







# TÀI LIỆU THỰC HÀNH

# **SPRING BOOT**

CHƯƠNG 07: XÂY DỤNG CHÚC NĂNG GIỎ HÀNG VÀ ĐẶT HÀNG

Giảng viên biên soạn: Ths. Nguyễn Minh Tân

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

#### I. MUC 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.

#### II. 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 =
```

#### 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.

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
     ID
     Name
     Price
     Description
     Category Name
     Add To Cart
  </thead>
  <a th:href="@{/products/edit/{id}(id=${product.id})}" class="btn</pre>
        <a th:href="@{/products/delete/{id}(id=${product.id}))}"</pre>
          onclick="return confirm('Are you sure?')">Delete</a>
     <form th:action="@{/cart/add}" method="post">
           <input type="number" name="quantity" min="1" value="1"</pre>
           <input type="hidden"</pre>
Cart</button>
        </form>
```

#### 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 backend và cập nhật mô hình dữ liệu:

#### 7.2.1 Tao 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.OrderDetail;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import java.util.List;
@Service
public class OrderService {
    private OrderRepository orderRepository;
    private OrderDetailRepository orderDetailRepository;
        Order order = new Order();
        order.setCustomerName(customerName);
        for (CartItem item : cartItems) {
            OrderDetail detail = new OrderDetail();
            detail.setOrder(order);
            detail.setProduct(item.getProduct());
            detail.setQuantity(item.getQuantity());
```

```
}
    // Optionally clear the cart after order placement
    cartService.clearCard();
    return order;
}
```

#### 7.2.2 Tao 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.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
    private OrderService orderService;
    private CartService cartService;
    public String checkout() {
        if (cartItems.isEmpty()) {
        orderService.createOrder(customerName, cartItems);
    public String orderConfirmation(Model model) {
```

#### 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`.

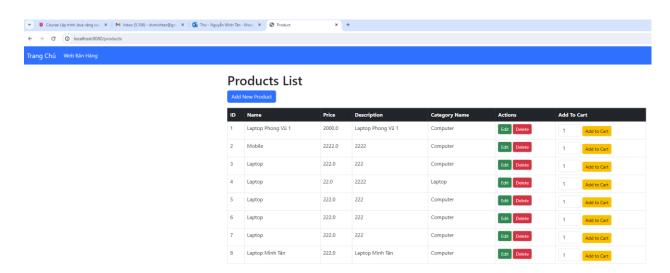
Form để người dùng nhập tên và gửi thông tin đơn hàng. File `checkout.html`:

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

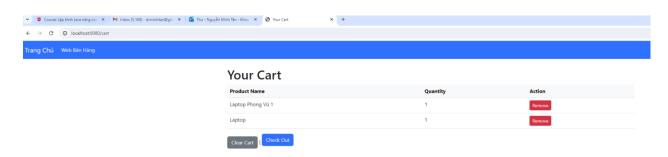
### 7.2.4 Tiến hành build lại dự án và kiểm tra kết quả:

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

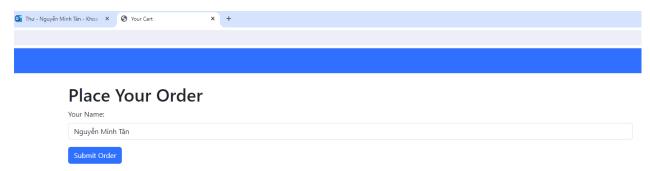
#### Khoa Công nghệ Thông tin



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



Chức năng Check out: <a href="http://localhost:8080/order/checkout">http://localhost:8080/order/checkout</a>



Thông báo đặt hàng thành công:

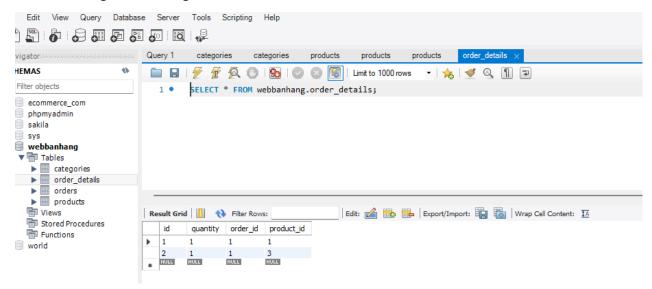


# **Order Confirmation**

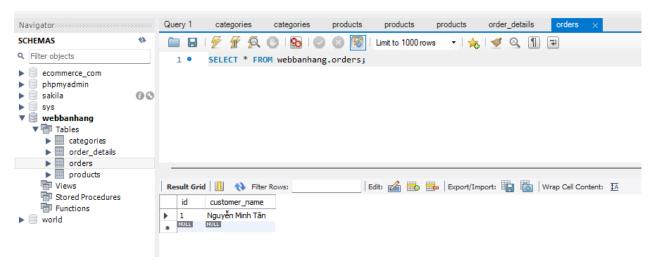
Your order has been successfully placed.

#### Kiểm tra thông tin tại database:

Thông tin tại bảng order\_detail



• Thông tin tại bảng order



#### Khoa Công nghệ Thông tin

→ 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.