**PHÁT TRIỂN ỨNG DỤNG WEB 2**

**BÁO CÁO DỰ ÁN MÔN HỌC**

**ĐỀ TÀI XÂY DỰNG WEBSITE QUẢN LÝ VĂN PHÒNG PHẨM NHÀ SÁCH FAHASA NHA TRANG**

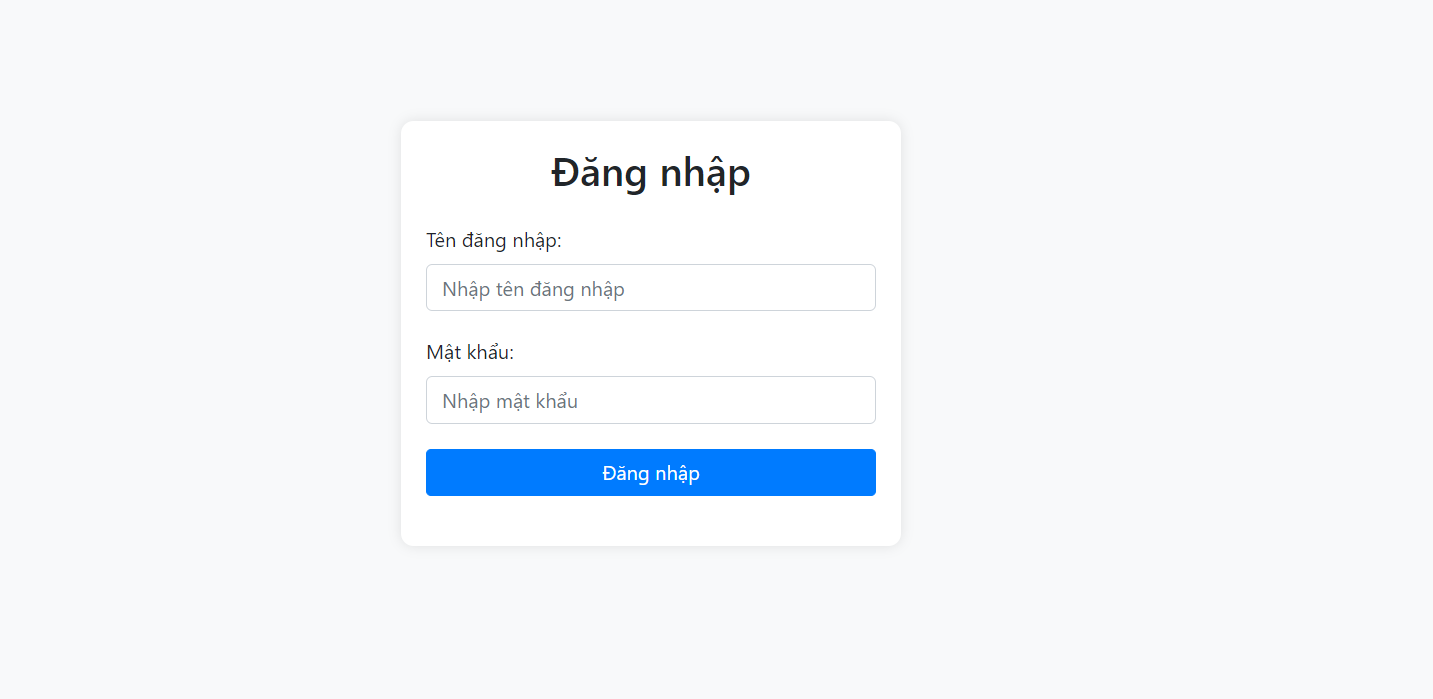
Sinh viên thực hiện: Phạm Thế Hùng\_63134128

**BÁO CÁO**

**Giao diện Trang chủ:**

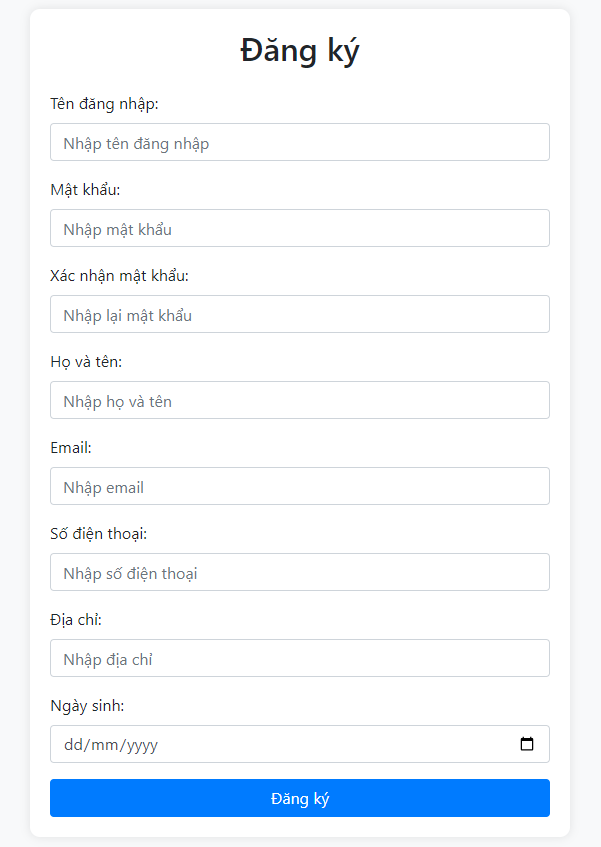


Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang chủ cho website quản lý văn phòng phẩm của nhà sách Fahasa Nha Trang. Em dùng Bootstrap để tích hợp các thành phần giao diện như button và container. Em sử dụng Font Awesome để thêm các biểu tượng vào trang. Em sử dụng CSS để tạo phong cách và giao diện cho trang, bao gồm các thành phần như hình nền, chữ, và nút bấm với các hiệu ứng chuyển đổi và bóng đổ. Em tích hợp các liên kết tới trang đăng nhập và đăng ký. Kết quả đạt được là một trang chủ với giao diện hấp dẫn, trực quan, và các nút bấm điều hướng rõ ràng giúp người dùng dễ dàng truy cập vào các chức năng đăng nhập và đăng ký.

**Giao diện trang đăng nhập**

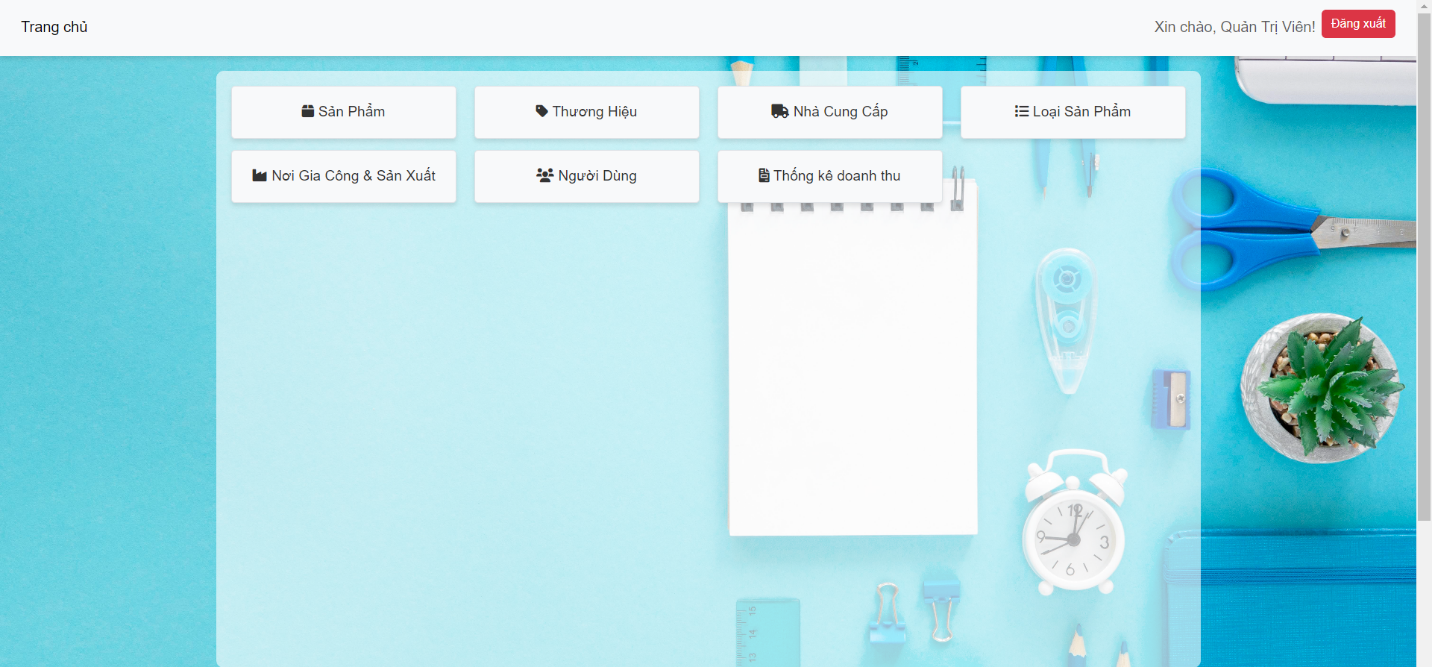
Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang đăng nhập. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, button và container. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng bóng đổ cho các thành phần, hiệu ứng khi hover và focus trên các input và button. Em sử dụng jQuery để thêm hiệu ứng ẩn dần các thông báo lỗi sau một khoảng thời gian. Kết quả đạt được là một trang đăng nhập với giao diện chuyên nghiệp, thân thiện với người dùng, và có các tính năng kiểm tra thông tin đăng nhập, hiển thị thông báo lỗi và hiệu ứng trực quan giúp cải thiện trải nghiệm người dùng.

**Giao diện trang đăng ký:**

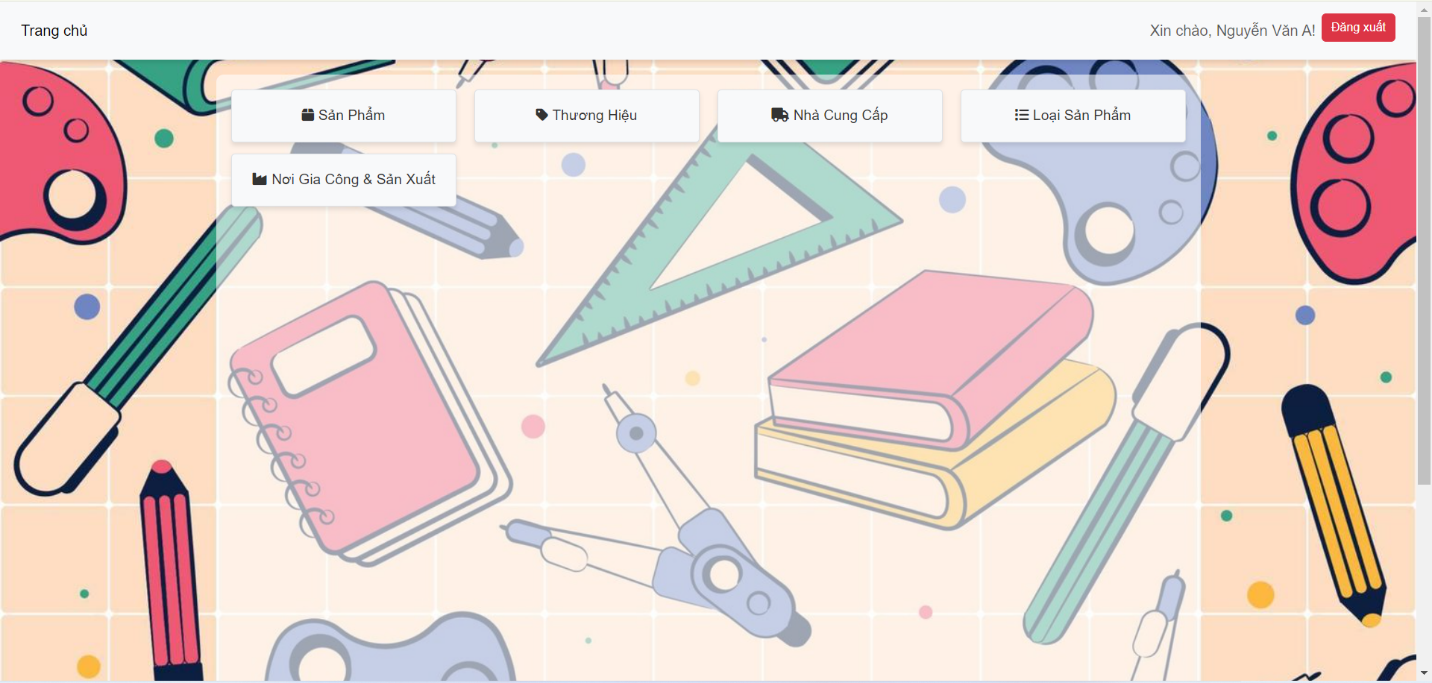


Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang đăng ký cho người dùng. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, button và container. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng bóng đổ cho các thành phần và hiệu ứng hiển thị trang đăng ký. Em sử dụng Thymeleaf để xử lý các yêu cầu động và liên kết dữ liệu người dùng nhập vào với các biến phía sever thông qua cú pháp ‘th’ và ‘th’. Em sử dụng jQuery để thêm hiệu ứng ẩn dần các thông báo lỗi sau một khoảng thời gian. Kết quả đạt được là một trang đăng ký với giao diện thân thiện, dễ sử dụng, và các tính năng kiểm tra thông tin đầu vào, hiển thị thông báo lỗi và hỗ trợ người dùng nhập thông tin một cách trực quan.

**Giao diện trang Quản trị viên:**



Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang chủ cho hệ thống quản lý sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như navbar, container và các thẻ điều hướng (nav-item). Em sử dụng Font Awesome để thêm các biểu tượng vào trang, giúp tăng tính trực quan và hấp dẫn. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hình nền, các hiệu ứng bóng đổ, hiệu ứng hover và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và tích hợp các thành phần như navbar và footer thông qua cú pháp ‘th’. Kết quả đạt được là một trang chủ với giao diện thân thiện, hiện đại, và dễ sử dụng, giúp người dùng dễ dàng điều hướng tới các phần quan trọng như sản phẩm, thương hiệu, nhà cung cấp, loại sản phẩm, nơi gia công và sản xuất, người dùng và thống kê doanh thu.

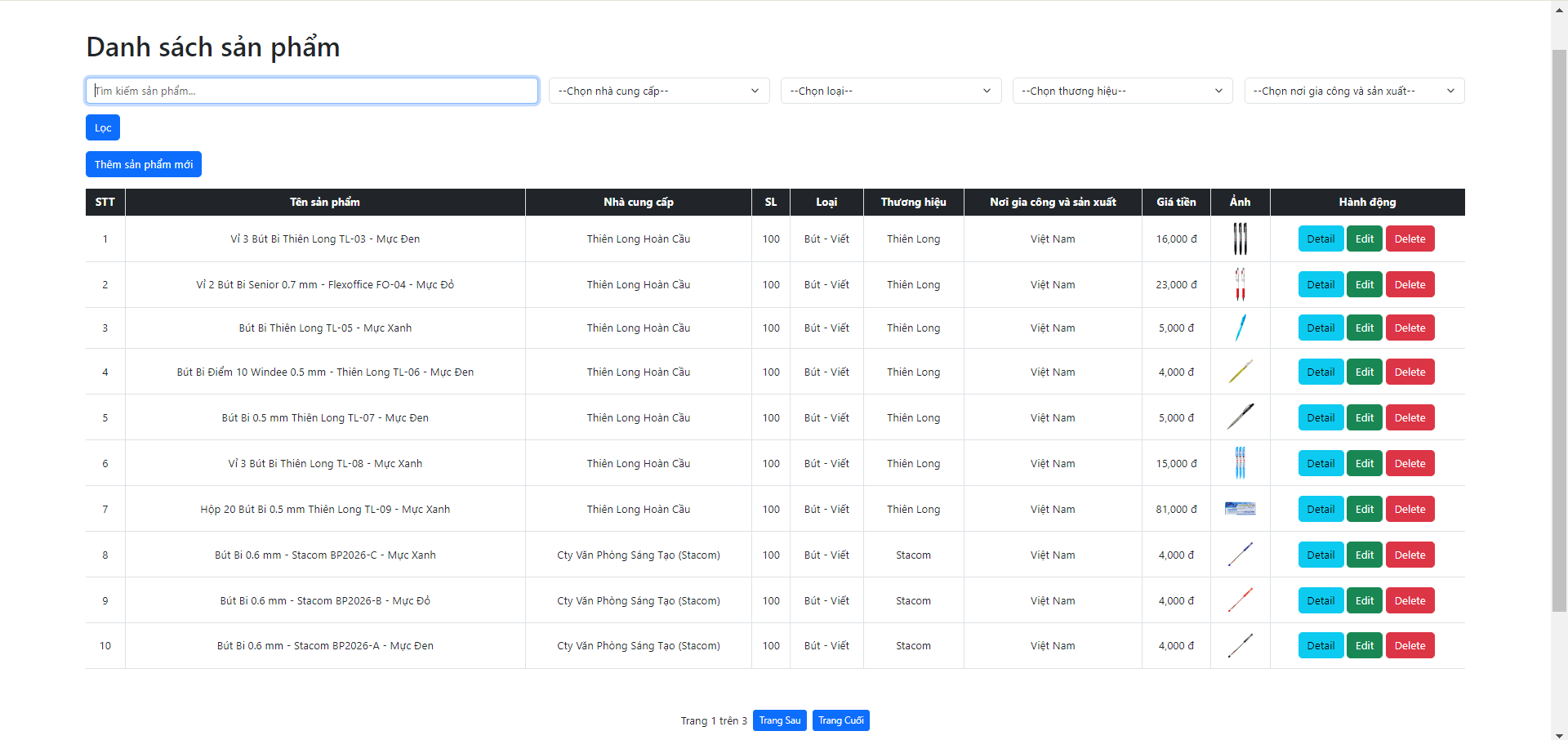
**Giao diện trang Nhân viên:**

Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang nhân viên. Em dùng Bootstrap để tích hợp các thành phần giao diện như navbar, container và các thẻ điều hướng (nav-item). Em sử dụng Font Awesome để thêm các biểu tượng vào trang, giúp tăng tính trực quan và hấp dẫn. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hình nền, các hiệu ứng bóng đổ, hiệu ứng hover và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và tích hợp các thành phần như navbar và footer thông qua cú pháp ‘th’. Kết quả đạt được là một trang nhân viên với giao diện thân thiện, hiện đại, và dễ sử dụng, giúp người dùng dễ dàng điều hướng tới các phần quan trọng như sản phẩm, thương hiệu, nhà cung cấp, loại sản phẩm, nơi gia công và sản xuất.

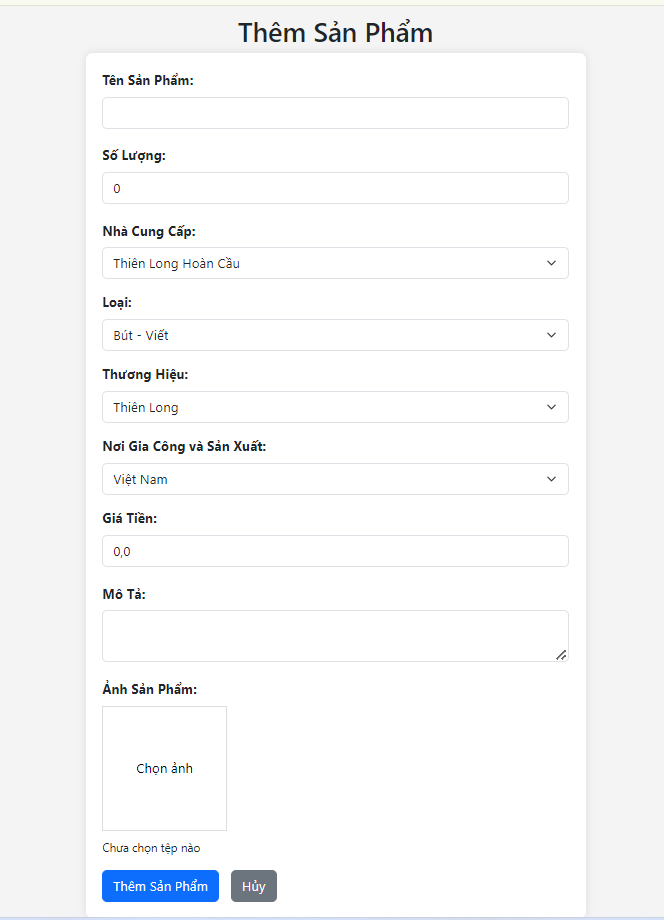
**Các danh mục trong trang Admin:**

1. Danh mục Sản phẩm:

Giao diện của danh mục Sản Phẩm:

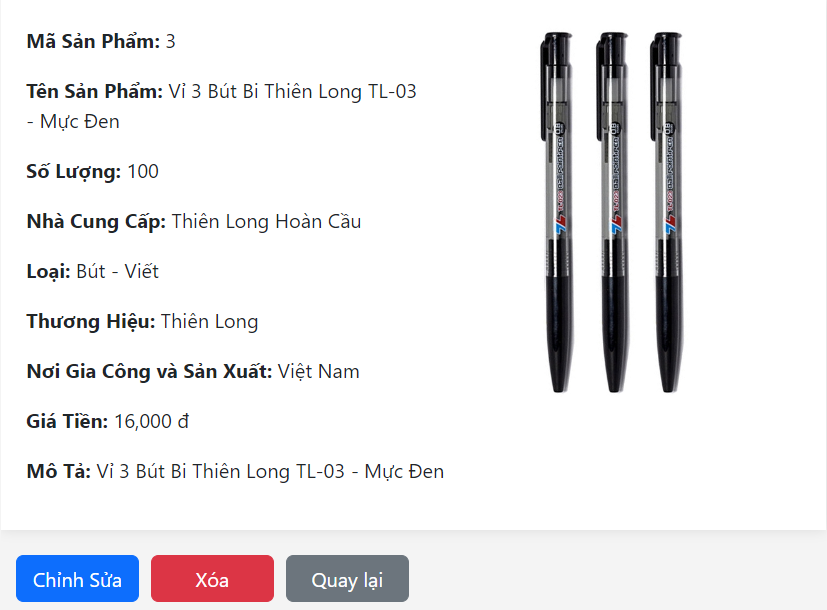
Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang danh sách sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form tìm kiếm, table hiển thị sản phẩm, và các nút điều hướng. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng phóng to hình ảnh sản phẩm khi hover, hiệu ứng khi focus trên các input, và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và liên kết dữ liệu sản phẩm với giao diện thông qua các cú pháp ‘th’, ‘th’, ‘th’ và ‘th’. Kết quả đạt được là một trang danh sách sản phẩm với giao diện trực quan, dễ sử dụng, và các tính năng tìm kiếm, lọc sản phẩm, hiển thị thông tin chi tiết, chỉnh sửa, và xóa sản phẩm, cùng với hệ thống phân trang giúp người dùng dễ dàng điều hướng qua các trang sản phẩm khác nhau.

* Chức năng thêm sản phẩm mới:



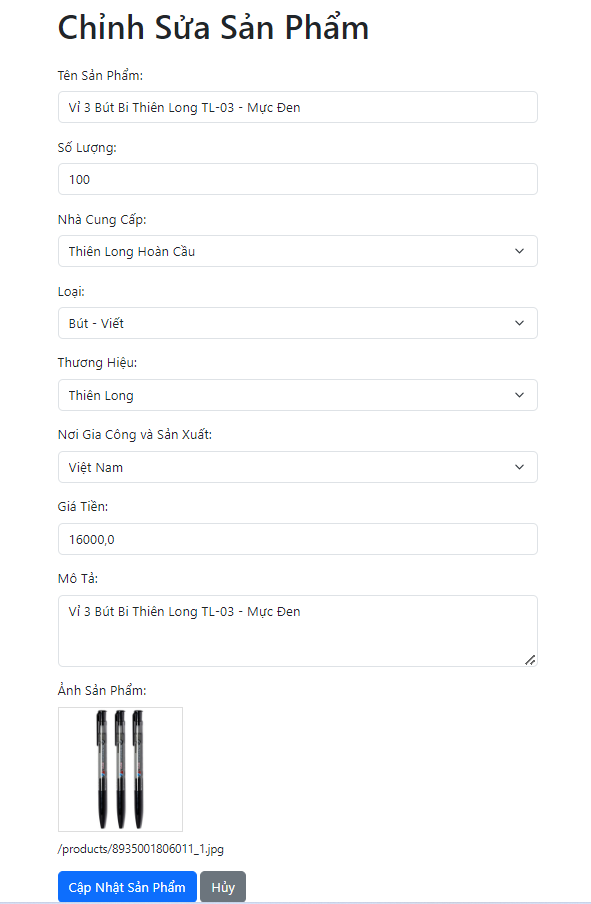
Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang thêm sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, select, và button. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng hiển thị và xem trước hình ảnh sản phẩm, các hiệu ứng bóng đổ, và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và liên kết dữ liệu sản phẩm với giao diện thông qua các cú pháp ‘th’ và ‘th’. Em tích hợp JavaScript để tạo hiệu ứng xem trước hình ảnh khi người dùng chọn tệp ảnh. Kết quả đạt được là một trang thêm sản phẩm với giao diện thân thiện, dễ sử dụng, và các tính năng giúp người dùng nhập thông tin sản phẩm một cách trực quan và chính xác.

* Chức năng xem chi tiết sản phẩm:



Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang chi tiết sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form hiển thị thông tin sản phẩm và các nút hành động. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng bóng đổ, căn chỉnh hình ảnh sản phẩm và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và liên kết dữ liệu sản phẩm với giao diện thông qua các cú pháp ‘th’, ‘th’ và ‘th’. Kết quả đạt được là một trang chi tiết sản phẩm với giao diện rõ ràng, đẹp mắt, và các tính năng hiển thị thông tin chi tiết của sản phẩm, cùng với các nút hành động cho phép người dùng chỉnh sửa, xóa sản phẩm, hoặc quay lại trang danh sách sản phẩm.

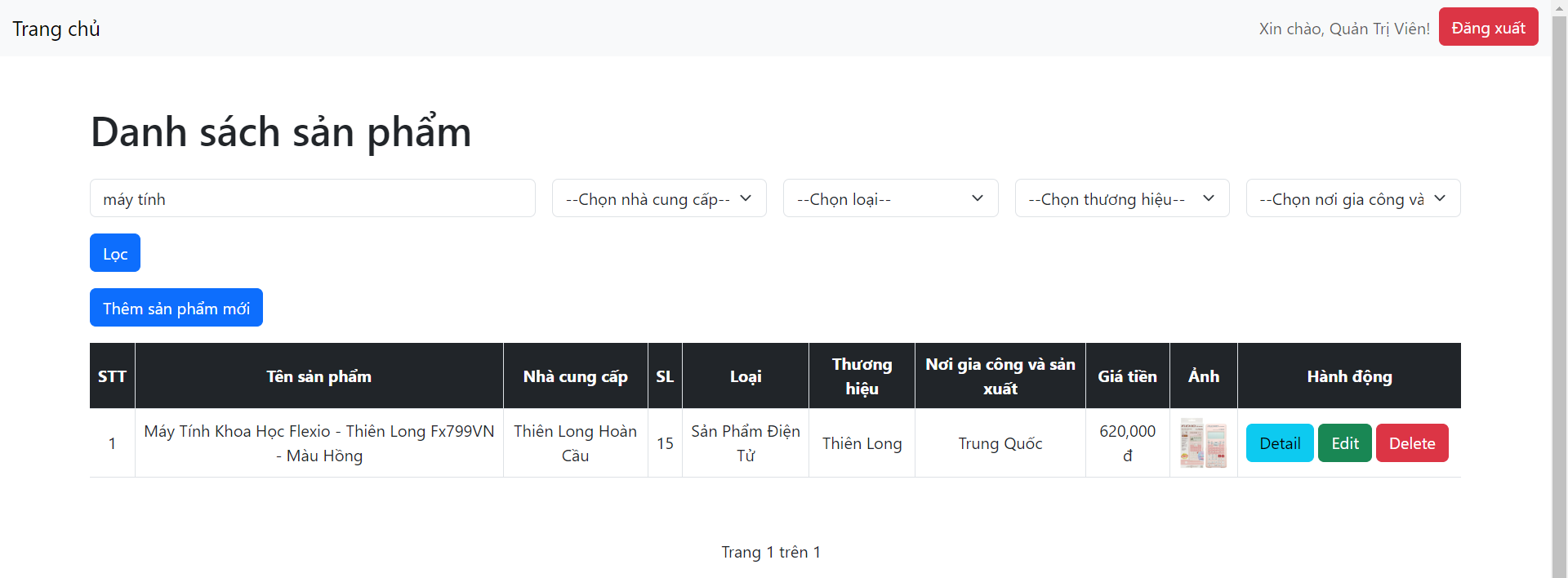
* Chức năng Chỉnh sửa sản phẩm:

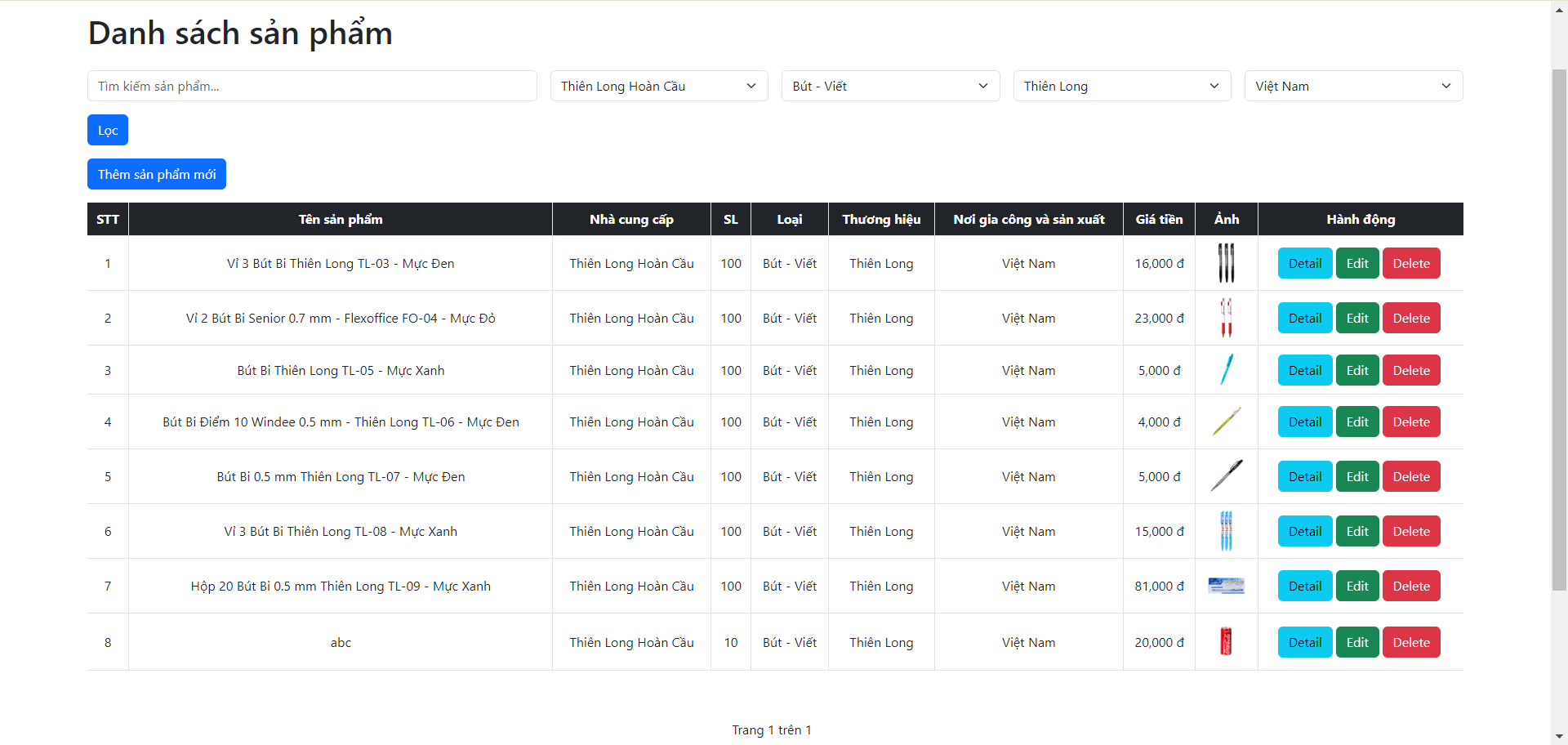


Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang chỉnh sửa sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, select, và button. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng hiển thị và xem trước hình ảnh sản phẩm, các hiệu ứng bóng đổ, và các thành phần trang trí khác. Em sử dụng Thymeleaf để xử lý các yêu cầu động và liên kết dữ liệu sản phẩm với giao diện thông qua các cú pháp ‘th’, ‘th’, và ‘th’. Em tích hợp JavaScript để tạo hiệu ứng xem trước hình ảnh khi người dùng chọn tệp ảnh. Kết quả đạt được là một trang chỉnh sửa sản phẩm với giao diện thân thiện, dễ sử dụng, và các tính năng giúp người dùng chỉnh sửa thông tin sản phẩm một cách trực quan và chính xác.

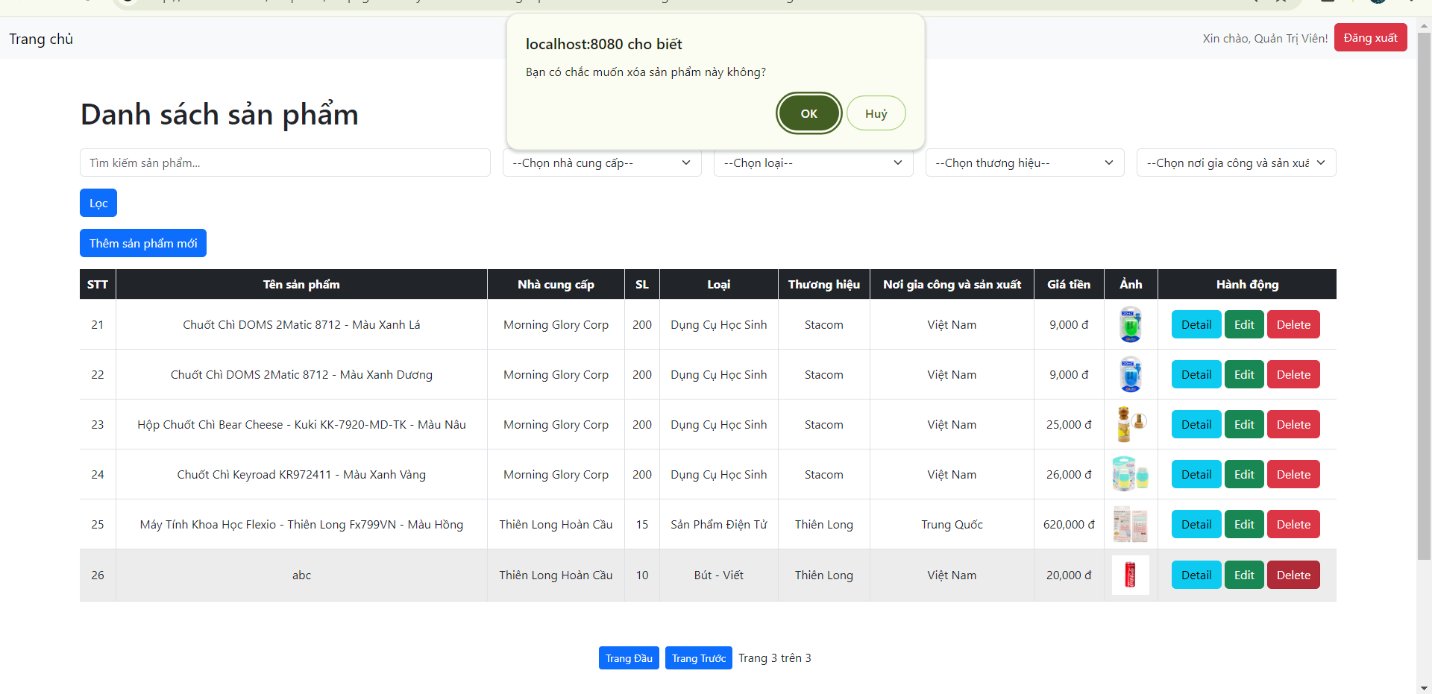
* Chức năng Tìm kiếm sản phẩm:

Để làm được chức năng tìm kiếm sản phẩm, em sử dụng Spring Framework để xây dựng các tầng xử lý trong ứng dụng. Em dùng Spring MVC trong **SanPhamController** để nhận yêu cầu từ người dùng và điều phối các dịch vụ liên quan. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, cho phép nhập từ khóa và các tiêu chí tìm kiếm. Em tái sử dụng các dịch vụ **SanPhamService**, **NhaCungCapService**, **LoaiService**, **ThuongHieuService**, và **NoiGiaCongVaSanXuatService** để lấy dữ liệu từ cơ sở dữ liệu và cung cấp cho giao diện. Kết quả đạt được là một chức năng tìm kiếm sản phẩm hiệu quả, cho phép người dùng dễ dàng tìm kiếm và lọc sản phẩm theo từ khóa, nhà cung cấp, loại, thương hiệu, và nơi gia công và sản xuất.

* Chức năng lọc các sản phẩm dựa trên nhà cung cấp, loại sản phẩm, thương hiệu, nơi gia công và sản xuất:

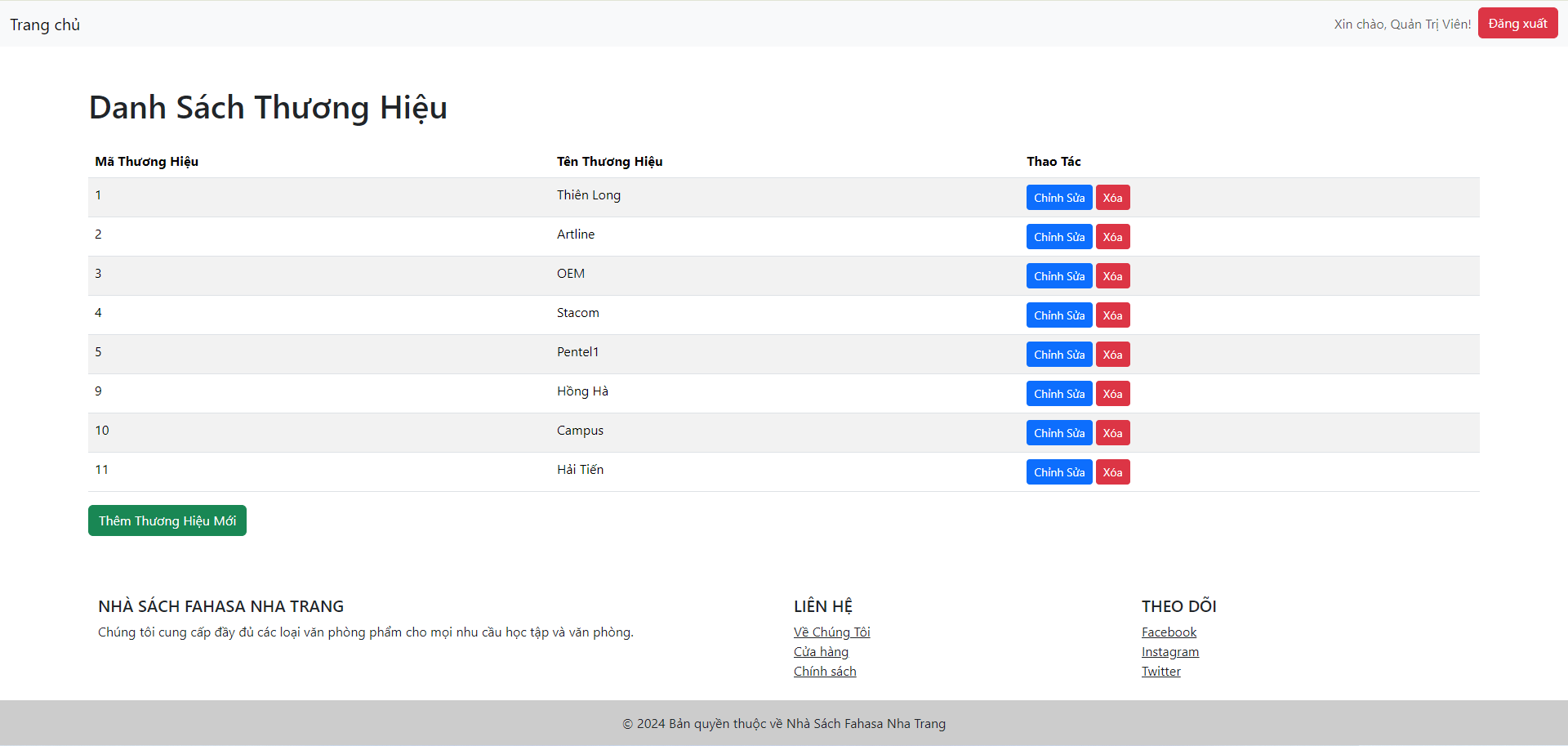
Để làm được chức năng lọc sản phẩm dựa trên nhà cung cấp, loại sản phẩm, thương hiệu, và nơi gia công và sản xuất, em sử dụng Spring Framework để xây dựng các tầng xử lý trong ứng dụng. Em dùng Spring MVC trong **SanPhamController** để nhận các tham số lọc từ yêu cầu của người dùng và điều phối các dịch vụ liên quan. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, cho phép chọn các tiêu chí lọc như nhà cung cấp, loại sản phẩm, thương hiệu, và nơi gia công và sản xuất. Em tái sử dụng các dịch vụ **SanPhamService**, **NhaCungCapService**, **LoaiService**, **ThuongHieuService**, và **NoiGiaCongVaSanXuatService** để truy vấn và lấy dữ liệu sản phẩm từ cơ sở dữ liệu dựa trên các tiêu chí lọc. Kết quả đạt được là một chức năng lọc sản phẩm hiệu quả, cho phép người dùng dễ dàng tìm kiếm và hiển thị các sản phẩm phù hợp với các tiêu chí được chọn, giúp nâng cao trải nghiệm người dùng và hiệu quả quản lý sản phẩm.

* Chức năng xóa sản phẩm:

Để làm được chức năng xóa sản phẩm, em sử dụng Spring Framework để xây dựng các tầng xử lý trong ứng dụng. Em dùng Spring MVC trong **SanPhamController** để nhận yêu cầu xóa sản phẩm từ người dùng thông qua đường dẫn có tham số ID của sản phẩm cần xóa. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, cho phép xác nhận hành động xóa. Em tái sử dụng dịch vụ **SanPhamService** để thực hiện xóa sản phẩm khỏi cơ sở dữ liệu bằng cách gọi phương thức **deleteSanPhamById** với ID của sản phẩm. Kết quả đạt được là sản phẩm được xóa khỏi hệ thống, giao diện người dùng được cập nhật để phản ánh thay đổi này, và người dùng có thể quản lý sản phẩm một cách hiệu quả hơn.

1. Danh mục Thương hiệu:

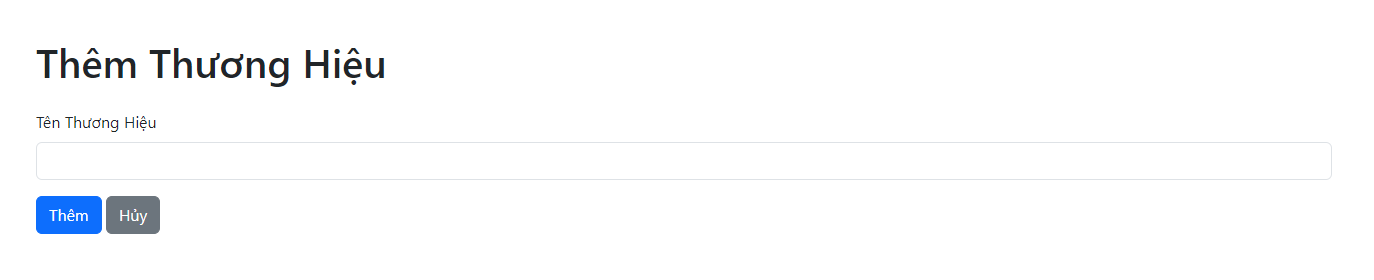
Giao diện của danh mục Thương Hiệu:



Để làm được trang "Danh sách thương hiệu", em sử dụng HTML để tạo cấu trúc trang và hiển thị danh sách các thương hiệu hiện có. Em dùng Bootstrap để tích hợp các thành phần giao diện như bảng, button và container, giúp tạo giao diện người dùng thân thiện và hiện đại. Em sử dụng Thymeleaf để hiển thị dữ liệu động từ server, bao gồm việc lặp qua danh sách thương hiệu và hiển thị từng thương hiệu trong bảng thông qua cú pháp **th:each**. Em tích hợp các liên kết điều hướng cho phép người dùng chỉnh sửa hoặc xóa thương hiệu thông qua các button với các đường dẫn động được tạo bởi Thymeleaf. Kết quả đạt được là một trang danh sách thương hiệu dễ sử dụng, cho phép người dùng quản lý các thương hiệu bằng cách thêm mới, chỉnh sửa hoặc xóa các thương hiệu hiện có.

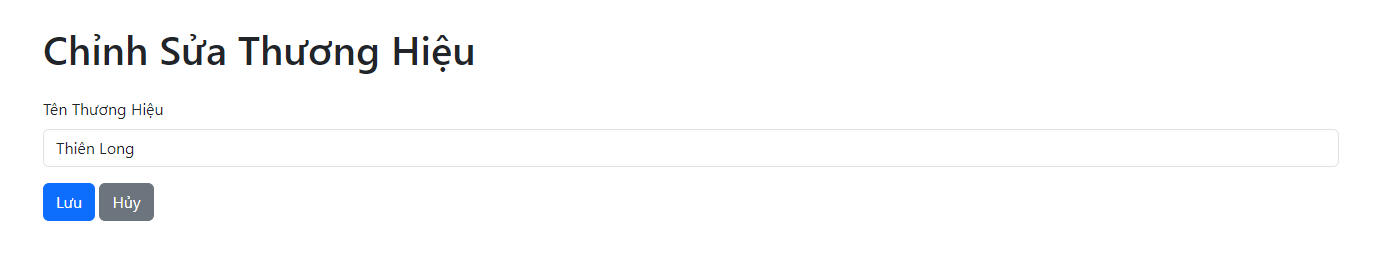
* Chức năng thêm thương hiệu mới:

Để làm được chức năng thêm thương hiệu, em sử dụng Spring Framework để xây dựng các tầng xử lý trong ứng dụng. Em dùng Spring MVC trong **ThuongHieuController** để nhận yêu cầu thêm thương hiệu từ người dùng và điều phối các dịch vụ liên quan. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, bao gồm một form cho phép người dùng nhập thông tin thương hiệu mới. Em tái sử dụng dịch vụ **ThuongHieuService** để lưu thông tin thương hiệu mới vào cơ sở dữ liệu bằng cách gọi phương thức **saveThuongHieu**. Kết quả đạt được là một chức năng thêm thương hiệu hiệu quả, cho phép người dùng dễ dàng thêm thương hiệu mới vào hệ thống thông qua giao diện đơn giản và trực quan.



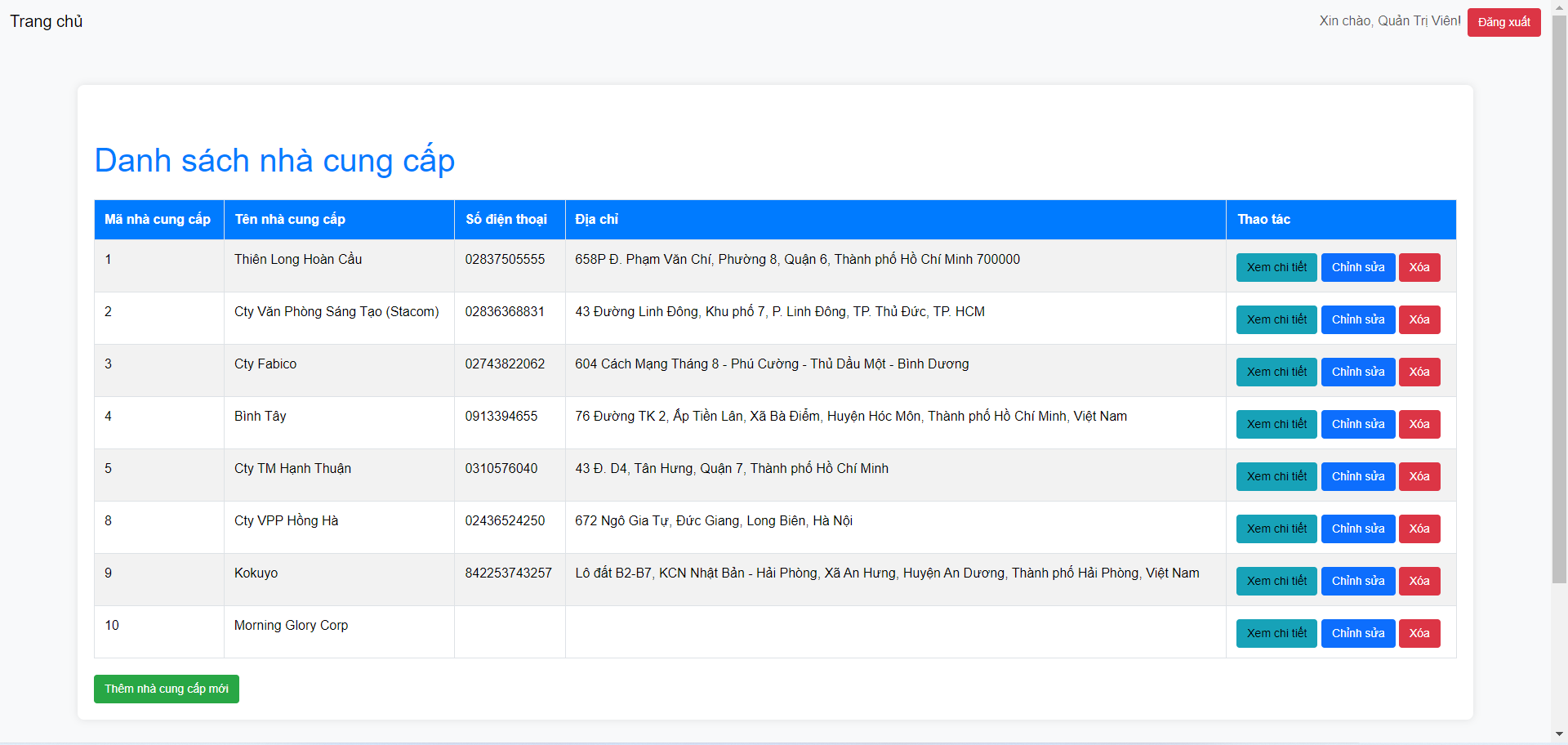
* Chức năng Chỉnh sửa thương hiệu:

Để làm được chức năng chỉnh sửa thương hiệu, em sử dụng Spring Framework để xây dựng các tầng xử lý trong ứng dụng. Em dùng Spring MVC trong **ThuongHieuController** để nhận yêu cầu chỉnh sửa thương hiệu từ người dùng và điều phối các dịch vụ liên quan. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, bao gồm một form hiển thị thông tin thương hiệu hiện tại và cho phép người dùng chỉnh sửa thông tin đó. Em tái sử dụng dịch vụ **ThuongHieuService** để lấy thông tin thương hiệu từ cơ sở dữ liệu bằng cách gọi phương thức **getThuongHieuById** và lưu các thay đổi bằng cách gọi phương thức **saveThuongHieu**. Kết quả đạt được là một chức năng chỉnh sửa thương hiệu hiệu quả, cho phép người dùng dễ dàng cập nhật thông tin thương hiệu thông qua giao diện đơn giản và trực quan, giúp duy trì dữ liệu thương hiệu luôn chính xác và cập nhật.



1. Danh mục Nhà cung cấp

Giao diện của danh mục Nhà cung cấp:



Để làm được trang "Danh sách nhà cung cấp", em sử dụng HTML để tạo cấu trúc trang và hiển thị danh sách các nhà cung cấp hiện có. Em dùng Bootstrap để tích hợp các thành phần giao diện như bảng, button và container, giúp tạo giao diện người dùng thân thiện và hiện đại. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thêm hiệu ứng hover cho các hàng trong bảng, thay đổi màu nền và màu chữ của bảng, và tạo các nút hành động với các màu sắc khác nhau cho từng chức năng. Em tích hợp Thymeleaf để hiển thị dữ liệu động từ server, bao gồm việc lặp qua danh sách nhà cung cấp và hiển thị từng nhà cung cấp trong bảng thông qua cú pháp **th:each**. Em cũng tích hợp các liên kết điều hướng cho phép người dùng xem chi tiết, chỉnh sửa hoặc xóa nhà cung cấp thông qua các button với các đường dẫn động được tạo bởi Thymeleaf. Kết quả đạt được là một trang danh sách nhà cung cấp dễ sử dụng, cho phép người dùng quản lý các nhà cung cấp bằng cách thêm mới, chỉnh sửa hoặc xóa các nhà cung cấp hiện có, giúp nâng cao hiệu quả quản lý và vận hành hệ thống.

* Chức năng thêm Nhà cung cấp:

Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang thêm nhà cung cấp. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng **NhaCungCap** thông qua cú pháp **th:field**. Em tái sử dụng dịch vụ **NhaCungCapService** để lưu thông tin nhà cung cấp mới vào cơ sở dữ liệu bằng cách gọi phương thức **saveNhaCungCap**. Kết quả đạt được là một trang thêm nhà cung cấp dễ sử dụng, cho phép người dùng nhập thông tin nhà cung cấp mới một cách trực quan và chính xác, và lưu trữ thông tin này vào hệ thống.

* Chức năng xem chi tiết nhà cung cấp:

Để làm được việc này, em sử dụng HTML để tạo cấu trúc trang xem chi tiết nhà cung cấp. Em dùng Bootstrap để tích hợp các thành phần giao diện như container, card, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để hiển thị dữ liệu động, bao gồm việc liên kết các thuộc tính của đối tượng **NhaCungCap** với các phần tử HTML thông qua cú pháp **th:text**. Em tái sử dụng dịch vụ **NhaCungCapService** để lấy thông tin chi tiết của nhà cung cấp từ cơ sở dữ liệu và hiển thị lên trang. Kết quả đạt được là một trang xem chi tiết nhà cung cấp dễ sử dụng, cho phép người dùng xem thông tin chi tiết của nhà cung cấp một cách trực quan và chính xác.



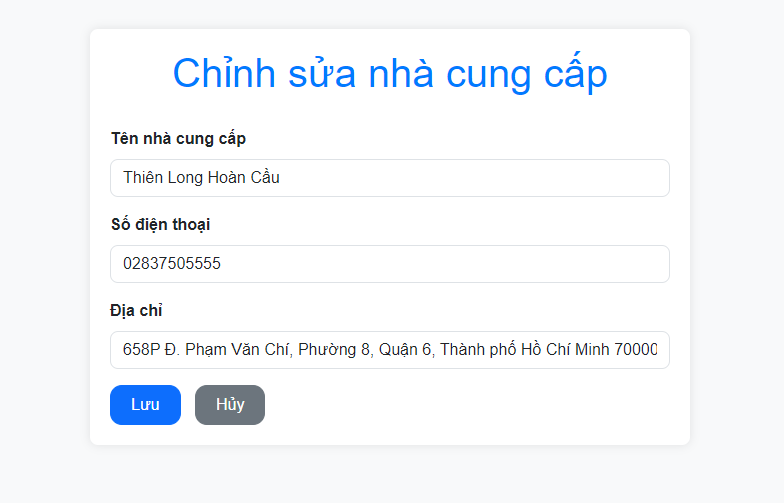
* Chức năng chỉnh sửa nhà cung cấp:

Để làm được chức năng chỉnh sửa nhà cung cấp, em sử dụng HTML để tạo cấu trúc trang chỉnh sửa nhà cung cấp. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng NhaCungCap thông qua cú pháp th:field.

Em sử dụng Spring MVC trong NhaCungCapController để nhận yêu cầu chỉnh sửa từ người dùng, lấy thông tin nhà cung cấp hiện tại từ cơ sở dữ liệu thông qua dịch vụ NhaCungCapService, và hiển thị lên giao diện. Khi người dùng gửi thông tin chỉnh sửa, controller sẽ nhận dữ liệu và gọi phương thức saveNhaCungCap của NhaCungCapService để cập nhật thông tin nhà cung cấp trong cơ sở dữ liệu.

Em tái sử dụng NhaCungCapRepository để truy vấn và lưu trữ thông tin nhà cung cấp. Trong NhaCungCapServiceImpl, em triển khai các phương thức của NhaCungCapService, bao gồm getNhaCungCapById để lấy thông tin nhà cung cấp theo ID và saveNhaCungCap để lưu thông tin nhà cung cấp đã chỉnh sửa.

Kết quả đạt được là một trang chỉnh sửa nhà cung cấp dễ sử dụng, cho phép người dùng cập nhật thông tin nhà cung cấp một cách trực quan và chính xác, đảm bảo dữ liệu luôn được duy trì đúng và cập nhật.



* Chức năng xóa nhà cung cấp:

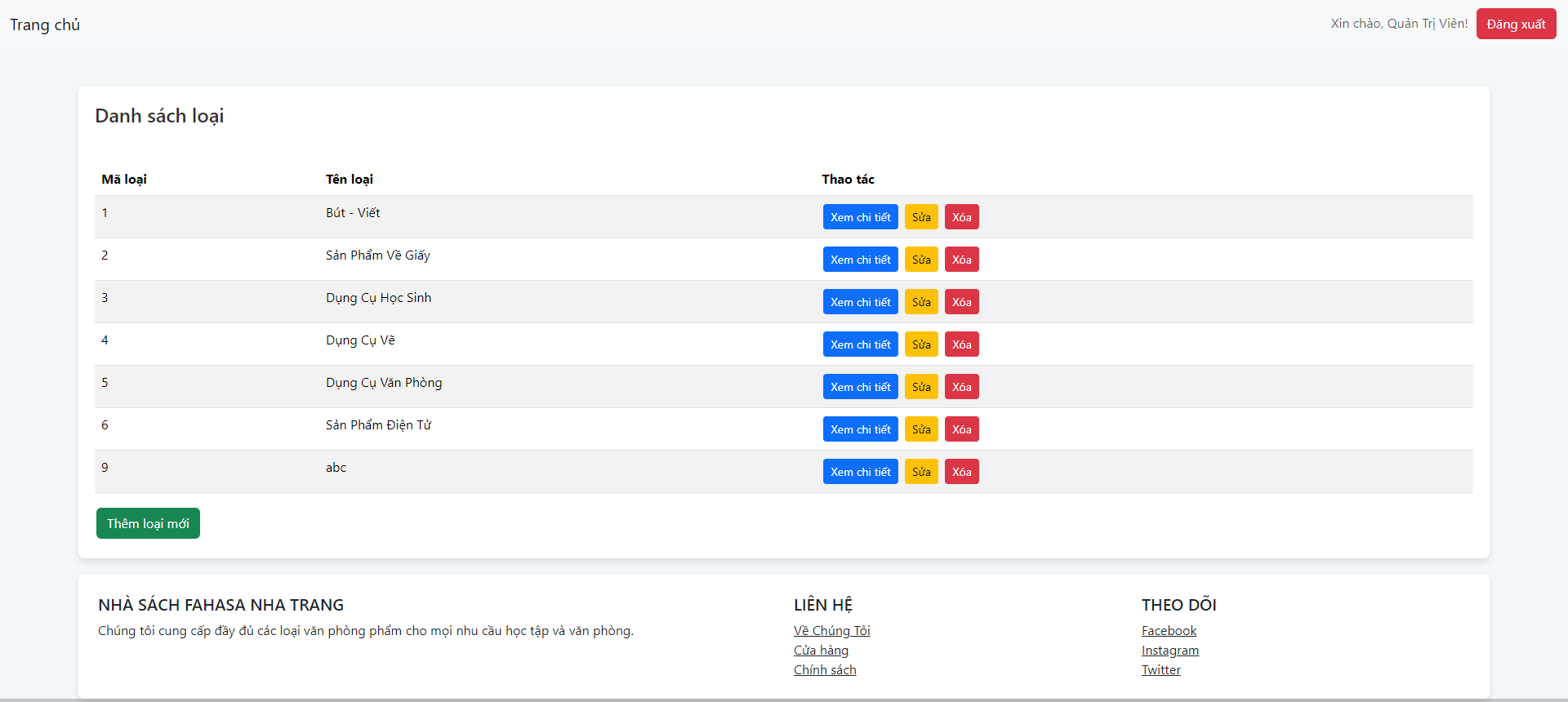
Để làm được chức năng xóa nhà cung cấp, em sử dụng Spring MVC trong NhaCungCapController để nhận yêu cầu xóa nhà cung cấp từ người dùng thông qua đường dẫn có tham số ID của nhà cung cấp cần xóa. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, bao gồm các nút xóa cho từng nhà cung cấp trong danh sách.

Khi người dùng yêu cầu xóa, controller sẽ gọi phương thức deleteNhaCungCapById của NhaCungCapService để thực hiện việc xóa nhà cung cấp khỏi cơ sở dữ liệu. Em tái sử dụng NhaCungCapRepository để truy vấn và xóa thông tin nhà cung cấp. Trong NhaCungCapServiceImpl, em triển khai các phương thức của NhaCungCapService, bao gồm deleteNhaCungCapById để xóa nhà cung cấp theo ID.

Kết quả đạt được là chức năng xóa nhà cung cấp hoạt động hiệu quả, cho phép người dùng dễ dàng xóa nhà cung cấp khỏi hệ thống thông qua giao diện đơn giản và trực quan. Dữ liệu nhà cung cấp được cập nhật kịp thời và chính xác trong cơ sở dữ liệu.

1. Danh mục Loại Sản Phẩm:

* Giao diện của danh mục Loại Sản Phẩm:



Để làm được giao diện "Danh sách loại sản phẩm", em sử dụng HTML để tạo cấu trúc trang hiển thị danh sách các loại sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như bảng, button và container, giúp tạo giao diện người dùng thân thiện và hiện đại. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc tạo hiệu ứng hover cho các nút, thay đổi màu nền và màu chữ của bảng, và thêm bóng đổ cho container.

Em tích hợp Thymeleaf để hiển thị dữ liệu động, bao gồm việc liên kết danh sách loại sản phẩm từ server với bảng thông qua cú pháp th:each. Các nút hành động như "Xem chi tiết", "Sửa", và "Xóa" được tạo động với các đường dẫn chứa ID của loại sản phẩm, cho phép người dùng thực hiện các thao tác này một cách trực quan và chính xác.

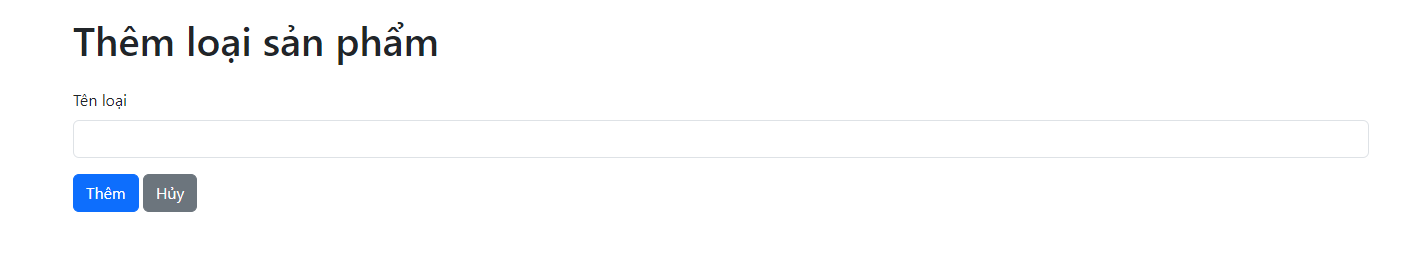
Kết quả đạt được là một giao diện "Danh sách loại sản phẩm" dễ sử dụng, cho phép người dùng quản lý các loại sản phẩm bằng cách xem chi tiết, chỉnh sửa hoặc xóa loại sản phẩm hiện có, và thêm loại sản phẩm mới vào hệ thống.

* Chức năng thêm loại sản phẩm mới:

Để làm được chức năng thêm loại sản phẩm mới, em sử dụng HTML để tạo cấu trúc trang thêm loại sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng Loai thông qua cú pháp th:field.

Trong controller LoaiController, em sử dụng phương thức addLoaiForm để hiển thị trang thêm loại sản phẩm, và phương thức addLoaiSubmit để xử lý yêu cầu POST khi người dùng gửi form. Em tái sử dụng dịch vụ LoaiService để lưu thông tin loại sản phẩm mới vào cơ sở dữ liệu bằng cách gọi phương thức saveLoai.

Kết quả đạt được là một chức năng thêm loại sản phẩm mới hiệu quả, cho phép người dùng dễ dàng nhập thông tin loại sản phẩm mới và lưu trữ vào hệ thống một cách trực quan và chính xác.



* Chức năng xem chi tiết loại sản phẩm:

Để làm được chức năng xem chi tiết loại sản phẩm, em sử dụng HTML để tạo cấu trúc trang hiển thị thông tin chi tiết của loại sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như container và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để hiển thị dữ liệu động, bao gồm việc liên kết các thuộc tính của đối tượng Loai với các phần tử HTML thông qua cú pháp th:text.

Trong controller LoaiController, em sử dụng phương thức getLoaiDetail để nhận yêu cầu từ người dùng, lấy thông tin chi tiết của loại sản phẩm từ cơ sở dữ liệu thông qua dịch vụ LoaiService, và thêm dữ liệu vào model để hiển thị trên giao diện.

Kết quả đạt được là một trang xem chi tiết loại sản phẩm dễ sử dụng, cho phép người dùng xem thông tin chi tiết của loại sản phẩm một cách trực quan và chính xác.

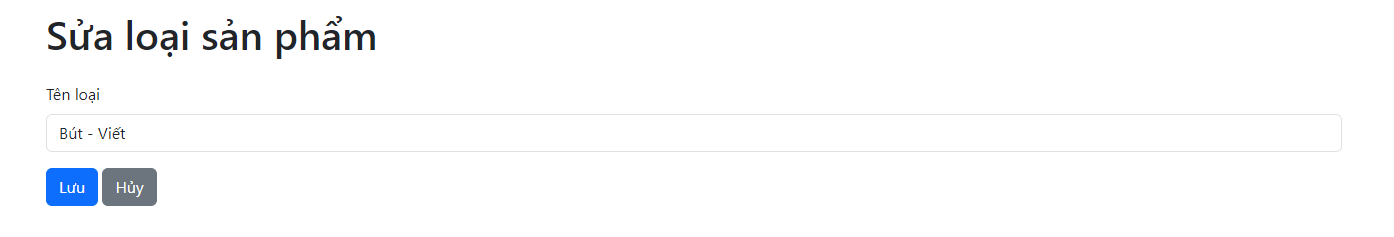


* Chức năng sửa loại sản phẩm:

Để làm được chức năng sửa loại sản phẩm, em sử dụng HTML để tạo cấu trúc trang chỉnh sửa thông tin loại sản phẩm. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng Loai thông qua cú pháp th:field.

Trong controller LoaiController, em sử dụng phương thức editLoaiForm để hiển thị trang chỉnh sửa thông tin loại sản phẩm, và phương thức editLoaiSubmit để xử lý yêu cầu POST khi người dùng gửi form. Em tái sử dụng dịch vụ LoaiService để lưu thông tin loại sản phẩm đã chỉnh sửa vào cơ sở dữ liệu bằng cách gọi phương thức saveLoai.

Kết quả đạt được là một chức năng sửa loại sản phẩm hiệu quả, cho phép người dùng dễ dàng chỉnh sửa thông tin loại sản phẩm và lưu trữ các thay đổi vào hệ thống một cách trực quan và chính xác.



Chức năng xóa loại sản phẩm:

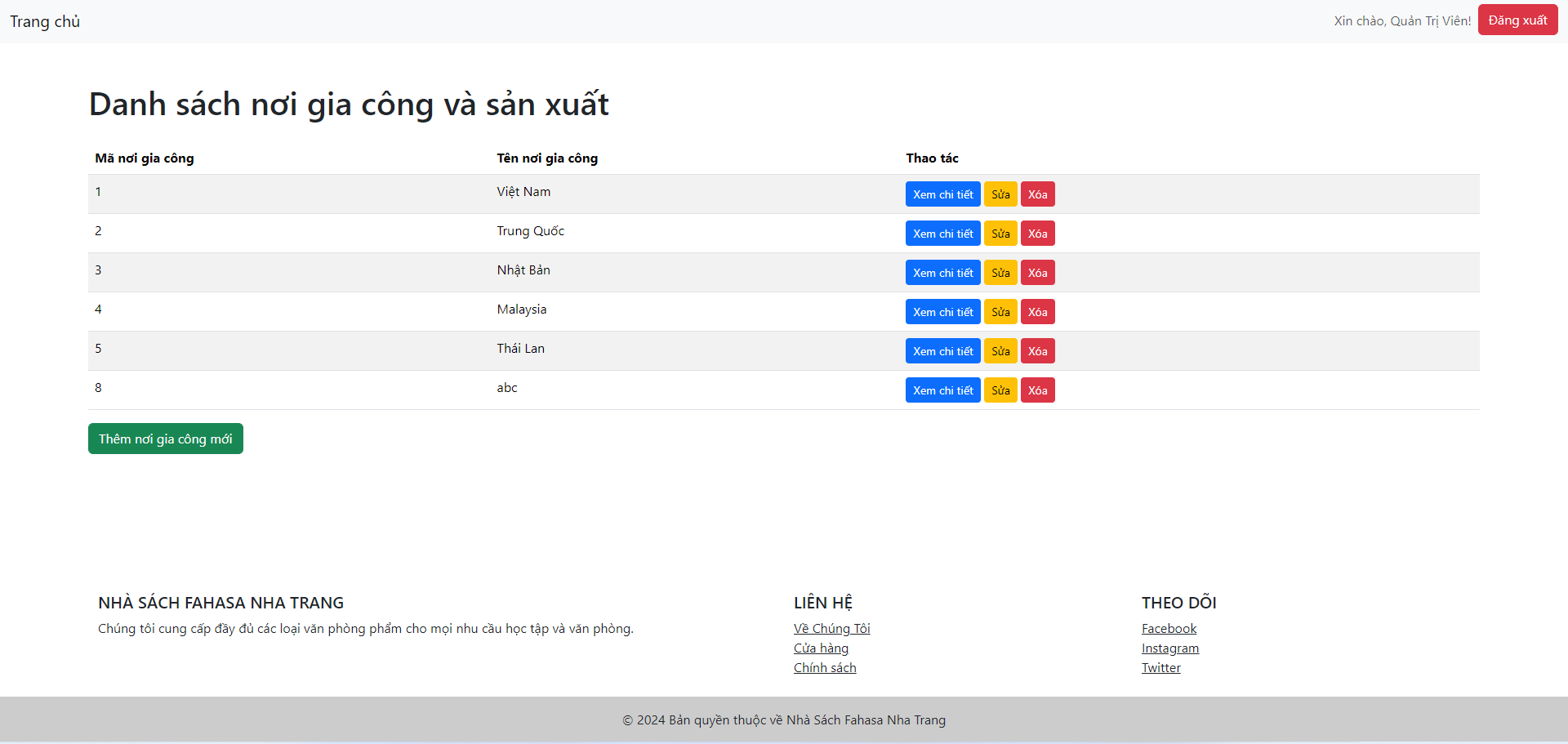
Để làm được chức năng xóa loại sản phẩm, em sử dụng Spring MVC trong LoaiController để nhận yêu cầu xóa loại sản phẩm từ người dùng thông qua đường dẫn có tham số ID của loại sản phẩm cần xóa. Em tích hợp Thymeleaf để hiển thị giao diện người dùng, bao gồm các nút xóa cho từng loại sản phẩm trong danh sách.

Khi người dùng yêu cầu xóa, controller sẽ gọi phương thức deleteLoaiById của LoaiService để thực hiện việc xóa loại sản phẩm khỏi cơ sở dữ liệu. Em tái sử dụng LoaiRepository để truy vấn và xóa thông tin loại sản phẩm. Trong LoaiServiceImpl, em triển khai các phương thức của LoaiService, bao gồm deleteLoaiById để xóa loại sản phẩm theo ID.

Kết quả đạt được là chức năng xóa loại sản phẩm hoạt động hiệu quả, cho phép người dùng dễ dàng xóa loại sản phẩm khỏi hệ thống thông qua giao diện đơn giản và trực quan. Dữ liệu loại sản phẩm được cập nhật kịp thời và chính xác trong cơ sở dữ liệu.

1. Danh mục Nơi gia công & Sản xuất:

Giao diện của danh mục Nơi gia công & Sản xuất:



Để làm được giao diện "Danh sách nơi gia công và sản xuất", em sử dụng HTML để tạo cấu trúc trang hiển thị danh sách các nơi gia công và sản xuất. Em dùng Bootstrap để tích hợp các thành phần giao diện như bảng, button và container, giúp tạo giao diện người dùng thân thiện và hiện đại. Em sử dụng CSS để tạo phong cách cho trang, bao gồm việc thiết lập chiều cao, khoảng cách và bố cục của trang.

Em tích hợp Thymeleaf để hiển thị dữ liệu động, bao gồm việc liên kết danh sách nơi gia công và sản xuất từ server với bảng thông qua cú pháp th:each. Các nút hành động như "Xem chi tiết", "Sửa", và "Xóa" được tạo động với các đường dẫn chứa ID của nơi gia công, cho phép người dùng thực hiện các thao tác này một cách trực quan và chính xác.

Trong controller NoiGiaCongVaSanXuatController, em sử dụng phương thức getAllNoiGiaCongVaSanXuat để nhận yêu cầu từ người dùng, lấy danh sách nơi gia công và sản xuất từ cơ sở dữ liệu thông qua dịch vụ NoiGiaCongVaSanXuatService, và thêm dữ liệu vào model để hiển thị trên giao diện.

Kết quả đạt được là một giao diện "Danh sách nơi gia công và sản xuất" dễ sử dụng, cho phép người dùng quản lý các nơi gia công và sản xuất bằng cách xem chi tiết, chỉnh sửa hoặc xóa nơi gia công hiện có, và thêm nơi gia công mới vào hệ thống.

* Chức năng thêm Nơi gia công & Sản xuất:

Để làm được chức năng thêm nơi gia công và sản xuất, em sử dụng HTML để tạo cấu trúc trang thêm nơi gia công và sản xuất. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng NoiGiaCongVaSanXuat thông qua cú pháp th:field.

Trong controller NoiGiaCongVaSanXuatController, em sử dụng phương thức addNoiGiaCongVaSanXuatForm để hiển thị trang thêm nơi gia công và sản xuất, và phương thức addNoiGiaCongVaSanXuatSubmit để xử lý yêu cầu POST khi người dùng gửi form. Em tái sử dụng dịch vụ NoiGiaCongVaSanXuatService để lưu thông tin nơi gia công và sản xuất mới vào cơ sở dữ liệu bằng cách gọi phương thức saveNoiGiaCongVaSanXuat.

Kết quả đạt được là một chức năng thêm nơi gia công và sản xuất hiệu quả, cho phép người dùng dễ dàng nhập thông tin nơi gia công và sản xuất mới và lưu trữ vào hệ thống một cách trực quan và chính xác.



* Chức năng xem chi tiết nơi gia công và sản xuất:

Để làm được chức năng xem chi tiết nơi gia công và sản xuất, em sử dụng HTML để tạo cấu trúc trang hiển thị thông tin chi tiết của nơi gia công và sản xuất. Em dùng Bootstrap để tích hợp các thành phần giao diện như container và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để hiển thị dữ liệu động, bao gồm việc liên kết các thuộc tính của đối tượng NoiGiaCongVaSanXuat với các phần tử HTML thông qua cú pháp th:text.

Trong controller NoiGiaCongVaSanXuatController, em sử dụng phương thức detailNoiGiaCongVaSanXuat để nhận yêu cầu từ người dùng, lấy thông tin chi tiết của nơi gia công và sản xuất từ cơ sở dữ liệu thông qua dịch vụ NoiGiaCongVaSanXuatService, và thêm dữ liệu vào model để hiển thị trên giao diện.

Kết quả đạt được là một trang xem chi tiết nơi gia công và sản xuất dễ sử dụng, cho phép người dùng xem thông tin chi tiết của nơi gia công và sản xuất một cách trực quan và chính xác.

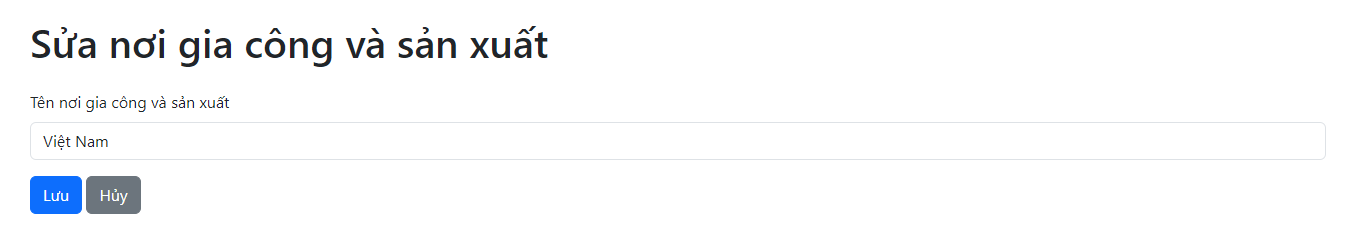


* Chức năng sửa nơi gia công và sản xuất:

Để làm được chức năng sửa nơi gia công và sản xuất, em sử dụng HTML để tạo cấu trúc trang sửa thông tin của nơi gia công và sản xuất. Em dùng Bootstrap để tích hợp các thành phần giao diện như form, input, và button, giúp tạo giao diện người dùng thân thiện và hiện đại. Em tích hợp Thymeleaf để xử lý dữ liệu động, bao gồm việc liên kết form với đối tượng NoiGiaCongVaSanXuat thông qua cú pháp th:field.

Trong controller NoiGiaCongVaSanXuatController, em sử dụng phương thức editNoiGiaCongVaSanXuatForm để hiển thị trang sửa thông tin nơi gia công và sản xuất, và phương thức editNoiGiaCongVaSanXuatSubmit để xử lý yêu cầu POST khi người dùng gửi form. Em tái sử dụng dịch vụ NoiGiaCongVaSanXuatService để lưu các thay đổi vào cơ sở dữ liệu bằng cách gọi phương thức saveNoiGiaCongVaSanXuat.

Kết quả đạt được là một chức năng sửa nơi gia công và sản xuất hiệu quả, cho phép người dùng dễ dàng cập nhật thông tin của nơi gia công và sản xuất hiện có và lưu trữ vào hệ thống một cách trực quan và chính xác.



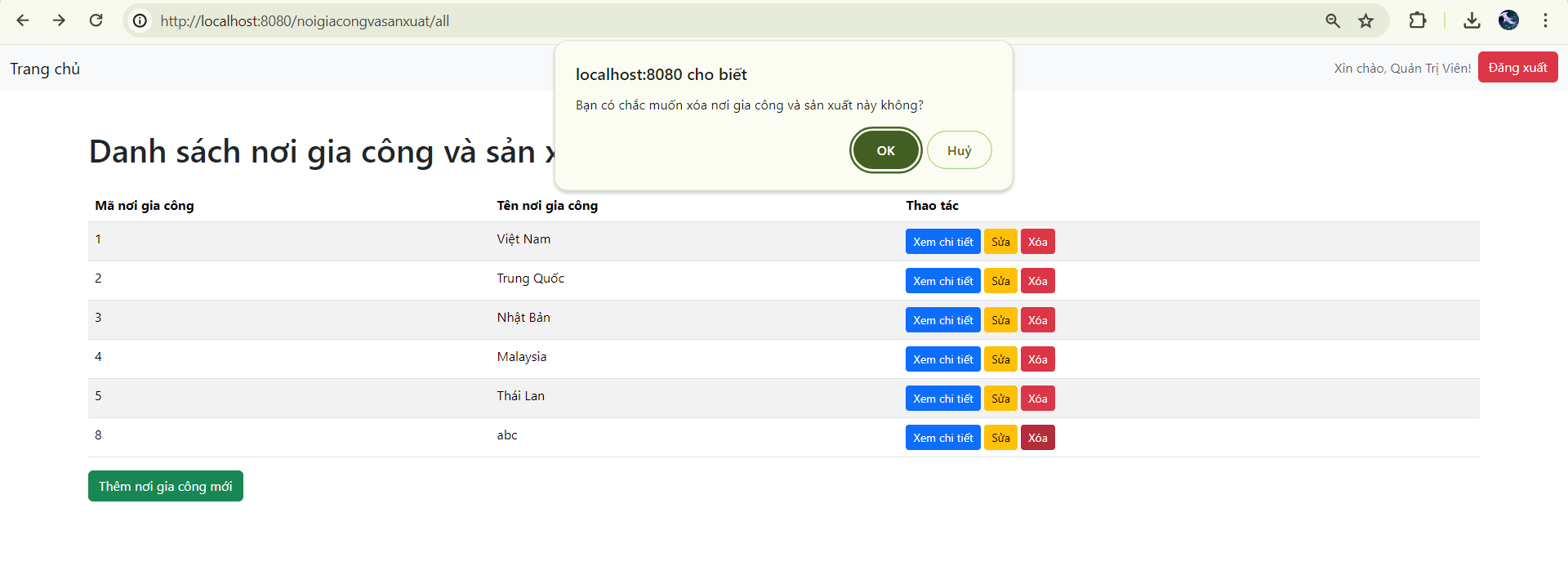
* Chức năng xóa nơi gia công & sản xuất:

Để làm được chức năng xóa nơi gia công và sản xuất, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu xóa. Em sử dụng phương thức deleteNoiGiaCongVaSanXuat trong controller NoiGiaCongVaSanXuatController để nhận yêu cầu từ người dùng. Phương thức này lấy thông tin id của nơi gia công và sản xuất cần xóa từ đường dẫn URL và gọi dịch vụ NoiGiaCongVaSanXuatService để thực hiện việc xóa đối tượng này khỏi cơ sở dữ liệu.

Trong lớp dịch vụ NoiGiaCongVaSanXuatService, em tích hợp phương thức deleteNoiGiaCongVaSanXuatById để thực hiện việc xóa nơi gia công và sản xuất bằng cách gọi phương thức tương ứng từ repository NoiGiaCongVaSanXuatRepository.

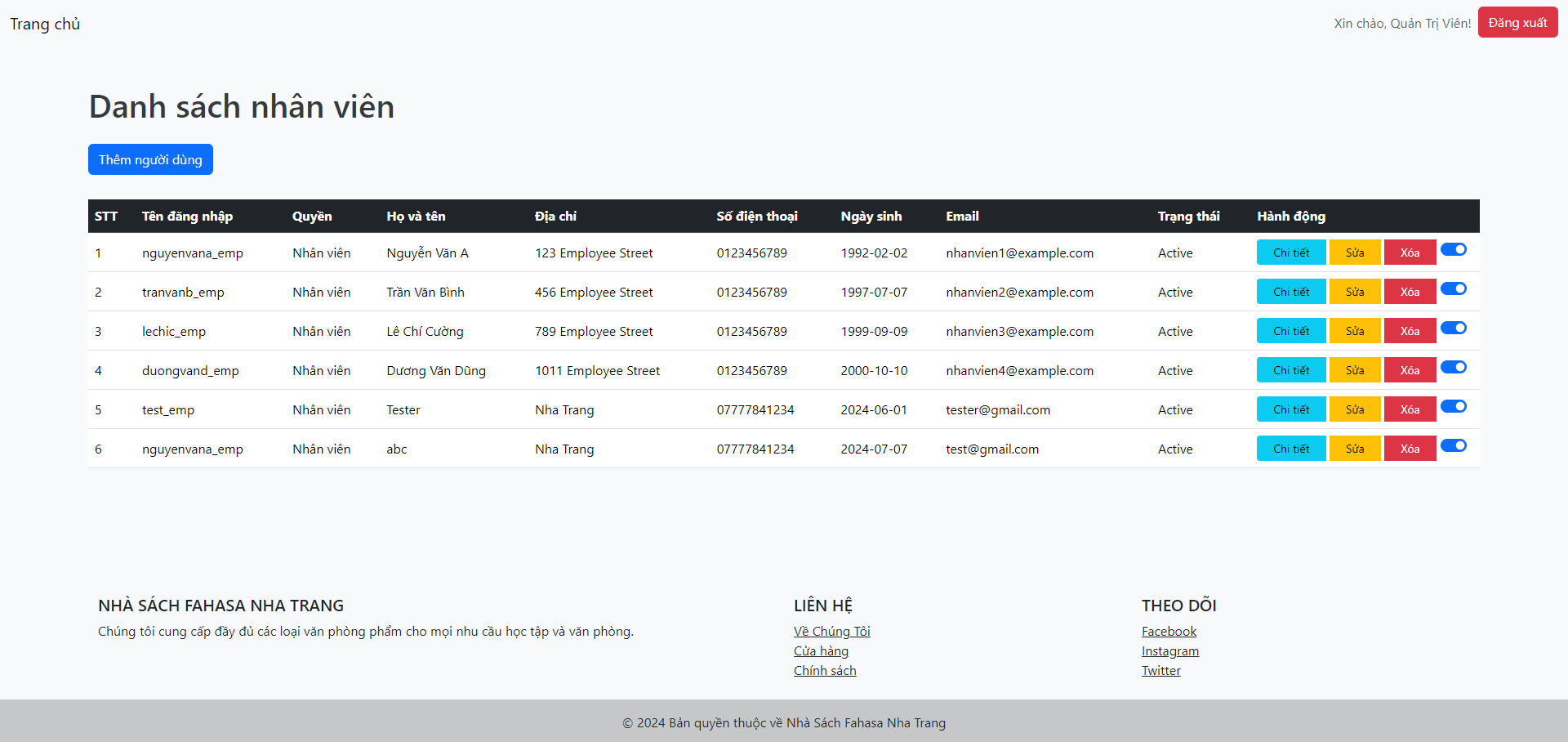
Em tích hợp Thymeleaf và Bootstrap trong trang HTML để tạo giao diện người dùng. Trên giao diện danh sách nơi gia công và sản xuất, em thêm các liên kết hoặc nút xóa, mỗi nút liên kết tới phương thức xóa của controller thông qua cú pháp th:href.

Kết quả đạt được là một chức năng xóa nơi gia công và sản xuất hiệu quả, cho phép người dùng dễ dàng xóa thông tin nơi gia công và sản xuất không cần thiết khỏi hệ thống một cách trực quan và chính xác.



1. Danh mục Người dùng (Quản lý Nhân Viên):

Giao diện của danh mục Người dùng (Danh sách Nhân Viên):



Để làm được chức năng hiển thị danh sách người dùng, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu lấy danh sách người dùng từ cơ sở dữ liệu. Em sử dụng phương thức getAllNguoiDung trong controller NguoiDungController để lấy danh sách người dùng từ dịch vụ NguoiDungService. Danh sách này được lọc để chỉ hiển thị các tài khoản không phải là "Admin" – tức “Nhân Viên”. Sau đó, danh sách này được đưa vào model để hiển thị trên trang.

Trong lớp dịch vụ NguoiDungService, em tích hợp phương thức getAllNguoiDungs để lấy tất cả người dùng từ repository NguoiDungRepository.

Em sử dụng Thymeleaf và Bootstrap để tạo giao diện người dùng. Trên trang HTML, em thiết kế giao diện để hiển thị danh sách người dùng dưới dạng bảng. Mỗi hàng trong bảng hiển thị thông tin chi tiết về một người dùng như tên đăng nhập, quyền, họ và tên, địa chỉ, số điện thoại, ngày sinh, email và trạng thái. Em tích hợp các nút hành động như "Chi tiết", "Sửa", "Xóa" và một nút chuyển đổi trạng thái.

Kết quả đạt được là một giao diện người dùng trực quan, dễ sử dụng cho phép quản trị viên có thể dễ dàng xem, chỉnh sửa, xóa hoặc khóa/mở khóa tài khoản người dùng.

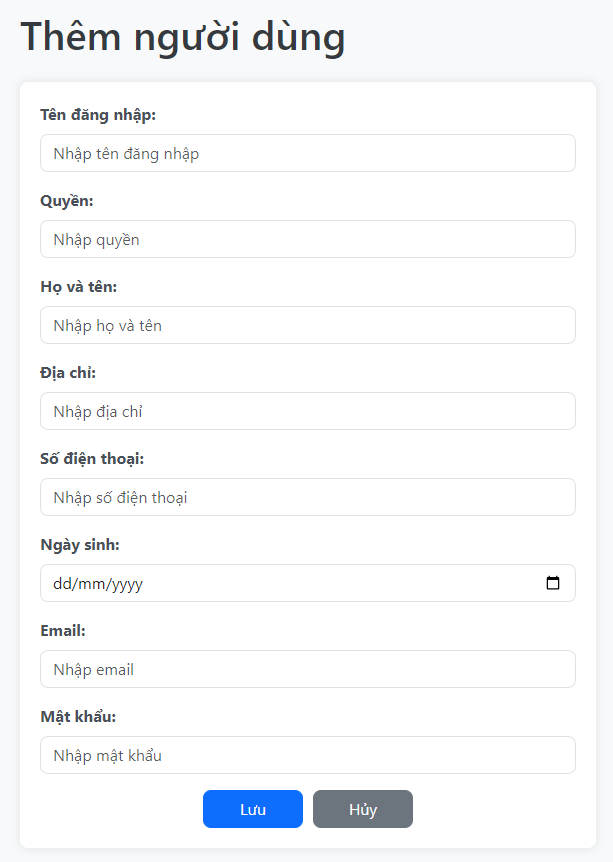
* Chức năng thêm người dùng (thêm nhân viên):

Để làm được chức năng thêm người dùng mới, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu thêm người dùng từ giao diện. Em sử dụng phương thức themNguoiDungForm trong controller NguoiDungController để trả về trang thêm người dùng. Trên trang này, em sử dụng Thymeleaf để hiển thị một form cho người dùng nhập thông tin.

Form này bao gồm các trường nhập liệu như tên đăng nhập, quyền, họ và tên, địa chỉ, số điện thoại, ngày sinh, email và mật khẩu. Khi người dùng điền xong thông tin và nhấn nút "Lưu", form sẽ gửi dữ liệu đến phương thức themNguoiDung trong NguoiDungController để xử lý việc lưu người dùng mới.

Em tích hợp phương thức saveNguoiDung trong NguoiDungService để lưu thông tin người dùng mới vào cơ sở dữ liệu thông qua repository NguoiDungRepository.

Kết quả đạt được là một người dùng mới được thêm vào hệ thống và hiển thị trên trang danh sách người dùng.



* Chức năng xem chi tiết người dùng:

Để làm được chức năng xem chi tiết người dùng, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu xem chi tiết người dùng từ giao diện. Em sử dụng phương thức chiTietNguoiDung trong controller NguoiDungController để lấy thông tin chi tiết của người dùng từ cơ sở dữ liệu.

Trên trang chi tiết, em sử dụng Thymeleaf để hiển thị thông tin chi tiết của người dùng. Giao diện bao gồm các thông tin như tên đăng nhập, quyền, họ và tên, địa chỉ, số điện thoại, ngày sinh và email. Các thông tin này được hiển thị dưới dạng các thẻ HTML và được định dạng bởi các lớp CSS của Bootstrap để tạo giao diện đẹp mắt và dễ nhìn.

Khi người dùng truy cập vào trang chi tiết người dùng, controller sẽ lấy thông tin từ NguoiDungService, phương thức này gọi đến repository NguoiDungRepository để truy vấn cơ sở dữ liệu và lấy thông tin người dùng theo ID. Sau đó, dữ liệu này được đưa vào model và trả về trang Thymeleaf để hiển thị.

Em tích hợp các thành phần như repository và service layer để đảm bảo việc lấy dữ liệu và hiển thị thông tin người dùng được thực hiện một cách hiệu quả.

Khi người dùng chọn xem chi tiết một người dùng nào đó, trang chi tiết sẽ hiển thị đầy đủ thông tin của người dùng đó một cách rõ ràng và dễ hiểu.



* Chức năng chỉnh sửa thông tin người dùng:

Để làm được chức năng sửa thông tin người dùng, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu sửa thông tin người dùng từ giao diện. Em sử dụng phương thức suaNguoiDungForm trong controller NguoiDungController để trả về trang sửa thông tin người dùng. Trên trang này, em sử dụng Thymeleaf để hiển thị một form cho người dùng nhập thông tin cần sửa.

Form này bao gồm các trường nhập liệu như tên đăng nhập, quyền, họ và tên, địa chỉ, số điện thoại, ngày sinh, email và mật khẩu. Khi người dùng điền xong thông tin và nhấn nút "Lưu", form sẽ gửi dữ liệu đến phương thức suaNguoiDung trong NguoiDungController để xử lý việc cập nhật thông tin người dùng.

Trong controller, em sử dụng phương thức getNguoiDungById của NguoiDungService để lấy thông tin người dùng hiện tại từ cơ sở dữ liệu thông qua repository NguoiDungRepository. Sau đó, em cập nhật các thông tin mới vào đối tượng người dùng và sử dụng phương thức saveNguoiDung để lưu lại thông tin đã được cập nhật vào cơ sở dữ liệu.

Người dùng có thể sửa đổi thông tin cá nhân một cách dễ dàng thông qua giao diện web.

Thông tin người dùng sau khi sửa đổi được lưu lại trong cơ sở dữ liệu và hiển thị trên trang danh sách người dùng.



* Chức năng khóa trạng thái tài khoản của người dùng:

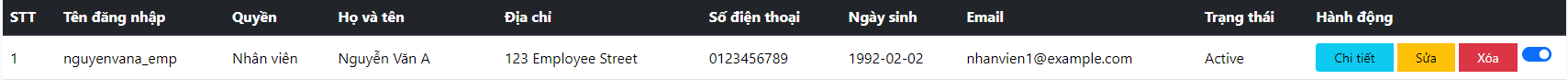
Để làm được chức năng khóa trạng thái của người dùng, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu khóa/mở khóa người dùng từ giao diện. Em sử dụng phương thức toggleNguoiDung trong controller NguoiDungController để xử lý việc thay đổi trạng thái của người dùng.

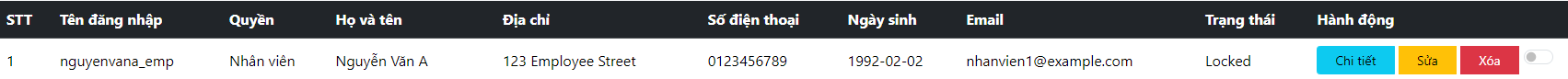
Trong controller, em lấy thông tin người dùng dựa trên userId thông qua phương thức getNguoiDungById của NguoiDungService. Nếu tìm thấy người dùng, em thay đổi trạng thái trangThai của người dùng và lưu lại thông tin đã thay đổi vào cơ sở dữ liệu thông qua phương thức saveNguoiDung.

Trên giao diện, em sử dụng một checkbox trong form để hiển thị và thay đổi trạng thái của người dùng. Khi người dùng thay đổi trạng thái của checkbox, form sẽ tự động gửi yêu cầu đến server để cập nhật trạng thái của người dùng.

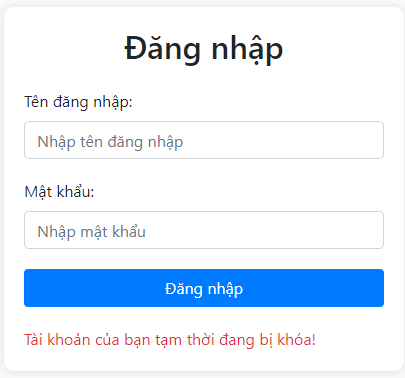
Quản trị viên có thể dễ dàng khóa hoặc mở khóa tài khoản người dùng thông qua giao diện web.

Trạng thái của người dùng được cập nhật ngay lập tức trong cơ sở dữ liệu và hiển thị đúng trên trang danh sách người dùng.





Người dùng không thể đăng nhập vào tài khoản khi đang bị khóa:



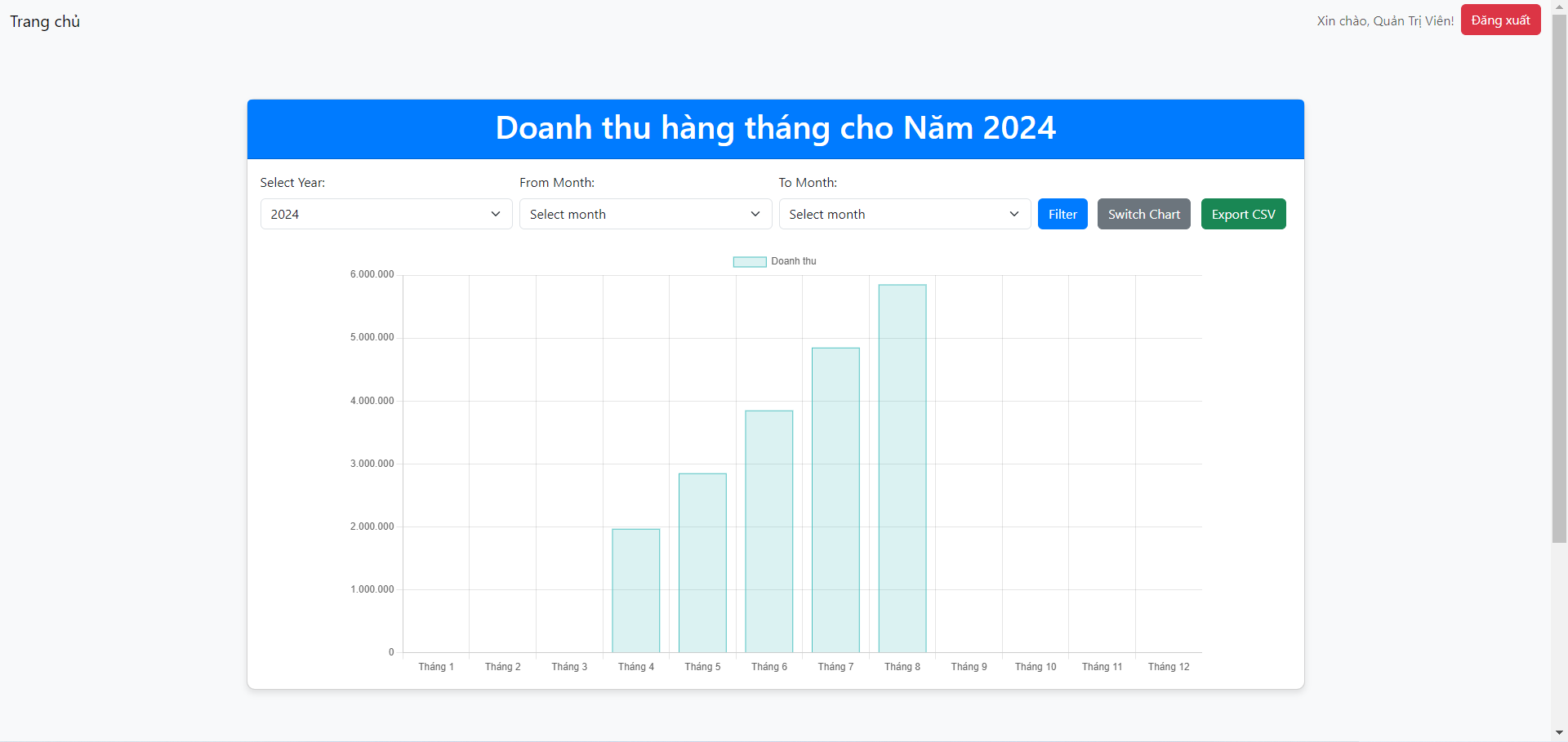
* Chức năng xóa người dùng:

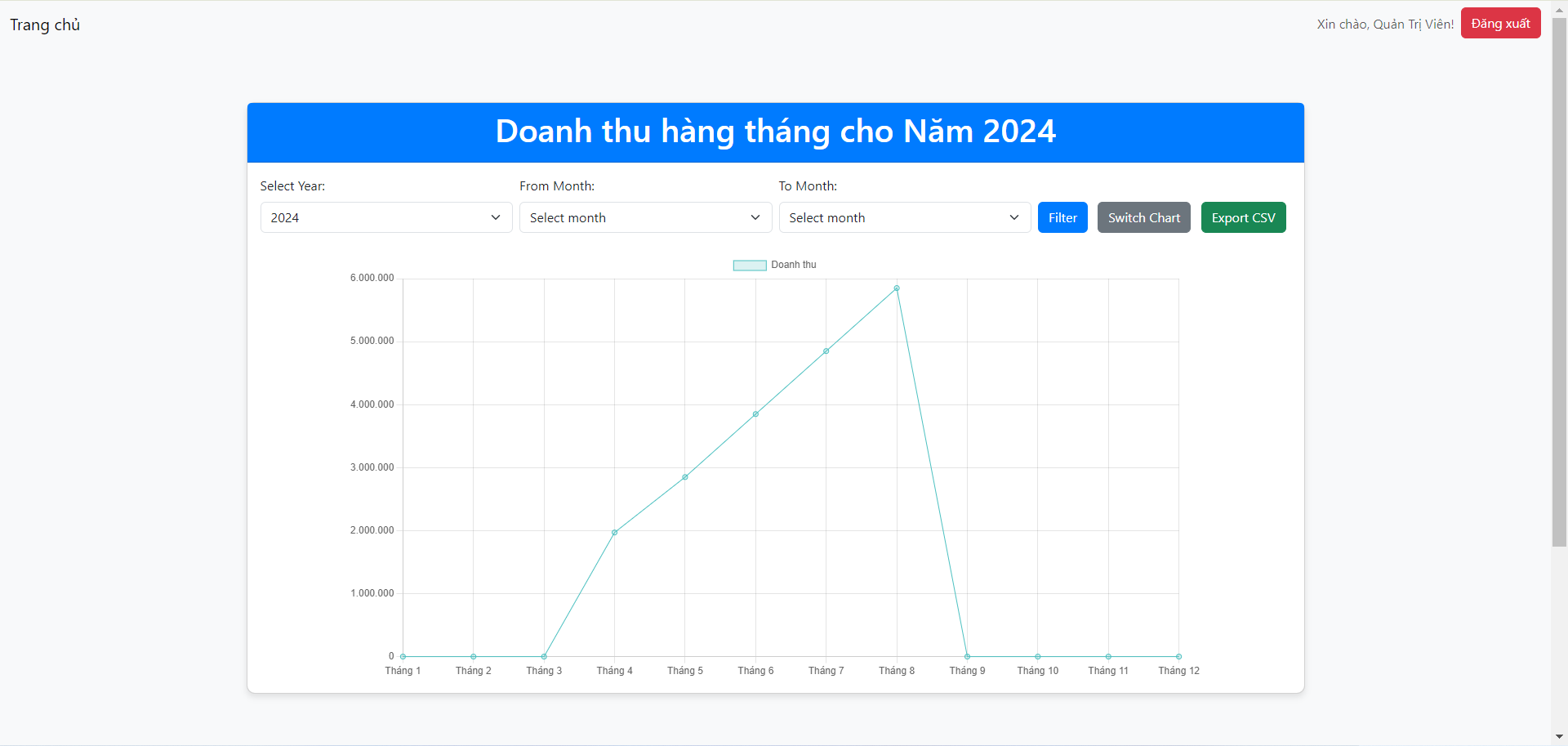
Để làm được chức năng xóa người dùng, em sử dụng Java Spring Boot để tạo controller xử lý yêu cầu xóa người dùng từ giao diện. Em sử dụng phương thức xoaNguoiDung trong controller NguoiDungController để xử lý việc xóa người dùng. Em tích hợp phương thức deleteNguoiDungById trong NguoiDungService để xóa người dùng khỏi cơ sở dữ liệu thông qua repository NguoiDungRepository.

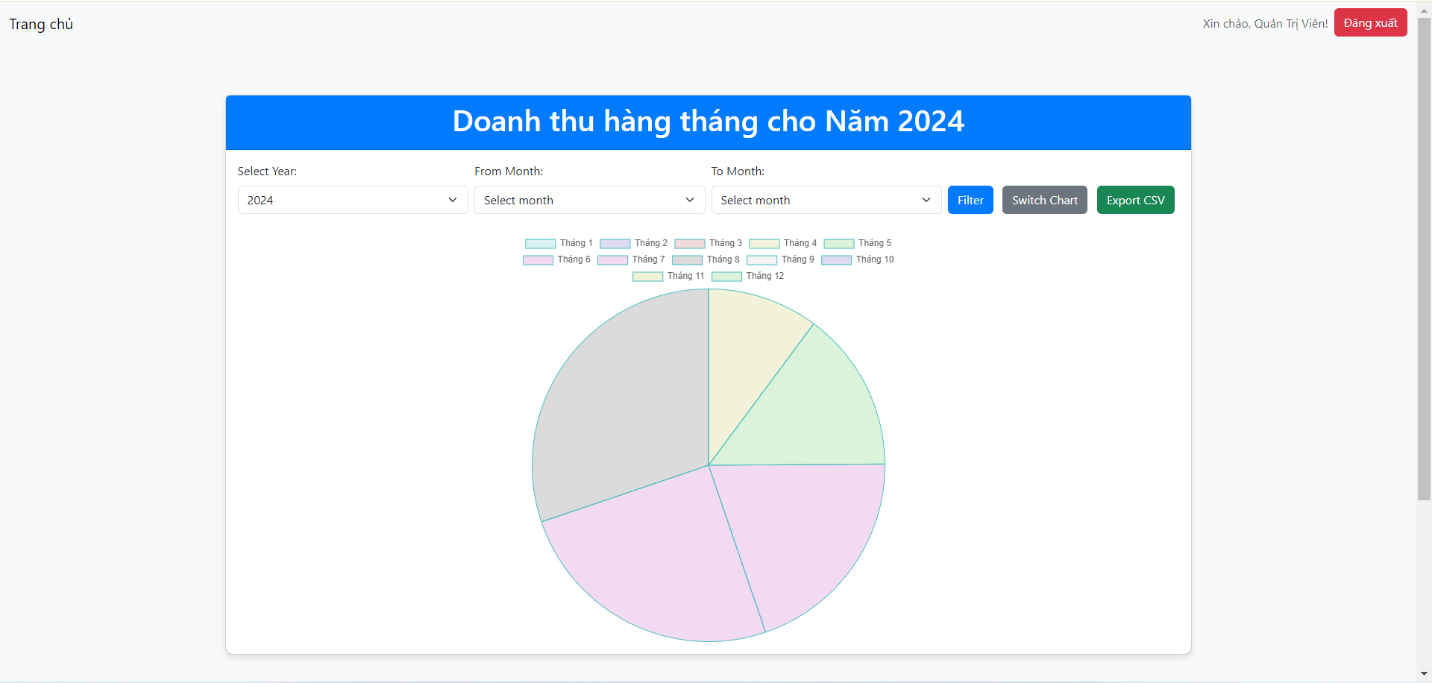
Kết quả đạt được là người dùng được xóa khỏi hệ thống và không còn xuất hiện trong danh sách người dùng.

1. Danh mục Thống kê doanh thu:

Giao diện của danh mục Thống kê doanh thu:







Để làm được việc này, em sử dụng Spring Framework để xây dựng backend và Thymeleaf để xây dựng giao diện người dùng.

* Controller: Em tạo một lớp HoaDonController để xử lý các request từ người dùng. Sử dụng annotation @GetMapping để định nghĩa hai endpoint: một để lấy tất cả hóa đơn và một để lọc doanh thu hàng tháng theo năm và khoảng tháng.
* Service: Em tạo interface HoaDonService và triển khai nó trong lớp HoaDonServiceImpl. Lớp này giao tiếp với HoaDonRepository để thực hiện các truy vấn dữ liệu từ cơ sở dữ liệu.
* Repository: Em sử dụng Spring Data JPA để tạo interface HoaDonRepository, trong đó có các phương thức tùy chỉnh với annotation @Query để truy vấn doanh thu hàng tháng và các năm khác nhau.
* Frontend: Em sử dụng Thymeleaf để xây dựng giao diện cho trang thống kê. Giao diện bao gồm một form để người dùng chọn năm và khoảng tháng, sau đó hiển thị biểu đồ doanh thu hàng tháng bằng Chart.js. Bootstrap được tích hợp để tạo giao diện người dùng thân thiện và responsive.
* Kết quả: Kết quả đạt được là một trang web hiển thị biểu đồ doanh thu hàng tháng theo năm và khoảng tháng mà người dùng chọn. Người dùng có thể chuyển đổi giữa các loại biểu đồ (bar, line, pie) và xuất dữ liệu ra file CSV. Giao diện được thiết kế đẹp mắt và dễ sử dụng nhờ sự hỗ trợ của Bootstrap và Chart.js.

**Các danh mục trong trang Nhân viên:**

* Tương tự như các danh mục trong trang Admin (ngoại trừ danh mục Người dùng và Thống kê doanh thu)