

## Lab 7 Practice

Name: Nguyễn Hữu Hoàng Nam

StudentID: ITCSIU23028

### Product Page

The screenshot shows a web browser window with the URL `localhost:8081/products`. The page title is "Product Management System". At the top left is a blue button labeled "+ Add New Product". To the right is a search bar with the placeholder "Search products..." and a "Search" button. Below the search bar is a table with the following data:

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	10	Electronics	Edit  Delete
2	P002	iPhone 15 Pro	\$999.99	25	Electronics	Edit  Delete
3	P003	Samsung Galaxy S24	\$899.99	20	Electronics	Edit  Delete
4	P004	Office Chair Ergonomic	\$199.99	50	Furniture	Edit  Delete
5	P005	Standing Desk	\$399.99	15	Furniture	Edit  Delete

At the bottom right of the page, there is a watermark: "Activate Windows Go to Settings to activate Windows."

Flow:

1. User accesses `/products`
2. Browser sends GET request to ProductController
3. Controller calls `productService.getAllProducts()`
4. Service calls `productRepository.findAll()`
5. Repository queries database and returns List<Product>
6. Controller adds products to Model
7. Returns "product-list" view
8. Thymeleaf renders product-list.html with product data
9. User sees table with all products

### Search Function

The screenshot shows a web browser window with the URL `localhost:8081/products`. The search bar contains the text "Laptop". The search results show one product in the table:

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	10	Electronics	Edit  Delete

Flow:

1. User types keyword in search box and clicks Search button
2. Browser sends GET request to `/products/search?keyword=laptop`
3. Controller receives keyword parameter
4. Controller calls `productService.searchProducts(keyword)`
5. Service calls `productRepository.findByNameContainingIgnoreCase(keyword)`
6. Repository searches database for products with name containing keyword
7. Controller adds filtered products and keyword to Model
8. Returns "product-list" view
9. User sees only matching products

### Add new Product feature

The screenshot shows a modal dialog titled "Add New Product" with a plus sign icon. The form contains fields for Product Code, Product Name, Price (\$), Quantity, Category, and Description. The "Category" field is a dropdown menu currently set to "Furniture". The "Description" field is a text area containing the word "nothing". At the bottom are "Save Product" and "Cancel" buttons.

Product Code *	009
Product Name *	LENOVO
Price (\$)	1000
Quantity *	10
Category *	Furniture
Description	nothing

Product Management System						
Actions		Product Details				
ID	Code	Name	Price	Quantity	Category	
1	P001	Laptop Dell XPS 13	\$1299.99	10	Electronics	<button>Edit</button> <button>Delete</button>
2	P002	iPhone 15 Pro	\$999.99	25	Electronics	<button>Edit</button> <button>Delete</button>
3	P003	Samsung Galaxy S24	\$899.99	20	Electronics	<button>Edit</button> <button>Delete</button>
4	P004	Office Chair Ergonomic	\$199.99	50	Furniture	<button>Edit</button> <button>Delete</button>
5	P005	Standing Desk	\$399.99	15	Furniture	<button>Edit</button> <button>Delete</button>
6	009	LENOVO	\$1000.00	10	Furniture	<button>Edit</button> <button>Delete</button>

Flow:

1. User fills in product details (code, name, price, quantity, category, description)
2. User clicks Save button
3. Browser sends POST request to `/products/save` with form data
4. Controller receives Product object with form data
5. Controller calls `productService.saveProduct(product)`
6. Service calls `productRepository.save(product)`
7. Repository inserts new product into database
9. Controller adds success message
10. Redirects to `/products`
11. User sees updated product list with new product

## Edit Product

## Edit Product

Product Code \*

Product Name \*

Price (\$) \*

Quantity \*

Category \*

Description

 Save Product Cancel

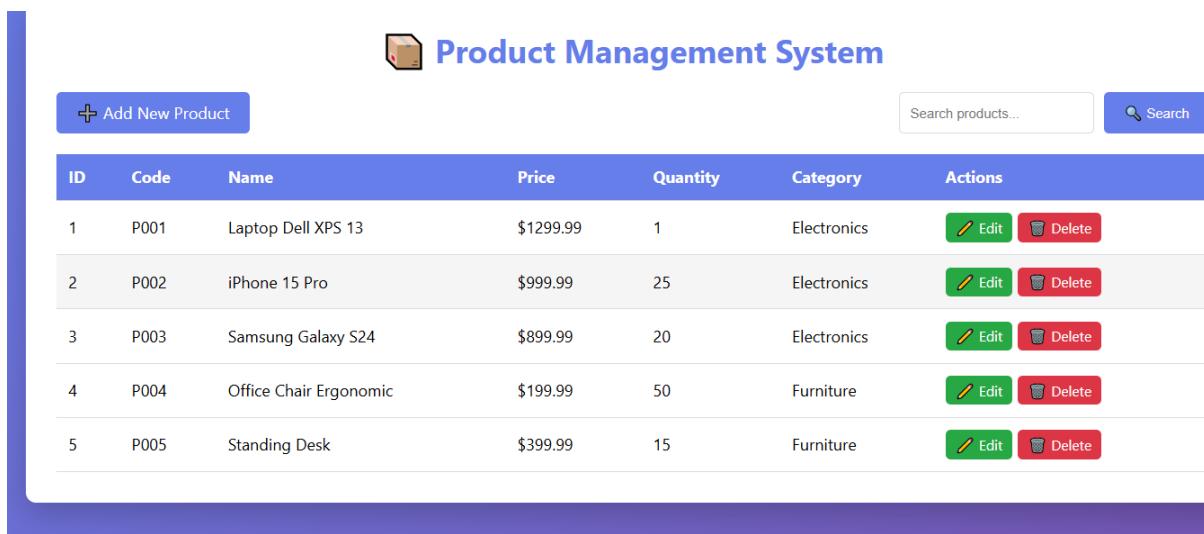
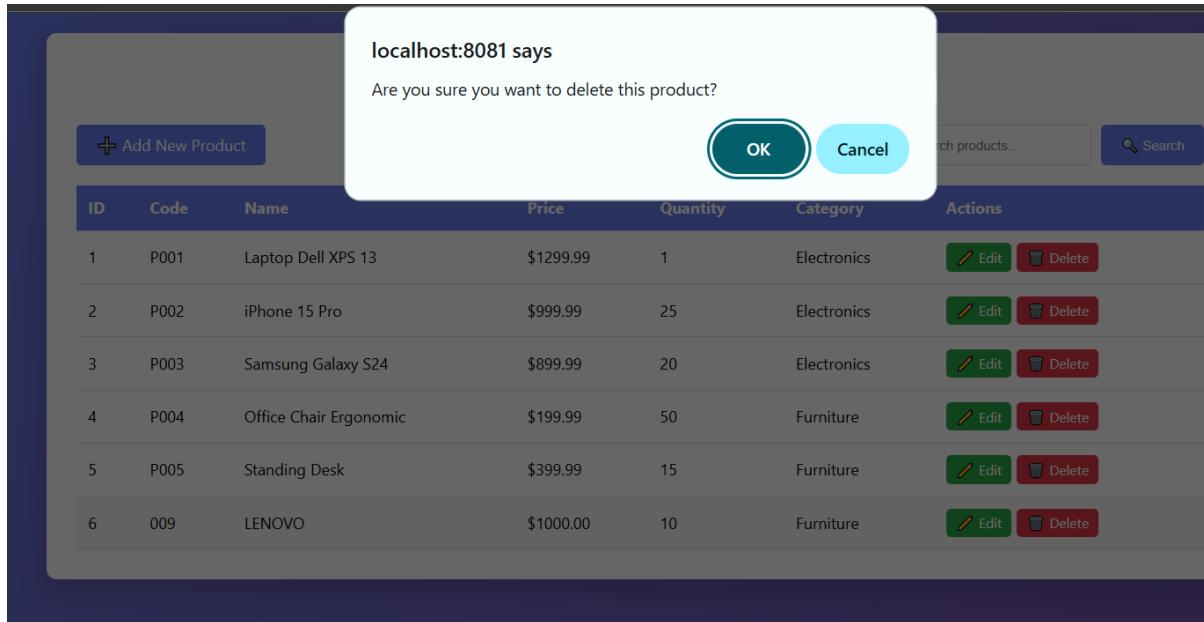
ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	1	Electronics	 

Flow:

1. User modifies product details in the form
2. User clicks Save button
3. Browser sends POST request to `/products/save` with updated data
4. Controller receives Product object with id and updated data
5. Controller calls `productService.saveProduct(product)`
6. Service calls `productRepository.save(product)`

7. Repository updates existing product in database (because id exists)
8. Controller adds success message
9. Redirects to `/products`
10. User sees updated product list

## Delete Product



Flow:

1. User clicks "Delete" button next to a product in the list
2. Browser sends GET request to `/products/delete/1` (where 1 is the product ID)
3. Controller receives id from URL path
4. Controller calls `productService.deleteProduct(id)`
5. Service calls `productRepository.deleteById(id)`

6. Repository deletes product from database
7. Controller adds success message "Product deleted successfully!"
8. Redirects to `/products`
9. User sees updated product list without the deleted product