# MỤC LỤC

[TÓM TẮT v](#_Toc167657226)

[MỤC LỤC vi](#_Toc167657227)

[CHƯƠNG 1 – TÌM HIỂU VỀ WEB DEVELOPMENT FRAMEWORK 1](#_Toc167657228)

[1.1 Khái niệm 1](#_Toc167657229)

[1.2 Ví dụ 1](#_Toc167657230)

[1.3 Ưu điểm 2](#_Toc167657231)

[1.4 Nhược điểm 2](#_Toc167657232)

[CHƯƠNG 2 – DJANGO 3](#_Toc167657233)

[2.1 Giới thiệu về Django 3](#_Toc167657234)

[2.2 Khái niệm 3](#_Toc167657235)

[2.3 Mục đích 3](#_Toc167657236)

[2.4 Ưu điểm 5](#_Toc167657237)

[2.5 Nhược điểm 6](#_Toc167657238)

[2.6 Tính năng 7](#_Toc167657239)

[CHƯƠNG 3 – KIẾN TRÚC MVT TRONG DJANGO 8](#_Toc167657240)

[3.1 Khái niệm 8](#_Toc167657241)

[3.2 Ví dụ 8](#_Toc167657242)

[CHƯƠNG 4 – MỘT VÀI METHOD CƠ BẢN 10](#_Toc167657243)

[4.1 Method \_\_init\_\_: 10](#_Toc167657244)

[4.2 Method \_\_str\_\_: 10](#_Toc167657245)

[4.3 Method \_\_len\_\_: 11](#_Toc167657246)

[4.4 Method \_\_getitem\_\_: 11](#_Toc167657247)

[4.5 Method \_\_setitem\_\_: 12](#_Toc167657248)

[4.6 Method \_\_del\_\_: 12](#_Toc167657249)

[4.7 Method \_\_call\_\_: 13](#_Toc167657250)

[4.8 Method \_\_contains\_\_: 13](#_Toc167657251)

[4.9 Method \_\_eq\_\_: 14](#_Toc167657252)

[4.10 Method models.Model 14](#_Toc167657253)

[4.11 Method CSKF trong django 15](#_Toc167657254)

[4.12 Method Path trong Django 15](#_Toc167657255)

[CHƯƠNG 5 – TIẾN ĐỘ WED 16](#_Toc167657256)

[5.1 Trang Login 16](#_Toc167657257)

[5.1.1 Xử lý code chính 16](#_Toc167657258)

[5.2 Trang chủ 17](#_Toc167657259)

[5.3 Thêm sản phẩm 17](#_Toc167657260)

[5.3.1 Xử lí code chính 19](#_Toc167657261)

[5.4 Xem sản phẩm 20](#_Toc167657262)

[5.4.1 Xử lí code chính 20](#_Toc167657263)

[5.5 Delete sản phẩm 20](#_Toc167657264)

[5.5.1 Xử lí code chính 21](#_Toc167657265)

[5.6 Đăng kí tài khoản 21](#_Toc167657266)

[5.6.1 Xử lí code chính 23](#_Toc167657267)

[5.7 Chức năng đổi mật khẩu và Username 24](#_Toc167657268)

[5.7.1 Xử lí code chính 26](#_Toc167657269)

[5.8 Logout 27](#_Toc167657270)

[5.8.1 Xử lí code chính 28](#_Toc167657271)

[5.9 Tạo giao dịch 28](#_Toc167657272)

[5.9.1 Xử lí code chính 33](#_Toc167657273)

[CHƯƠNG 6 – ERD 36](#_Toc167657274)

[TÀI LIỆU THAM KHẢO 38](#_Toc167657275)

# CHƯƠNG 1 – TÌM HIỂU VỀ WEB DEVELOPMENT FRAMEWORK

## 1.1 Khái niệm

- Web development framework là một bộ công cụ và thư viện phần mềm được sử dụng để xây dựng và phát triển các ứng dụng web. Framework cung cấp một cách tiếp cận cấu trúc hóa và tổ chức cho việc phát triển ứng dụng web, giúp tăng tốc độ phát triển, giảm thiểu mã lặp lại và quản lý mã nguồn dễ dàng hơn.

## 1.2 Ví dụ

- Một số ví dụ về các web development framework phổ biến:

* ***Django***: Django là một framework phát triển web mạnh mẽ và linh hoạt được viết bằng ngôn ngữ lập trình Python. Nó cung cấp một hệ thống ORM mạnh mẽ, hỗ trợ tích hợp sẵn cho các tính năng bảo mật và giao diện quản trị tự động, và có một cộng đồng lớn và phong phú.
* ***Ruby on Rails***: Rails là một framework phát triển web được viết bằng ngôn ngữ lập trình Ruby. Nó tuân thủ theo triết lý "Convention over Configuration" và "Don't Repeat Yourself" (DRY), giúp tối ưu hóa quá trình phát triển và giảm thiểu mã lặp lại.
* ***Express.js****:* Express.js là một framework phát triển web Node.js mạnh mẽ, dễ sử dụng và linh hoạt. Nó cung cấp một cách tiếp cận minimalist và không cấu trúc, cho phép nhà phát triển xây dựng các ứng dụng web và API một cách nhanh chóng.
* ***ASP.NET MVC****:* ASP.NET MVC là một framework phát triển web được phát triển bởi Microsoft và sử dụng ngôn ngữ lập trình C#. Nó cung cấp một cách tiếp cận mô hình MVC (Model-View-Controller) để phát triển các ứng dụng web đa nền tảng và hiệu quả.
* ***Spring Framework****:* Spring là một framework phát triển ứng dụng Java mạnh mẽ và linh hoạt. Nó cung cấp một loạt các module và công cụ để phát triển ứng dụng web, bao gồm Spring MVC cho mô hình MVC và Spring Boot cho phát triển ứng dụng độc lập tự động.
  1. **Ưu điểm**
* ***Hỗ trợ đa nền tảng:*** Các Framework được thiết kế để có thể chạy trên được nhiều nền tảng nên có thể dễ dàng để sử dụng và phát triển.
* ***Độ bảo mật và tin cậy cao:*** Framework được phát triển và thử nghiệm nhiều bởi các developer nên rủi ro bảo mật thường được kiểm tra và xử lý thường xuyên.
* ***Hạn chế trùng lặp code:*** hạn chế lặp đi lặp lại code trong nhiều ứng dụng giúp tang hiệu suất công việc.
* ***Tiết kiệm thời gian và công sức:*** Framework được phát triển và tích hợp sẵn các mã cơ bản.
  1. **Nhược điểm**
* ***Đòi hỏi kiến thức và kĩ thuật cao:*** Để có thể dùng thành thạo Framwork thì người dùng cần phải bỏ ra nhiều thời gian và công sức luyện tập nó
* ***Bị giới hạn bởi tính linh hoạt và tùy chỉnh:*** Nếu các tính năng không phù hợp với tính năng có sẵn của Framework sẽ gây ra khó khăn khi phát triển ứng dụng
* ***Tốc độ hoạt động của ứng dụng bị chậm:*** Do các Framwork thường sử dụng nhiều lớp trừu tượng nên sẽ làm chậm tốc độ khi xử lý các ứng dụng lớn.

# CHƯƠNG 2 – DJANGO

## 2.1 Giới thiệu về Django

- Django là 1 web framework khá nổi tiếng được viết hoàn toàn bằng ngôn ngữ Python. Nó là 1 framework với đầu đủ các thư viện, module hỗ trợ các web-developer. Django sử dụng mô hình MVC và được phát triển bởi Django Software Foundation (DSF một tổ chức phi lợi nhuận độc lập) Mục tiêu chính của Django là đơn giản hóa việc tạo các website phức tạp có sử dụng cơ sở dữ liệu. Django tập trung vào tính năng “có thể tái sử dụng” và “có thể tự chạy” của các component, tính năng phát triển nhanh, không làm lại những gì đã làm. Một số website phổ biến được xây dựng từ Django là Pinterest, Instagram, Mozilla, và Bitbucket.

## 2.2 Khái niệm

- Django là một framework phát triển web mạnh mẽ được viết bằng Python. Được phát triển bởi Django Software Foundation, Django cung cấp một cách nhanh chóng và linh hoạt để xây dựng các ứng dụng web, từ các trang web nhỏ đến các ứng dụng web lớn và phức tạp.

- Django được thiết kế theo mô hình "MTV" (Model-Template-View).

* Model: Định nghĩa cấu trúc cơ sở dữ liệu.
* Template: Xác định giao diện người dùng.
* View: Xử lý logic ứng dụng và tương tác với dữ liệu.

## 2.3 Mục đích

- Mục đích chính của Django là cung cấp một framework phát triển web mạnh mẽ, linh hoạt và hiệu quả cho các nhà phát triển. Dưới đây là các mục đích cụ thể của Django:

* ***Tính linh hoạt và tùy chỉnh:*** Django cung cấp một cấu trúc linh hoạt và dễ tùy chỉnh, cho phép nhà phát triển điều chỉnh các thành phần của framework để phù hợp với nhu cầu cụ thể của dự án mà không cần phải bỏ qua sự tiện ích và tính chắc chắn của Django.
* ***Hỗ trợ đa dạng cơ sở dữ liệu*:** Django hỗ trợ nhiều loại cơ sở dữ liệu phổ biến như MySQL, PostgreSQL, SQLite, và Oracle, cho phép lựa chọn cơ sở dữ liệu phù hợp với yêu cầu của dự án.
* ***Quản lý URL hiệu quả*:** Django cung cấp một hệ thống quản lý URL mạnh mẽ, giúp tổ chức và quản lý các URL của ứng dụng một cách hiệu quả, giúp dễ dàng theo dõi và bảo trì hệ thống routing của ứng dụng.
* ***Hỗ trợ API RESTful*:** Django Rest Framework cung cấp một cách tiện lợi để phát triển và quản lý các API RESTful, giúp tạo ra các ứng dụng web dựa trên API một cách hiệu quả và linh hoạt.
* ***Tính khả dụng và mở rộng*:** Django có tính khả dụng cao và dễ mở rộng, cho phép phát triển và triển khai các ứng dụng web với quy mô từ nhỏ đến lớn mà không gặp phải các vấn đề về hiệu suất và tính mở rộng.
* ***Tích hợp dễ dàng với các dịch vụ bên ngoài*:** Django cho phép tích hợp dễ dàng với các dịch vụ và công nghệ bên ngoài như AWS, Google Cloud, và các dịch vụ khác, giúp mở rộng và tăng cường khả năng của ứng dụng.
* ***Hỗ trợ phát triển ứng dụng theo mô hình MVC (Model-View-Controller):*** Django sử dụng mô hình MTV (Model-Template-View) tương tự với mô hình MVC, giúp phân tách logic dữ liệu, giao diện người dùng và xử lý yêu cầu. Điều này giúp tổ chức mã nguồn ứng dụng một cách logic và dễ quản lý.
* ***Dễ bảo trì****:* Cấu trúc rõ ràng và chuẩn mực của Django giúp dễ dàng bảo trì và mở rộng ứng dụng.
* ***Bảo mật:*** Django đi kèm với nhiều tính năng bảo mật tích hợp, giúp người phát triển giảm thiểu các lỗ hổng bảo mật phổ biến.

## 2.4 Ưu điểm

* ***Tích hợp đầy đủ:*** Django đi kèm với nhiều tính năng tích hợp sẵn như quản lý người dùng, quản lý phiên, quản lý admin, và nhiều tính năng khác, giúp giảm thiểu việc phải tìm kiếm và tích hợp các thư viện bên ngoài.
* ***Tính dễ mở rộng*:** Django cho phép bạn mở rộng ứng dụng một cách dễ dàng thông qua việc sử dụng ứng dụng con, gói ứng dụng, và các tiện ích bên ngoài.
* ***Hệ sinh thái mạnh mẽ***: Django có một cộng đồng lớn và mạnh mẽ, với nhiều tài nguyên học tập, tài liệu và các gói mở rộng sẵn có, giúp giảm thiểu thời gian tìm kiếm giải pháp cho các vấn đề cụ thể.
* ***Tích hợp tốt với Python***: Django được viết bằng Python và hoàn toàn tương thích với ngôn ngữ này. Điều này giúp cho các nhà phát triển Python tận dụng được kiến thức và kinh nghiệm của mình mà không cần phải học một ngôn ngữ mới.
* ***Tính linh hoạt****:* Mặc dù Django có cấu trúc mô hình cụ thể, nhưng nó vẫn linh hoạt đủ để cho phép bạn tạo ra các ứng dụng web có tính động cao và tương thích với nhiều loại dự án khác nhau.
* ***Hiệu suất cao***: Django được tối ưu hóa để cung cấp hiệu suất cao, từ việc cache dữ liệu cho đến việc xử lý yêu cầu của người dùng.
* ***Tài liệu phong phú****:* Mặc dù đã từng được đề cập là không phong phú, nhưng Django vẫn có một cộng đồng lớn và đa dạng, với nhiều tài liệu, hướng dẫn và ví dụ phong phú để bạn tham khảo.
* ***Tiêu chuẩn hoá và tuân thủ:*** Django tuân thủ các tiêu chuẩn và quy tắc phát triển chung, giúp đảm bảo tính ổn định và bảo mật của ứng dụng
* ***Tính dễ bảo trì và nâng cấp*:** Cấu trúc rõ ràng và chuẩn mực của Django, cùng với tính năng tự động sinh mã, giúp dễ dàng bảo trì và nâng cấp ứng dụng một cách hiệu quả.

## 2.5 Nhược điểm

* ***Học phức tạp ban đầu***: Django có một độ dốc học ban đầu khá cao, đặc biệt đối với những người mới bắt đầu với framework này hoặc người không quen với Python. Việc hiểu và áp dụng đúng các khái niệm như MTV (Model-Template-View) và các thành phần khác có thể mất một thời gian.
* ***Giới hạn về linh hoạt*:** Mặc dù Django cung cấp nhiều tính năng mạnh mẽ, nhưng đôi khi việc tuân thủ theo cấu trúc của nó có thể hạn chế tính linh hoạt của ứng dụng. Điều này có thể dẫn đến việc gặp khó khăn khi phát triển các tính năng phức tạp hoặc đòi hỏi tùy chỉnh cao.
* ***Tính tương thích giữa các phiên bản****:* Mỗi khi có phiên bản mới của Django được phát hành, việc nâng cấp từ phiên bản cũ có thể gây ra các vấn đề tương thích với mã nguồn hiện có. Điều này đặc biệt đúng đối với các dự án lớn hoặc dự án đã hoạt động từ lâu.
* ***Hiệu suất không tốt trong các ứng dụng có lưu lượng lớn*:** Mặc dù Django có thể xử lý lưu lượng truy cập lớn, nhưng nó không phải lúc nào cũng là lựa chọn tốt nhất cho các ứng dụng yêu cầu hiệu suất cực cao. Các framework như FastAPI có thể phù hợp hơn cho những tác vụ đòi hỏi xử lý nhanh và lưu lượng lớn.
* ***Khả năng mở rộng đối với các ứng dụng phức tạp***: Trong các ứng dụng lớn và phức tạp, việc mở rộng và bảo trì có thể trở nên khó khăn hơn do sự phụ thuộc vào các thành phần cốt lõi của Django. Điều này có thể yêu cầu các giải pháp tùy chỉnh hoặc sự chia tách thành các dịch vụ độc lập.
* ***Tài liệu không đồng nhất:*** Mặc dù Django có một cộng đồng lớn, nhưng tài liệu thường không đồng nhất và không phải lúc nào cũng cập nhật đầy đủ. Điều này có thể gây khó khăn cho những người mới bắt đầu hoặc những dự án đòi hỏi kiến thức sâu rộng về Django.
* ***Tính linh hoạt trong việc lựa chọn cơ sở dữ liệu:***Mặc dù Django hỗ trợ nhiều loại cơ sở dữ liệu, nhưng việc thay đổi hoặc chuyển đổi giữa các loại cơ sở dữ liệu có thể gặp một số khó khăn. Điều này có thể là vấn đề nếu yêu cầu của dự án thay đổi hoặc khi cần phải tối ưu hóa hiệu suất dự án.

## 2.6 Tính năng

* ***Hỗ trợ ORM mạnh mẽ:*** cho phép tương tác với cơ sở dữ liệu một cách dễ dàng mà không cần phải viết các truy vấn SQL trực tiếp.
* ***Hệ thống Routing linh hoạt*:** Định tuyến các URL đến các hàm xử lý tương ứng.
* ***Hệ thống Template:*** Cho phép tạo và tái sử dụng các giao diện người dùng dễ dàng.
* ***Bảo mật tích hợp:*** Bao gồm cả việc xử lý CSRF (Cross-Site Request Forgery) và XSS (Cross-Site Scripting).
* ***Hệ thống Admin tự động***: Django đi kèm với một giao diện quản trị tự động được tạo ra từ các model của bạn mà không cần phải viết mã. Giao diện này cho phép quản trị viên thao tác với dữ liệu một cách dễ dàng, bao gồm việc thêm, sửa, xóa và tìm kiếm.
* ***Hỗ trợ cho Đa ngôn ngữ và Quốc gia:*** Django cung cấp các công cụ và tính năng để phát triển các ứng dụng đa ngôn ngữ và đa quốc gia, bao gồm việc xác định, dịch và hiển thị nội dung theo ngôn ngữ và vùng miền cụ thể.

# CHƯƠNG 3 – KIẾN TRÚC MVT TRONG DJANGO

## 3.1 Khái niệm

- Trong Django, MVT là viết tắt của Model-View-Template, một biến thể của mô hình MVC (Model-View-Controller). Kiến trúc này giúp tổ chức và phân chia logic của ứng dụng web thành các thành phần riêng biệt, giúp mã nguồn dễ bảo trì và mở rộng.

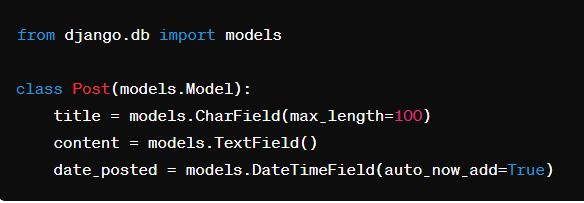
## 3.2 Ví dụ

- Ý nghĩa và ví dụ cụ thể về MVT trong Django:

**Model (M)**:

* Model đại diện cho lớp đối tượng trong cơ sở dữ liệu của bạn. Mỗi model trong Django thường tương ứng với một bảng trong cơ sở dữ liệu.
* Model định nghĩa cấu trúc dữ liệu và quan hệ giữa các đối tượng. Nó chịu trách nhiệm cho việc tương tác với cơ sở dữ liệu để thêm, sửa đổi hoặc truy vấn dữ liệu.

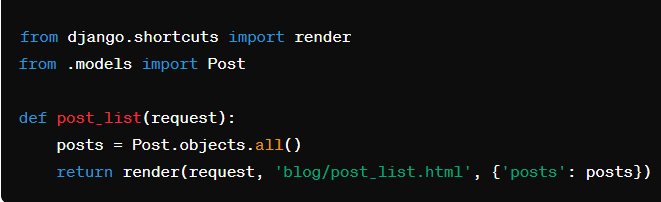
Ví dụ: Xem xét một ứng dụng blog trong Django. Một model cho bài đăng có thể được định nghĩa như sau:



**View (V)**:

* View là thành phần xử lý logic của ứng dụng. Nó nhận yêu cầu từ người dùng, tương tác với model để lấy hoặc cập nhật dữ liệu, sau đó render ra một template để trả về cho người dùng.
* View thường làm nhiệm vụ xử lý dữ liệu đầu vào, xử lý logic kinh doanh và chuẩn bị dữ liệu để hiển thị cho người dùng.

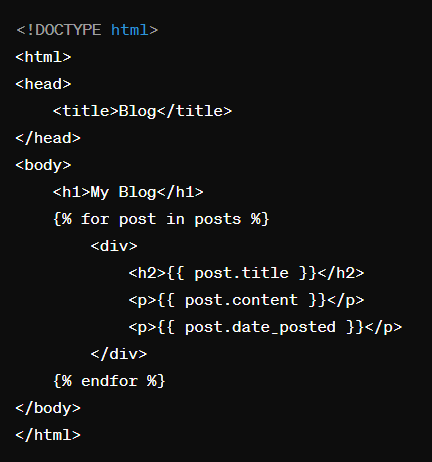
Ví dụ: Dưới đây là một view đơn giản để hiển thị danh sách bài đăng trong ứng dụng blog:



**Template (T)**:

* Template là giao diện người dùng của ứng dụng, định nghĩa cách dữ liệu được hiển thị cho người dùng cuối.
* Template sử dụng ngôn ngữ template của Django để tương tác với dữ liệu được truyền từ view và hiển thị nó ra trình duyệt web.

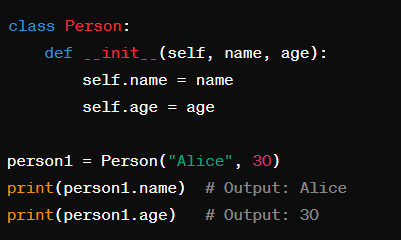
Ví dụ: Dưới đây là một template đơn giản để hiển thị danh sách bài đăng:



# CHƯƠNG 4 – MỘT VÀI METHOD CƠ BẢN

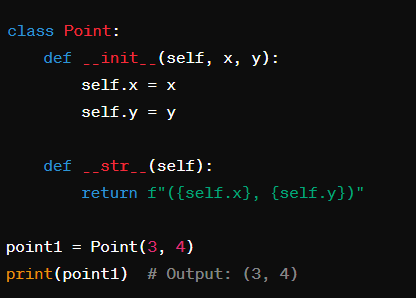
## 4.1 Method \_\_init\_\_:

* Chức năng: Được sử dụng để khởi tạo một đối tượng khi nó được tạo ra.
* Ví dụ:



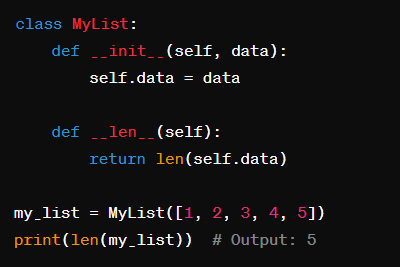
## 4.2 Method \_\_str\_\_:

* Chức năng:Được sử dụng để định dạng một đối tượng thành một chuỗi để in ra.
* Ví dụ:



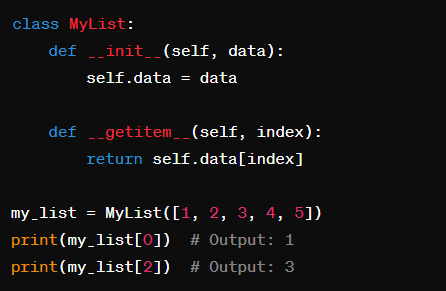
## 4.3 Method \_\_len\_\_:

* Chức năng: Trả về độ dài hoặc số lượng phần tử trong một đối tượng.
* Ví dụ:



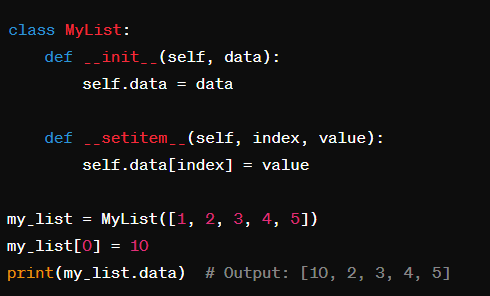
## 4.4 Method \_\_getitem\_\_:

* Chức năng: Cho phép truy cập các phần tử của một đối tượng như một chuỗi hoặc mảng.
* Ví dụ:



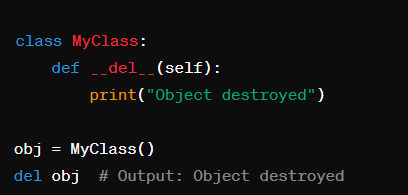
## 4.5 Method \_\_setitem\_\_:

* Chức năng: Cho phép thiết lập giá trị của một phần tử trong một đối tượng.
* Ví dụ:



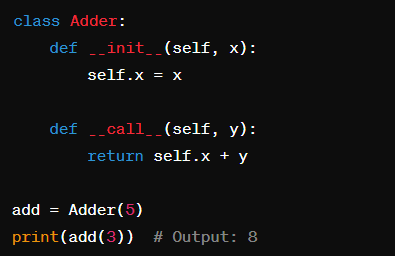
## 4.6 Method \_\_del\_\_:

* Chức năng: Được gọi khi một đối tượng bị xóa hoặc giải phóng khỏi bộ nhớ.
* Ví dụ:



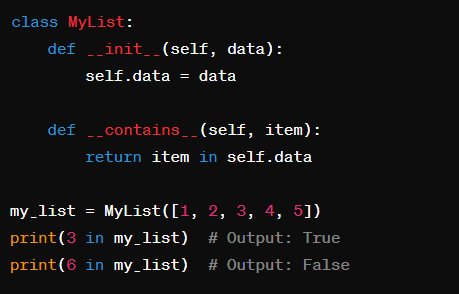
## 4.7 Method \_\_call\_\_:

* Chức năng: Cho phép đối tượng được gọi như một hàm.
* Ví dụ:



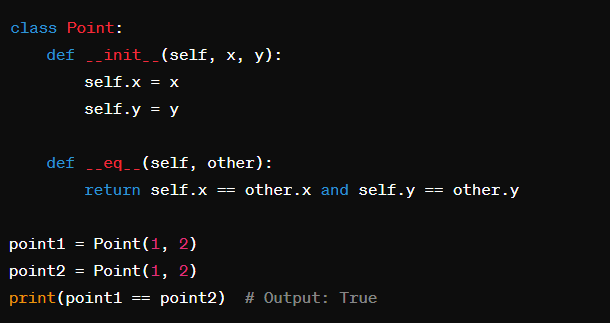
## **4.8 Method \_\_contains**\_\_:

* Chức năng: Kiểm tra xem một giá trị có tồn tại trong đối tượng hay không.
* Ví dụ:



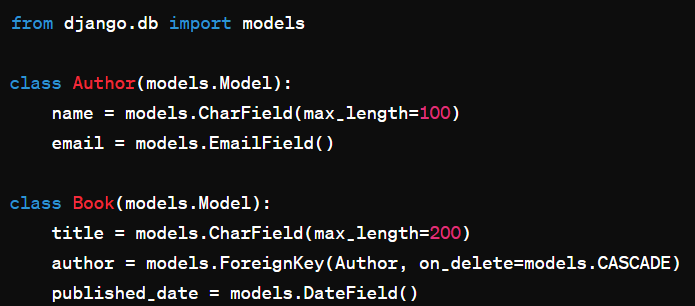
## **4.9 Method \_\_eq**\_\_:

* Chức năng: So sánh hai đối tượng để xác định xem chúng có bằng nhau không.
* Ví dụ:



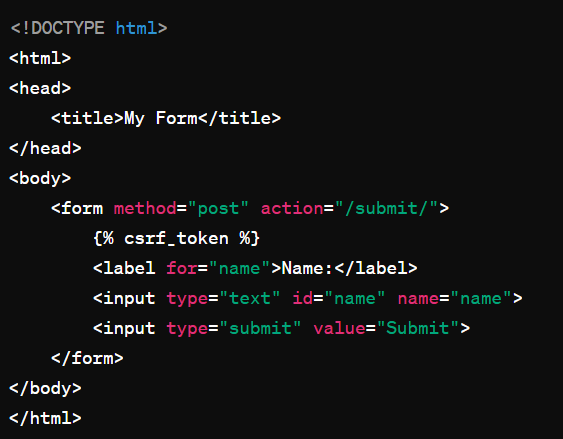
## 4.10 Method models.Model

* Chức năng : Một model là nguồn thông tin duy nhất và quyết định về dữ liệu của bạn. Nó chứa các trường và hành vi cần thiết của dữ liệu bạn đang lưu trữ. Thông thường, mỗi model tương ứng với một bảng cơ sở dữ liệu.
* Các điều cơ bản:
* Mỗi model là một lớp Python mà kế thừa từ django.db.models.Model.
* Mỗi thuộc tính của model đại diện cho một trường cơ sở dữ liệu.
* Ví dụ:



## 4.11 Method CSKF trong django

* Chức năng : CSRF (Cross-Site Request Forgery) token là một cơ chế bảo mật giúp bảo vệ ứng dụng web khỏi các cuộc tấn công CSRF. Các cuộc tấn công CSRF xảy ra khi một trang độc hại thực hiện các hành động không mong muốn thay cho người dùng đã xác thực trên một trang web khác. CSRF token giúp đảm bảo rằng các yêu cầu gửi đến server thực sự đến từ người dùng hợp lệ.
* Ví dụ : **Thêm CSRF Token vào Form**

****

## 4.12 Method Path trong Django

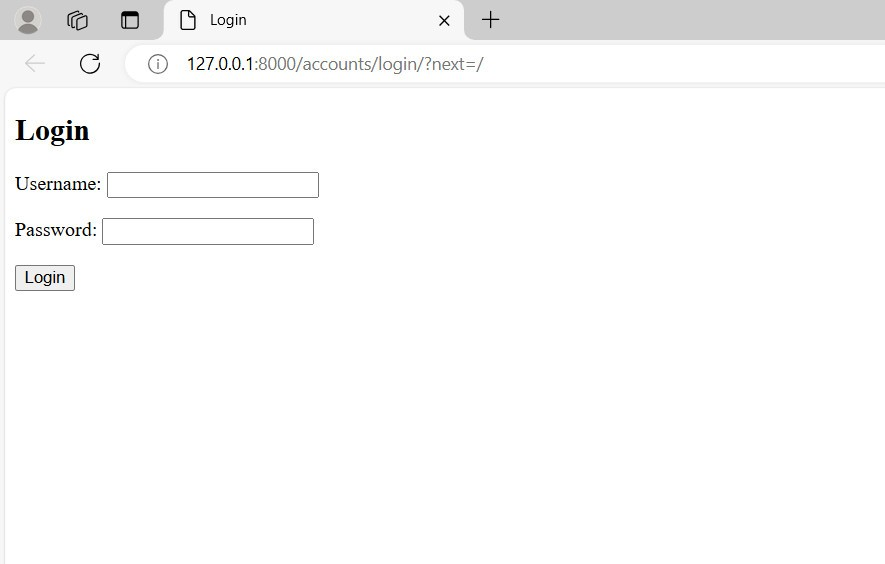
* Chức năng: Trong Django, path là một hàm được sử dụng để định nghĩa các URL pattern trong ứng dụng của bạn. Nó nằm trong module django.urls và giúp ánh xạ các URL tới các view cụ thể. Khi người dùng truy cập một URL, Django sẽ sử dụng các pattern này để xác định view nào sẽ xử lý yêu cầu đó
* Ví dụ:

****

# CHƯƠNG 5 – TIẾN ĐỘ WED

## 5.1 Trang Login

* Trang login bắt buộc login để sang home nếu không sẽ trả về lỗi hoặc quay ngược lại login. Nếu sai mật khẩu tự động load lại trang:

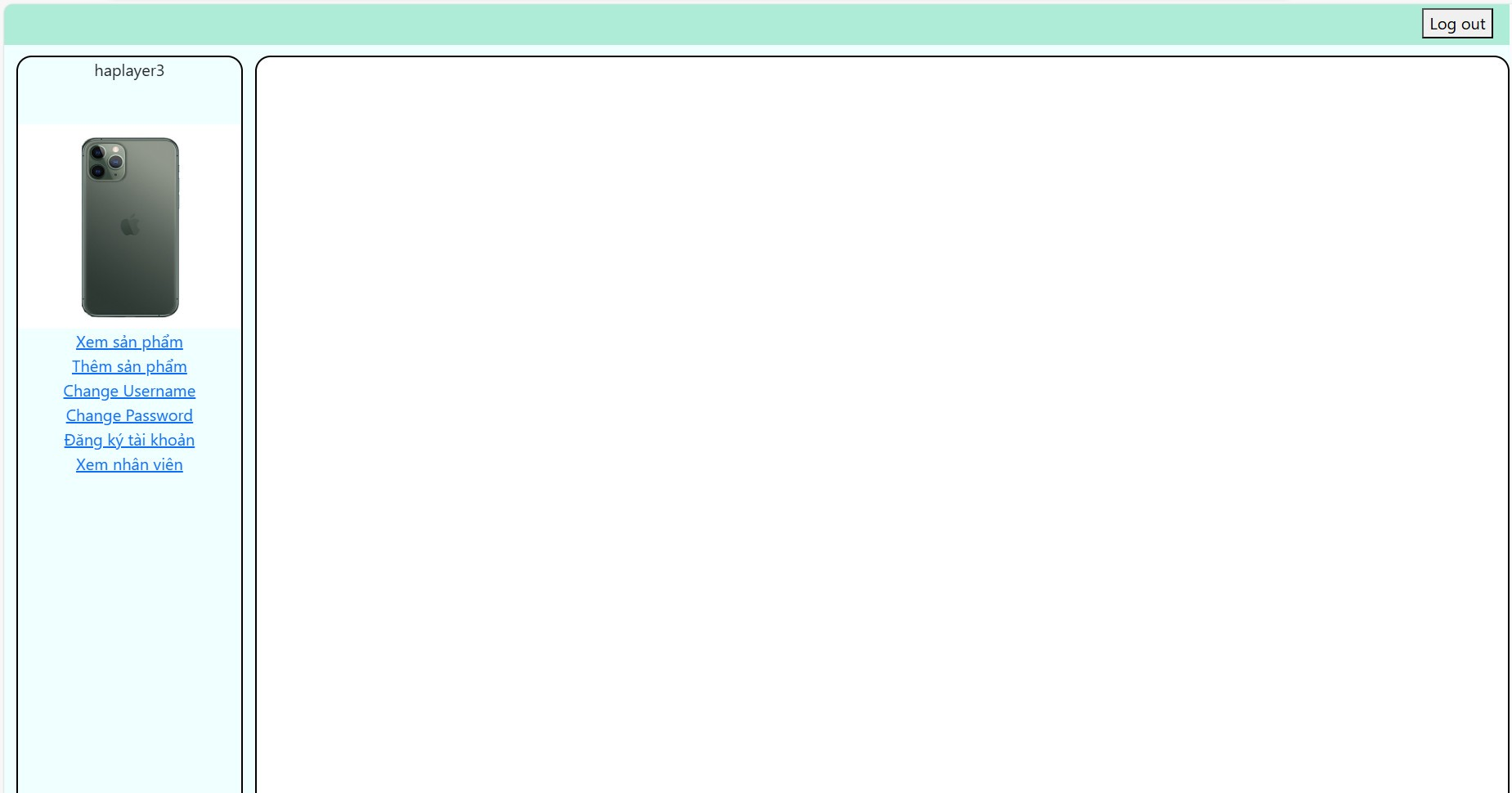


### 5.1.1 Xử lý code chính



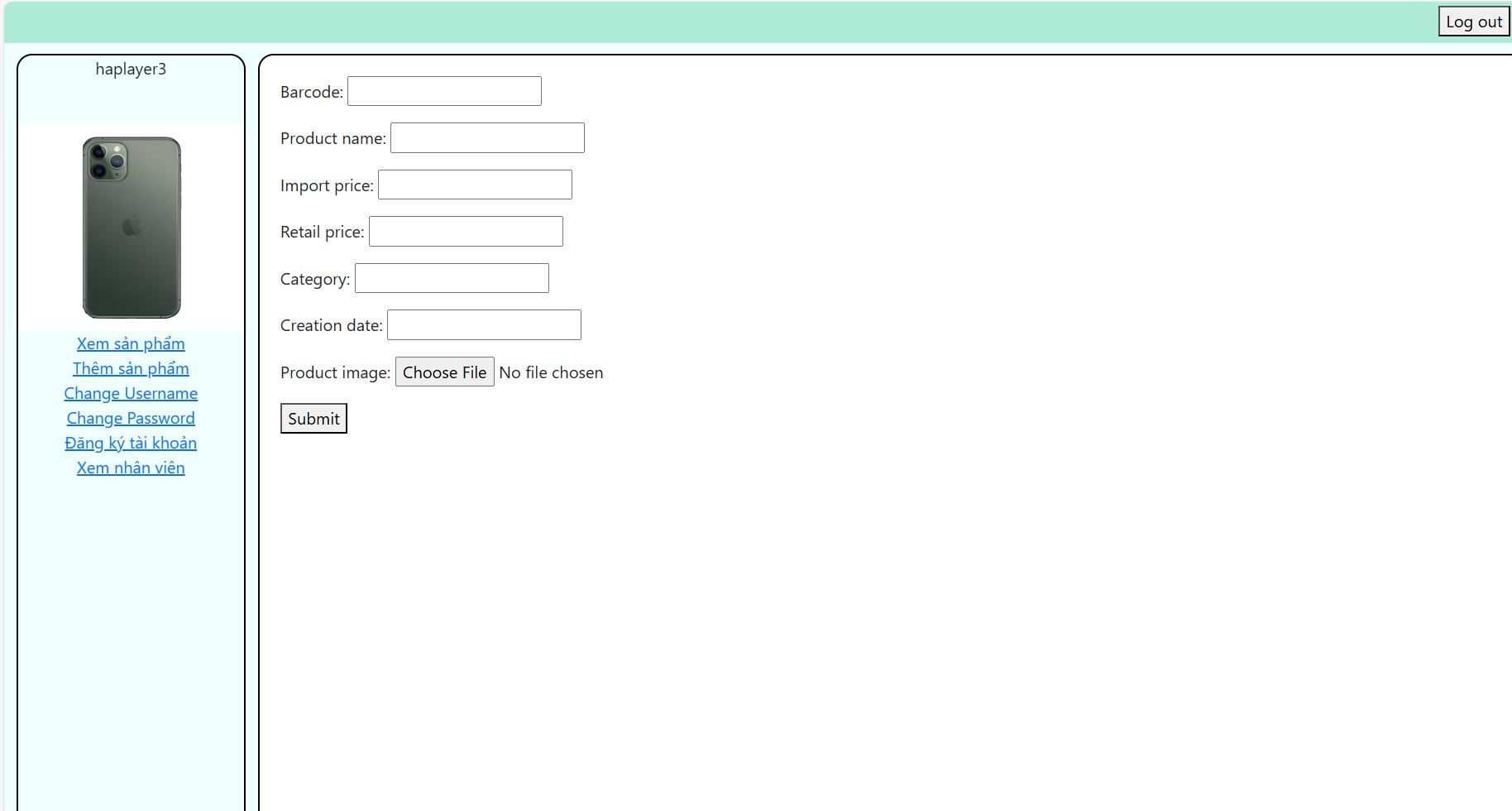
## 5.2 Trang chủ

* Sau khi đăng nhập thành công sẽ tự động chuyển đến trang Home:
* Barside menu bên trái sẽ dc fix với viewport
* Barside menu bên trái và header sẽ dc fix với viewport
* Vùng trắng sẽ là content view được hiển thị cũng như tưởng tác ajax
* Các chức năng như:
* Xem sản phẩm
* Thêm sản phẩm
* Change Username
* Change Password
* Đăng ký tài khoản
* Xem nhân viên
* Trong đó có 2 chức năng là:
* Đăng ký tài khoản
* Xem nhân viên sẽ chỉ hiển thị nếu tại khoản đăng nhập là admin

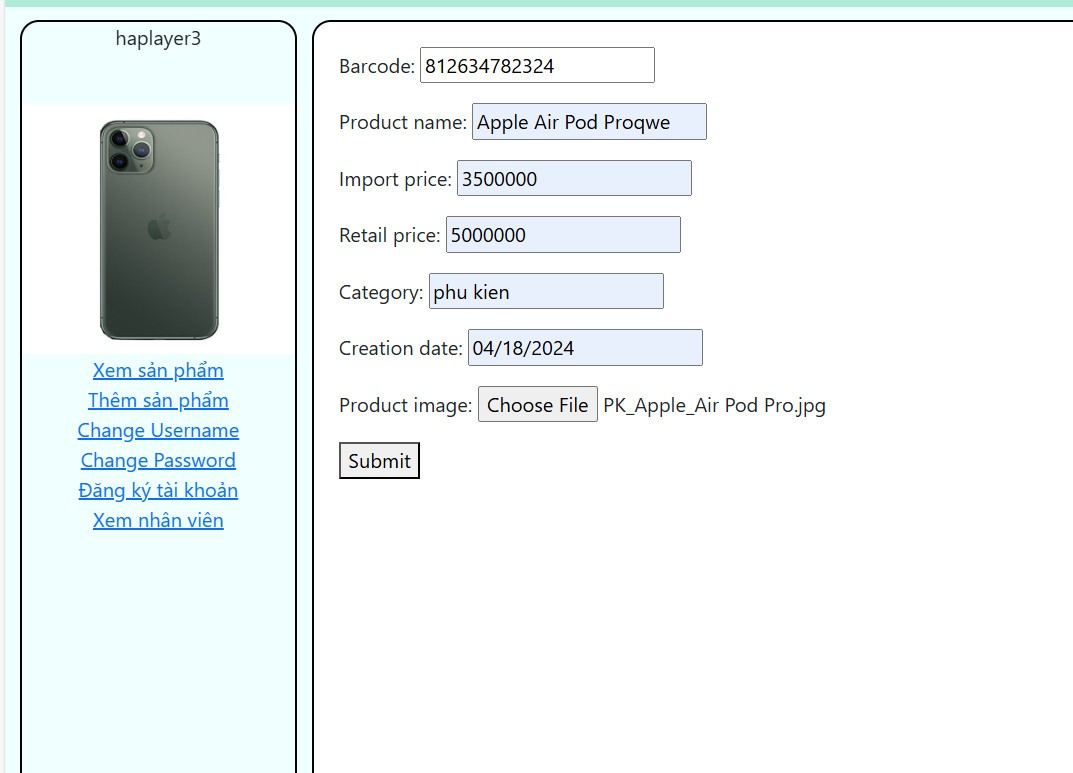


## 5.3 Thêm sản phẩm

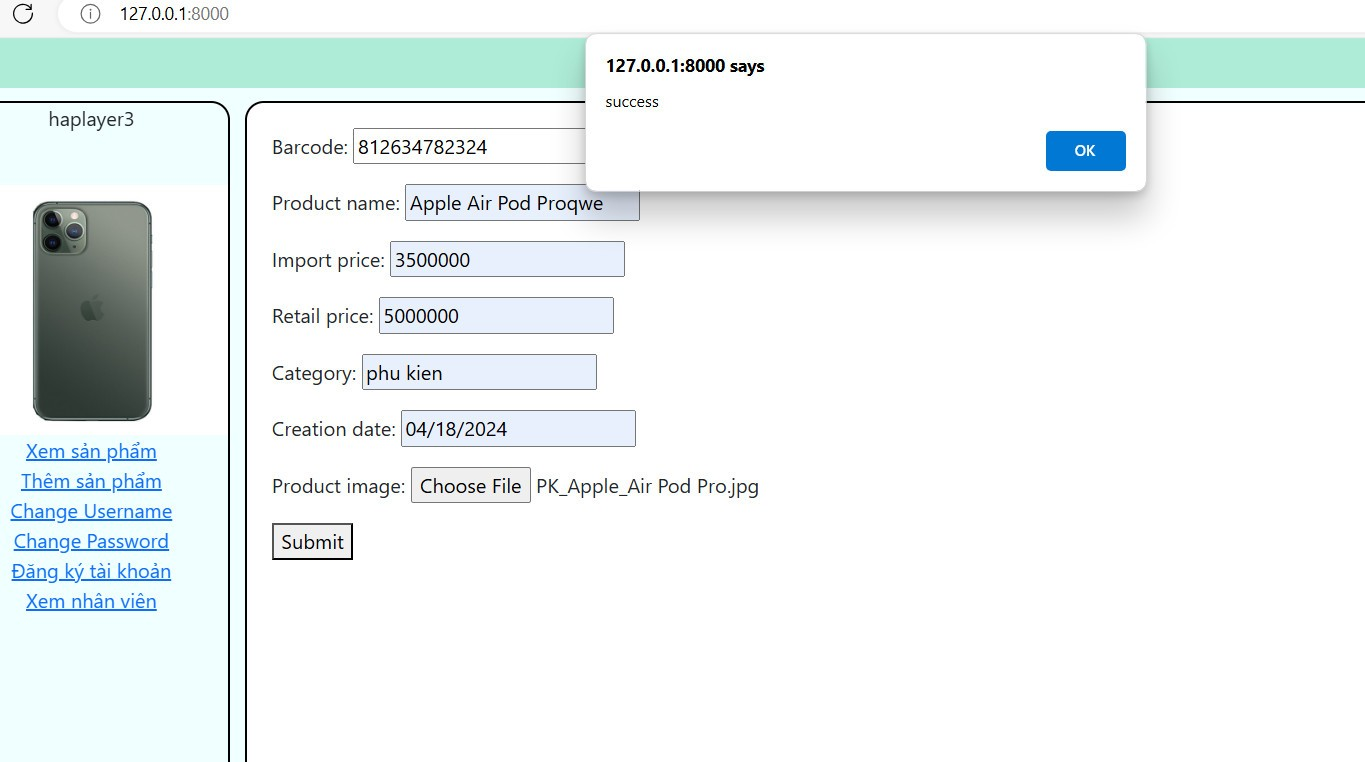
* Form thêm sản phẩm:



* Điền đầy đủ thông tin, sau đó nhâns submit để thêm sản phẩm :

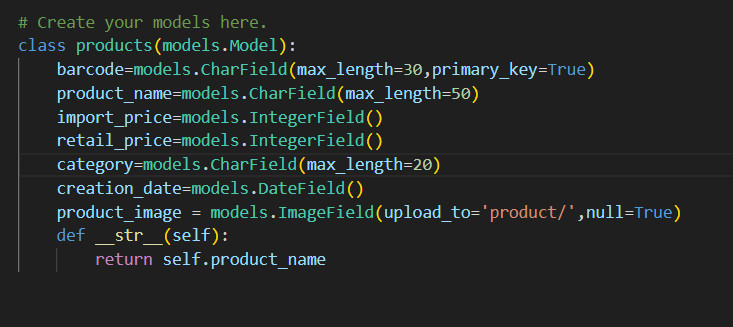


* Sau khi Submit thành công sẽ nhân được alert success :

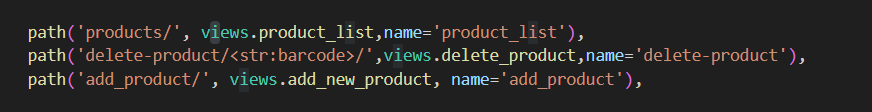


### 5.3.1 Xử lí code chính

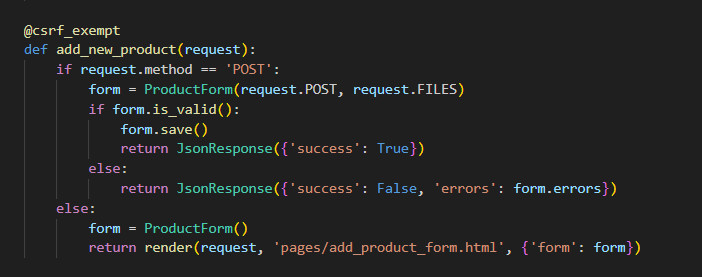
- Database:



- Điều hướng xử lý:

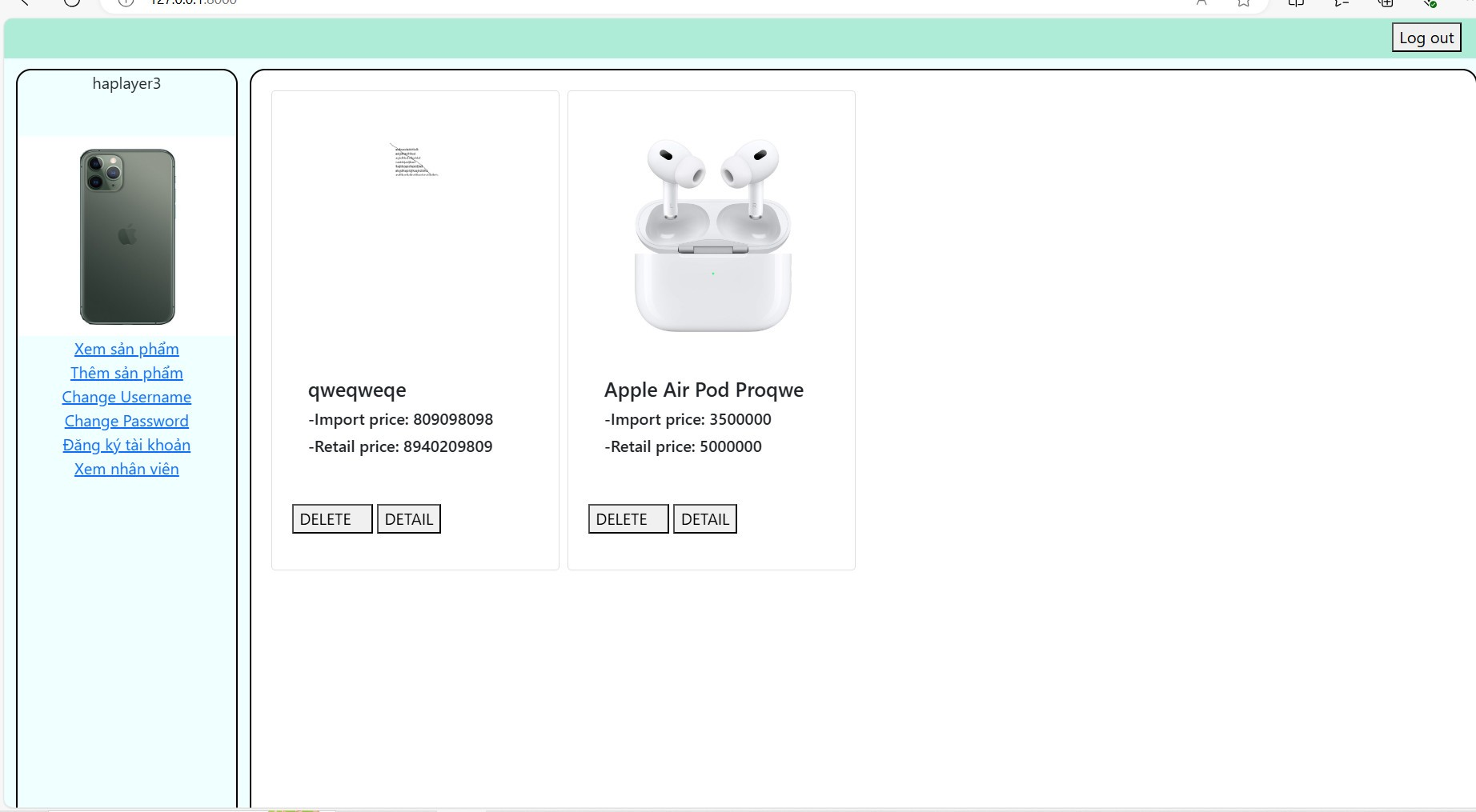


- Thêm sản phẩm:

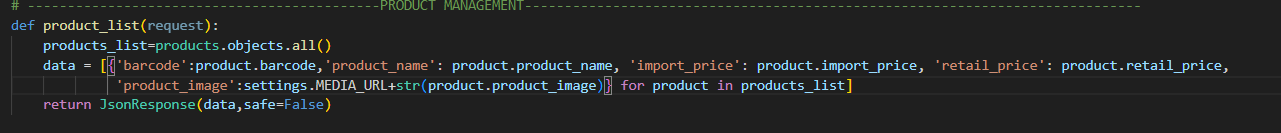


## 5.4 Xem sản phẩm

* Tiếp theo sẽ chuyển qua chức năng xem sản phẩm để xem sản phẩm đã dc thêm vào chưa:

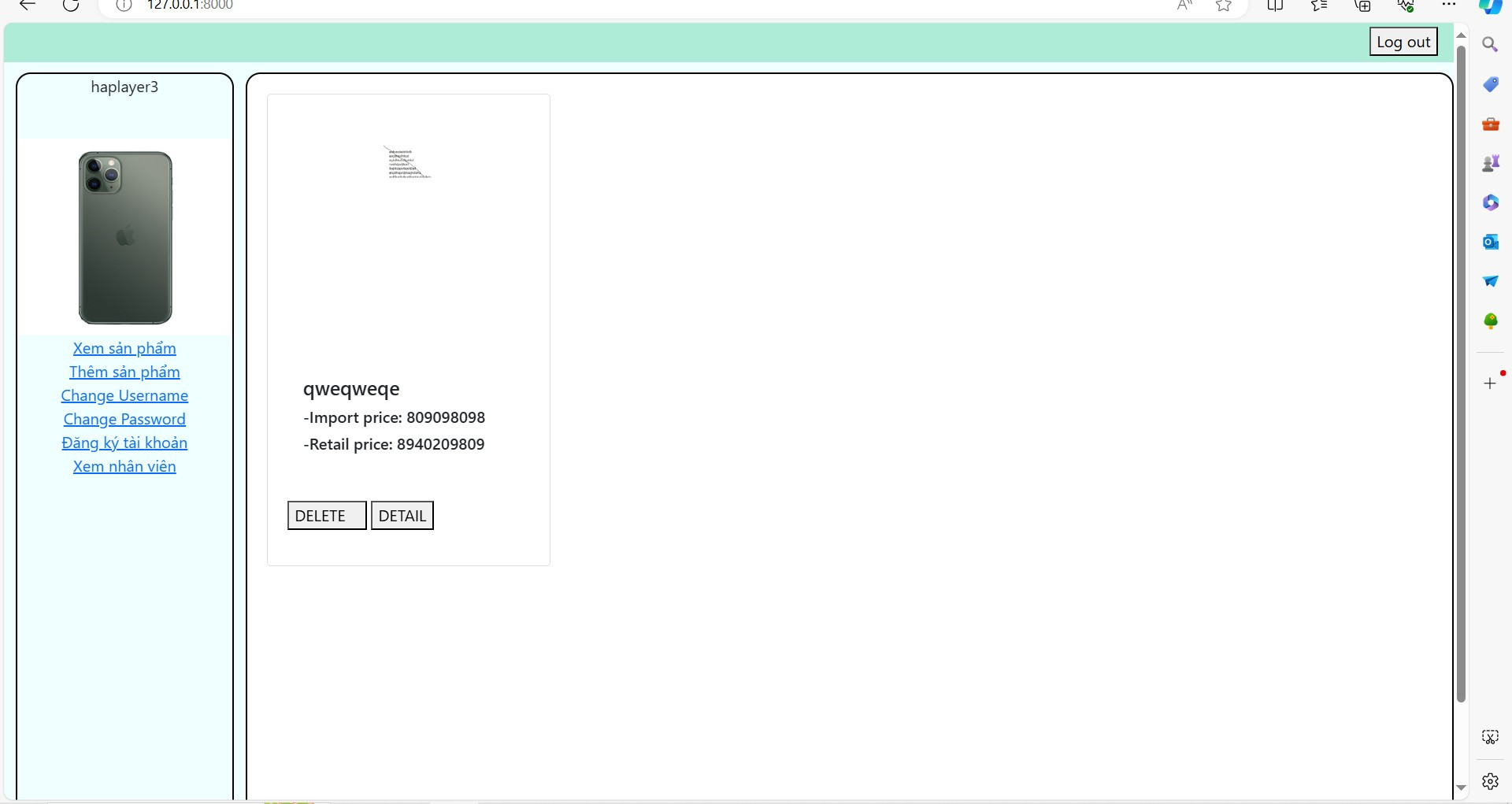


### 5.4.1 Xử lí code chính



## 5.5 Delete sản phẩm

* Chúng ta có thể delete sản phẩm bằng cách nhấn delete (vì là ajax nên content sẽ dc cập nhập tức thì) :

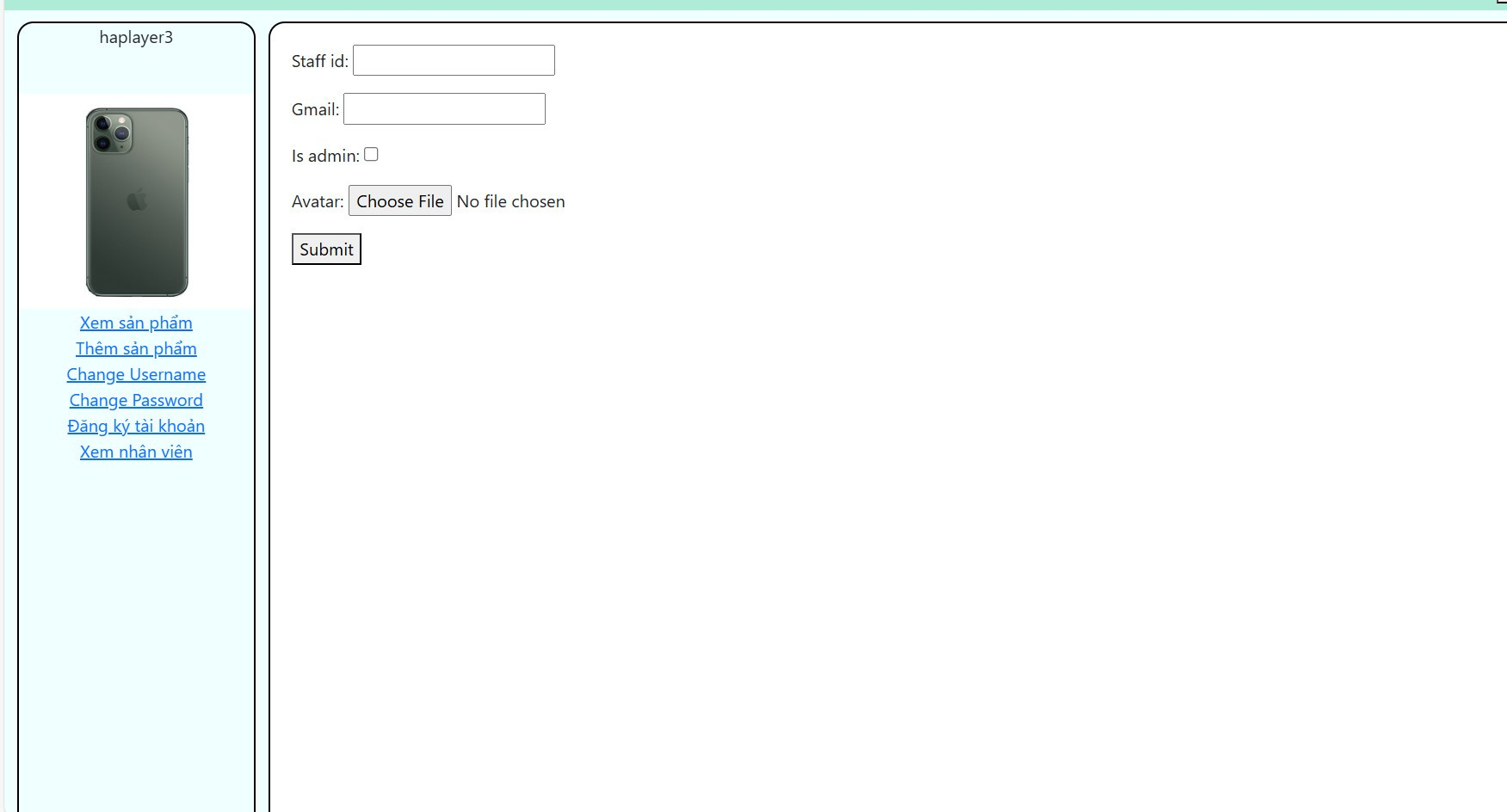


### 5.5.1 Xử lí code chính

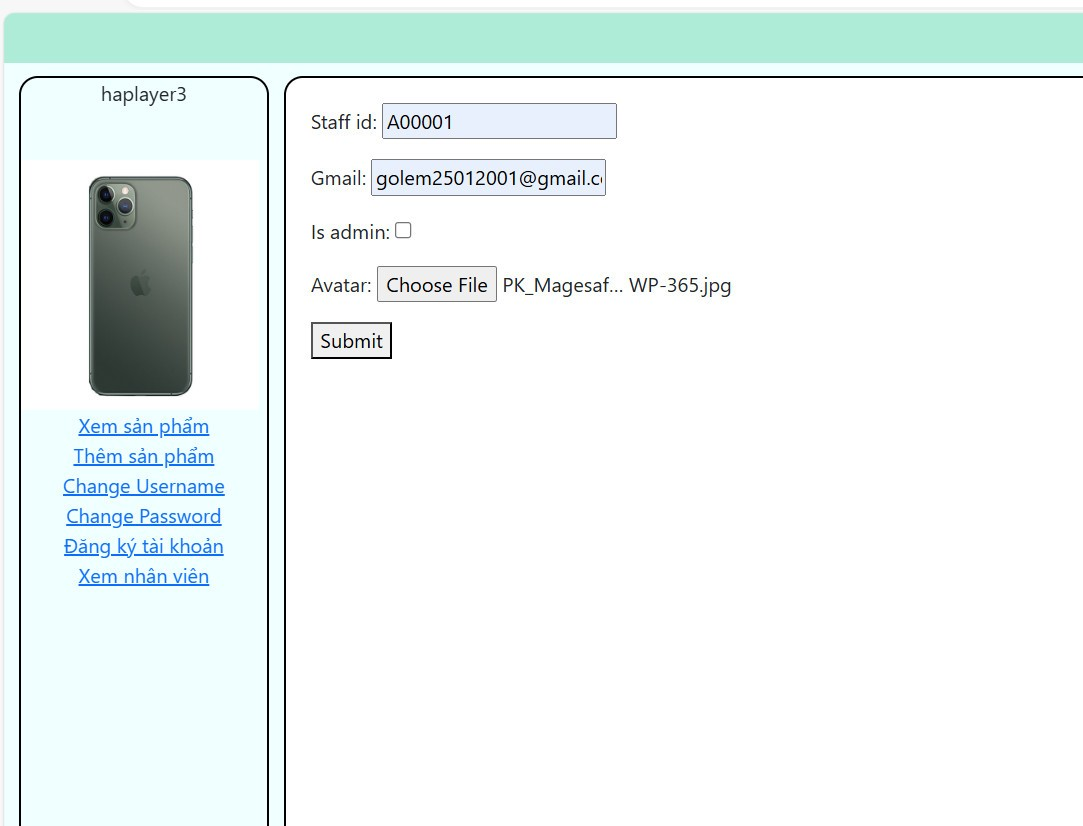


## 5.6 Đăng kí tài khoản

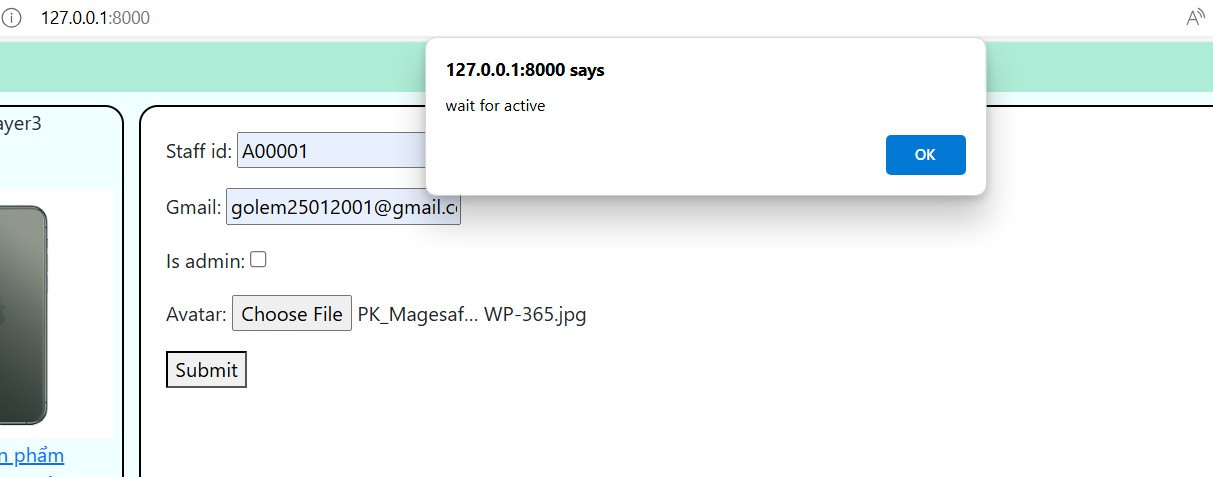
* Form Đăng kí Tài khoản:



* Theo như yêu cầu thì sau khi tạo tài khoản sẽ có 1 đường dẫn được gửi đến mail đã cung cấp
* Người được đăng ký sẽ phải xác nhận trong vòng 5 phút kể từ khi dữ liệu dc tạo. Vì 1 phút quá ngắn ko đủ thời gian xử lý gửi active nên chúng ta set thành 5 phút.

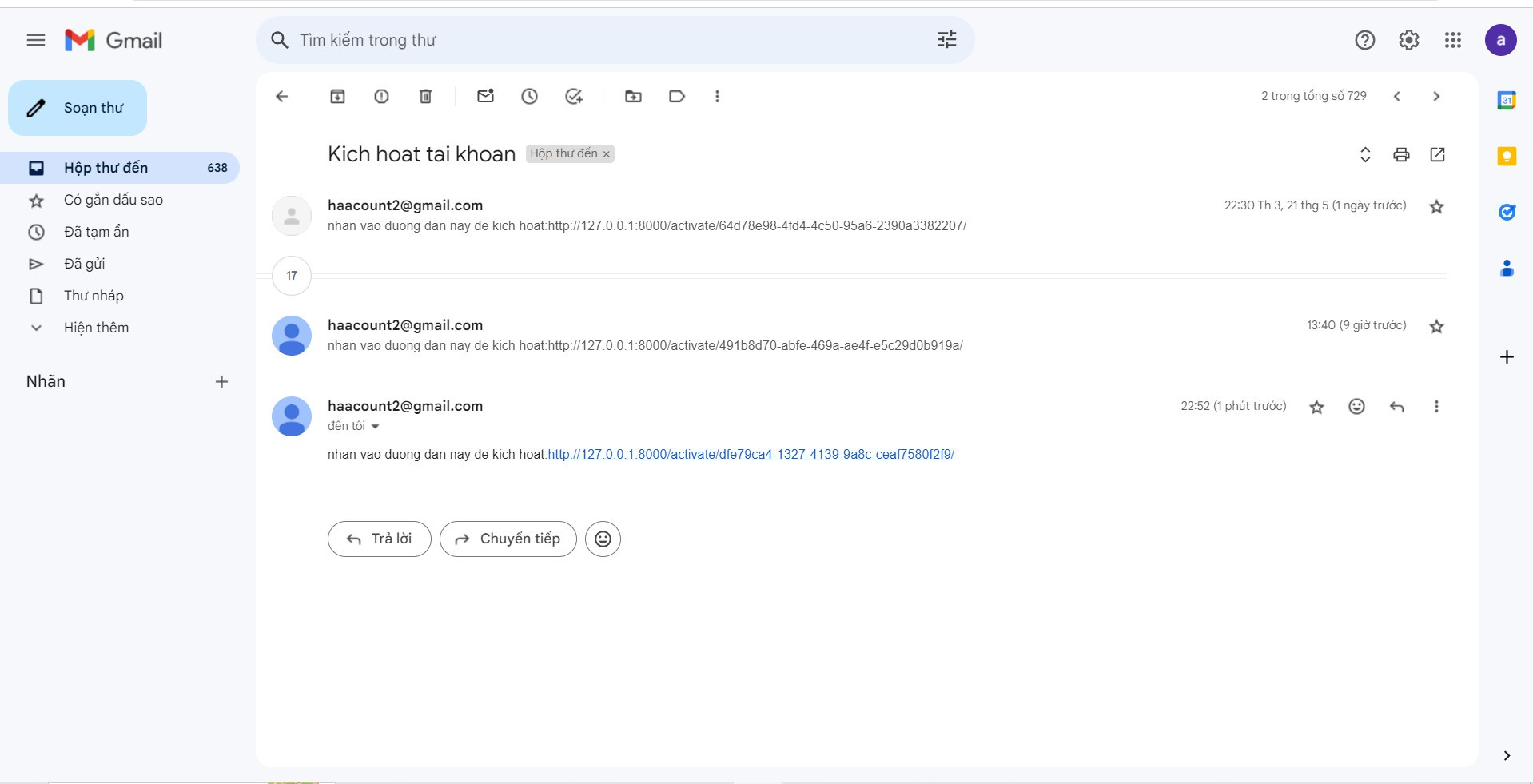


Bắt đầu đếm đợi active:

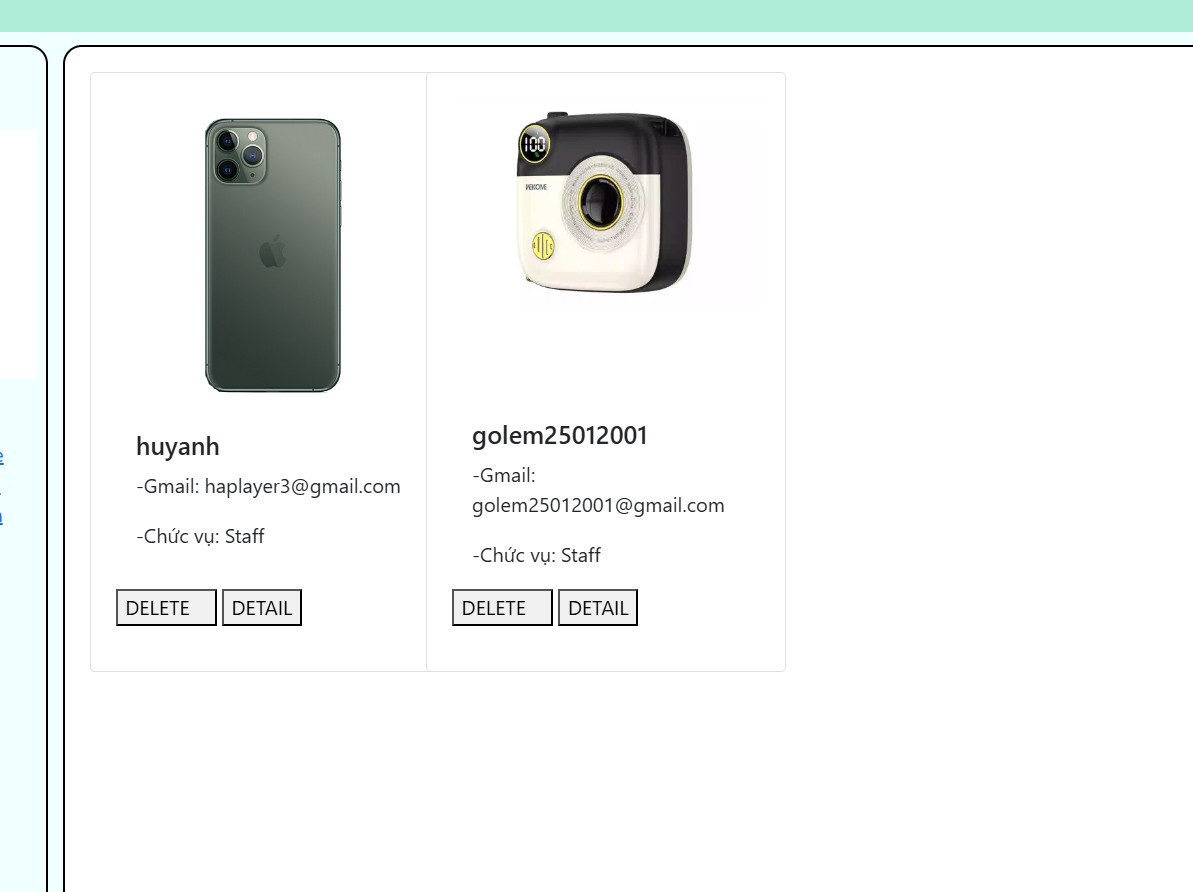


Email nhận được:



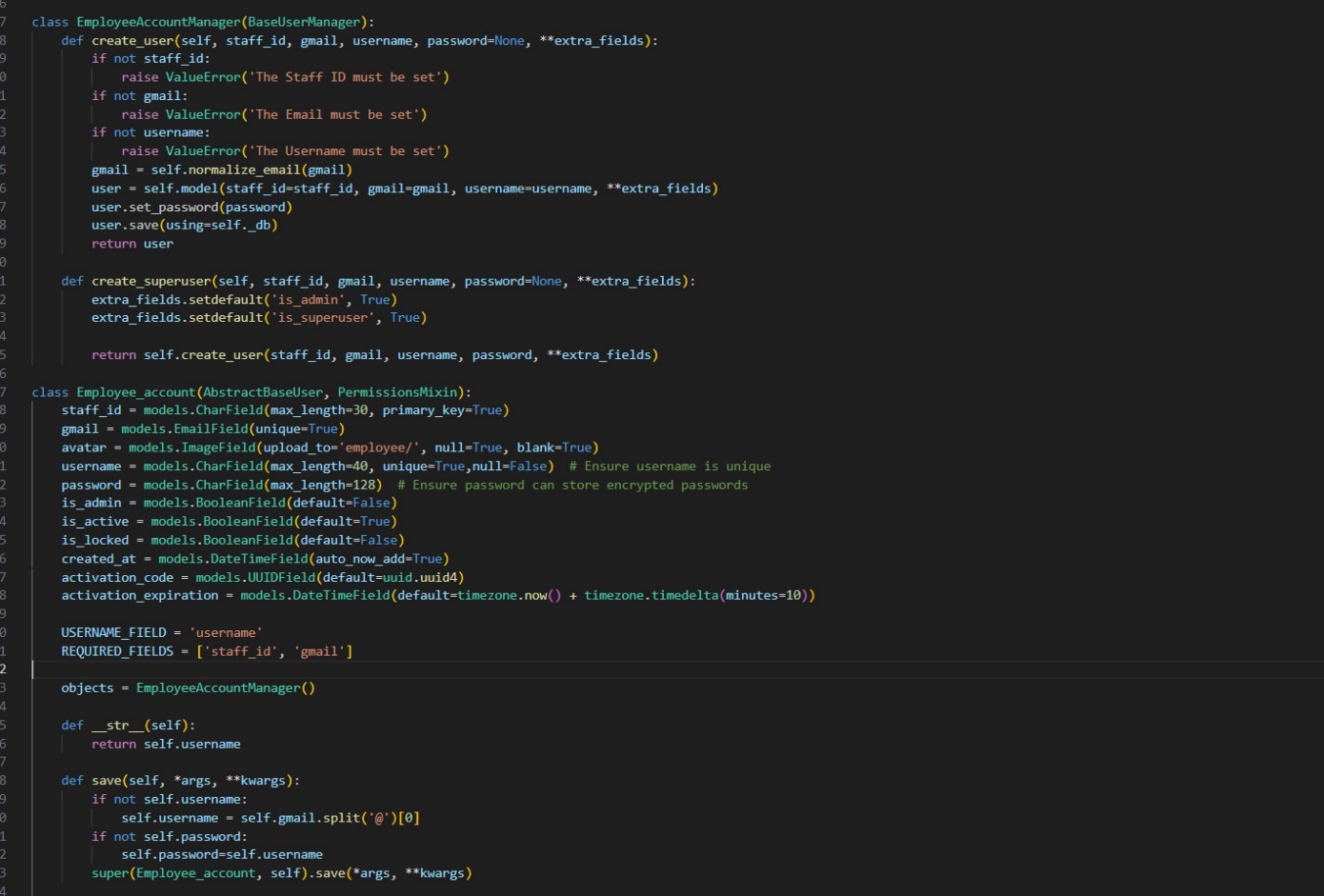


* Sau khi đăng kí thành công:



### 5.6.1 Xử lí code chính

- Database account:

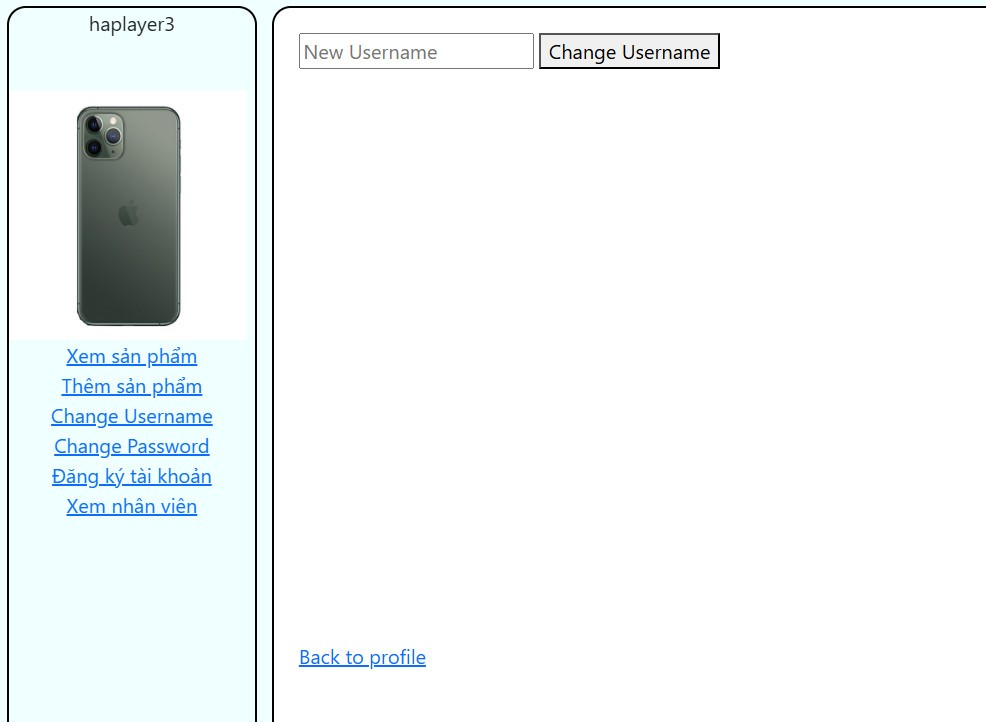


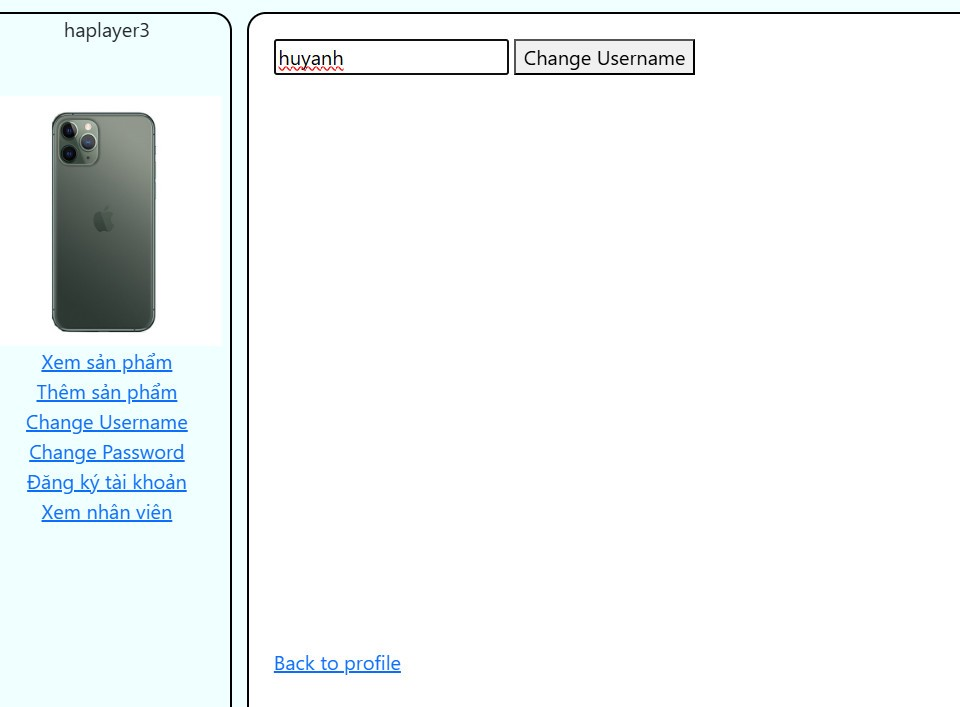
- Đăng kí tài khoản:

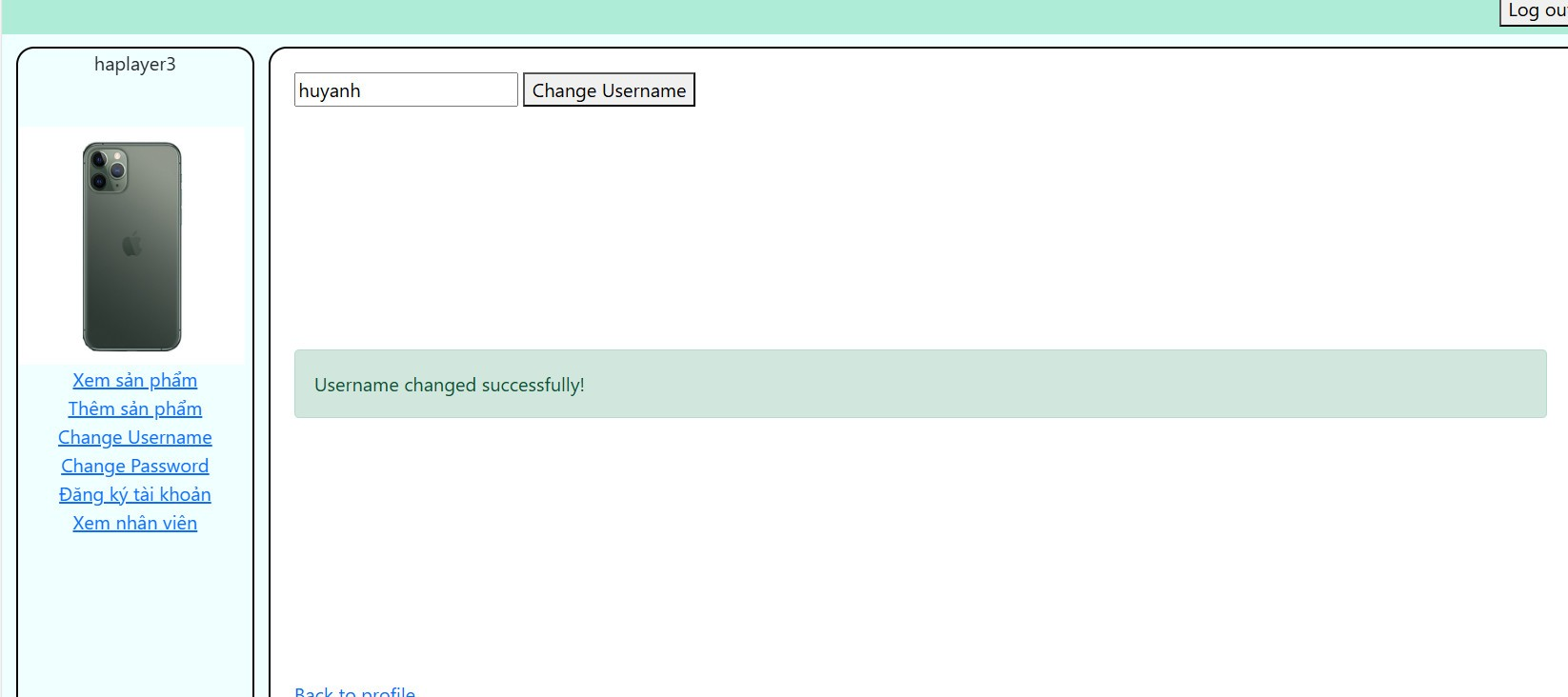


## 5.7 Chức năng đổi mật khẩu và Username

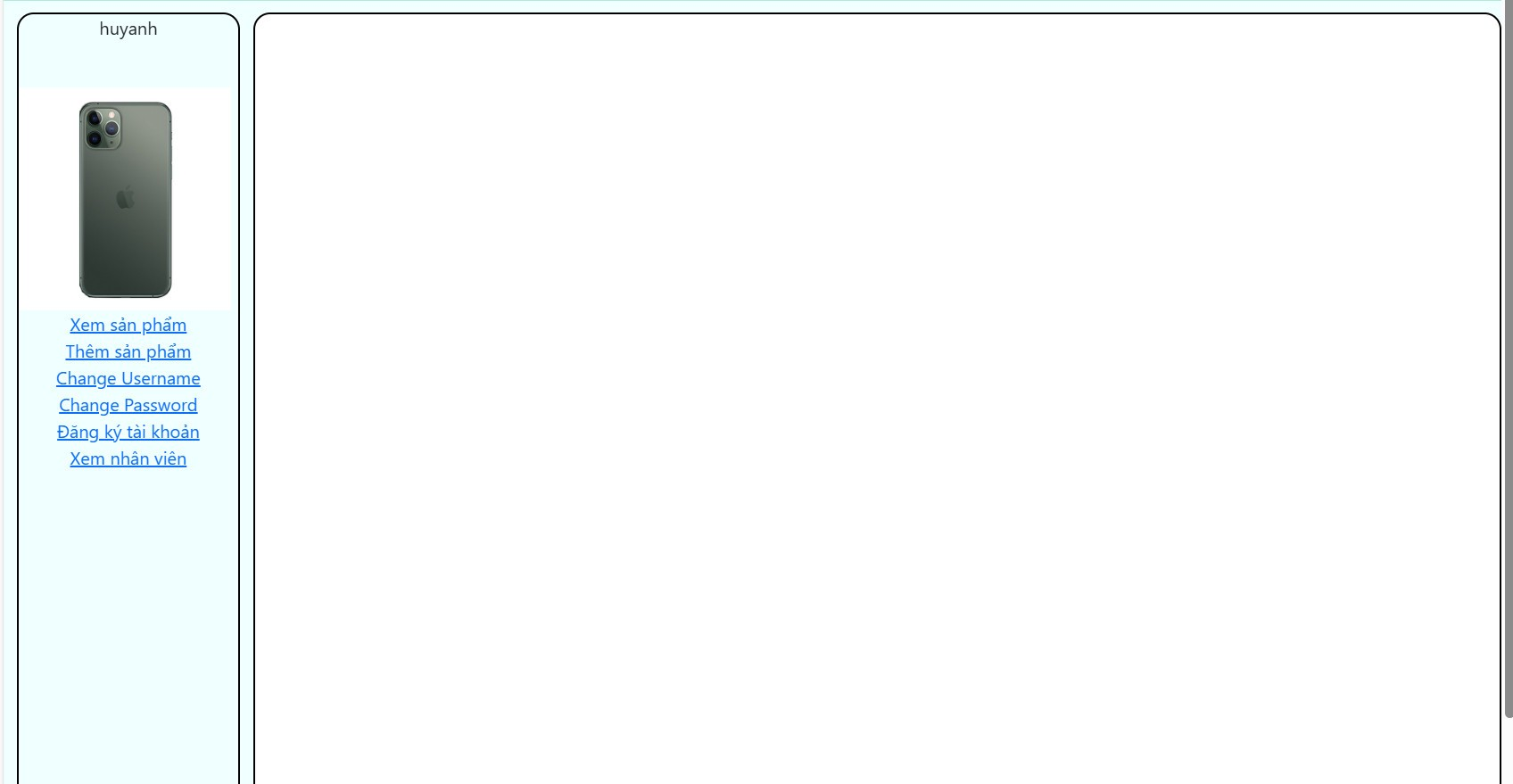
* Đổi Username:





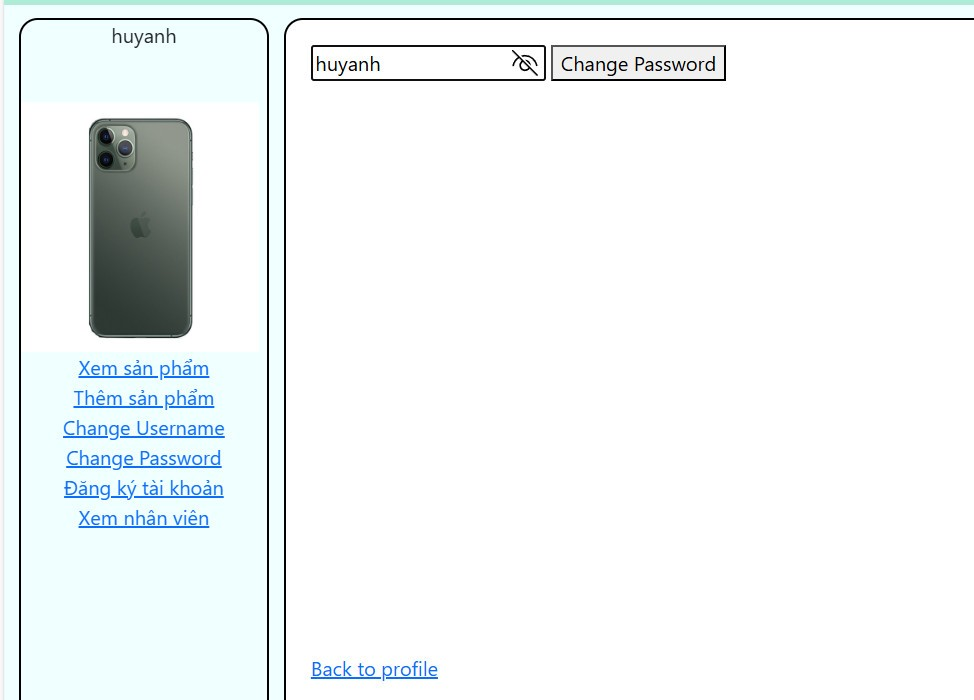


(4)

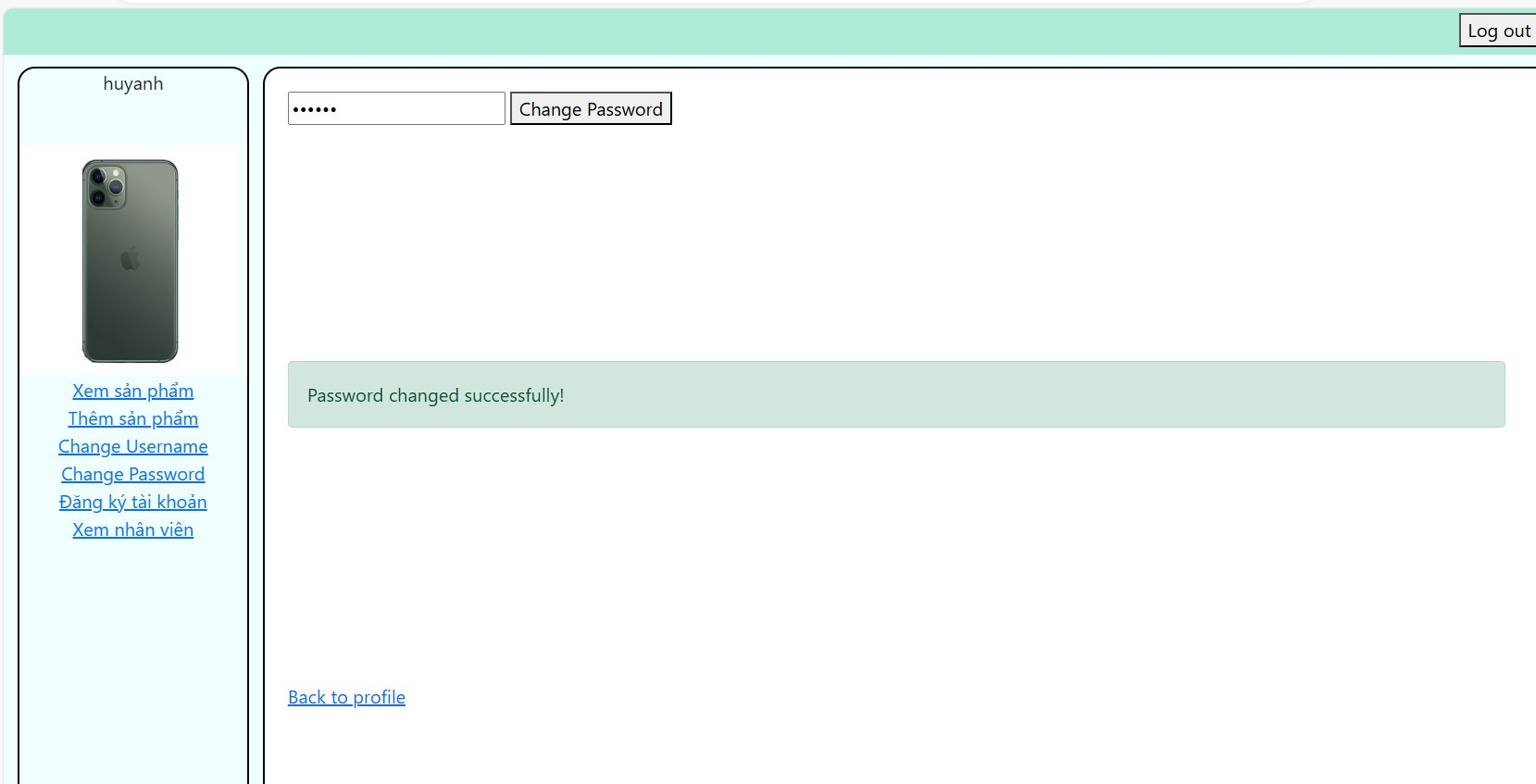


* Đổi mật khẩu :
* Việc đổi mật khẩu cũng tương tự nhưng khó làm rõ hơn. Vì trong django mật khẩu theo chuẩn sẽ được lưu dưới dạng encoder

(1)

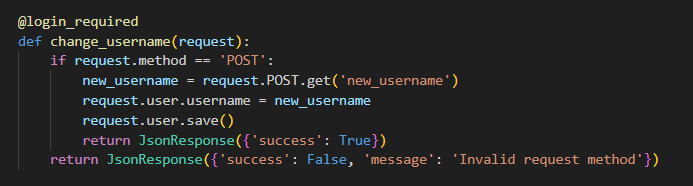


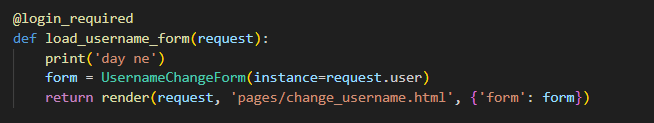
(2)



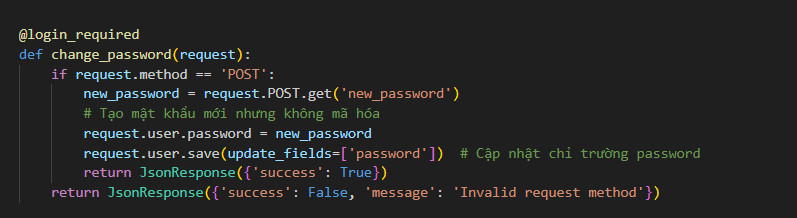
### 5.7.1 Xử lí code chính

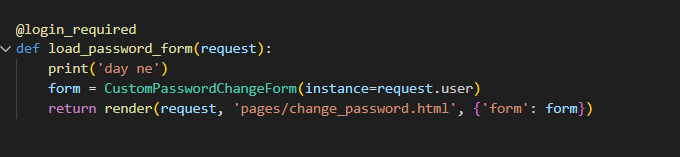
- Đổi username:



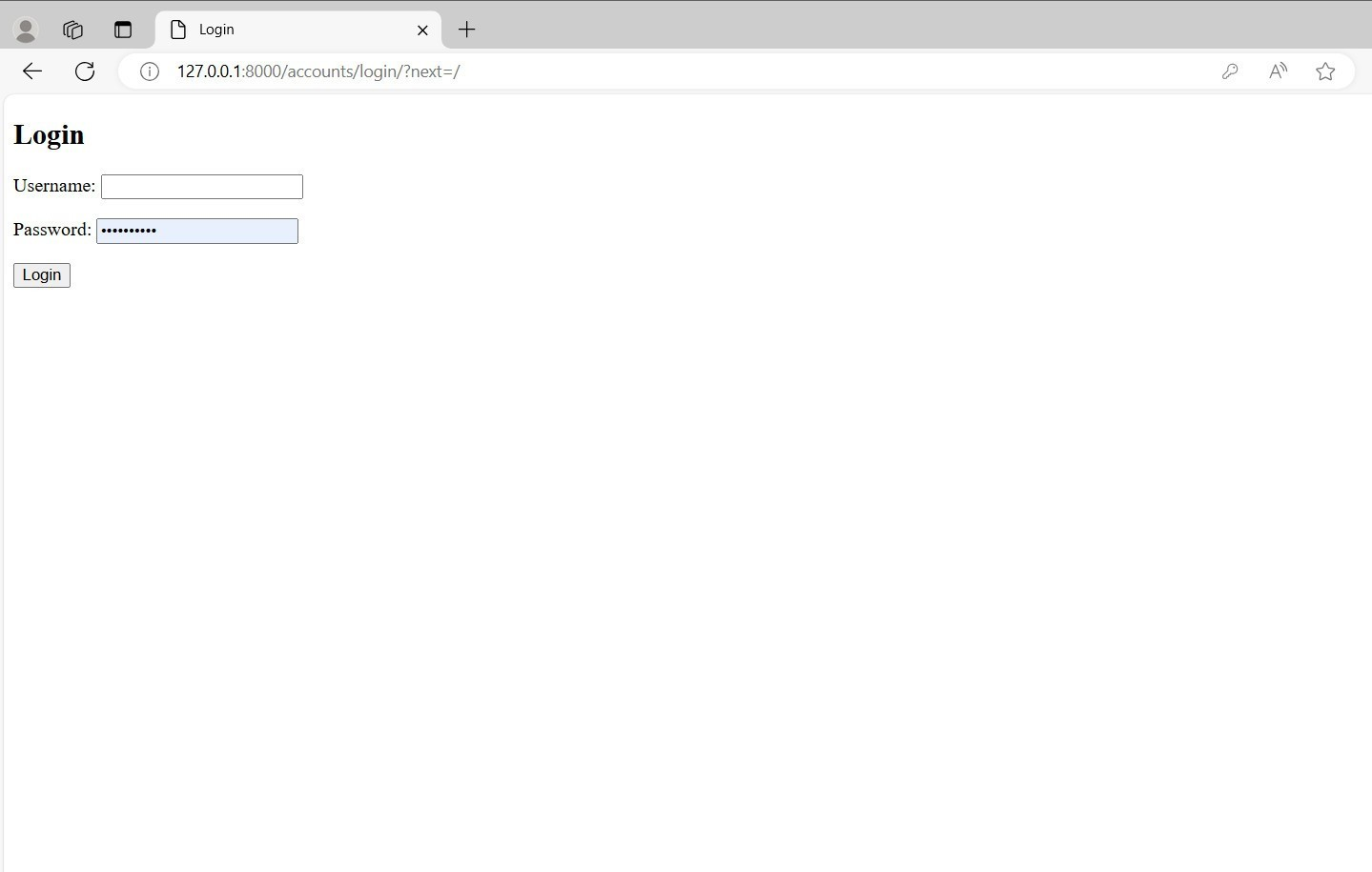


- Đổi mật khẩu:

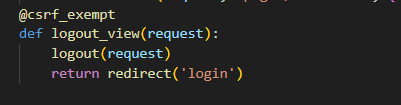




## 5.8 Logout

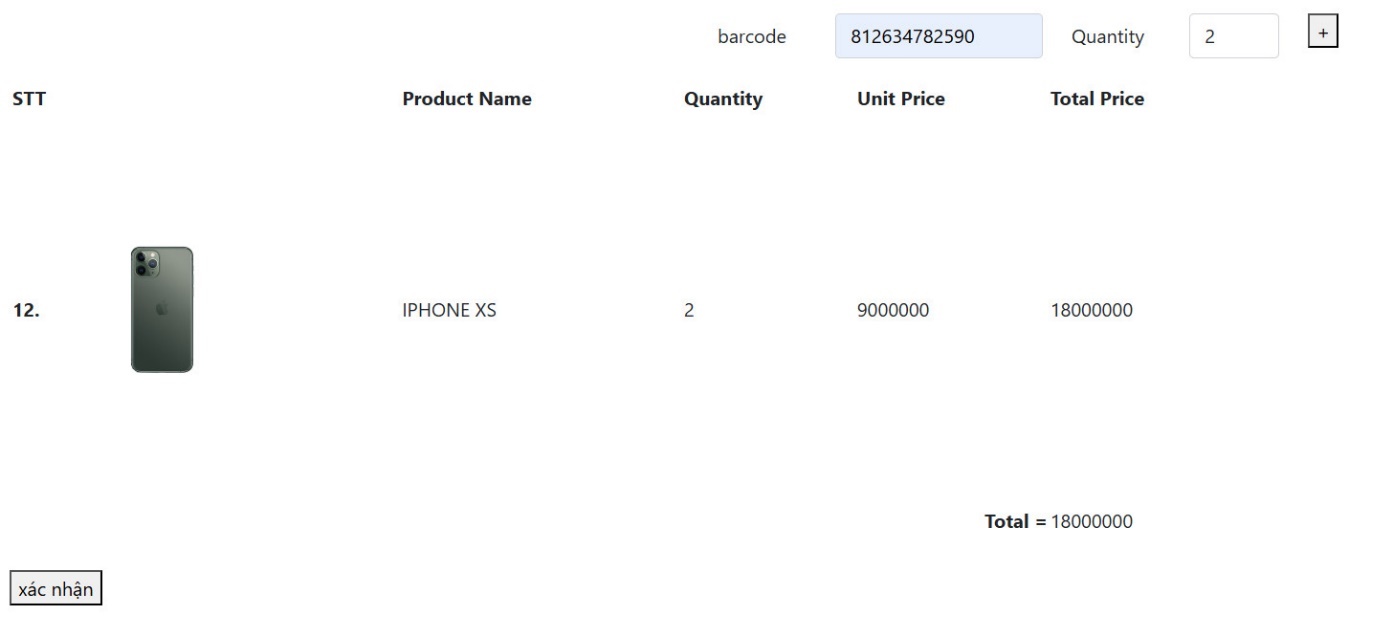


### 5.8.1 Xử lí code chính



## 5.9 Tạo giao dịch

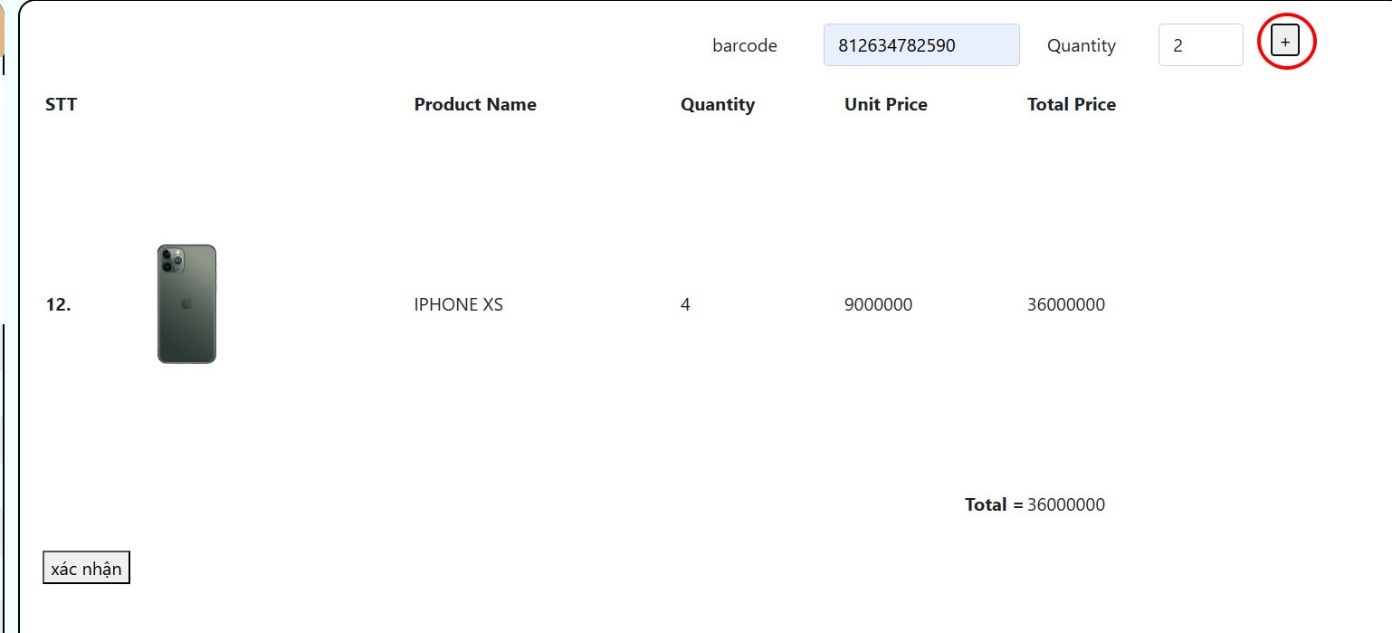
**-** Nhập barcode và quantity để thêm sản phẩm kèm số lượng mua vào đơn hàng.

****

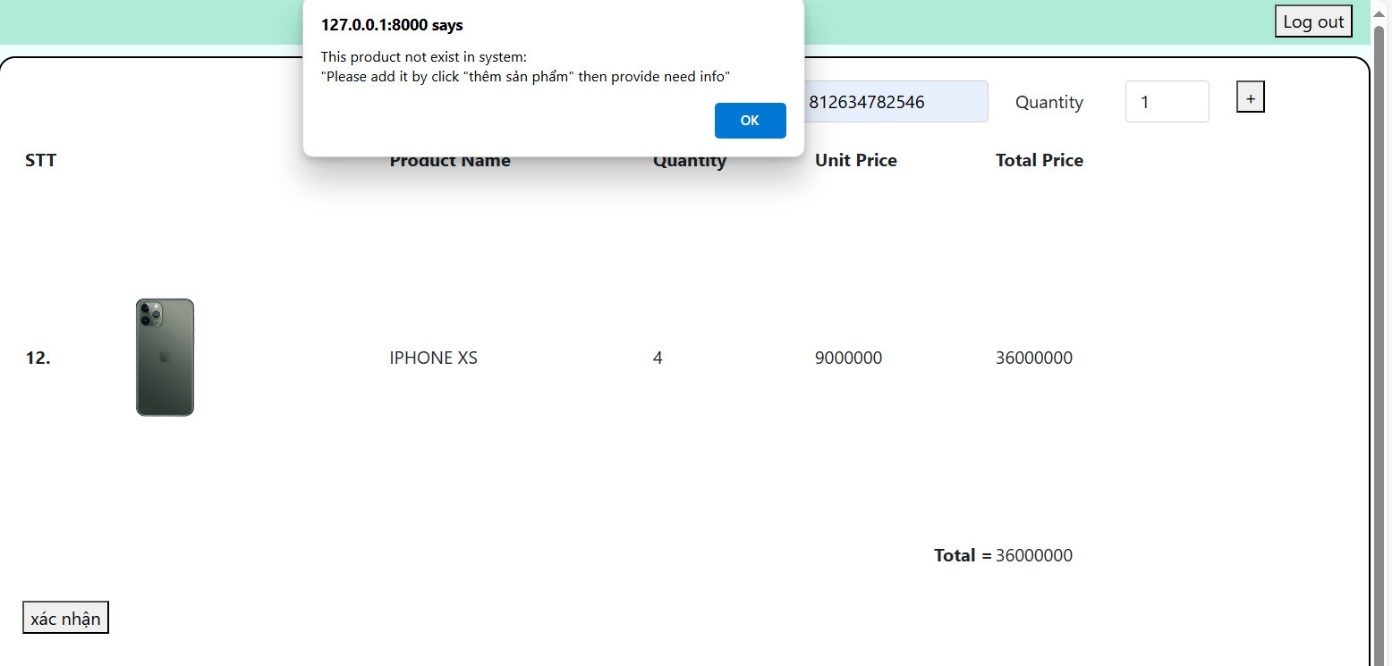
- Sau khi nhập và thêm sản phẩm mua, sản phẩm sẽ được hiện tại vùng content với các dữ liệu như: STT, Product Name, Quantity, Unit Price, Total Price.

- Total và Total Price sẽ đc tự động cập nhập.

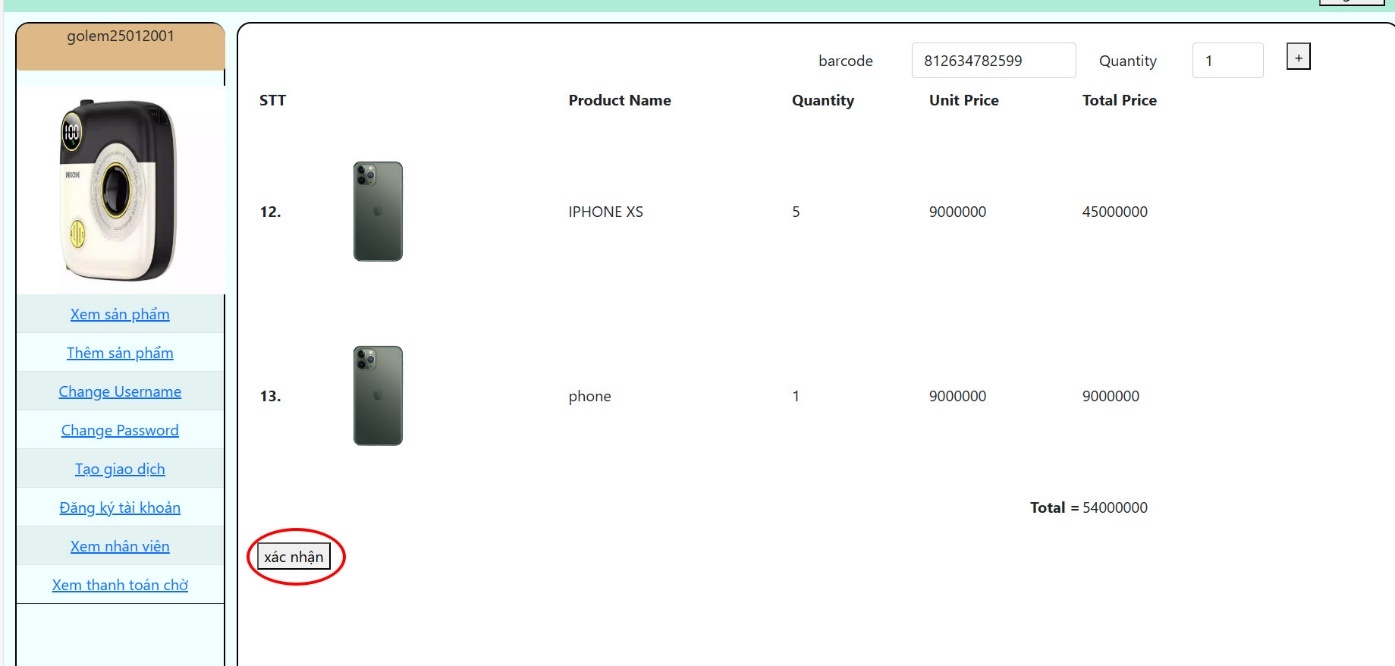
- Nếu nhập cùng sản phẩm số lượng sẽ tăng lên chứ ko thêm hàng mới.



- Nếu nhập sản phẩm không có trong dữ liệu sẽ thông báo cho người dùng.

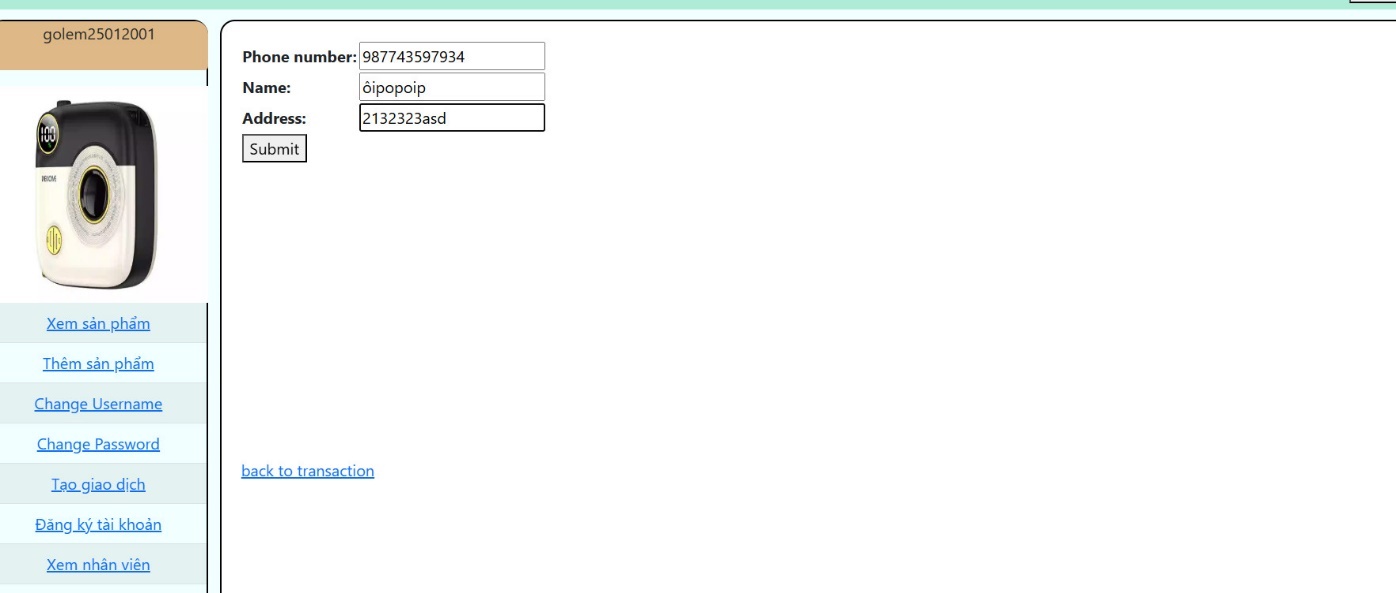


- Sau khi đã đăng ký đơn hàng đầy đủ ấn xác nhận để tiếp tục giao dịch và dẫn đến trang cung cấp thông tin khách hàng.



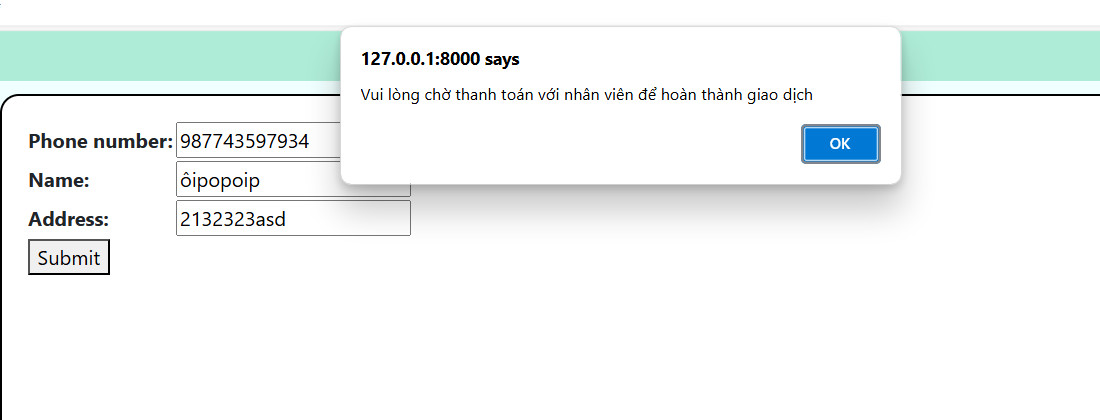


- Thông tin khách hàng nếu chưa có trong database thì sẽ tự động được cập nhập.

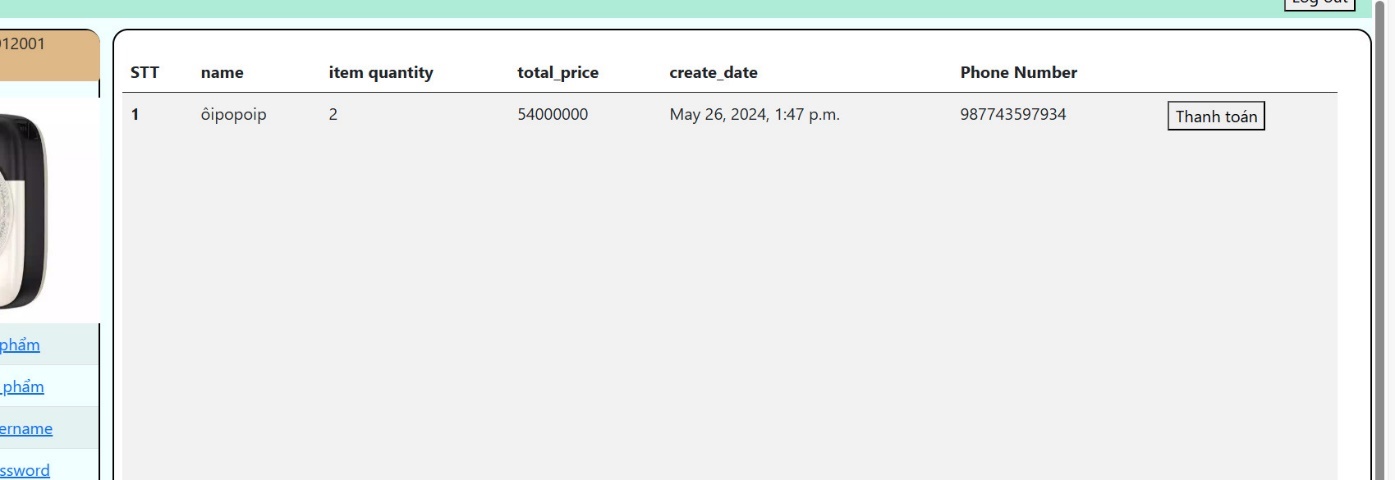


- Ấn submit để thực hiện tạo đơn hàng hoặc back to transaction để quay về.

- Thông báo tạo đơn thành công sẽ tự chuyển đến trang xem thanh toán.



- Đây là trang đơn hàng của người mua gồm những thông tin: STT, Name, Item quantity, Total price, Create date, Phone number.

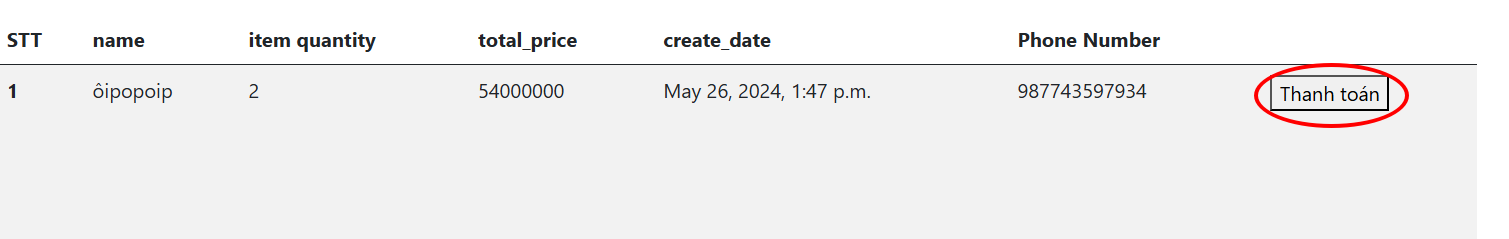


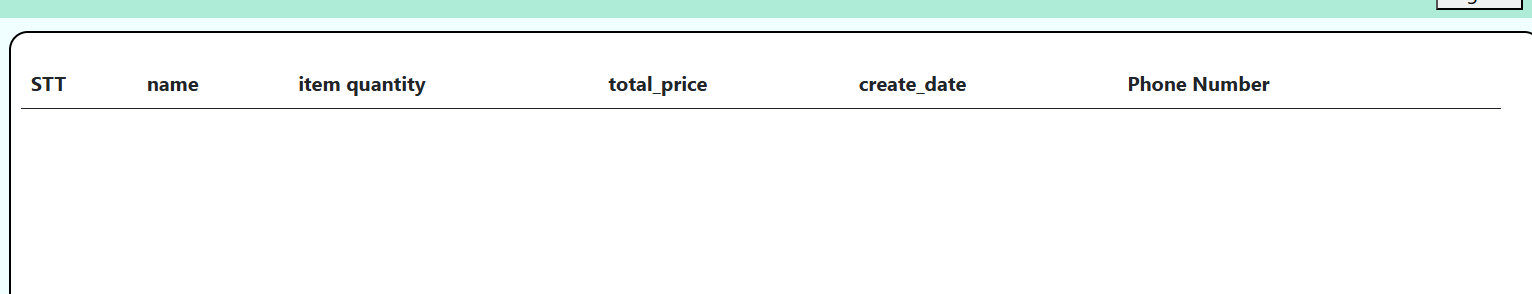
- Kiểm tra thành công thì sẽ ấn thanh toán để hoàn thành quá trình giao dịch.

- Lưu ý trong qua trình tạo đơn mà không thanh toán sẽ tự tạo 1 object trong bộ dữ liệu nhưng sẽ được set là chưa thanh toán.

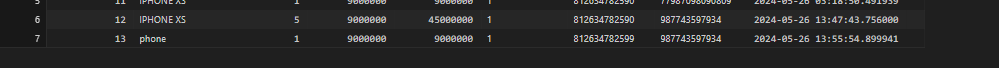
- Trong vòng 10p nếu ko thực hiện thanh toán đơn sẽ bị xóa.

- Sau khi ấn thanh toán.



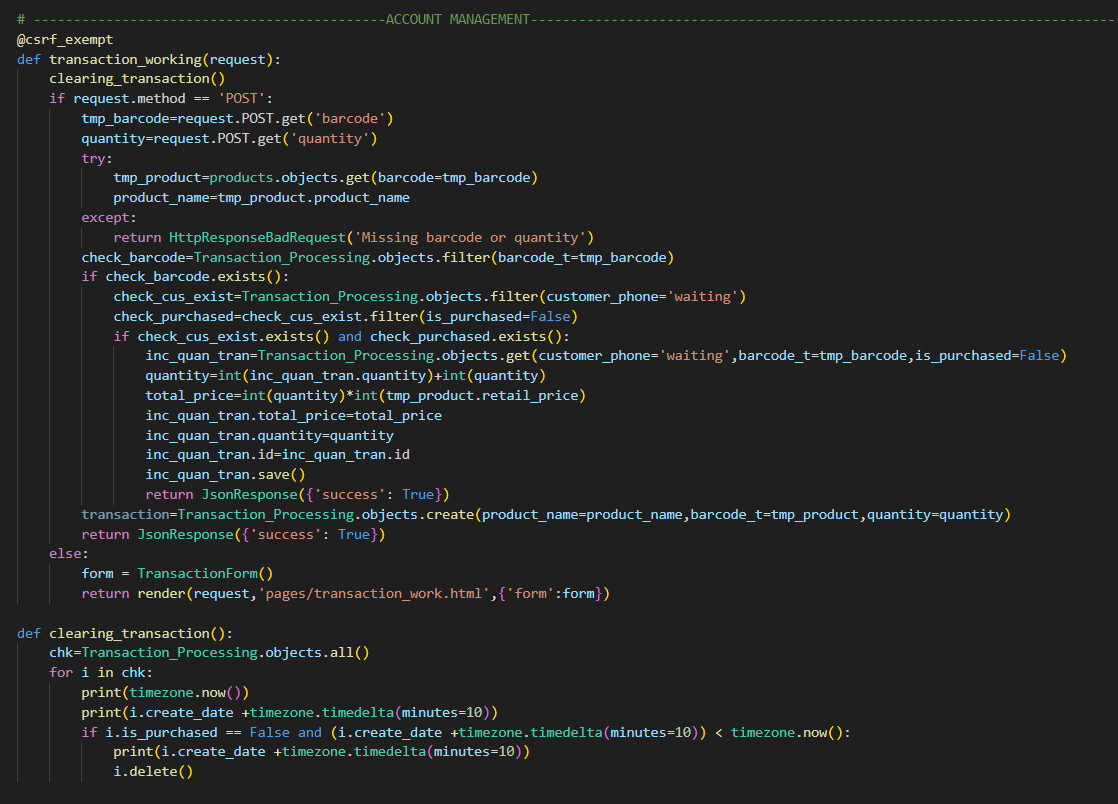


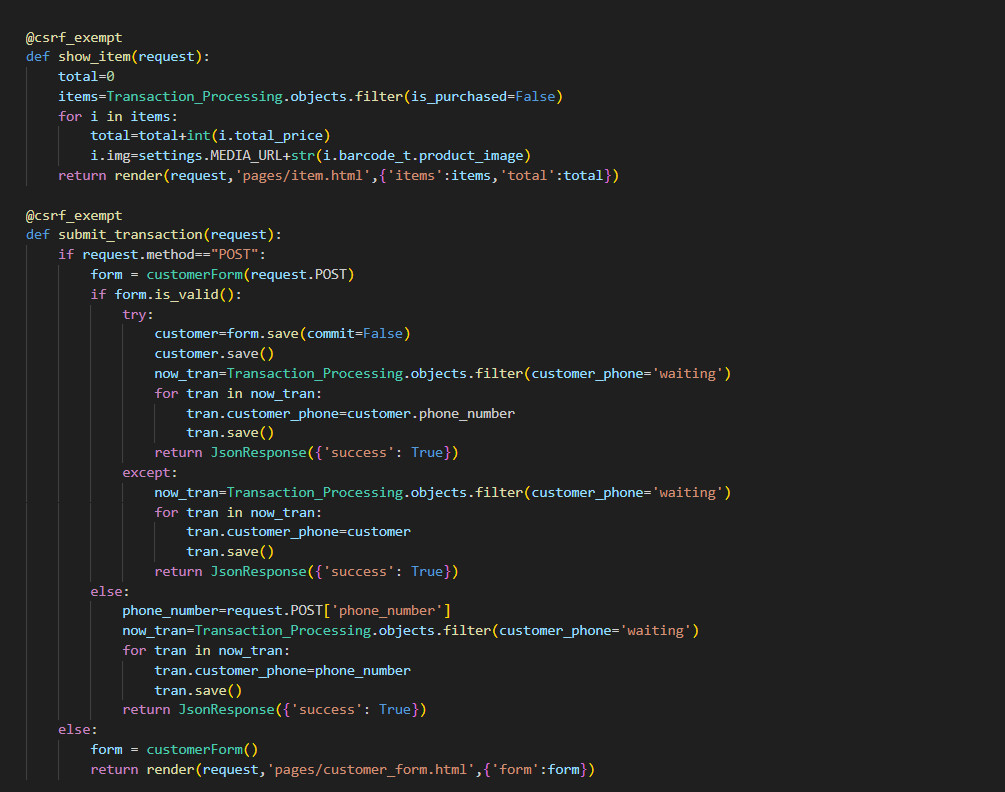
- Đây là dữ liệu về đơn hàng vừa thực hiện.

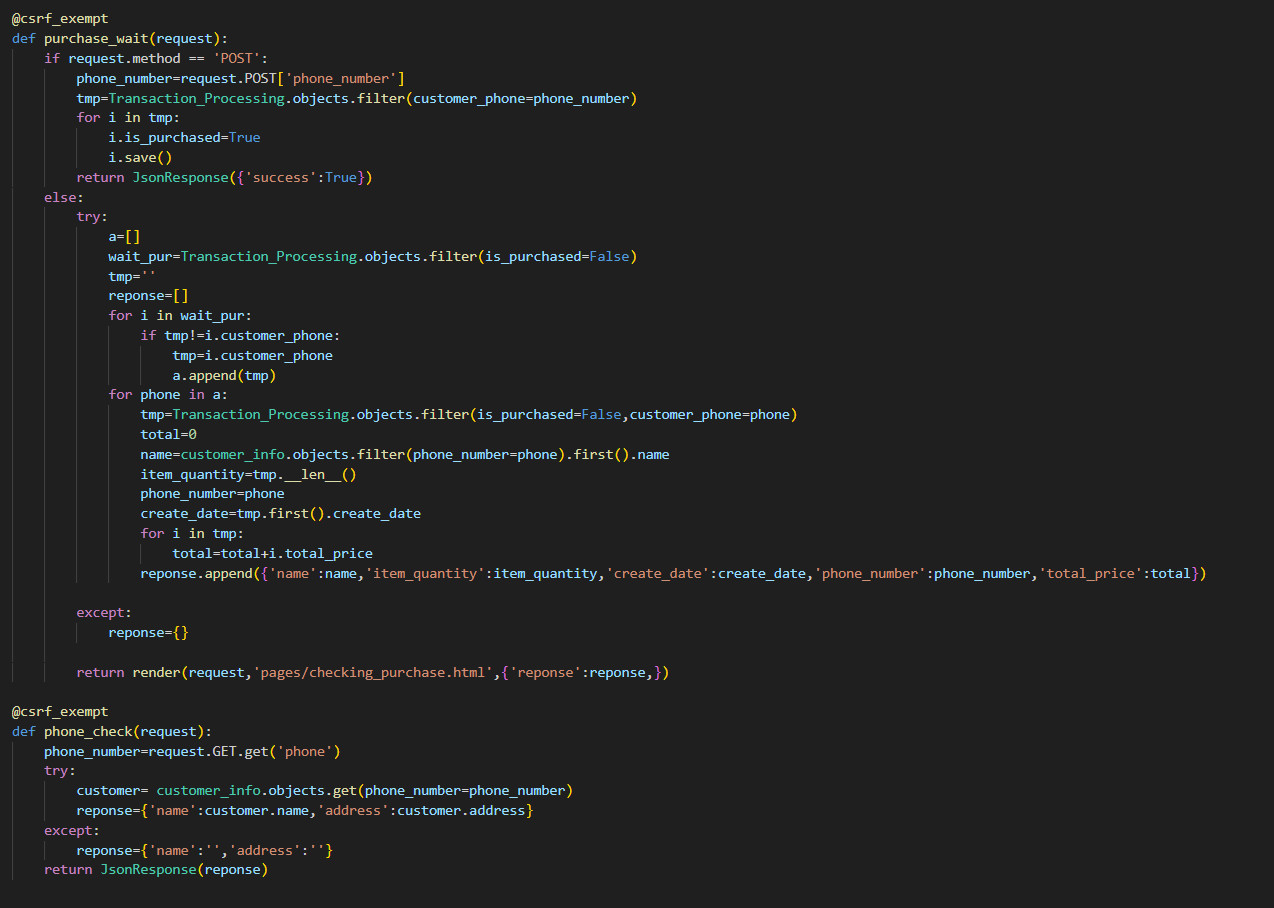
- Vì khách hàng chưa có trong database nên sẽ được thêm vào.



### 5.9.1 Xử lí code chính







# CHƯƠNG 6 – ERD

