineDAO.list(getCart().getId());  
 }  
   
 **public** String manageCartLine(**int** cartLineId, **int** count) {  
 CartLine cartLine = **this**.cartLineDAO.get(cartLineId);  
 **double** oldTotal = cartLine.getTotal();  
 Product product = cartLine.getProduct();  
 **if** (product.getQuantity() < count)  
 **return** "result=unavailable";   
 cartLine.setProductCount(count);  
 cartLine.setBuyingPrice(product.getUnitPrice());  
 cartLine.setTotal(product.getUnitPrice() \* count);  
 **this**.cartLineDAO.update(cartLine);  
 Cart cart = getCart();  
 cart.setGrandTotal(cart.getGrandTotal() - oldTotal + cartLine.getTotal());  
 **this**.cartLineDAO.updateCart(cart);  
 **return** "result=updated";  
 }  
   
 **public** String addCartLine(**int** productId) {  
 Cart cart = getCart();  
 String response = **null**;  
 CartLine cartLine = **this**.cartLineDAO.getByCartAndProduct(cart.getId(), productId);  
 **if** (cartLine == **null**) {  
 cartLine = **new** CartLine();  
 Product product = **this**.productDAO.get(productId);  
 cartLine.setCartId(cart.getId());  
 cartLine.setProduct(product);  
 cartLine.setProductCount(1);  
 cartLine.setBuyingPrice(product.getUnitPrice());  
 cartLine.setTotal(product.getUnitPrice());  
 **this**.cartLineDAO.add(cartLine);  
 cart.setGrandTotal(cart.getGrandTotal() + cartLine.getTotal());  
 cart.setCartLines(cart.getCartLines() + 1);  
 **this**.cartLineDAO.updateCart(cart);  
 response = "result=added";  
 } **else** **if** (cartLine.getProductCount() < 3) {  
 response = manageCartLine(cartLine.getId(), cartLine.getProductCount() + 1);  
 } **else** {  
 response = "result=maximum";  
 }   
 **return** response;  
 }  
   
 **private** Cart getCart() {  
 **return** ((UserModel)**this**.session.getAttribute("userModel")).getCart();  
 }  
   
 **public** String removeCartLine(**int** cartLineId) {  
 CartLine cartLine = **this**.cartLineDAO.get(cartLineId);  
 Cart cart = getCart();  
 cart.setGrandTotal(cart.getGrandTotal() - cartLine.getTotal());  
 cart.setCartLines(cart.getCartLines() - 1);  
 **this**.cartLineDAO.updateCart(cart);  
 **this**.cartLineDAO.remove(cartLine);  
 **return** "result=deleted";  
 }  
   
 **public** String validateCartLine() {  
 Cart cart = getCart();  
 List<CartLine> cartLines = **this**.cartLineDAO.list(cart.getId());  
 **double** grandTotal = 0.0D;  
 **int** lineCount = 0;  
 String response = "result=success";  
 **boolean** changed = **false**;  
 Product product = **null**;  
 **for** (CartLine cartLine : cartLines) {  
 product = cartLine.getProduct();  
 changed = **false**;  
 **if** (!product.isActive() && product.getQuantity() == 0 && cartLine.isAvailable()) {  
 cartLine.setAvailable(**false**);  
 changed = **true**;  
 }   
 **if** (product.isActive() && product.getQuantity() > 0 && !cartLine.isAvailable()) {  
 cartLine.setAvailable(**true**);  
 changed = **true**;  
 }   
 **if** (cartLine.getBuyingPrice() != product.getUnitPrice()) {  
 cartLine.setBuyingPrice(product.getUnitPrice());  
 cartLine.setTotal(cartLine.getProductCount() \* product.getUnitPrice());  
 changed = **true**;  
 }   
 **if** (cartLine.getProductCount() > product.getQuantity()) {  
 cartLine.setProductCount(product.getQuantity());  
 cartLine.setTotal(cartLine.getProductCount() \* product.getUnitPrice());  
 changed = **true**;  
 }   
 **if** (changed) {  
 **this**.cartLineDAO.update(cartLine);  
 response = "result=modified";  
 }   
 grandTotal += cartLine.getTotal();  
 lineCount++;  
 }   
 cart.setCartLines(lineCount++);  
 cart.setGrandTotal(grandTotal);  
 **this**.cartLineDAO.updateCart(cart);  
 **return** response;  
 }  
}

**FileUTIL**

**package** com.fsd.foodbox.util;  
  
**import** java.io.File;  
**import** java.io.IOException;  
**import** java.net.URL;  
**import** javax.servlet.http.HttpServletRequest;  
**import** org.slf4j.Logger;  
**import** org.slf4j.LoggerFactory;  
**import** org.springframework.web.multipart.MultipartFile;  
  
**public** **class** FileUtil {  
 @SuppressWarnings("unused")  
**private** **static** **final** String ABS\_PATH = "C:/Users/Dell/eclipse-workspace\_1/com\_foodbox\_FSD2/assets/images/";  
   
 **private** **static** String REAL\_PATH = **null**;  
   
 **private** **static** **final** Logger logger = LoggerFactory.getLogger(com.fsd.foodbox.util.FileUtil.**class**);  
   
 **public** **static** **boolean** uploadFile(HttpServletRequest request, MultipartFile file, String code) {  
 REAL\_PATH = request.getSession().getServletContext().getRealPath("/assets/images/");  
 logger.info(REAL\_PATH);  
 **if** (!(**new** File(REAL\_PATH)).exists())  
 (**new** File(REAL\_PATH)).mkdirs();   
 **if** (!(**new** File("C:/Users/Dell/eclipse-workspace\_1/com\_foodbox\_FSD2/assets/images/")).exists())  
 (**new** File("C:/Users/Dell/eclipse-workspace\_1/com\_foodbox\_FSD2/assets/images/")).mkdirs();   
 **try** {  
 file.transferTo(**new** File(String.valueOf(REAL\_PATH) + code + ".jpg"));  
 file.transferTo(**new** File("C:/Users/Dell/eclipse-workspace\_1/com\_foodbox\_FSD2/assets/images" + code + ".jpg"));  
 } **catch** (IOException ex) {  
 ex.printStackTrace();  
 }   
 **return** **true**;  
 }  
   
   
}

**Validator**

**package** com.fsd.foodbox.validator;  
  
**import** com.fsd.foodboxbackend.dto.Product;  
**import** org.springframework.validation.Errors;  
**import** org.springframework.validation.Validator;  
  
**public** **class** ProductValidator **implements** Validator {  
 **public** **boolean** supports(Class<?> clazz) {  
 **return** Product.**class**.equals(clazz);  
 }  
   
 **public** **void** validate(Object target, Errors errors) {  
 Product product = (Product)target;  
 **if** (product.getFile() == **null** || product.getFile().getOriginalFilename().equals("")) {  
 errors.rejectValue("file", **null**, "Please select a file to upload!");  
 **return**;  
 }   
 **if** ((!product.getFile().getContentType().equals("image/jpeg") &&   
 !product.getFile().getContentType().equals("image/png")) ||   
 product.getFile().getContentType().equals("image/gif")) {  
 errors.rejectValue("file", **null**, "Please select an image file to upload!");  
 **return**;  
 }   
 }  
}

**Config**

**package** com.fsd.foodboxbackend.config;  
  
  
**import** java.util.Properties;  
**import** javax.sql.DataSource;  
**import** org.apache.commons.dbcp2.BasicDataSource;  
**import** org.hibernate.SessionFactory;  
**import** org.springframework.beans.factory.annotation.Value;  
**import** org.springframework.context.annotation.Bean;  
**import** org.springframework.context.annotation.ComponentScan;  
**import** org.springframework.context.annotation.Configuration;  
**import** org.springframework.context.annotation.PropertySource;  
**import** org.springframework.orm.hibernate5.HibernateTransactionManager;  
**import** org.springframework.orm.hibernate5.LocalSessionFactoryBuilder;  
**import** org.springframework.transaction.annotation.EnableTransactionManagement;  
  
@Configuration  
@ComponentScan(basePackages = {"net.fsd.foodboxbackend.dto"})  
@EnableTransactionManagement  
@PropertySource({"classpath:connection.properties"})  
**public** **class** HibernateConfig {  
 @Value("${jdbc.url}")  
 **private** String DATABASE\_URL;  
   
 @Value("${jdbc.driver}")  
 **private** String DATABASE\_DRIVER;  
   
 @Value("${jdbc.dialect}")  
 **private** String DATABASE\_DIALECT;  
   
 @Value("${jdbc.username}")  
 **private** String DATABASE\_USERNAME;  
   
 @Value("${jdbc.password}")  
 **private** String DATABASE\_PASSWORD;  
   
 @Bean({"dataSource"})  
 **public** DataSource getDataSource() {  
 BasicDataSource dataSource = **new** BasicDataSource();  
 dataSource.setDriverClassName(**this**.DATABASE\_DRIVER);  
 dataSource.setUrl(**this**.DATABASE\_URL);  
 dataSource.setUsername(**this**.DATABASE\_USERNAME);  
 dataSource.setPassword(**this**.DATABASE\_PASSWORD);  
 **return** (DataSource)dataSource;  
 }  
   
 @Bean  
 **public** SessionFactory getSessionFactory(DataSource dataSource) {  
 LocalSessionFactoryBuilder builder = **new** LocalSessionFactoryBuilder(dataSource);  
 builder.addProperties(getHibernateProperties());  
 builder.scanPackages(**new** String[] { "net.fsd.foodboxbackend.dto" });  
 **return** builder.buildSessionFactory();  
 }  
   
 **private** Properties getHibernateProperties() {  
 Properties properties = **new** Properties();  
 properties.put("hibernate.dialect", **this**.DATABASE\_DIALECT);  
 properties.put("hibernate.show\_sql", "true");  
 properties.put("hibernate.format\_sql", "true");  
 **return** properties;  
 }  
   
 @Bean  
 **public** HibernateTransactionManager getTransactionManager(SessionFactory sessionFactory) {  
 HibernateTransactionManager transactionManager = **new** HibernateTransactionManager(sessionFactory);  
 **return** transactionManager;  
 }  
}

**DAO**

**package** com.fsd.foodboxbackend.dao;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.CartLine;  
**import** com.fsd.foodboxbackend.dto.OrderDetail;  
  
**public** **interface** CartLineDAO {  
 List<CartLine> list(**int** paramInt);  
   
 CartLine get(**int** paramInt);  
   
 **boolean** add(CartLine paramCartLine);  
   
 **boolean** update(CartLine paramCartLine);  
   
 **boolean** remove(CartLine paramCartLine);  
   
 CartLine getByCartAndProduct(**int** paramInt1, **int** paramInt2);  
   
 **boolean** updateCart(Cart paramCart);  
   
 List<CartLine> listAvailable(**int** paramInt);  
   
 **boolean** addOrderDetail(OrderDetail paramOrderDetail);  
}

**package** com.fsd.foodboxbackend.dao;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dto.Cuisine;  
  
**public** **interface** CuisineDAO {  
 Cuisine get(**int** paramInt);  
   
 List<Cuisine> list();  
   
 **boolean** add(Cuisine paramCuisine);  
   
 **boolean** update(Cuisine paramCuisine);  
   
 **boolean** delete(Cuisine paramCuisine);  
}

**package** com.fsd.foodboxbackend.dao;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dto.Product;  
  
**public** **interface** ProductDAO {  
 Product get(**int** paramInt);  
   
 List<Product> list();  
   
 **boolean** add(Product paramProduct);  
   
 **boolean** update(Product paramProduct);  
   
 **boolean** delete(Product paramProduct);  
   
 List<Product> getProductsByParam(String paramString, **int** paramInt);  
   
 List<Product> listActiveProducts();  
   
 List<Product> listActiveProductsByCuisine(**int** paramInt);  
   
 List<Product> getLatestActiveProducts(**int** paramInt);  
}

**package** com.fsd.foodboxbackend.dao;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dto.Address;  
**import** com.fsd.foodboxbackend.dto.User;  
  
**public** **interface** UserDAO {  
 User getByEmail(String paramString);  
   
 User get(**int** paramInt);  
   
 **boolean** add(User paramUser);  
   
 Address getAddress(**int** paramInt);  
   
 **boolean** addAddress(Address paramAddress);  
   
 **boolean** updateAddress(Address paramAddress);  
   
 Address getBillingAddress(**int** paramInt);  
   
 List<Address> listShippingAddresses(**int** paramInt);  
}

DAO IML

**package** com.fsd.foodboxbackend.daoimpl;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dao.CartLineDAO;  
**import** com.fsd.foodboxbackend.dto.Cart;  
**import** com.fsd.foodboxbackend.dto.CartLine;  
**import** com.fsd.foodboxbackend.dto.OrderDetail;  
**import** org.hibernate.SessionFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Repository;  
**import** org.springframework.transaction.annotation.Transactional;  
  
@Repository("cartLineDAO")  
@Transactional  
**public** **class** CartLineDAOImpl **implements** CartLineDAO {  
 @Autowired  
 **private** SessionFactory sessionFactory;  
   
 **public** CartLine getByCartAndProduct(**int** cartId, **int** productId) {  
 String query = "FROM CartLine WHERE cartId = :cartId AND product.id = :productId";  
 **try** {  
 **return** (CartLine)**this**.sessionFactory.getCurrentSession()  
 .createQuery(query, CartLine.**class**)  
 .setParameter("cartId", Integer.valueOf(cartId))  
 .setParameter("productId", Integer.valueOf(productId))  
 .getSingleResult();  
 } **catch** (Exception ex) {  
 **return** **null**;  
 }   
 }  
   
 **public** **boolean** add(CartLine cartLine) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().persist(cartLine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** update(CartLine cartLine) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().update(cartLine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** remove(CartLine cartLine) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().delete(cartLine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
   
 **public** List<CartLine> list(**int** cartId) {  
 String query = "FROM CartLine WHERE cartId = :cartId";  
 **return** **this**.sessionFactory.getCurrentSession()  
 .createQuery(query, CartLine.**class**)  
 .setParameter("cartId", Integer.valueOf(cartId))  
 .getResultList();  
 }  
   
 **public** CartLine get(**int** id) {  
 **return** (CartLine)**this**.sessionFactory.getCurrentSession().get(CartLine.**class**, Integer.valueOf(id));  
 }  
   
 **public** **boolean** updateCart(Cart cart) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().update(cart);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
   
 **public** List<CartLine> listAvailable(**int** cartId) {  
 String query = "FROM CartLine WHERE cartId = :cartId AND available = :available";  
 **return** **this**.sessionFactory.getCurrentSession()  
 .createQuery(query, CartLine.**class**)  
 .setParameter("cartId", Integer.valueOf(cartId))  
 .setParameter("available", Boolean.valueOf(**true**))  
 .getResultList();  
 }  
   
 **public** **boolean** addOrderDetail(OrderDetail orderDetail) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().persist(orderDetail);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
}

**package** com.fsd.foodboxbackend.daoimpl;  
  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dao.CuisineDAO;  
**import** com.fsd.foodboxbackend.dto.Cuisine;  
**import** org.hibernate.SessionFactory;  
**import** org.hibernate.query.Query;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Repository;  
**import** org.springframework.transaction.annotation.Transactional;  
  
@Repository("cuisineDAO")  
@Transactional  
**public** **class** CuisineDAOImpl **implements** CuisineDAO {  
 @Autowired  
 **private** SessionFactory sessionFactory;  
   
 **public** List<Cuisine> list() {  
 String selectActiveCuisine = "FROM Cuisine WHERE active = :active";  
 Query query = **this**.sessionFactory.getCurrentSession().createQuery(selectActiveCuisine);  
 query.setParameter("active", Boolean.valueOf(**true**));  
 **return** query.getResultList();  
 }  
   
 **public** Cuisine get(**int** id) {  
 **return** (Cuisine)**this**.sessionFactory.getCurrentSession().get(Cuisine.**class**, Integer.valueOf(id));  
 }  
   
 **public** **boolean** add(Cuisine cuisine) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().persist(cuisine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** update(Cuisine cuisine) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().update(cuisine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** delete(Cuisine cuisine) {  
 cuisine.setActive(**false**);  
 **try** {  
 **this**.sessionFactory.getCurrentSession().update(cuisine);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
}

**package** com.fsd.foodboxbackend.daoimpl;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dao.ProductDAO;  
**import** com.fsd.foodboxbackend.dto.Product;  
**import** org.hibernate.SessionFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Repository;  
**import** org.springframework.transaction.annotation.Transactional;  
  
@Repository("productDAO")  
@Transactional  
**public** **class** ProductDAOImpl **implements** ProductDAO {  
 @Autowired  
 **private** SessionFactory sessionFactory;  
   
 **public** Product get(**int** productId) {  
 **try** {  
 **return** (Product)**this**.sessionFactory  
 .getCurrentSession()  
 .get(Product.**class**, Integer.valueOf(productId));  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **null**;  
 }   
 }  
   
 **public** List<Product> list() {  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery("FROM Product", Product.**class**)  
 .getResultList();  
 }  
   
 **public** **boolean** add(Product product) {  
 **try** {  
 **this**.sessionFactory  
 .getCurrentSession()  
 .persist(product);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** update(Product product) {  
 **try** {  
 **this**.sessionFactory  
 .getCurrentSession()  
 .update(product);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** delete(Product product) {  
 **try** {  
 product.setActive(**false**);  
 **return** update(product);  
 } **catch** (Exception ex) {  
 ex.printStackTrace();  
 **return** **false**;  
 }   
 }  
   
 **public** List<Product> listActiveProducts() {  
 String selectActiveProducts = "FROM Product WHERE active = :active";  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(selectActiveProducts, Product.**class**)  
 .setParameter("active", Boolean.valueOf(**true**))  
 .getResultList();  
 }  
   
 **public** List<Product> listActiveProductsByCuisine(**int** cuisineId) {  
 String selectActiveProductsByCuisine = "FROM Product WHERE active = :active AND cuisineId = :cuisineId";  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(selectActiveProductsByCuisine, Product.**class**)  
 .setParameter("active", Boolean.valueOf(**true**))  
 .setParameter("cuisineId", Integer.valueOf(cuisineId))  
 .getResultList();  
 }  
   
 **public** List<Product> getLatestActiveProducts(**int** count) {  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery("FROM Product WHERE active = :active ORDER BY id", Product.**class**)  
 .setParameter("active", Boolean.valueOf(**true**))  
 .setFirstResult(0)  
 .setMaxResults(count)  
 .getResultList();  
 }  
   
 **public** List<Product> getProductsByParam(String param, **int** count) {  
 String query = "FROM Product WHERE active = true ORDER BY " + param + " DESC";  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(query, Product.**class**)  
 .setFirstResult(0)  
 .setMaxResults(count)  
 .getResultList();  
 }  
}

**package** com.fsd.foodboxbackend.daoimpl;  
  
**import** java.util.List;  
**import** com.fsd.foodboxbackend.dao.UserDAO;  
**import** com.fsd.foodboxbackend.dto.Address;  
**import** com.fsd.foodboxbackend.dto.User;  
**import** org.hibernate.SessionFactory;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Repository;  
**import** org.springframework.transaction.annotation.Transactional;  
  
@Repository("userDAO")  
@Transactional  
**public** **class** UserDAOImpl **implements** UserDAO {  
 @Autowired  
 **private** SessionFactory sessionFactory;  
   
 **public** User getByEmail(String email) {  
 String selectQuery = "FROM User WHERE email = :email";  
 **try** {  
 **return** (User)**this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(selectQuery, User.**class**)  
 .setParameter("email", email)  
 .getSingleResult();  
 } **catch** (Exception ex) {  
 **return** **null**;  
 }   
 }  
   
 **public** **boolean** add(User user) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().persist(user);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** addAddress(Address address) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().persist(address);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
   
 **public** **boolean** updateAddress(Address address) {  
 **try** {  
 **this**.sessionFactory.getCurrentSession().update(address);  
 **return** **true**;  
 } **catch** (Exception ex) {  
 **return** **false**;  
 }   
 }  
   
 **public** List<Address> listShippingAddresses(**int** userId) {  
 String selectQuery = "FROM Address WHERE userId = :userId AND shipping = :isShipping ORDER BY id DESC";  
 **return** **this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(selectQuery, Address.**class**)  
 .setParameter("userId", Integer.valueOf(userId))  
 .setParameter("isShipping", Boolean.valueOf(**true**))  
 .getResultList();  
 }  
   
 **public** Address getBillingAddress(**int** userId) {  
 String selectQuery = "FROM Address WHERE userId = :userId AND billing = :isBilling";  
 **try** {  
 **return** (Address)**this**.sessionFactory  
 .getCurrentSession()  
 .createQuery(selectQuery, Address.**class**)  
 .setParameter("userId", Integer.valueOf(userId))  
 .setParameter("isBilling", Boolean.valueOf(**true**))  
 .getSingleResult();  
 } **catch** (Exception ex) {  
 **return** **null**;  
 }   
 }  
   
 **public** User get(**int** id) {  
 **try** {  
 **return** (User)**this**.sessionFactory.getCurrentSession().get(User.**class**, Integer.valueOf(id));  
 } **catch** (Exception ex) {  
 System.out.println(ex.getMessage());  
 **return** **null**;  
 }   
 }  
   
 **public** Address getAddress(**int** addressId) {  
 **try** {  
 **return** (Address)**this**.sessionFactory.getCurrentSession().get(Address.**class**, Integer.valueOf(addressId));  
 } **catch** (Exception ex) {  
 System.out.println(ex.getMessage());  
 **return** **null**;  
 }   
 }  
}