

Lab 7 Homework  
Name: Nguyễn Hữu Hoàng Nam  
StudentID: ITCSIU23028

Exercise 5:

Add New Product

Search products...

Search

Advanced Search

Product Name:

Category:

Min Price:

Max Price:

Search by name...

All Categories

Min price

Max price

Advanced Search

Clear Filters

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



Product Management System

Add New Product

Search products...

Search

Advanced Search

Product Name:

Category:

Min Price:

Max Price:

E

All Categories

0.09

Max price

Advanced Search

Clear Filters

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	1	Electronics	<div><div></div>Edit</div> <div><div></div>Delete</div>
2	P002	iPhone 15 Pro	\$999.99	25	Electronics	<div><div></div>Edit</div> <div><div></div>Delete</div>
4	P004	Office Chair Ergonomic	\$199.99	50	Furniture	<div><div></div>Edit</div> <div><div></div>Delete</div>
5	P005	Standing Desk	\$399.99	15	Furniture	<div><div></div>Edit</div> <div><div></div>Delete</div>

## Product Management System

[+ Add New Product](#)

Search products...

[Search](#)

### Advanced Search

Product Name:

E

Category:

Electronics

Min Price:

0.09

Max Price:

Max price

[Advanced Search](#)

[Clear Filters](#)

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	1	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>
2	P002	iPhone 15 Pro	\$999.99	25	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>

## Product Management System

[+ Add New Product](#)

Search products...

[Search](#)

### Advanced Search

Product Name:

Search by name...

Category:

All Categories

Min Price:

Min price

Max Price:

Max price

[Advanced Search](#)

[Clear Filters](#)

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	1	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>

[« Previous](#) [1](#) [2](#) [3](#) [Next »](#)

Showing page 1 of 5 (Total: 5 products)

### Flow(Advanced Search):

1. User fills in advanced search form with criteria.
2. User clicks "Advanced Search" button.
3. Browser sends GET request to `/products/advanced-search` with all parameters.
4. Controller receives parameters (name, category, minPrice, maxPrice).
6. Controller calls `productService.advancedSearch`.
7. Service calls `productRepository.searchProducts`.
8. Repository executes custom JPQL query .
9. Repository returns filtered List<Product>.
10. Controller adds filtered products to Model.
11. Returns "product-list" view.
12. User sees only products matching ALL criteria.

**Flow(Search with pagination):**

1. User types keyword in search box and clicks Search
2. Browser sends GET request to `/products/search?keyword=laptop&page=0&size=10`
3. Controller receives parameters:
5. Controller calls `productService.searchProducts(keyword, pageable)`
6. Service calls `productRepository.findByNameContainingIgnoreCase(keyword, pageable)`
7. Repository executes paginated query:
9. Controller extracts data from Page object and adds to Model:
10. Returns "product-list" view

**Exercise 6:**

## Add New Product

Product Code \*

1

Product code must be 3-20 characters  
Product code must start with P followed by numbers

Product Name \*

2

Name must be 3-100 characters

Price (\$) \*

3

Quantity \*

4

Category \*

Electronics



Description

4



Save Product



Cancel

### Flow:

1. User fills in product form with valid data:
2. User clicks Save button
3. Browser sends POST request to `/products/save` with form data
4. Controller receives Product object
5. `@Valid` annotation triggers validation

6. Spring Validator checks all constraints - ALL PASS
7. BindingResult contains no errors
8. Controller checks: ``if (result.hasErrors())`` → FALSE
9. Controller proceeds to save:
  - Calls ``productService.saveProduct(product)``
  - Service calls ``productRepository.save(product)``
  - Repository saves to database
10. Controller adds success message: ``"Product saved successfully!"``
11. Redirects to ``/products``
12. User sees product list with success message

### Exercise 7:

The screenshot shows a web application titled "Product Management System". It features a dashboard with a "Dashboard" button and an "Add New Product" button. A search bar is present with the placeholder "Search products...". Below this, there's a "Filter by Category" section with buttons for "All Categories", "Electronics", "Clothing", "Books", "Furniture", "Food", and "Sports". The "Electronics" button is highlighted. An "Advanced Search" section includes input fields for "Product Name", "Category" (a dropdown menu currently showing "All Categories"), "Min Price", and "Max Price". There are "Advanced Search" and "Clear Filters" buttons. Below the search section is a table of products.

ID	Code	Name	Price	Quantity	Category	Actions
1	P001	Laptop Dell XPS 13	\$1299.99	1	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>
2	P002	iPhone 15 Pro	\$999.99	25	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>
3	P003	Samsung Galaxy S24	\$899.99	20	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>
7	1	2	\$3.00	4	Electronics	<a href="#">Edit</a> <a href="#">Delete</a>

### Flow

1. User selects category "Electronics" AND sorts by "Price" ascending
2. Browser sends GET request with both parameters
3. Controller receives category, sortBy, sortDir
4. If category provided:
  - Calls ``productService.getProductsByCategory(category)``
  - Applies manual sorting using Stream API and Comparator
5. If no category (all products):

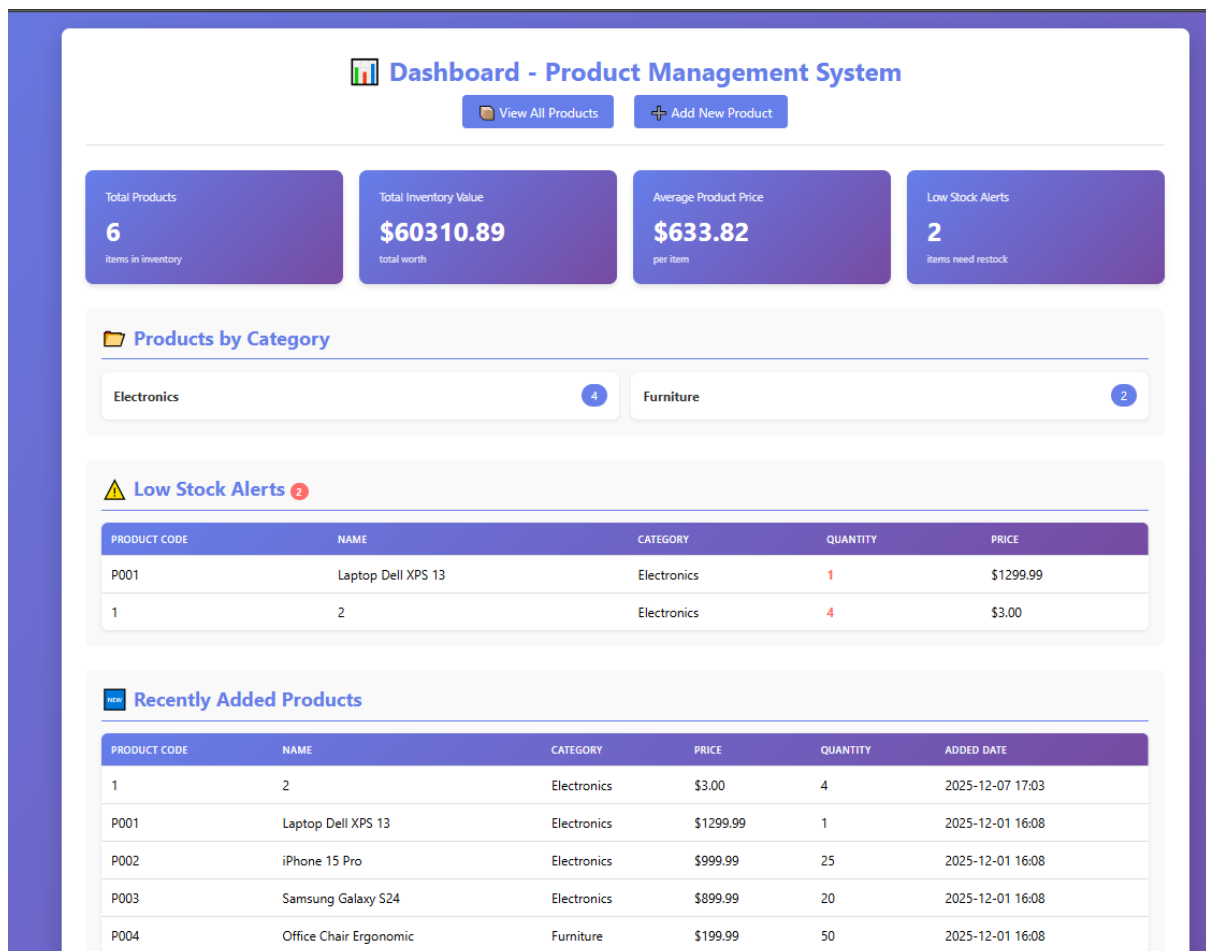
- Creates Sort object
- Calls `productService.getAllProducts(sort)`

6. Returns sorted and filtered List<Product>

7. Controller adds products, category, sortBy, sortDir to Model

8. Returns "product-list" view

### Exercise 8:



### Flow:

1. User clicks "Dashboard" button from product list.
2. Browser sends GET request to `/dashboard`.
3. Controller gathers 6 types of statistics from ProductService.
4. Service calls repository methods with @Query annotations
5. Repository executes SQL queries .
6. Controller adds all statistics to Model.
7. Returns "dashboard" view.