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 06: XÂY DỰNG CÁC CHỨC NĂNG HIỂN THỊ/THÊM/XÓA/SỬA CHO PRODUCT**

Giảng viên biên soạn: Ths. Nguyễn Minh Tân

2024

**BUỔI 6: XÂY DỰNG CÁC CHỨC NĂNG HIỂN THỊ/THÊM/XÓA/SỬA CHO PRODUCT**

1. **MỤC TIÊU**

**Sau khi học xong bài này, học viên có thể nắm được:**

- Hiểu rõ nguyên lý hoạt động của mô hình MVC và vai trò của Controllers trong kiến trúc này.

- Có kỹ năng thiết kế và cài đặt Controllers để xử lý các yêu cầu từ người dùng (HTTP requests).

- Học cách sử dụng annotations để định nghĩa các routes và actions trong Controller.

- Hiểu cách truyền dữ liệu từ Controller đến View sử dụng Model hoặc ModelAndView.

- Biết cách nhận và xử lý dữ liệu đầu vào từ người dùng thông qua form submissions.

- Phát triển kỹ năng xử lý các tình huống lỗi trong ứng dụng web.

- Biết cách gửi phản hồi thích hợp đến người dùng thông qua HTTP status codes và messages.

- Học cách tích hợp Controllers với các Services để xử lý nghiệp vụ như xác thực người dùng, quản lý giỏ hàng, và thực hiện thanh toán.

- Hiểu được cách sử dụng Dependency Injection để quản lý các phụ thuộc trong ứng dụng.

1. **NỘI DUNG THỰC HÀNH:**

**6.1 Yêu cầu bổ sung**

Xây dựng chức năng **Hiển thị/Thêm/Xóa/Sửa** cho **Product**

**Bước 1: Tạo ProductService.java**

Trong đường dẫn src/main/java/com.vanlang.webbanhang/service xây dựng ProductService.Java:

|  |
| --- |
| package com.vanlang.webbanhang.service;  import com.vanlang.webbanhang.model.Product; import com.vanlang.webbanhang.repository.ProductRepository; import lombok.NonNull; import lombok.RequiredArgsConstructor; import org.springframework.stereotype.Service; import org.springframework.transaction.annotation.Transactional;  import java.util.List; import java.util.Optional;  @Service @RequiredArgsConstructor @Transactional public class ProductService {  private final ProductRepository productRepository;  // Retrieve all products from the database  public List<Product> getAllProducts() {  return productRepository.findAll();  }  // Retrieve a product by its id  public Optional<Product> getProductById(Long id) {  return productRepository.findById(id);  }  // Add a new product to the database  public Product addProduct(Product product) {  return productRepository.save(product);  }  // Update an existing product  public Product updateProduct(@NonNull Product product) {  Product existingProduct = productRepository.findById(product.getId()) .orElseThrow(() -> new IllegalStateException("Product with ID " + product.getId() + " does not exist."));  existingProduct.setName(product.getName());  existingProduct.setPrice(product.getPrice());  existingProduct.setDescription(product.getDescription());  existingProduct.setCategory(product.getCategory());  return productRepository.save(existingProduct);  }  // Delete a product by its id  public void deleteProductById(Long id) {  if(!productRepository.existsById(id)){  throw new IllegalStateException("Product with ID " + id + " does not exist.");  }  productRepository.deleteById(id);  } } |

**Bước 2: Tạo ProductController.java**

Trong đường dẫn **src/main/java/com.vanlang.webbanhang/controller** tạo một class **ProductController.java**

|  |
| --- |
| package com.vanlang.webbanhang.controller;  import com.vanlang.webbanhang.model.Product; import com.vanlang.webbanhang.service.CategoryService; import com.vanlang.webbanhang.service.ProductService; import jakarta.validation.Valid; 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.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.PathVariable; import org.springframework.web.bind.annotation.PostMapping; import org.springframework.web.bind.annotation.RequestMapping;  @Controller @RequestMapping("/products") public class ProductController {  @Autowired  private ProductService productService;  @Autowired  private CategoryService categoryService;// Đảm bảo bạn đã inject CategoryService  // Display a list of all products  @GetMapping  public String showProducts(Model model) {  model.addAttribute("products",productService.getAllProducts());  return "/products/products-list";  }  // For adding a new product  @GetMapping("/add")  public String showAddForm(Model model) {  model.addAttribute("product", new Product());  model.addAttribute("categories",categoryService.getAllCategories());// Load categories  return "/products/add-product";  }  // Process the form for adding a new product  @PostMapping("/add")  public String addProduct(@Valid Product product, BindingResult bindingResult) {  if(bindingResult.hasErrors()){  return "products/add-product";  }  productService.addProduct(product);  return "redirect:/products";  }  // For editing a product  @GetMapping("/edit/{id}")  public String showEditForm(@PathVariable("id") Long id, Model model) {  Product product = productService.getProductById(id).orElseThrow(() -> new IllegalArgumentException("Invalid product Id:" + id));  model.addAttribute("product", product);  model.addAttribute("categories",categoryService.getAllCategories());//Load categories  return "/products/edit-product";  }  // Process the form for updating a product  @PostMapping("/update/{id}")  public String updateProduct(@PathVariable("id") Long id, @Valid Product product, BindingResult bindingResult) {  if(bindingResult.hasErrors()){  product.setId(id);//// set id to keep it in the form in case of errors  return "/products/update-product";  }  productService.updateProduct(product);  return "redirect:/products";  }   // Handle request to delete a product  @GetMapping("/delete/{id}")  public String deleteProduct(@PathVariable("id") Long id) {  productService.deleteProductById(id);  return "redirect:/products";  } } |

**Bước 3: Tạo view cho product**

Trong đường dẫn **src/main/resources/templates** tạo thêm directory **products**. Sau đó tạo thêm 3 file html: **`add-product.html`,`products-list.html`,`update-product`**

File **`add-product.html`:**

|  |
| --- |
| <!DOCTYPE html> <html xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"  layout:decorate="~{layout}"> <head><title>Add Product</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-5"><h1>Add New Product</h1>  <form th:action="@{/products/add}" th:object="${product}" method="post" class="needs-validation" novalidate>  <div class="mb-3">  <label for="name" class="form-label">Name:</label>  <input type="text" th:field="\*{name}" class="form-control" id="name" required>  <div class="text-danger" th:if="${#fields.hasErrors('name')}" th:errors="\*{name}"></div>  </div>  <div class="mb-3">  <label for="price" class="form-label">Price:</label>  <input type="text" th:field="\*{price}" class="form-control" id="price" required></div>  <div class="mb-3">  <label for="category" class="form-label">Category:</label>  <select th:field="\*{category}" class="form-control" id="category">  <option th:each="category : ${categories}" th:value="${category.id}" th:text="${category.name}"></option>  </select>  </div>  <div class="mb-3">  <label for="description" class="form-label">Description:</label>   <textarea th:field="\*{description}" class="form-control" id="description" required></textarea>  </div>  <button type="submit" class="btn btn-primary">Add Product</button>  <a th:href="@{/products}" class="btn btn-link">Cancel</a></form> </section> <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script> </body> </html> |

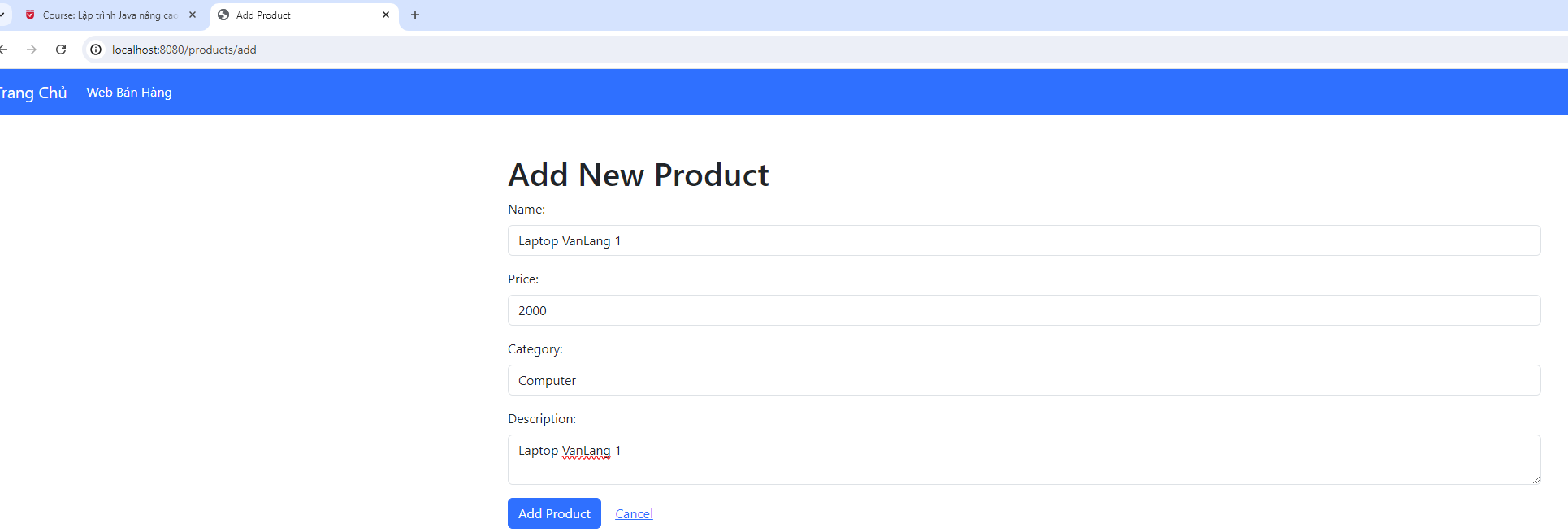
File **`products-list.html`:**

|  |
| --- |
| <!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} ?: 'Product'">Product</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"><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> </section> </body> </html> |

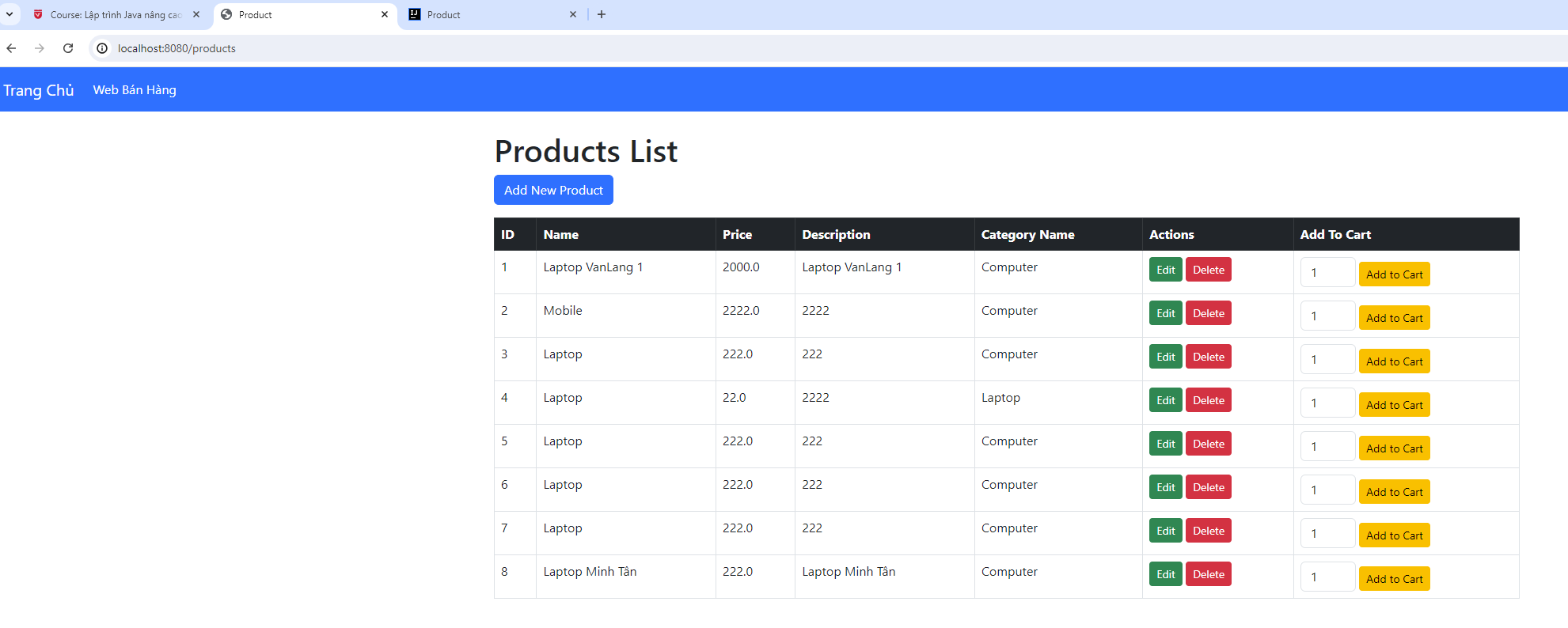
File **`update-product.html`:**

|  |
| --- |
| <!DOCTYPE html> <html xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"  layout:decorate="~{layout}"> <head><title>Update Product</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-5"><h1>Edit Product</h1>  <form th:action="@{/products/update/{id}(id=${product.id})}" th:object="${product}" method="post"  class="needs-validation" novalidate>  <div class="mb-3">  <label for="name" class="form-label">Name:</label>  <input type="text" th:field="\*{name}"  class="form-control" id="name"  required>  <div class="text-danger" th:if="${#fields.hasErrors('name')}" th:errors="\*{name}"></div>  </div>  <div class="mb-3">  <label for="price" class="form-label">Price:</label>  <input type="text" th:field="\*{price}"  class="form-control" id="price"  required></div>  <div class="mb-3">  <label for="category" class="form-label">Category:</label>  <select th:field="\*{category}"  class="form-control"  id="category">  <option th:each="category : ${categories}" th:value="${category.id}" th:text="${category.name}"  th:selected="${category.id == product.category.id}"></option>  </select>  </div>  <div class="mb-3">  <label for="description" class="form-label">Description:</label>  <textarea  th:field="\*{description}" class="form-control" id="description" required>   </textarea>  </div>  <button type="submit" class="btn btn-primary">Save Changes</button>  <a th:href="@{/products}" class="btn btn-link">Cancel</a></form> </section> <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script> </body> </html> |

**Bước 4: Tiến hành build dự án và kiểm tra kết quả:** <http://localhost:8080/products/add>



Thêm thành công và hiển thị danh sách category: <http://localhost:8080/categories>



Chức năng **Edit:**

A screenshot of a computer

Description automatically generated

Hiển thị danh sách sản phẩm sau khi đã sửa:

A screenshot of a computer screen

Description automatically generated

Kiểm tra Database cho thấy kết quả sản phẩm đã tự động lưu, cập nhật vào trong Database:

* *Kiểm tra Database:*

A screenshot of a computer

Description automatically generated