**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------

A red circle with a star and a book

Description automatically generated

**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “ Xây dựng website bán đồng hồ”**

**GV hướng dẫn : Ths. Huỳnh Trung Trụ**

**Sinh viên thực hiện : LÂM NGỌC TÚ**

**Mã số sinh viên : N19DCCN174**

**Sinh viên thực hiện : HUỲNH NGÔ ANH THÁI**

**Mã số sinh viên : N19DCCN188**

**Khoá** **: 2019-2024**

**Hệ** **: ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 11/2023**

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

-----------------------------

A red circle with a star and a book

Description automatically generated

**BÁO CÁO THỰC TẬP**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “ Xây dựng website bán đồng hồ”**

**GV hướng dẫn : Ths. Huỳnh Trung Trụ**

**Sinh viên thực hiện : LÂM NGỌC TÚ**

**Mã số sinh viên : N19DCCN174**

**Sinh viên thực hiện : HUỲNH NGÔ ANH THÁI**

**Mã số sinh viên : N19DCCN188**

**Khoá** **: 2019-2024**

**Hệ** **: ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 11/2023**

# LỜI CẢM ƠN

Đầu tiên, em xin gửi lời cảm ơn chân thành đến tất cả thầy cô đã giảng dạy và cho em những kiến thức vô cùng quan trọng và quý báo trong quá trình học tập tại Học viện để chúng em có những kỹ năng và những kiến thức cần thiết để hoàn thành đề tài tốt nghiệp này.

Cảm ơn tất cả những người bạn đã ít nhiều cho mình những kiến thức và trải nghiệm đáng nhớ, luôn đồng hành và sát cánh trong quá trình học tập và rèn luyện các kỹ năng. Quan trọng hơn hết là động lực và niềm vui mà mình chắc chỉ có các bạn mới có thể mang lại. Nhờ vậy, mình đã có thể vượt qua những khó khăn và áp lực trong học tập cũng như trong cuộc sống.

Báo cáo đã được em hoàn thành đúng tiến độ và đảm bảo nội dung của đề tài. Mặc dù, em đã cố gắng hoàn thiện đề tài với tất cả sự cố gắng nhưng không tránh khỏi những sai phạm không mong muốn. Mong thầy giúp em đóng góp ý kiến để em có nhiều hơn những kinh nghiệm và kỹ năng để hoàn thành tốt công tác nghiên cứu, làm việc sau này. Một lần nữa em xin cảm ơn thầy và các bạn rất nhiều.

# **MỤC LỤC**

[**LỜI CẢM ƠN 14**](#_mqla3aue6bjz)

[**MỤC LỤC 14**](#_nql6xvlokhfg)

[**MỤC LỤC BẢNG 16**](#_8ujxu0vi0nd3)

[**MỤC LỤC HÌNH 17**](#_kehelru4gq6r)

[**KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT 19**](#_ffe8akaawton)

[**CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1**](#_30j0zll)

[1.1 Tìm hiểu “Xây dựng website bán đồng hồ”: 1](#)

[1.2Nền tảng công nghệ: 1](#)

[1.3Công cụ lập trình và công nghệ được sử dụng: 1](#)

[1.4 Giới thiệu về NodeJS 1](#)

[1.5 Giới thiệu về SQLite 2](#)

[1.6 Các thư viện – framework hỗ trợ: 2](#)

[1.6.1 NestJS 2](#)

[1.6.2 RESTful Web Service 3](#)

[1.6.3 React 3](#)

[1.6.4 Tailwind 4](#_7voxm716fyud)

[**CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG 5**](#_1fob9te)

[2.1. Hiện trạng thực tế 5](#)

[2.2 Giải pháp nâng cao hiệu quả 5](#)

[2.3 Mô hình nghiệp vụ của hệ thống: 5](#)

[2.3.1 Sơ đồ cơ cấu tổ chức: 5](#)

[2.3.2 Chức năng từng bộ phận 5](#)

[2.3.3 Quy trình nghiệp vụ 5](#)

[2.4 Yêu cầu chức năng: 6](#)

[**CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 7**](#_3znysh7)

[3.1 Usecase Tổng quát 7](#_9bp4wo1p6fpi)

[3.2 Usecase 7](#_2j6d7tnx1az4)

[3.2.1. Usercase Quản lý giỏ hàng 7](#_2nlmif1jvg4p)

[3.2.2. Quản lý thông tin cá nhân 9](#_dzu1es1kmpbf)

[3.2.3. Usercase xem sản phẩm 12](#_oywa7r5bs6m3)

[3.2.4. Usercase đăng ký 15](#_ps8fuvlnekst)

[3.2.5. Quản lý đơn hàng 18](#_lefs2cx5kfnz)

[3.2.6. Quản lý sản phẩm 20](#_622p10fluznc)

[3.2.7.Usercase Quản lý tài khoản 24](#_mi6ys2ppz3xw)

[3.2.8. Usercase đăng nhập 27](#_yuk49ttosd4g)

[3.2.9. Usercase đặt hàng 30](#_rr7hxal7tqjy)

[3.2.10. xem lịch sử đơn hàng 33](#_tkzu26yvudyj)

[3.3. Lược đồ bảng 35](#_cod78gfzi2r0)

[Bảng 3.1 database sản phẩm 35](#_mz3wzhn58t1m)

[Bảng 3.2 database người dùng 35](#_1xlq4krpvo7i)

[Bảng 3.3 database giỏ hàng 36](#_pxintnsd18zc)

[Bảng 3.4 database đặt hàng 36](#_pe8psxk54ea9)

[Bảng 3.5 database sản phẩm trong đơn hàng 36](#_rvr3wk1hi2a1)

[**CHƯƠNG 4: CHƯƠNG TRÌNH 37**](#_3dy6vkm)

[4.1. Giao diện dành cho nhóm khách hàng 37](#_su4e6rmya4vv)

[4.2 Giao diện dành cho nhóm Nhân viên 46](#_2mvtsas6nymp)

[4.3 Giao diện dành cho nhóm quản lý 49](#_5x3npkxriqay)

[**KẾT LUẬN 51**](#_1t3h5sf)

[**DANH MUC TÀI LIỆU THAM KHẢO 51**](#_4d34og8)

# MỤC LỤC BẢNG

# 

[Bảng 3.1 database sản phẩm 35](#_mz3wzhn58t1m)

[Bảng 3.2 database người dùng 35](#_1xlq4krpvo7i)

[Bảng 3.3 database giỏ hàng 36](#_pxintnsd18zc)

[Bảng 3.4 database đặt hàng 36](#_pe8psxk54ea9)

[Bảng 3.5 database sản phẩm trong đơn hàng 36](#_rvr3wk1hi2a1)

# 

# 

# 

# MỤC LỤC HÌNH

# 

[Hình 2.1 sơ đồ tổ chức 5](#_jj4awm56o0s)

[Hình 3.1 usace tổng quát 7](#_gowvsivq5ut)

[Hình 3.1 Usecase Quản lý giỏ hàng 7](#_in8bvxyjgn8e)

[Hình 3.2 Tuần tự quản lý giỏ hàng 8](#_421dz79rquuj)

[Hình 3.3 Usecase Quản lý thông tin cá nhân 10](#_fx2ld0h569ff)

[Hình 3.4 Tuần tự thay đổi thông tin cá nhân 11](#_fg766l9ct7if)

[Hình 3.5 Usecase xem chi tiết sản phẩm 12](#_6hoo73gtfkhc)

[Hình 3.6 Tuần tự xem sản phẩm 13](#_rmo5q3gnvmy0)

[Hình 3.7 Usecase Đăng ký 15](#_n8qy426mn1rb)

[Hình 3.8 Tuần tự đăng ký 16](#_1qgeg2sdyuhs)

[Hình 3,9 Usecase quản lý đơn hàng 18](#_vlfrr6keoa1n)

[Hình 3.10 Tuần tự Quản lý đơn hàng 19](#_3ltqoh2oamjo)

[Hình 3.11 Usecase quản lý sản phẩm 21](#_5x36ybs2lsa6)

[Hình 3.12 Tuần tự quản lý sản phẩm 22](#_jtlrt1p73xf9)

[Hình 3.13 Usecase Quản lý tài khoản 24](#_5v7nr7g9vi5q)

[Hình 3.14 Tuần tự quản lý tài khoản 25](#_yxij0v6kjbry)

[Hình 3.15 Usecase đăng nhập 27](#_k90unimvnkk7)

[Hình 3.16 Tuần tự đăng nhập 28](#_dlmu642kfc2k)

[Hình 3.17 Usecase đặt hàng 30](#_pqovl0jyqub6)

[Hình 3.18 Tuần tự đặt hàng 31](#_eucaifp5sxro)

[Hình 3.19 Usecase lịch sử đơn hàng 33](#_lt3vvt9s4neu)

[Hình 3.20 Tuần tự lịch sử đơn hàng 33](#_568nthqmt4sg)

[Hình 4.1 Trang chủ website 37](#_22ubf1vyb1hr)

[Hình 4.2 Giao diện trang sản phẩm 38](#_y30oz3rvi1a0)

[Hình 4.3 Giao diện chi tiết sản phẩm 39](#_oitgxsbkqg8u)

[Hình 4.4 Giao diện thêm sản phẩm chưa đăng nhập 40](#_ncskiop243dp)

[Hình 4.5 Giao diện thêm sản phẩm đã đăng nhập 41](#_9tvuaq4aiz0r)

[Hình 4.6 Giao diện đăng nhập thành công 42](#_n31hn98x0q3c)

[Hình 4.7 Giao diện đăng ký thất bại 43](#_xa9zt6cpp9hc)

[Hình 4.8 Giao diện giỏ hàng 44](#_kyoshq5tb5is)

[Hình 4.9 Giao diện cập nhật thông tin cá nhân 45](#_3fdih9o12ind)

[Hình 4.10 Giao diện lịch sử đơn hàng 45](#_lwh6bt5oxs0c)

[Hình 4.11 Giao diện chi tiết đơn hàng 46](#_fck71avaf3hp)

[Hình 4.12 Giao diện quản lý sản phẩm 47](#_bbv839fv5fkt)

[Hình 4.13 Giao diện thêm sản phẩm 47](#_rdqjua9mv3cu)

[Hình 4.14 Giao diện cập nhật sản phẩm 48](#_4mq5nwyd76rq)

[Hình 4.15 Giao diện quản lý đơn hàng 48](#_9ttg0vf0su8q)

[Hình 4.16 Giao diện cập nhật đơn hàng 49](#_q2gzq5jjegjx)

[Hình 4.17 Giao diện quản lý tài khoản 49](#_p9xyzjxy7orw)

[Hình 4.17 Giao diện phần quyền 50](#_o34zuuv2b0ad)

# 

# 

# KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Tiếng anh** | **Giải nghĩa** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**MỞ ĐẦU**

# CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

## 1.1 Tìm hiểu “Xây dựng website bán đồng hồ”:

Trong thời đại công nghệ 4.0, việc kinh doanh trực tuyến và xây dựng các trang web thương mại điện tử đã trở nên ngày càng quan trọng và phổ biến. Một trong những lĩnh vực thúc đẩy sự phát triển này là kinh doanh đồng hồ trực tuyến. Cùng với sự thịnh hành của việc mua sắm trực tuyến, nhiều người dùng trên khắp thế giới đang tìm kiếm các trang web chuyên cung cấp đồng hồ với sự đa dạng về mẫu mã và chất lượng.

Xây dựng một trang web bán đồng hồ có thể mang lại nhiều cơ hội kinh doanh, nhưng đòi hỏi sự đầu tư về thời gian, kiến thức và tài chính để đạt được sự thành công mong muốn.

## 1.2Nền tảng công nghệ:

Client: Sử dụng React, javascript, css, tailwind, html,…

Server: Sử dụng NestJS, typescript, nodejs.

## 1.3Công cụ lập trình và công nghệ được sử dụng:

Hệ quản trị cơ sở dữ liệu: MySQL Server Workbench.

Công cụ lập trình: Visual Studio Code.

Ngôn ngữ lập trình: NodeJS.

Xây dựng client: HTML, CSS, Tailwind, Javascript.

Xây dựng server: NodeJS, Typescript

Framework NestJS.

Xây dựng web với React.

## 1.4 Giới thiệu về NodeJS

Node.js là một môi trường chạy mã JavaScript phía máy chủ (server-side) được xây dựng dựa trên JavaScript Engine V8 của Google. Nó cho phép bạn thực hiện các tác vụ máy chủ, xây dựng ứng dụng web và ứng dụng mạng khác mà sử dụng JavaScript làm ngôn ngữ lập trình. Dưới đây là một số điểm quan trọng cần biết về Node.js:

* JavaScript Everywhere: Node.js cho phép bạn sử dụng JavaScript cả ở phía máy chủ và phía client (trình duyệt web). Điều này làm giảm sự phân chia giữa code backend và frontend và tạo điều kiện thuận lợi cho việc chia sẻ mã nguồn và kiến thức giữa các phần của dự án.
* Single-Threaded, Non-Blocking: Node.js sử dụng mô hình chạy đơn luồng (single-threaded), không chặn (non-blocking), cho phép xử lý nhiều yêu cầu đồng thời mà không gây chậm trễ. Điều này đạt được bằng cách sử dụng sự kiện (event) và các luồng nhẹ (lightweight threads) để xử lý yêu cầu.
* Nhiều Mô-đun và Thư Viện: Node.js có một cộng đồng lớn và phong phú, với nhiều mô-đun và thư viện có sẵn qua npm (Node Package Manager). Điều này giúp bạn tận dụng được nhiều tính năng và chức năng mà không cần phải xây dựng từ đầu.
* Phát triển Nhanh Chóng: Node.js thúc đẩy phát triển ứng dụng nhanh chóng nhờ vào quá trình phát triển chung giữa client và server và sử dụng JavaScript, ngôn ngữ phổ biến và dễ học.
* Phù hợp cho ứng dụng thời gian thực: Node.js thường được sử dụng trong các ứng dụng yêu cầu xử lý dữ liệu thời gian thực, chẳng hạn như ứng dụng trò chơi trực tuyến, ứng dụng chat trực tuyến, ứng dụng xử lý sự kiện thời gian thực, v.v.
* Cộng Đồng Mạnh Mẽ: Node.js có một cộng đồng lớn và nhiệt tình, với nhiều tài liệu, ví dụ và hỗ trợ trực tuyến. Điều này giúp bạn giải quyết vấn đề nhanh chóng và tìm hiểu kiến thức mới.
* Cross-Platform: Node.js hỗ trợ nhiều hệ điều hành, giúp bạn triển khai ứng dụng trên nhiều nền tảng khác nhau.

Node.js đã trở thành một công cụ phổ biến trong phát triển ứng dụng web và ứng dụng máy chủ. Nó cho phép bạn xây dựng các ứng dụng hiệu suất cao và phản hồi nhanh với sự linh hoạt trong việc xử lý yêu cầu và sự đồng nhất trong việc sử dụng JavaScript.

## 1.5 Giới thiệu về SQLite

SQLite là hệ quả trị cơ sở dữ liệu (DBMS) quan hệ tương tự như Mysql, ... Đặc điểm nổi bật của SQLite so với các DBMS khác là gọn, nhẹ, đơn giản, đặt biệt không cần mô hình server-client, không cần cài đặt, cấu hình hay khởi động nên không có khái niệm user, password hay quyền hạn trong SQLite Database. Dữ liệu cũng được lưu ở một file duy nhất.

SQLite thường không được sử dụng với các hệ thống lớn nhưng với những hệ thống ở quy mô vùa và nhỏ thì SQLite không thua các DBMS khác về chức năng hay tốc độ. Vì không cần cài đặt hay cấu hình nên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm … vì tránh được những rắc rối trong quá trình cài đặt.

## 1.6 Các thư viện – framework hỗ trợ:

### 1.6.1 NestJS

NestJS là một khung (framework) phát triển ứng dụng server-side được xây dựng trên nền tảng Node.js. NestJS là một lựa chọn mạnh mẽ cho việc xây dựng các ứng dụng web và API đa dạng, đặc biệt là các ứng dụng có tính phức tạp. Dưới đây là một số điểm quan trọng về NestJS:

* Kiến trúc Module-Based: NestJS sử dụng kiến trúc module để tạo ứng dụng. Điều này giúp tạo ra mã nguồn sạch sẽ và dễ quản lý, cho phép bạn chia nhỏ ứng dụng thành các module riêng biệt có thể được kết hợp lại với nhau.
* Sử dụng TypeScript: NestJS được viết bằng TypeScript, một biến thể của JavaScript với kiểu dữ liệu tĩnh. Điều này giúp tạo ra mã nguồn an toàn hơn và giúp bạn tránh được nhiều lỗi thường gặp trong quá trình phát triển.
* Dependency Injection: NestJS sử dụng mô hình Dependency Injection (DI) để quản lý các thành phần của ứng dụng. Điều này giúp bạn dễ dàng quản lý và tái sử dụng mã nguồn, cũng như tạo các đơn vị kiểm tra đơn lẻ (unit tests).
* HTTP Framework mạnh mẽ: NestJS cung cấp một HTTP framework mạnh mẽ cho việc xây dựng các API RESTful hoặc GraphQL. Bạn có thể định nghĩa các điểm cuối (endpoints) và xử lý các yêu cầu HTTP một cách dễ dàng.
* Middleware và Pipes: NestJS cho phép bạn sử dụng middleware và pipes để xử lý yêu cầu trước khi chúng đến đích. Điều này hữu ích để thực hiện xác thực, xử lý lỗi, và kiểm tra dữ liệu trước khi nó được xử lý bởi các bộ điều khiển (controllers).
* WebSockets và Microservices: NestJS hỗ trợ cả WebSockets và microservices, cho phép bạn xây dựng các ứng dụng thời gian thực và phân tán.
* Cộng đồng lớn và Tài liệu phong phú: NestJS có một cộng đồng lớn và đa dạng, với nhiều tài liệu, ví dụ và thư viện hỗ trợ. Điều này giúp bạn giải quyết vấn đề nhanh chóng và tìm hiểu kiến thức mới.
* Hỗ trợ cho TypeORM và Mongoose: NestJS tích hợp dễ dàng với các ORM (Object-Relational Mapping) như TypeORM cho SQL databases và Mongoose cho MongoDB.

NestJS là một công cụ mạnh mẽ để xây dựng các ứng dụng phía máy chủ hiệu suất cao và dễ bảo trì. Nó cung cấp cấu trúc và quy tắc tốt giúp bạn tạo ra mã nguồn dễ đọc và bảo trì, và nó phù hợp cho nhiều loại dự án từ nhỏ đến lớn.

### 1.6.2 RESTful Web Service

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng

dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú

trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu

động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua

HTTP.

Diễn giải các thành phần:

* API (Application Programming Interface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML.
* REST (Representational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.
* RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.
* Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API. JavaScript là một ngôn ngữ lập trình của HTML và WEB. Nó là nhẹ và được sử dụng phổ biến nhất như là một phần của các trang web, mà sự thi hành của chúng cho phép Client-Side script tương tác với người sử dụng và tạo các trang web động. Nó là một ngôn ngữ chương trình thông dịch với các khả năng hướng đối tượng

### 1.6.3 React

React là một thư viện JavaScript phía người dùng (frontend) được phát triển bởi Facebook. Nó là một trong những công cụ phổ biến nhất để xây dựng giao diện người dùng (UI) tương tác trên web và ứng dụng di động. Dưới đây là một số điểm quan trọng về React:

* Thư viện JavaScript cho UI: React tập trung vào việc xây dựng giao diện người dùng (UI) và quản lý sự thay đổi của nó khi dữ liệu thay đổi. Nó không phải là một framework hoàn chỉnh, cho phép bạn tự do lựa chọn các thư viện và công cụ bổ sung.
* Công nghệ Virtual DOM: Một trong những đặc điểm nổi bật của React là Virtual DOM (DOM ảo). React tạo ra một bản sao ảo của DOM và so sánh nó với DOM thực sự. Khi có sự thay đổi, React cập nhật chỉ những phần thay đổi thực sự trên trang web thay vì cập nhật toàn bộ DOM, điều này làm tăng hiệu suất ứng dụng.
* Component-Based: React sử dụng mô hình phát triển dựa trên thành phần (component-based development). Bạn có thể tạo ra các component độc lập, tái sử dụng chúng và kết hợp chúng lại với nhau để xây dựng giao diện người dùng phức tạp.
* React Native: React Native là một phần mở rộng của React, cho phép bạn xây dựng ứng dụng di động bằng JavaScript và React. Điều này giúp bạn chia sẻ mã nguồn và logic kinh doanh giữa ứng dụng web và ứng dụng di động.
* Học dễ dàng: React có một cộng đồng lớn và nhiều tài liệu học tập, giúp người mới học và các nhà phát triển có kinh nghiệm dễ dàng tiếp cận và sử dụng nó.
* Hỗ trợ công cụ phát triển: React có nhiều công cụ phát triển mạnh mẽ như React DevTools để debug và tối ưu hóa ứng dụng, Redux cho quản lý trạng thái ứng dụng, và nhiều thư viện và framework bên ngoài khác để giúp bạn xây dựng ứng dụng phức tạp hơn.
* Cộng đồng mạnh mẽ: React có một cộng đồng lớn và nhiệt tình, với nhiều người dùng chia sẻ tài liệu, ví dụ, và góp phần phát triển các dự án liên quan đến React.

React đã trở thành một công cụ quan trọng trong việc phát triển giao diện người dùng cho nhiều ứng dụng web và di động. Nó được sử dụng bởi nhiều công ty lớn và tổ chức để xây dựng các sản phẩm và dự án công nghiệp.

### 1.6.4 Tailwind

Tailwind CSS là một thư viện CSS nhẹ và mạnh mẽ được thiết kế để giúp bạn xây dựng giao diện người dùng (UI) nhanh chóng và dễ dàng. Thay vì tạo ra các lớp CSS tùy chỉnh riêng, Tailwind CSS cung cấp một tập hợp các lớp CSS đã được định nghĩa trước, giúp bạn tạo ra giao diện tùy chỉnh bằng cách áp dụng các lớp này trực tiếp trong mã HTML.

Dưới đây là một số điểm quan trọng về Tailwind CSS:

* Lớp CSS Utility: Tailwind CSS cung cấp một loạt các lớp CSS như bg-blue-500, text-white, p-4, và nhiều lớp khác. Mỗi lớp thực hiện một tính năng cụ thể, chẳng hạn như màu nền, màu văn bản, khoảng cách lề, và tính căn giữa.
* Cấu hình Tùy chỉnh: Bạn có thể tùy chỉnh Tailwind CSS bằng cách chỉnh sửa tệp cấu hình (configuration file) để đáp ứng nhu cầu cụ thể của dự án. Bạn có thể thêm, sửa đổi hoặc xóa các lớp CSS, màu sắc, kích thước, và nhiều hơn nữa.
* Dễ Học và Sử Dụng: Tailwind CSS dễ học và sử dụng, đặc biệt đối với các nhà phát triển mới học CSS. Mỗi lớp CSS có tên rất mô tả, giúp bạn dễ dàng hiểu chức năng của nó.
* Tích hợp tốt với Frameworks và Thư viện Khác: Tailwind CSS có thể tích hợp tốt với các framework JavaScript như React, Vue.js, và Angular. Nó cũng hoạt động tốt với các thư viện UI như Bootstrap và Material-UI.
* Tích hợp với Build Tools: Tailwind CSS có thể được tích hợp vào các công cụ xây dựng (build tools) như Webpack và Laravel Mix để tối ưu hóa và biên dịch CSS của bạn.
* Cộng Đồng Lớn: Tailwind CSS có một cộng đồng lớn, với nhiều tài liệu, ví dụ, và plugins được tạo bởi cộng đồng. Điều này giúp bạn tìm kiếm giải pháp cho các vấn đề cụ thể và học hỏi từ người khác.

Tailwind CSS đã trở thành một công cụ phổ biến trong việc xây dựng giao diện người dùng hiện đại và tùy chỉnh. Nó giúp tiết kiệm thời gian và tăng nhanh quá trình phát triển front-end của dự án, đặc biệt là cho các nhà phát triển không chuyên về CSS.

# CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG

## 2.1. Hiện trạng thực tế

Nhu cầu khách hàng phân bố không đều tại các khu vực khác nhau đã tạo ra một số thách thức đối với cả cửa hàng đồng hồ truyền thống và các trang web bán đồng hồ. Cửa hàng đồng hồ truyền thống thường phải đối mặt với các chi phí cao về mặt bằng và nhân sự do phải duy trì nhiều vị trí vật lý. Điều này đồng nghĩa với việc hạn chế sự tiếp cận của khách hàng và gây bất lợi cho họ khi muốn mua sản phẩm tại nhà. Quản lý hàng tồn kho và đơn hàng cũng trở nên phức tạp và thường dẫn đến khó khăn trong việc đánh giá khách hàng một cách khách quan.

## 2.2 Giải pháp nâng cao hiệu quả

|  |  |
| --- | --- |
| **Vấn đề** | **Giải pháp** |
| Khó khăn trong việc theo dõi lượng hàng tồn kho và quản lý đơn đặt hàng | Quản lý hàng tồn kho và đơn đặt hàng bằng cách sử dụng phần mềm quản lý kho và hệ thống quản lý đơn đặt hàng trực tuyến. |
| Gặp khó khăn trong việc quản lý đánh giá và phản hồi từ khách hàng | Cung cấp khả năng cho khách hàng đánh giá sản phẩm và dịch vụ trên trang web. Theo dõi những đánh giá này và phản hồi khách hàng để cải thiện chất lượng sản phẩm và dịch vụ. |
| Không theo dõi được hoạt động của các tài khoản | Xây dựng hệ thống có thể ghi log trên các thao tác đặc biệt |

## 2.3 Mô hình nghiệp vụ của hệ thống:

### 2.3.1 Sơ đồ cơ cấu tổ chức:

A white rectangle with black text

Description automatically generated

#### Hình 2.1 sơ đồ tổ chức

### 2.3.2 Chức năng từng bộ phận

Quản lý: Quản lý trực tiếp website bán đồng hồ, mọi vấn đề của website điều phải thông qua quản lý và quản lý có quyền quyết định mọi thông tin trong cửa hàng.

Nhân viên: Quản lý sản phẩm, quản lý đơn hàng, quản lý kho, quản lý đánh giá

Khách hàng: Quản lý giỏ hàng, xem lịch sử đơn hàng, đặt hàng, đánh giá sản phẩm

### 2.3.3 Quy trình nghiệp vụ

Nghiệp vụ bán hàng

* Xem sản phẩm
* Quản lý giỏ hàng
* Đặt hàng

Nghiệp vụ quản lý sản phẩm

* Xem danh sách sản phẩm
* Thêm, xóa, sửa sản phẩm

Nghiệp vụ quản lý đơn hàng

* Xem danh sách đơn hàng
* Xem chi tiết đơn hàng
* Cập nhật trạng thái đơn hàng

Nghiệp vụ quản lý kho

* Cập nhật số lượng sản phẩm khi nhập hàng

Nghiệp vụ quản lý tài khoản

* Phân quyền
* Chỉnh sửa tài khoản

Nghiệp vụ tài khoản người dùng

* Đăng ký, đăng nhập, quên mật khẩu
* Thay đổi thông tin cá nhân và mật khẩu.
* Xem lịch sử mua hàng

## 2.4 Yêu cầu chức năng:

Hệ thống chạy đúng, tốt, ổn định.

Giao diện thân thiện, dễ sử dụng.

Đáp ứng hầu hết các yêu cầu của người dùng.

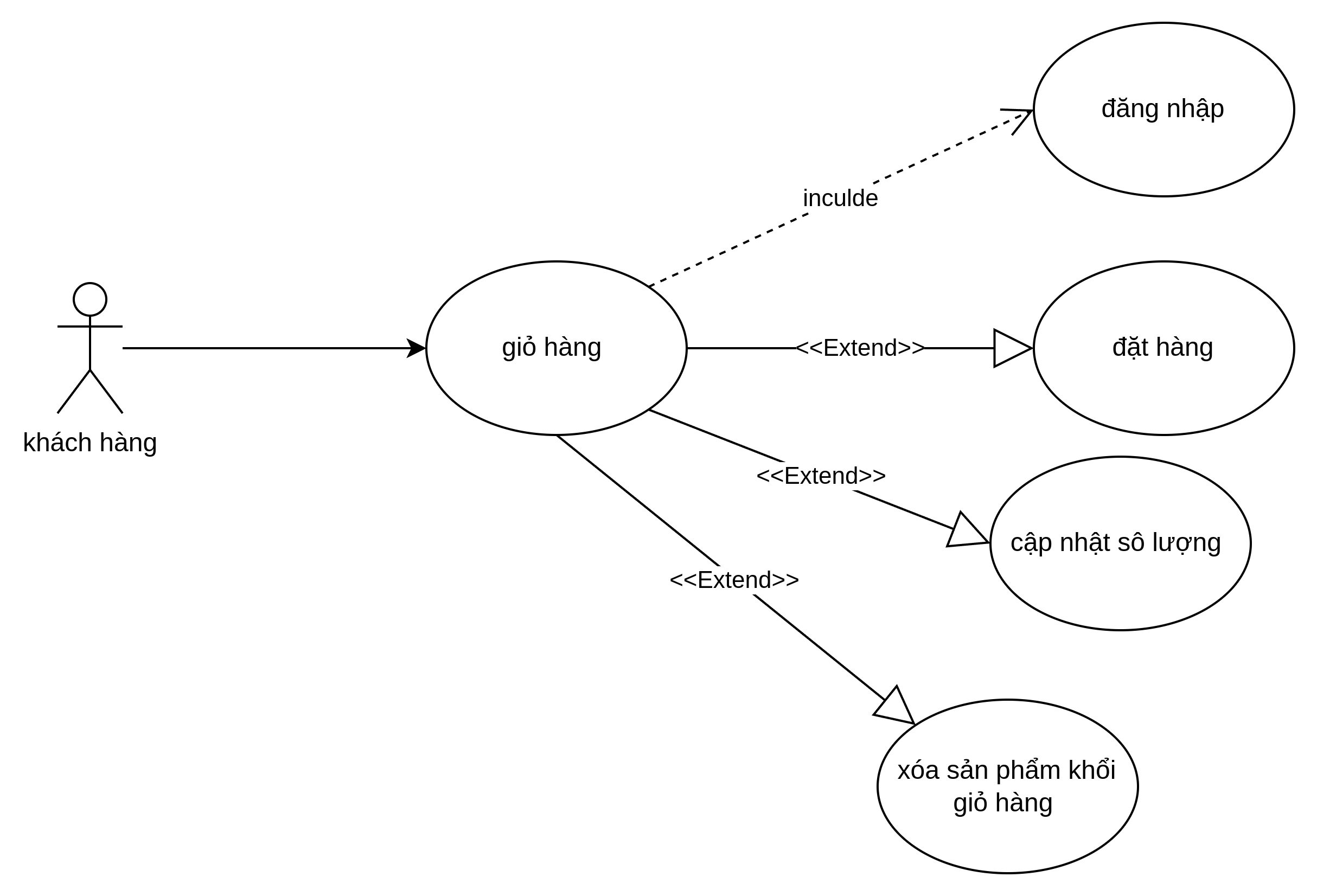
# CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 3.1 Usecase Tổng quát

#### Hình 3.1 usace tổng quát

## 3.2 Usecase

### 3.2.1. Usercase Quản lý giỏ hàng



#### Hình 3.1 Usecase Quản lý giỏ hàng

#### Hình 3.2 Tuần tự quản lý giỏ hàng

|  |  |
| --- | --- |
| Tên usecase | Quản lý giỏ hàng |
| Tác nhân chính | Khách hàng không có tài khoản, khách hàng có tài khoản |
| Điều kiện trước |  |
| Đảm bảo tối thiểu | Hệ thống hiển thị giao diện giỏ hàng với danh sách sản phẩm và thông tin chi tiết.  Tác nhân có khả năng thay đổi số lượng sản phẩm trong giỏ hàng và thực hiện đặt hàng. |
| Điều kiện sau |  |
| Chuỗi sự kiện chính:  1. Tác nhân truy cập vào chức năng "Quản lý giỏ hàng".  2. Hệ thống hiển thị giao diện giỏ hàng với danh sách sản phẩm đã được thêm vào giỏ hàng.  3. Tác nhân có khả năng thay đổi số lượng sản phẩm trong giỏ hàng bằng cách cập nhật số lượng tương ứng hoặc xóa sản phẩm khỏi giỏ hàng.  4. Tác nhân có khả năng thực hiện đặt hàng bằng cách chọn nhận đặt hàng.  5. Hệ thống yêu cầu nhập thông tin.  6. Hệ thống yêu cầu xác nhận và đặt hàng. | |

### 3.2.2. Quản lý thông tin cá nhân

#### Hình 3.3 Usecase Quản lý thông tin cá nhân

#### Hình 3.4 Tuần tự thay đổi thông tin cá nhân

|  |  |
| --- | --- |
| Tên usecase | Thay đổi thông tin cá nhân |
| Tác nhân chính | Khách hàng có tài khoản |
| Điều kiện trước | Khách hàng đã đăng nhập vào hệ thống. |
| Đảm bảo tối thiểu | Hệ thống hiển thị giao diện xem thông tin cá nhân.  Khách hàng có khả năng cập nhật thông tin cá nhân. |
| Điều kiện sau | Thông tin cá nhân đã được cập nhật. |
| Chuỗi sự kiện chính:  1. Khách hàng truy cập vào chức năng "Quản lý thông tin cá nhân".  2. Hệ thống hiển thị giao diện xem thông tin cá nhân hiện tại của khách hàng.  3. Khách hàng có khả năng cập nhật các trường thông tin cá nhân cần thiết  4. Sau khi hoàn thành việc cập nhật thông tin, khách hàng xác nhận để lưu thay đổi.  5. Hệ thống kiểm tra và cập nhật thông tin cá nhân. | |

### 3.2.3. Usercase xem sản phẩm

#### Hình 3.5 Usecase xem chi tiết sản phẩm

#### Hình 3.6 Tuần tự xem sản phẩm

|  |  |
| --- | --- |
| Tên usecase | Xem sản phẩm |
| Tác nhân chính | Tất cả |
| Điều kiện trước |  |
| Đảm bảo tối thiểu | Hệ thống hiển thị danh sách sản phẩm . |
| Điều kiện sau |  |
| Chuỗi sự kiện chính:  1. Người dùng truy cập vào trang sản phẩm  2. Hệ thông hiển thị danh sách sản phẩm  3. Người dùng có thể truy cập vào sản phẩm mông muốn để xem chi tiết.  4. Sau khi vào trang chi tiết sản phẩm khách hàng có thể thêm vào giỏ hàng  5. Hệ thống kiểm tra thông tin và thông báo thêm vào giỏ hàng  Ngoại lệ  1. Thông báo lối thêm vào giỏ hàng khi chưa đăng nhập và chuyển trang đăng nhập | |

### 3.2.4. Usercase đăng ký

#### Hình 3.7 Usecase Đăng ký

#### Hình 3.8 Tuần tự đăng ký

|  |  |
| --- | --- |
| Tên usecase | Đăng ký |
| Tác nhân chính | Khách hàng chưa có tài khoản |
| Điều kiện trước | Khách hàng chưa có tài khoản trong hệ thống. |
| Đảm bảo tối thiểu | Hệ thống hiển thị giao diện đăng ký.  Khách hàng có khả năng nhập các thông tin cần thiết để tạo tài khoản. |
| Điều kiện sau | Tài khoản khách hàng đã được tạo thành công. |
| Chuỗi sự kiện chính:  1. Khách hàng truy cập vào chức năng "Đăng ký".  2. Hệ thống hiển thị giao diện đăng ký với các trường thông tin cần thiết  3. Khách hàng nhập thông tin vào các trường cần thiết.  4. Sau khi hoàn thành việc nhập thông tin, khách hàng xác nhận để tiến hành đăng ký.  5. Hệ thống kiểm tra thông tin và tạo tài khoản cho khách hàng.  Ngoại lệ  1.Thông báo lỗi nếu đăng ký sai | |

### 3.2.5. Quản lý đơn hàng

#### Hình 3,9 Usecase quản lý đơn hàng

#### Hình 3.10 Tuần tự Quản lý đơn hàng

|  |  |
| --- | --- |
| Tên usecase | Quản lý đơn hàng |
| Tác nhân chính | Quản lý, nhân viên |
| Điều kiện trước | Tài khoản quản trị viên hoặc nhân viên đã đăng nhập vào hệ thống với quyền hạn tương ứng. |
| Đảm bảo tối thiểu | Hệ thống hiển thị danh sách đơn hàng.  Tác nhân (quản trị viên và nhân viên) có khả năng xem chi tiết đơn hàng.  Tác nhân có khả năng cập nhật trạng thái của đơn hàng. |
| Điều kiện sau | Trạng thái của đơn hàng đã được cập nhật. |
| Chuỗi sự kiện chính:  1. Tác nhân truy cập vào chức năng "Quản lý đơn hàng".  2. Hệ thống hiển thị danh sách đơn hàng hiện có với các thông tin cơ bản.  3. Tác nhân có khả năng chọn một đơn hàng từ danh sách để xem chi tiết.  4. Hệ thống hiển thị thông tin chi tiết của đơn hàng bao gồm thông tin khách hàng, sản phẩm và trạng thái hiện tại.  5. Tác nhân có khả năng cập nhật trạng thái của đơn hàng bằng cách chọn trạng thái mới từ danh sách.  6. Sau khi cập nhật, tác nhân xác nhận để lưu thay đổi.  7. Hệ thống kiểm tra và cập nhật trạng thái của đơn hàng. | |

### 3.2.6. Quản lý sản phẩm

#### Hình 3.11 Usecase quản lý sản phẩm

#### Hình 3.12 Tuần tự quản lý sản phẩm

|  |  |
| --- | --- |
| Tên usecase | Quản lý sản phẩm |
| Tác nhân chính | Quản lý, nhân viên |
| Điều kiện trước | Đăng nhập với quyền quản trị viên, nhân viên |
| Đảm bảo tối thiểu | Hệ thống hiển thị danh sách sản phẩm.  Tác nhân (quản trị viên và nhân viên) có khả năng thêm mới sản phẩm.  Tác nhân có khả năng chỉnh sửa thông tin của sản phẩm.  Tác nhân có khả năng xóa sản phẩm. |
| Điều kiện sau | Thông tin về sản phẩm đã được cập nhật. |
| Chuỗi sự kiện chính:  1. Tác nhân truy cập vào chức năng "Quản lý sản phẩm".  2. Hệ thống hiển thị danh sách sản phẩm hiện có với các thông tin cơ bản.  3. Tác nhân có khả năng thêm mới sản phẩm bằng cách nhập thông tin cần thiết.  4. Tác nhân có khả năng chọn một sản phẩm từ danh sách để chỉnh sửa thông tin.  5. Hệ thống hiển thị thông tin chi tiết của sản phẩm và cho phép tác nhân thực hiện các thay đổi cần thiết.  6. Tác nhân có khả năng xóa một sản phẩm khỏi danh sách.  7. Sau khi thực hiện các thay đổi (thêm mới, chỉnh sửa, xóa), tác nhân xác nhận để lưu thay đổi.  8. Hệ thống kiểm tra và cập nhật thông tin của sản phẩm. | |

### 3.2.7.Usercase Quản lý tài khoản

#### Hình 3.13 Usecase Quản lý tài khoản

#### Hình 3.14 Tuần tự quản lý tài khoản

|  |  |
| --- | --- |
| Tên usecase | Quản lý tài khoản |
| Tác nhân chính | Quản lý |
| Điều kiện trước | Tài khoản quản trị viên đã đăng nhập vào hệ thống với quyền hạn tương ứng. |
| Đảm bảo tối thiểu | Hệ thống hiển thị danh sách các tài khoản.  Quản trị viên có thể xem thông tin, cập nhật và thiết lập quyền hạn cho các tài khoản.  Hệ thống thực hiện thay đổi và cung cấp phản hồi. |
| Điều kiện sau | Dữ liệu tài khoản được cập nhật và quản lý |
| Chuỗi sự kiện chính:  1. Quản trị viên truy cập chức năng "Quản lý tài khoản".  2. Hệ thống hiển thị danh sách tài khoản gồm các thông tin cơ bản về mỗi tài khoản.  3. Quản trị viên chọn một tài khoản trong danh sách để xem chi tiết.  4. Hệ thống hiển thị thông tin chi tiết của tài khoản bao gồm tên, địa chỉ, email, và quyền hạn hiện tại.  5. Quản trị viên có khả năng cập nhật thông tin của tài khoản, bao gồm cả việc chỉnh sửa thông tin cá nhân và thiết lập quyền hạn mới.  6. Quản trị viên xác nhận để lưu các thay đổi.  7. Hệ thống kiểm tra và thực hiện cập nhật thông tin và quyền hạn của tài khoản.  8. Hệ thống trả về thông báo thành công và hiển thị thông tin đã được cập nhật. | |

### 

### 3.2.8. Usercase đăng nhập

#### Hình 3.15 Usecase đăng nhập

#### Hình 3.16 Tuần tự đăng nhập

|  |  |
| --- | --- |
| Tên use case | Đăng nhập |
| Tác nhân chính | Người dùng, Quản lý, Nhân Viên |
| Điều kiện trước |  |
| Đảm bảo tối thiểu | Người dùng nhập đầy đủ chính xác email và mật khẩu đã đăng ký trước đó. |
| Điều kiện sau | Đăng nhập thành công.  chuyển trang home |
| Chuỗi sự kiện chính:  1. Người dùng nhập đầy đủ và chính xác email và password để đăng nhập.  2. Người dùng nhấp chọn nút đăng nhập trên màn hình hoặc Enter.  3. Nếu thất bại, người dùng sẽ nhận được thông báo lỗi.  4. Nếu thành công, chuyển vào trang home. | |
| Ngoại lệ:   * Thông báo lỗi khi đăng nhập bị trống 1 trong tất cả các trường thông tin. | |

### 3.2.9. Usercase đặt hàng

#### Hình 3.17 Usecase đặt hàng

#### Hình 3.18 Tuần tự đặt hàng

|  |  |
| --- | --- |
| Tên use case | Đặt hàng |
| Tác nhân chính | khách hàng đã có tài khoản |
| Điều kiện trước | khách hàng đã đăng nhập vào hệ thống |
| Đảm bảo tối thiểu |  |
| Điều kiện sau | thông tin đặt hàng không được để trống |
| 1. Tác nhân truy cập vào chức năng "Đặt hàng".  2. nhập thông tin đặt hàng  3. thông báo nếu có lỗi  4. thông báo nếu đặt hàng thành công | |

### 3.2.10. xem lịch sử đơn hàng

#### Hình 3.19 Usecase lịch sử đơn hàng

#### Hình 3.20 Tuần tự lịch sử đơn hàng

|  |  |
| --- | --- |
| Tên use case | Lịch sử đơn hàng |
| Tác nhân chính | Khách hàng |
| Điều kiện trước | Khách hàng đã đặng nhập vào hệ thống |
| Đảm bảo tối thiểu | Hiển thị danh sách đơn hàng |
| Điều kiện sau | khách hàng đã đăng nhập vào hệ thống |
| 1. Tác nhân truy cập vào chức năng "xem lịch sử đơn hàng“  2. Hiện thị danh sách các đơn hàng đã được ghị là của người dùng | |

|  |  |
| --- | --- |
| Tên use case | xem chi tiết đơn hàng |
| Tác nhân chính | Khách hàng |
| Điều kiện trước | Khách hàng đã đặng nhập vào hệ thống |
| Đảm bảo tối thiểu | Hiển thị chi tiết đơn hàng |
| Điều kiện sau |  |
| 1. Tác nhân truy cập vào chức năng "xem lịch sử đơn hàng“  2. Hiện thị danh sách các đơn hàng đã được ghị là của người dùng  3. Chọn vào đơn hàng muốn xem chi tiết  4. Hiển thị thông tin chi tiết của đơn hàng đã chọn | |

## 3.3. Lược đồ bảng

|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | | | |
| **Tên trường** | **Kiểu** | **Khóa** | **Ghi chú** |
| id | int | Khóa chính | NOT NULL |
| image | varchar(50) |  | NOT NULL |
| price | int |  | NOT NULL |
| description | varchar(500) |  |  |
| status | varchar(10) |  | NOT NULL |

##### Bảng 3.1 database sản phẩm

|  |  |  |  |
| --- | --- | --- | --- |
| **User** | | | |
| **Tên trường** | **Kiểu** | **Khóa** | **Ghi chú** |
| id | int | Khóa chính | NOT NULL |
| name | varchar(50) |  | NOT NULL |
| email | varchar(500) |  | NOT NULL |
| Password | varchar(50) |  | NOT NULL |
| role | varchar(10) |  | NOT NULL |

##### Bảng 3.2 database người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| **Cart\_item** | | | |
| **Tên trường** | **Kiểu** | **Khóa** | **Ghi chú** |
| id | int | Khóa chính | NOT NULL |
| userId | int | Khóa phụ | NOT NULL |
| productId | int | Khóa phụ | NOT NULL |
| Quatity | int |  | NOT NULL |

##### Bảng 3.3 database giỏ hàng

|  |  |  |  |
| --- | --- | --- | --- |
| **Order** | | | |
| **Tên trường** | **Kiểu** | **Khóa** | **Ghi chú** |
| id | int | Khóa chính | NOT NULL |
| userId | int | Khóa phụ | NOT NULL |
| name | varchar(300) |  | NOT NULL |
| phone | varchar(10) |  | NOT NULL |
| address | varchar(500) |  | NOT NULL |
| total | int |  | NOT NULL |
| status | varchar(10) |  | NOT NULL |
| cteateAt | datetime |  | NOT NULL |

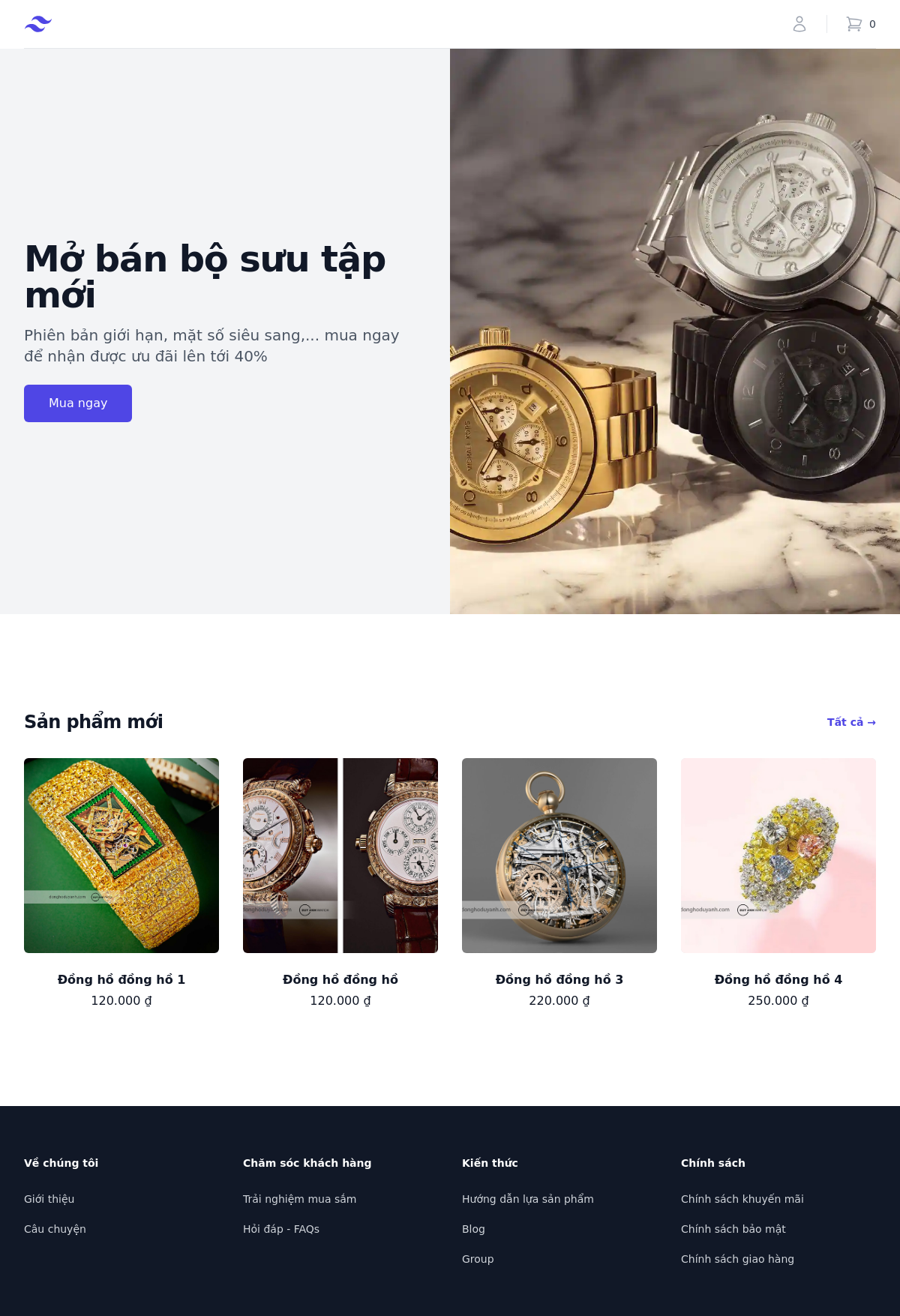
##### Bảng 3.4 database đặt hàng

|  |  |  |  |
| --- | --- | --- | --- |
| **Order\_item** | | | |
| **Tên trường** | **Kiểu** | **Khóa** | **Ghi chú** |
| id | int | Khóa chính | NOT NULL |
| orderId | int | Khóa phụ | NOT NULL |
| productId | int | Khóa phụ | NOT NULL |
| Quatity | int |  | NOT NULL |

##### Bảng 3.5 database sản phẩm trong đơn hàng

# CHƯƠNG 4: CHƯƠNG TRÌNH

## 4.1. Giao diện dành cho nhóm khách hàng

* Giao diện trang chủ: Khi truy cập vào cửa hàng, khách hàng có thể xem

+ Chọn vào mua ngay

#### Hình 4.1 Trang chủ website

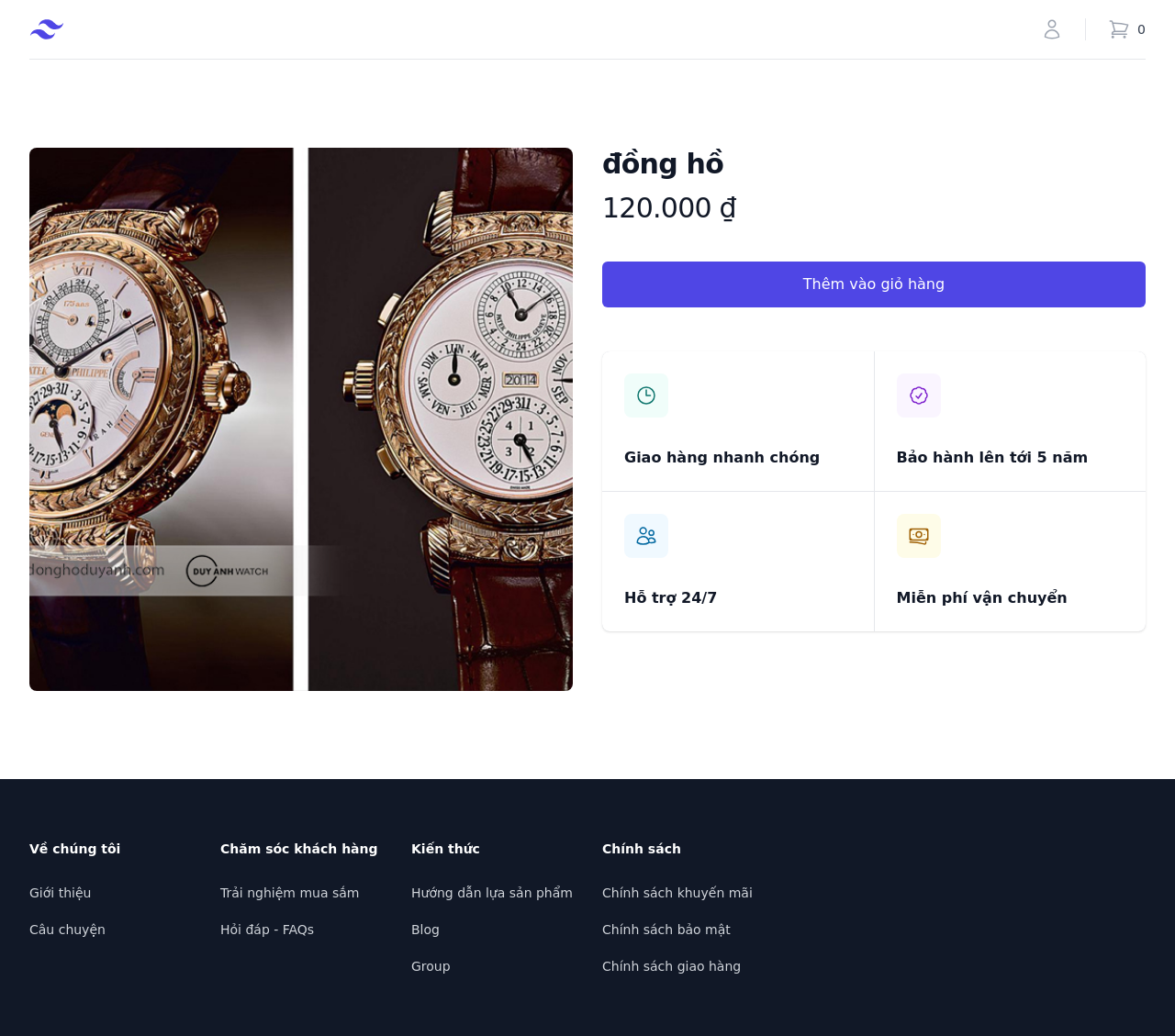
Thực hiện goị API lấy tất cả sản phẩm trong cở sở dữ liệu hiện thị ra cho khách hàng

+ Chọn vào tất cả

#### Hình 4.2 Giao diện trang sản phẩm

Cũng như “mua ngay” thực hiện gọi API lấy tất cả sản phẩm trong cở sơ dữ liệu ra

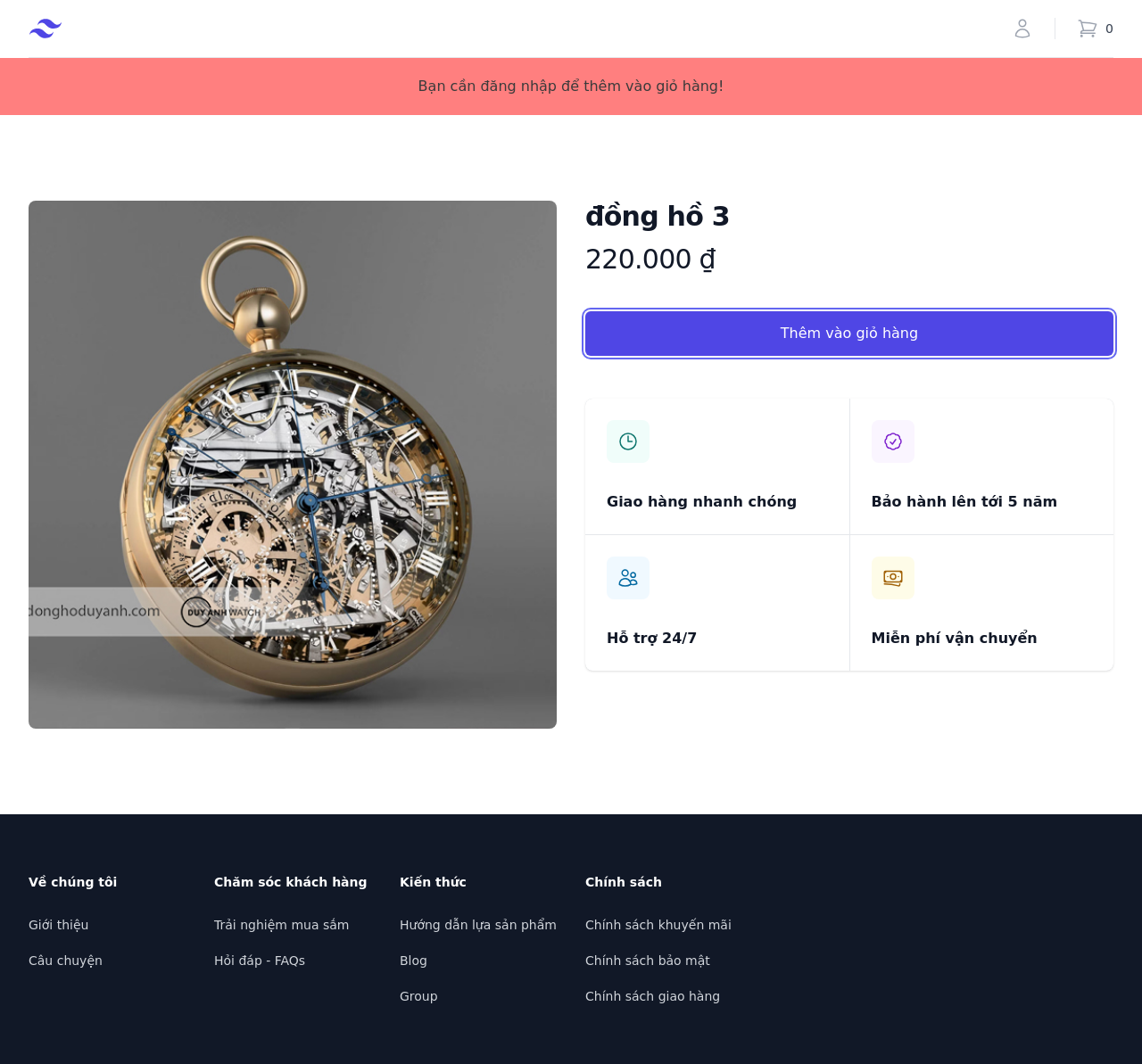
+ Chọn vào sản phẩm



#### Hình 4.3 Giao diện chi tiết sản phẩm

Thực hiện gọi API chi tiết sản phẩm để lấy thông tin chi tiết của một sản phẩm, nếu đã login vào hệ thống thì thì có thể thực hiện thêm vào giỏ hàng, nếu chưa thì thông báo yêu cầu đăng nhập và chuyển đến trang đăng nhập

+ Thêm vào giỏ hàng khi chưa đăng nhập



#### Hình 4.4 Giao diện thêm sản phẩm chưa đăng nhập

Thực hiện API thêm vào giỏ hàng chưa đăng nhập thông báo lỗi và chuyển trang đăng nhập

+ Thêm vào giỏ hàng khi đã đăng nhập

#### Hình 4.5 Giao diện thêm sản phẩm đã đăng nhập

Thực hiện API thêm vào giở hàng đã đăng nhập thông báo thành công số lượng sản phẩm được tăn lên

+ Đăng nhập

#### Hình 4.6 Giao diện đăng nhập thành công

Thực hiện API đăng nhập nhập thông tin hệ thống xác thực và trả về kết quả

+ Đăng ký

#### Hình 4.7 Giao diện đăng ký thất bại

Thực hiện API đăng ký hệ thống xác thực và trả về kết quả

+ Giỏ hàng

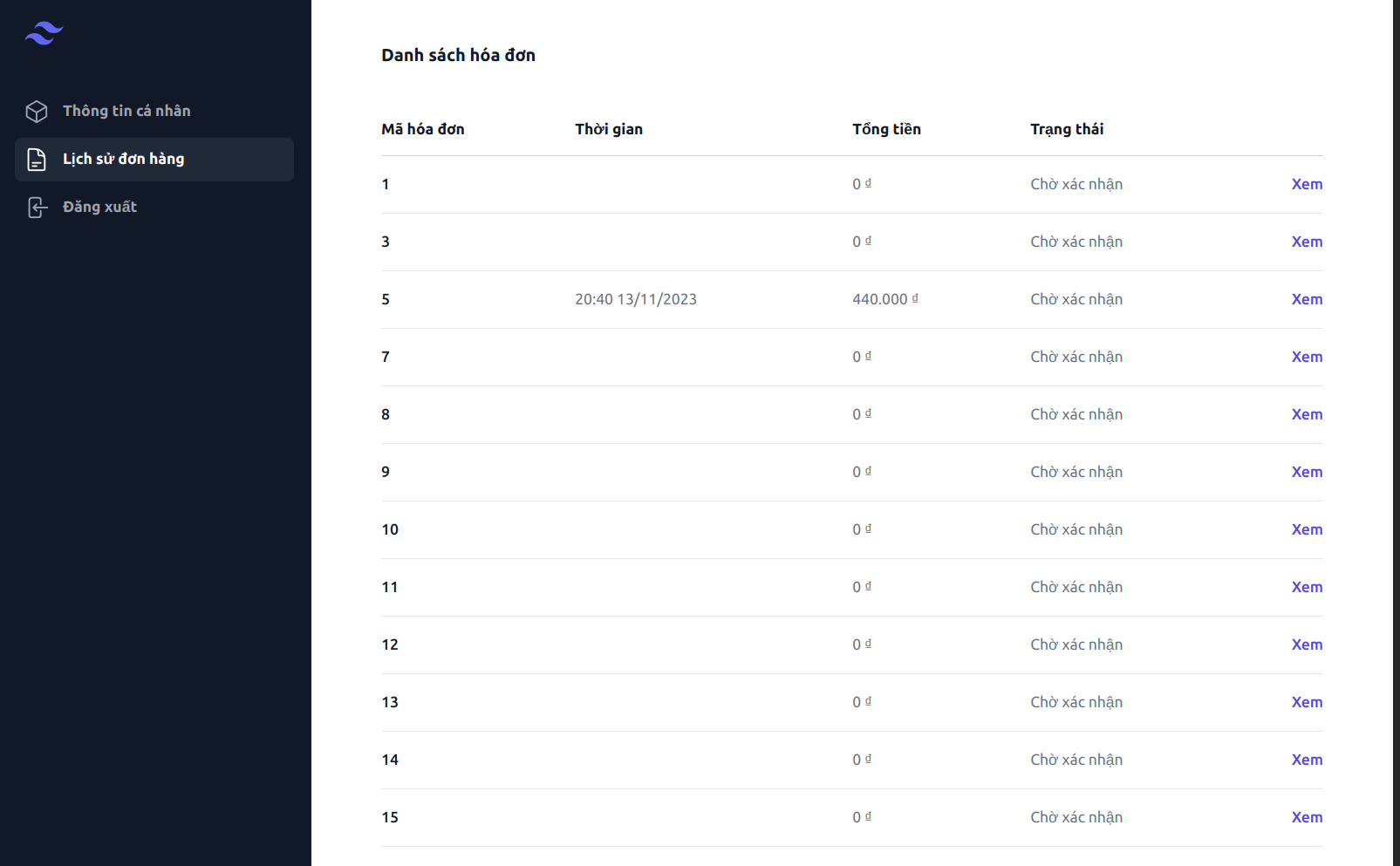
#### Hình 4.8 Giao diện giỏ hàng

Thực hiện gọi API giỏ hàng để lấy ra các sản phẩm đã thêm vào giỏ hàng của user đó

+ Quản lý thông tin cá nhân

#### Hình 4.9 Giao diện cập nhật thông tin cá nhân

Thực hiện gọi API lấy thông tin cá nhân, có thể thực hiện cập nhật thông tài khoản gọi api cập nhật tài khoản, có thể đổi mật khẩu gọi API đổi mật khẩu.

+Lịch sử đơn hàng

#### Hình 4.10 Giao diện lịch sử đơn hàng

Thực hiện API lấy lịch sử đơn hàng của user đó, có thể xem thông tin chi tiết của đơn hàng

+ Thông tin chi tiết

#### Hình 4.11 Giao diện chi tiết đơn hàng

Thực hiện gọi API lấy thông tin chi tiết đơn hàng nếu đang ở trạng thái chờ xác nhật thì có thể hủy đơn

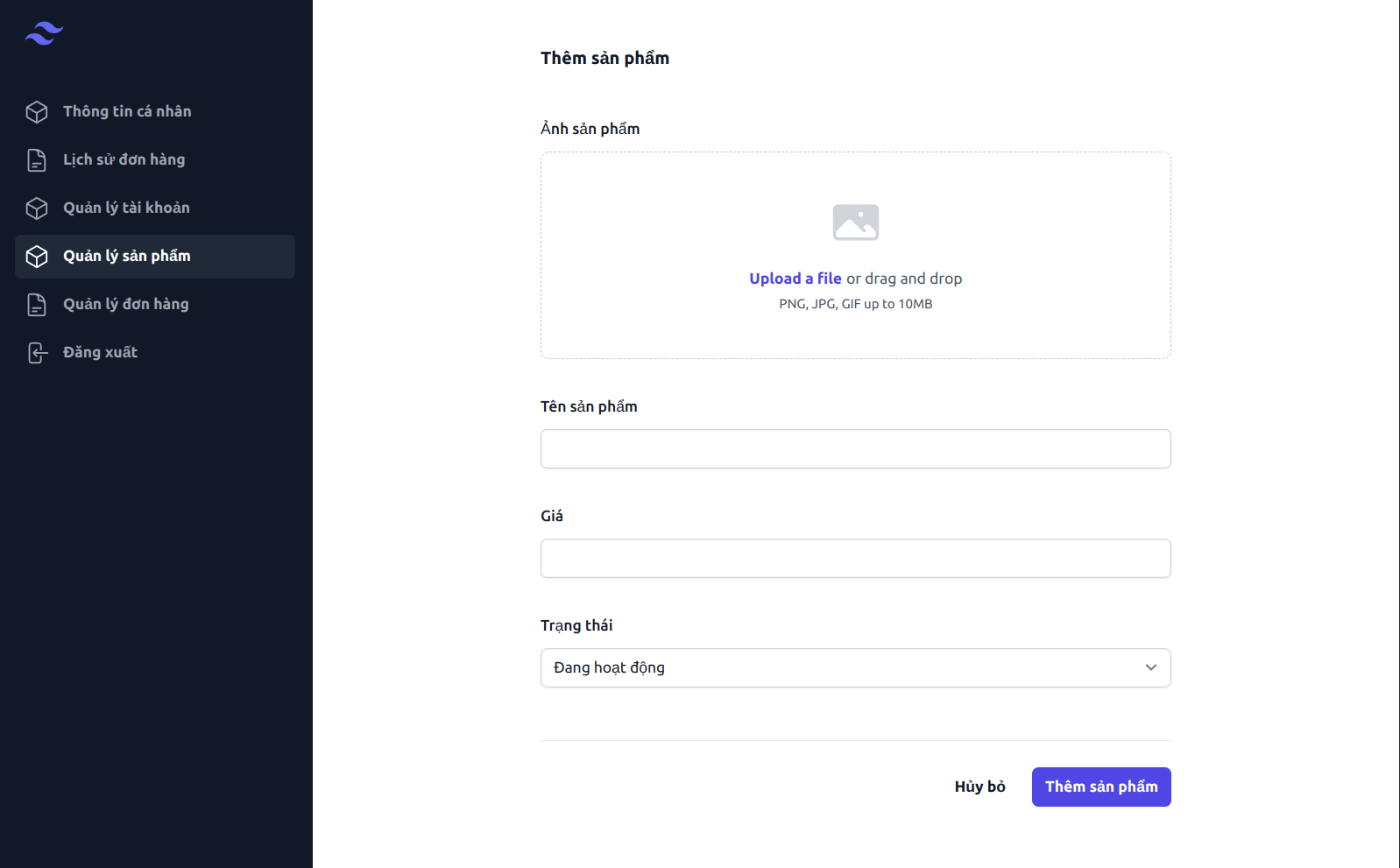
## 4.2 Giao diện dành cho nhóm Nhân viên

+ Quản lý sản phẩm

#### Hình 4.12 Giao diện quản lý sản phẩm

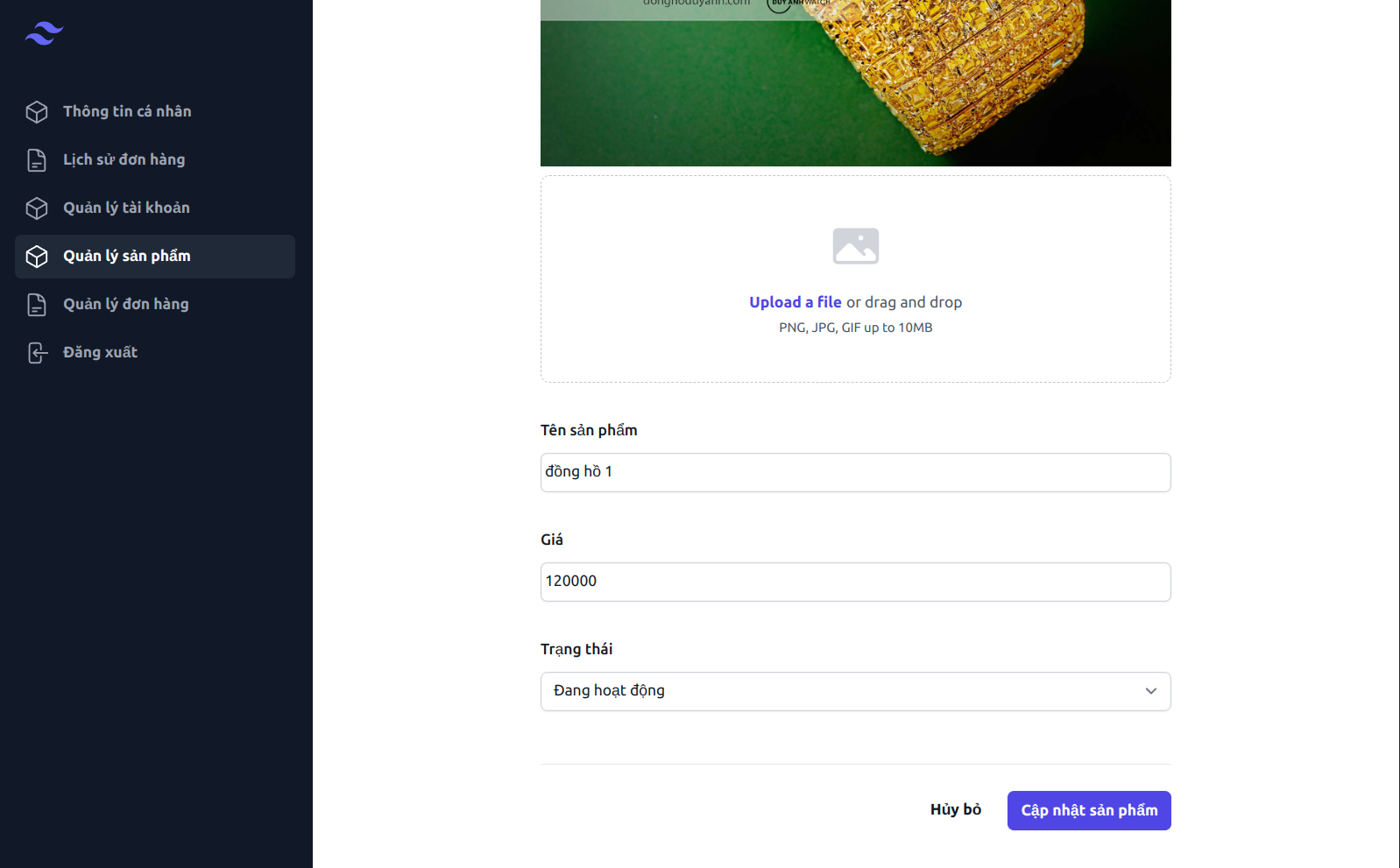
Thực hiện gọi API lấy danh sách sản phẩm, có thể thêm một sản phẩm mời hoăc cập nhật sản phẩm

+Thêm sản phẩm



#### Hình 4.13 Giao diện thêm sản phẩm

+Cập nhật sản phẩm



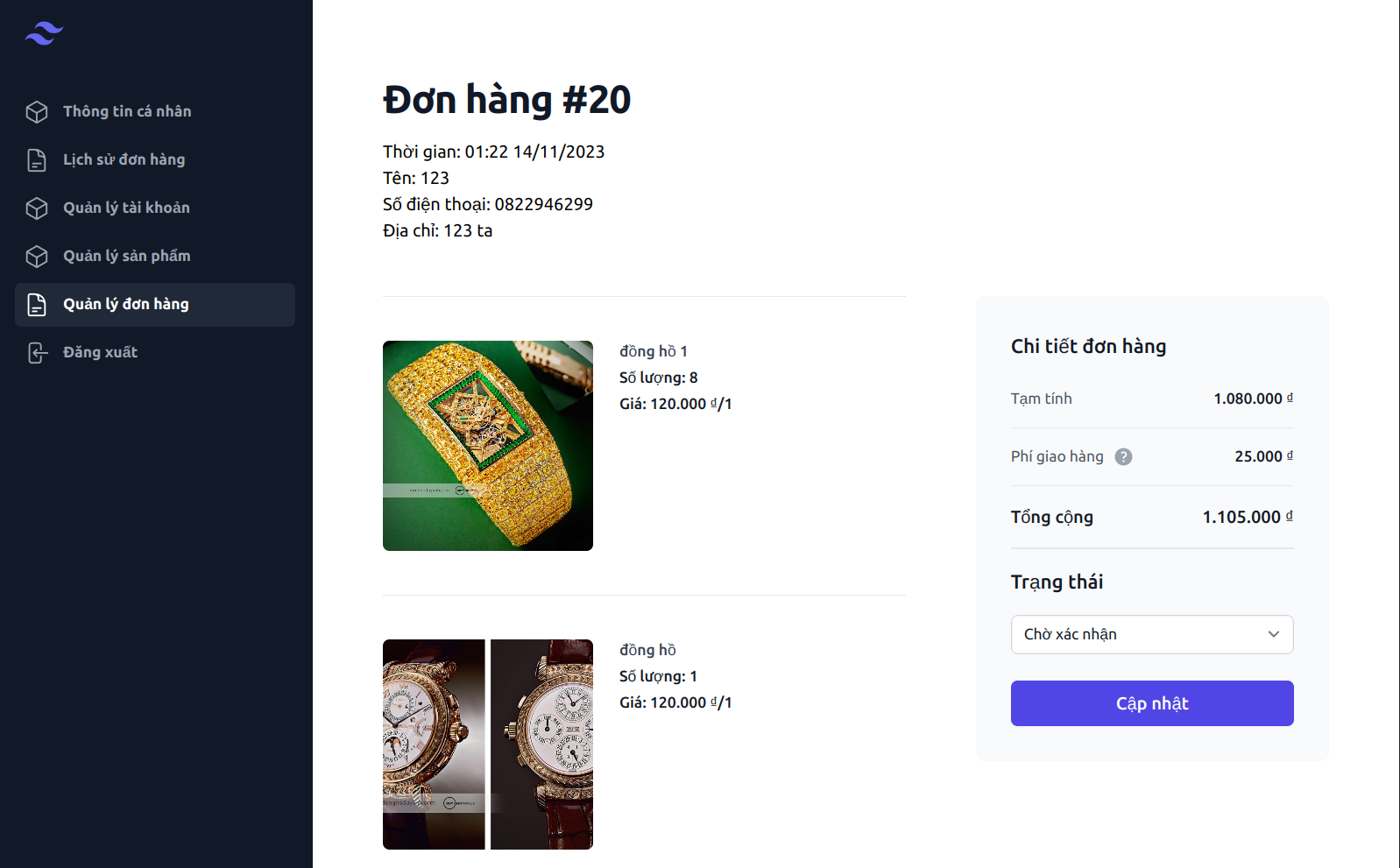
#### Hình 4.14 Giao diện cập nhật sản phẩm

+ Quản lý đơn hàng

#### Hình 4.15 Giao diện quản lý đơn hàng

Thực hiện gọi API lấy tất cả đơn hàng, có thể thực hiện cập nhật đơn hàng khi này ta có thể thay đổi trạng thái của một đơn hàng

+Cập nhật đơn hàng

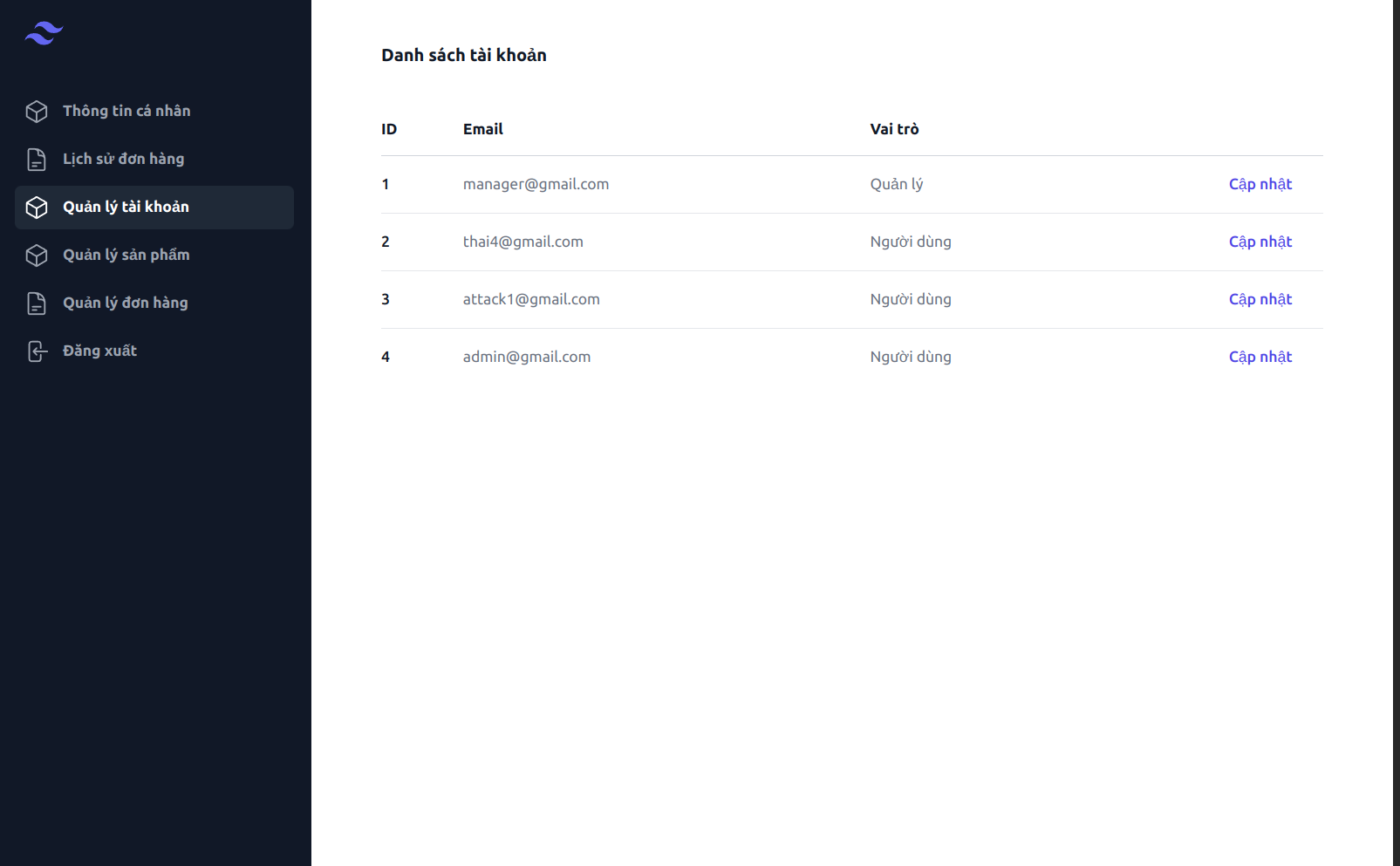


#### Hình 4.16 Giao diện cập nhật đơn hàng

## 4.3 Giao diện dành cho nhóm quản lý

Bản điều khiển của quản lý có thêm quản lý tài khoản

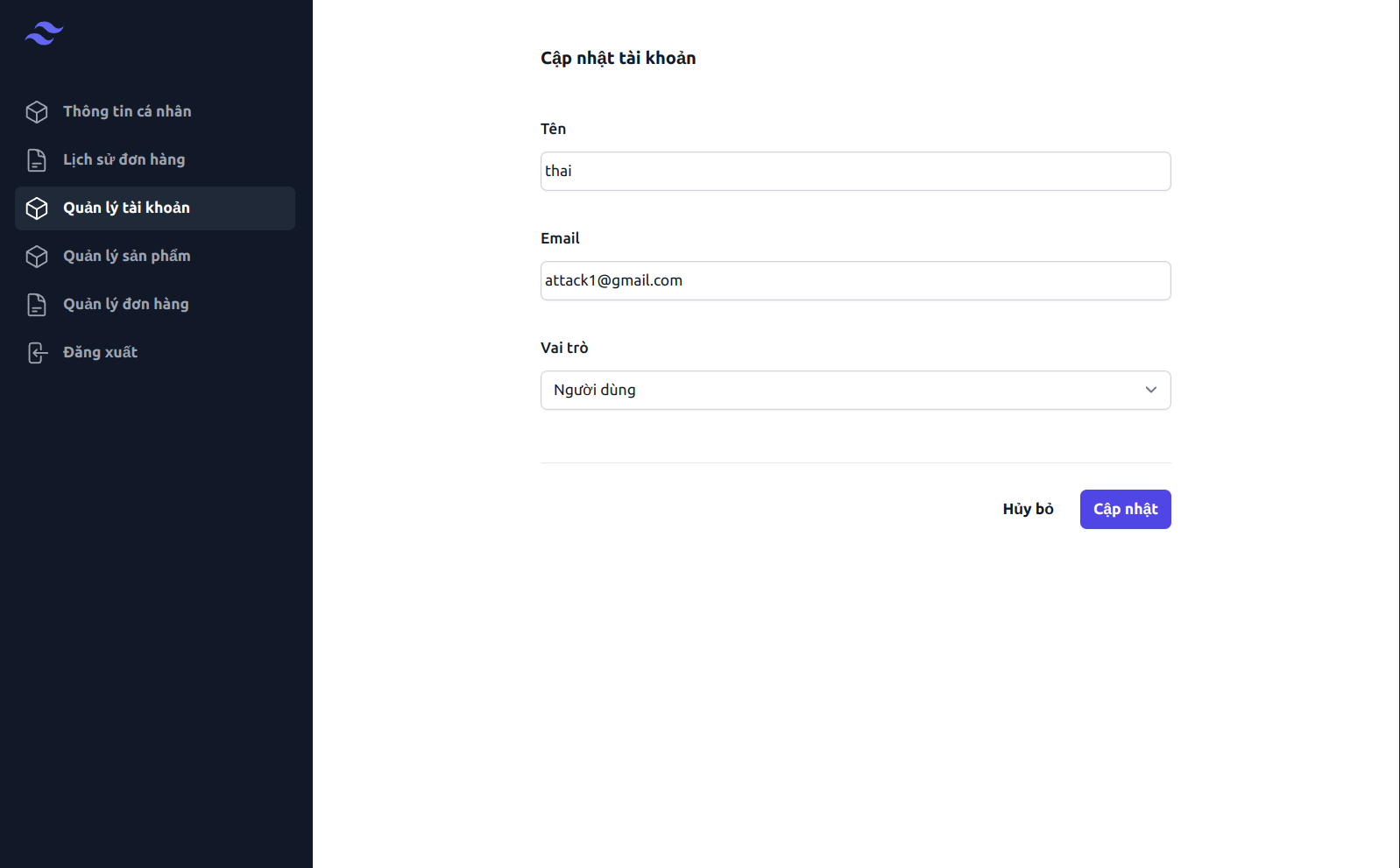
+ Quản lý tài khoản



#### Hình 4.17 Giao diện quản lý tài khoản

Thực hiện API lấy tất cả user của hệ thống, có thể phân quyền cho các user đố chỉ có thể phần một là “khách hàng” hay “Nhân viên”.

+ Phân quyền cho tài khoản



#### Hình 4.17 Giao diện phần quyền

# KẾT LUẬN

Đề tài đã xây dựng được hệ thống Website bán quần áo online

* Có giao diện website
* Có server chạy NestJS
* Có sử dụng SQLite
* Áp dụng reactjs

Hạn chế:

Hạn chế của đề tài này là:

* Chưa tối ưu phần mềm
* Cần thêm nhiều tính năng để hỗ trợ người dùng

Hướng giải quyết:

Để giải quyết các mặt hạn chế, bài nghiên cứu cần bổ sung thêm:

* Tối ưu hóa phần mềm dùng thêm caching để tăng tốc độ
* Thêm tính năng hỗ trợ người dùng

## 

## 

# DANH MUC TÀI LIỆU THAM KHẢO

Tiếng Việt:

1. *Ebook: Lập trình Node.js thật đơn giản - VNTALKING.COM*

Tiếng Anh:

1. Node.js By Example – Krasimir Tsonev
2. Web Development with Node and Express – Ethan Brown

Danh mục các Website tham khảo:

1. SQLite: https://viblo.asia/p/sqlite-la-gi-E375zVVR5GW
2. nodejs: <https://thewebsmithsite.wordpress.com/2015/11/21/makes-nothing-sense-with-asyncronous-programming-node-js/>
3. lý thuyết Api: https://aws.amazon.com/vi/what-is/api/
4. lý thuyết NestJS: https://viblo.asia/p/nestjs-framework-than-thanh-cho-nodejs-RQqKLL7OK7z
5. lý thuyết ReactJS: <https://blog.hubspot.com/website/react-js>