TRƯỜNG ĐẠI HỌC HỌC VĂN LANG

**KHOA CÔNG NGHỆ THÔNG TIN**

A red and black shield with white logo

Description automatically generated

**TÀI LIỆU THỰC HÀNH**

**SPRING BOOT**

**CHƯƠNG 07: XÂY DỰNG CHỨC NĂNG GIỎ HÀNG VÀ ĐẶT HÀNG**

Giảng viên biên soạn: Ths. Nguyễn Minh Tân

2024

**BUỔI 7: XÂY DỰNG CHỨC NĂNG GIỎ HÀNG VÀ ĐẶT HÀNG**

1. **MỤC TIÊU**

…Xây dựng chức năng giỏ hàng và thanh toán bằng cách sử dụng Session.

Để xây dựng chức năng giỏ hàng sử dụng session trong một ứng dụng web Spring Boot, ta lưu trữ thông tin giỏ hàng trong session người dùng. Điều này giúp dữ liệu giỏ hàng được duy trì suốt phiên làm việc của người dùng mà không cần lưu vào cơ sở dữ liệu ngay lập tức. Chức năng giỏ hàng được xây dựng dựa trên session cho phép giữ dữ liệu giỏ hàng qua nhiều yêu cầu mà không cần lưu trữ ngay lập tức vào cơ sở dữ liệu.

Dưới đây là một hướng dẫn từng bước để xây dựng chức năng giỏ hàng sử dụng session.

1. **NỘI DUNG THỰC HÀNH:**

**7.1 Xây dựng chức năng Giỏ hàng**

**7.1.1 Thiết kế mô hình dữ liệu – tạo class ‘CartItem’**

Dự án không cần tạo mô hình dữ liệu mới trong cơ sở dữ liệu vì giỏ hàng sẽ được lưu trữ trong session. Tuy nhiên, chúng ta cần một class ‘**CartItem**’ để đại diện cho các mặt hàng trong giỏ hàng.

• **Mục đích**: Thay vì tạo mô hình dữ liệu mới trong cơ sở dữ liệu, chúng ta sẽ lưu trữ giỏ hàng tạm thời trong session. Để quản lý các mặt hàng trong giỏ hàng, cần có một class **CartItem**.

•Thực hiện: Trong package **model** tại đường dẫn **src/main/java/com.vanlang.webbanhang/model**, tạo lớp **CartItem.java** với hai thuộc tính là sản phẩm (**Product**) và số lượng (**quantity**). Lớp này cũng cần các phương thức getter và setter để truy cập và cập nhật thuộc tính

|  |
| --- |
| package com.vanlang.webbanhang.model;  public class CartItem {  private Product product;  private int quantity;  //constructor  public CartItem(Product product, int quantity) {  this.product = product;  this.quantity = quantity;  }  //getter & setter   public Product getProduct() {  return product;  }   public void setProduct(Product product) {  this.product = product;  }   public int getQuantity() {  return quantity;  }   public void setQuantity(int quantity) {  this.quantity = quantity;  } } |

**7.1.2 Xây Dựng CartService.java**

Để quản lý các hoạt động liên quan đến giỏ hàng như thêm sản phẩm, xóa sản phẩm khỏi giỏ hàng, và xóa sạch giỏ hàng.

Tạo **CartService.java** trong package **service** trong đường dẫn **src/main/java/com.vanlang.webbanhang/service** với các phương thức để thêm sản phẩm vào giỏ hàng, lấy tất cả sản phẩm trong giỏ, xóa sản phẩm khỏi giỏ và xóa sạch giỏ hàng. Dịch vụ này được định nghĩa với phạm vi session để duy trì thông tin giỏ hàng trong suốt phiên làm việc của người dùng.

|  |
| --- |
| package com.vanlang.webbanhang.service;  import com.vanlang.webbanhang.model.CartItem; import com.vanlang.webbanhang.model.Product; import com.vanlang.webbanhang.repository.ProductRepository; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Service; import org.springframework.web.context.annotation.SessionScope;  import java.util.ArrayList; import java.util.List;  @Service @SessionScope public class CartService {  private List<CartItem> cartItems = new ArrayList<CartItem>();   @Autowired  private ProductRepository productRepository;   public void addToCart(Long productId, int quantity) {  Product product = productRepository.findById(productId).orElseThrow(() -> new IllegalArgumentException("Product not found: " + productId));  cartItems.add(new CartItem(product, quantity));  }   public List<CartItem> getCartItems() {  return cartItems;  }   public void removeFromCart(Long productId) {  cartItems.removeIf(cartItem -> cartItem.getProduct().getId().equals(productId));  }   public void clearCard() {  cartItems.clear();  } } |

**7.1.3 Xây Dựng CartController.java**

• **Mục đích**: Xử lý các yêu cầu liên quan đến giỏ hàng từ người dùng.

• **Thực hiện**: Trong **CartController.java**, cung cấp các đường dẫn để xem giỏ hàng, thêm sản phẩm vào giỏ hàng, loại bỏ sản phẩm từ giỏ hàng, và xóa sạch giỏ hàng. Mỗi hành động này sẽ tương ứng với một phương thức trong controller.

Tạo một file mới có tên ‘**CartController.java’** trong đường dẫn **src/main/java/com.vanlang.webbanhang/controller**

|  |
| --- |
| package com.vanlang.webbanhang.controller;  import com.vanlang.webbanhang.service.CartService; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Controller; import org.springframework.ui.Model; import org.springframework.web.bind.annotation.\*;  @Controller @RequestMapping("/cart") public class CartController {  @Autowired  private CartService cartService;   @GetMapping  public String showCart(Model model) {  model.addAttribute("cartItems", cartService.getCartItems());  return "/cart/cart";  }  @PostMapping("/add")  public String addToCart(@RequestParam Long productId, @RequestParam int quantity) {  cartService.addToCart(productId, quantity);  return "redirect:/cart";  }  @GetMapping("/remove/{productId}")  public String removeFromCart(@PathVariable Long productId) {  cartService.removeFromCart(productId);  return "redirect:/cart";  }  @GetMapping("/clear")  public String clearCart() {  cartService.clearCard();  return "redirect:/cart";  } } |

**7.1.4 Tạo View cho giỏ hàng**

Hiển thị giỏ hàng cho người dùng với các tùy chọn thay đổi số lượng sản phẩm hoặc xóa sản phẩm khỏi giỏ hàng.

Trong đường dẫn **src/main/resources/templates** tạo thêm directory **cart**. Sau đó tạo thêm file html: **`cart.html`** để hiển thị các sản phẩm trong giỏ hàng. Trang này bao gồm một bảng với các sản phẩm được liệt kê, mỗi sản phẩm có tùy chọn để thay đổi số lượng hoặc xóa khỏi giỏ.

|  |
| --- |
| <!DOCTYPE html> <html xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"  layout:decorate="~{layout}"> <head><title th:text="${title} ?: 'Your Cart'">Your Cart</title>  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"> </head> <body> <section layout:fragment="content" class="container mt-3"><h1>Your Cart</h1>  <div th:if="${cartItems.isEmpty()}" class="alert alert-info">Your cart is empty.</div>  <table class="table" th:unless="${cartItems.isEmpty()}">  <thead class="table-light">  <tr>  <th>Product Name</th>  <th>Quantity</th>  <th>Action</th>  </tr>  </thead>  <tbody>  <tr th:each="item : ${cartItems}">  <td th:text="${item.product.name}"></td>  <td th:text="${item.quantity}"></td>  <td>  <a th:href="@{/cart/remove/{productId}(productId=${item.product.id})}" class="btn btn-danger btn-sm">Remove</a>  </td>  </tr>  </tbody>  </table>  <div class="mt-3">  <a th:href="@{/cart/clear}" class="btn btn-secondary">Clear Cart</a> |  <a th:href="@{/order/checkout}" class="btn btn-primary mb-3">Check Out</a>  </div> </section> </body> </html> |

Cập nhật **`products-list.html`** trong directory **products** để tạo nút thêm giỏ hàng “**Add to Cart”:**

|  |
| --- |
| <h1>Products List</h1> <div><a th:href="@{/products/add}" class="btn btn-primary mb-3">Add New Product</a></div> <table class="table table-bordered table-hover">  <thead class="table-dark">  <tr>  <th>ID</th>  <th>Name</th>  <th>Price</th>  <th>Description</th>  <th>Category Name</th>  <th>Actions</th>  <th>Add To Cart</th>  </tr>  </thead>  <tbody>  <tr th:each="product : ${products}">  <td th:text="${product.id}"></td>  <td th:text="${product.name}"></td>  <td th:text="${product.price}"></td>  <td th:text="${product.description}"></td>  <td th:text="${product.category.name}"></td>  <td>  <a th:href="@{/products/edit/{id}(id=${product.id})}" class="btn btn-success btn-sm">Edit</a>  <a th:href="@{/products/delete/{id}(id=${product.id})}" class="btn btn-danger btn-sm"  onclick="return confirm('Are you sure?')">Delete</a>  </td>  <td>  <form th:action="@{/cart/add}" method="post">  <input type="number" name="quantity" min="1" value="1"  class="form-control d-inline-block"  style="width: 70px;">  <input type="hidden"  th:value="${product.id}"  name="productId"/>  <button type="submit" class="btn btn-warning btn-sm">Add to Cart</button>  </form>  </td>  </tr>  </tbody> </table> |

**7.2 Xây dựng chức năng Đặt hàng**

Để xây dựng chức năng đặt hàng trong một ứng dụng Spring Boot, ta cần lưu trữ thông tin về đơn hàng (Order) và các chi tiết đơn hàng (OrderDetail) dựa vào thông tin trong giỏ hàng. Dưới đây là hướng dẫn từng bước để triển khai chức năng này, bao gồm xử lý back-end và cập nhật mô hình dữ liệu:

**7.2.1 Tạo OrderService.java**

Dịch vụ này sẽ xử lý logic để tạo mới đơn hàng từ giỏ hàng. Trong package **service** tạo `**OrderService.java`**

|  |
| --- |
| package com.vanlang.webbanhang.service;  import com.vanlang.webbanhang.model.CartItem; import com.vanlang.webbanhang.model.Order; import com.vanlang.webbanhang.model.OrderDetail; import com.vanlang.webbanhang.repository.OrderDetailRepository; import com.vanlang.webbanhang.repository.OrderRepository; import lombok.RequiredArgsConstructor; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Service; import org.springframework.transaction.annotation.Transactional;  import java.util.List;  @Service @RequiredArgsConstructor @Transactional public class OrderService {  @Autowired  private OrderRepository orderRepository;  @Autowired  private OrderDetailRepository orderDetailRepository;  @Autowired  private CartService cartService; // Assuming you have a CartService   @Transactional  public Order createOrder(String customerName, List<CartItem> cartItems) {  Order order = new Order();  order.setCustomerName(customerName);  order = orderRepository.save(order);  for (CartItem item : cartItems) {  OrderDetail detail = new OrderDetail();  detail.setOrder(order);  detail.setProduct(item.getProduct());  detail.setQuantity(item.getQuantity());  orderDetailRepository.save(detail);  }  // Optionally clear the cart after order placement  cartService.clearCard();  return order;  } } |

**7.2.2 Tạo OrderController.java**

Controller này sẽ xử lý các yêu cầu HTTP để đặt hàng.

|  |
| --- |
| package com.vanlang.webbanhang.controller;  import com.vanlang.webbanhang.model.CartItem; import com.vanlang.webbanhang.service.CartService; import com.vanlang.webbanhang.service.OrderService; 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.PostMapping; import org.springframework.web.bind.annotation.RequestMapping;  import java.util.List;  @Controller @RequestMapping("/order") public class OrderController {   @Autowired  private OrderService orderService;  @Autowired  private CartService cartService;   @GetMapping("/checkout")  public String checkout() {  return "cart/checkout";  }   @PostMapping("/submit")  public String submitOrder(String customerName) {  List<CartItem> cartItems = cartService.getCartItems();  if (cartItems.isEmpty()) {  return "redirect:/cart"; // Redirect if cart is empty  }  orderService.createOrder(customerName, cartItems);  return "redirect:/order/confirmation";  }   @GetMapping("/confirmation")  public String orderConfirmation(Model model) {  model.addAttribute("message", "Your order has been successfully placed.");  return "cart/order-confirmation";  } } |

**7.2.3 Tạo View cho phần đặt hàng**

Trong đường dẫn **src/main/resources/templates/cart** tạo thêm 2 file html: `**checkout.html`, `order-confirmation.html`**.

Form để người dùng nhập tên và gửi thông tin đơn hàng. File `**checkout.html`:**

|  |
| --- |
| <!DOCTYPE html> <html xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"  layout:decorate="~{layout}"> <head><title th:text="${title} ?: 'Your Cart'">Place Order</title>  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"> </head> <body> <section layout:fragment="content" class="container mt-3"><h1>Place Your Order</h1>  <form th:action="@{/order/submit}" method="post">  <div class="mb-3">  <label for="customerName" class="form-label">Your Name:</label>  <input type="text" id="customerName"  name="customerName"  class="form-control"  required>  </div>  <button type="submit" class="btn btn-primary">Submit Order</button>  </form> </section> </body> </html> |

Phần xác nhận hoàn tất đặt hàng, file **`order-confirmation.html`**:

|  |
| --- |
| <!DOCTYPE html> <html xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"  layout:decorate="~{layout}"> <head><title th:text="${title} ?: 'Your Cart'">Order Confirmation</title>  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"> </head> <body> <section layout:fragment="content" class="container mt-3"><h1>Order Confirmation</h1>  <p th:text="${message}"></p></section> </body> </html> |

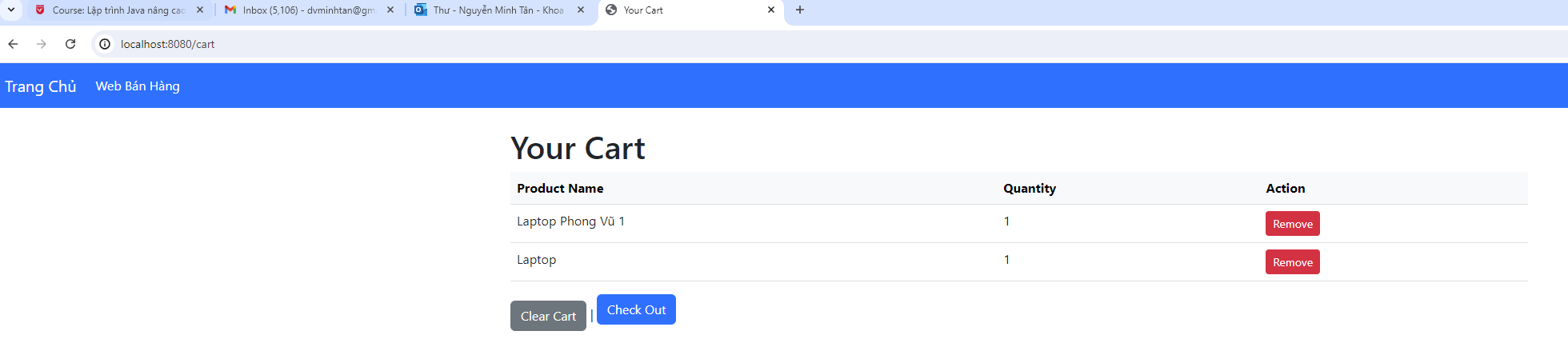
**7.2.4 Tiến hành build lại dự án và kiểm tra kết quả:**

Trang danh sách sản phẩm đã thêm nút **Add to Cart**:

A screenshot of a computer

Description automatically generated

Chức năng Giỏ hàng: *http://localhost:8080/cart*



Chức năng Check out: [*http://localhost:8080/order/checkout*](http://localhost:8080/order/checkout)

A blue and white line

Description automatically generated

Thông báo đặt hàng thành công:

A screenshot of a computer

Description automatically generated

**Kiểm tra thông tin tại database:**

* Thông tin tại bảng order\_detail

A screenshot of a computer

Description automatically generated

* Thông tin tại bảng order

A screenshot of a computer

Description automatically generated

➔ Hoàn tất chức năng giỏ hàng và đặt hàng

**7.3 Yêu cầu bổ sung**

Thêm các thông tin bổ sung cho đơn đặt hàng như địa chỉ giao hàng, số điện thoại, email, ghi chú, phương thức thanh toán.