|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NGUYỄN VĂN DŨNG  Nguyễn Văn Dũng | **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **KHOA CÔNG NGHỆ THÔNG TIN**  **======\*\*\*======**  **E:\dh-cn-hanoi.png**  **BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**  **NGÀNH: CÔNG NGHỆ THÔNG TIN**  **ĐỀ TÀI: XÂY DỰNG TRANG THƯƠNG MẠI ĐIỆN TỬ TỰ ĐỘNG ĐỒNG BỘ CÁC SẢN PHẨM TỪ NỀN TẢNG THỨ BA**   |  |  | | --- | --- | | Cán bộ hướng dẫn | **: ThS. Nguyễn Xuân Hoàng** | | Sinh viên thực hiện | **: Nguyễn Văn Dũng** | | Mã số sinh viên  Lớp | **: 2020603502**  **: CNTT03-K15** |   ***Hà Nội – Năm 2024*** |
| CÔNG NGH Ệ THÔNG TIN |

# MỤC LỤC

[MỤC LỤC 1](#_Toc167021978)

[LỜI CẢM ƠN 4](#_Toc167021979)

[LỜI MỞ ĐẦU 5](#_Toc167021980)

[CHƯƠNG 1. TỔNG QUAN VỀ CÔNG NGHỆ VÀ CƠ SỞ LÝ THUYẾT 7](#_Toc167021981)

[1.1. NestJS 7](#_Toc167021982)

[1.1.1. Giới thiệu về NestJS 7](#_Toc167021983)

[1.1.2. Lịch sử phát triển của NestJS 7](#_Toc167021984)

[1.1.3. Ưu điểm của NestJS 8](#_Toc167021985)

[1.1.4. Nhược điểm của NestJS 9](#_Toc167021986)

[1.1.5. So sánh NestJS với công nghệ khác 9](#_Toc167021987)

[1.2. PostgreSQL 10](#_Toc167021988)

[1.2.1. Giới thiệu về PostgreQL 10](#_Toc167021989)

[1.2.2. Lịch sử phát triển của PostgreQL 10](#_Toc167021990)

[1.2.3. Ưu điểm của PostgreQL 11](#_Toc167021991)

[1.2.4. Nhược điểm của PostgreSQL 12](#_Toc167021992)

[1.2.5. So sánh PostgreSQL với các hệ quản trị cơ sở dữ liệu khác 12](#_Toc167021993)

[1.3. ReactJS 13](#_Toc167021994)

[1.3.1. Giới thiệu ReactJs 13](#_Toc167021995)

[1.3.2. Lịch sử phát triển 13](#_Toc167021996)

[1.3.3. Ưu điểm của ReactJS 13](#_Toc167021997)

[1.3.4. Nhược điểm của ReactJS 14](#_Toc167021998)

[1.3.5. So sánh ReactJS với các công nghệ khác 14](#_Toc167021999)

[CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU 15](#_Toc167022000)

[2.1. Phân tích vấn đề đồng bộ 15](#_Toc167022001)

[2.1.1. Tại sao cần đồng bộ sản phẩm từ bên thứ ba về hệ thống 15](#_Toc167022002)

[2.1.2. Khó khăn khi tích hợp đồng bộ sản phẩm 16](#_Toc167022003)

[2.2. Xác định các yêu cầu chính của hệ thống 17](#_Toc167022004)

[2.2.1. Phân tích các yêu cầu 17](#_Toc167022005)

[2.2.2. Các yêu cầu phi chức năng 17](#_Toc167022006)

[2.3. Xây dựng biểu đồ use case 17](#_Toc167022007)

[2.3.1. Các tác nhân 17](#_Toc167022008)

[2.3.2. Các use case của hệ thống 17](#_Toc167022009)

[2.3.3. Biểu đồ use case 19](#_Toc167022010)

[2.3.4. Mô tả chi tiết các use case 20](#_Toc167022011)

[2.3.5. Phân tích các use case 35](#_Toc167022012)

[2.4. Xây dựng cơ cở dữ liệu 57](#_Toc167022013)

[2.4.1. Biểu đồ Entity Relationship Diagram 57](#_Toc167022014)

[2.4.2. Các bảng trong cơ sở dữ liệu 57](#_Toc167022015)

[CHƯƠNG 3. KẾT QUẢ ĐẠT ĐƯỢC 60](#_Toc167022016)

[3.1. Giao diện chương trình 60](#_Toc167022017)

[3.1.1. Giao diện phía khách hàng 60](#_Toc167022018)

[3.1.2. Giao diện phía quản lí 68](#_Toc167022019)

[3.2. Kiểm thử hệ thống 72](#_Toc167022020)

[TÀI LIỆU THAM KHẢO 74](#_Toc167022021)

# DANH MỤC HÌNH ẢNH

[Hình 2.3.1. Các use case phía khách hàng 20](#_Toc167022171)

[Hình 2.3.2. Các use case phía người quản trị 21](#_Toc167022172)

[Hình 2.3.3. Biểu đồ trình tự use case Đăng nhập 36](#_Toc167022173)

[Hình 2.3.4. Biểu đồ trình tự use case Quản lý địa chỉ 37](#_Toc167022174)

[Hình 2.3.5. Biểu đồ trình tự use case Xem sản phẩm 38](#_Toc167022175)

[Hình 2.3.6. Biểu đồ trình tự use case Tìm kiếm sản phẩm 39](#_Toc167022176)

[Hình 2.3.7 Biểu đồ trình tự use case Thêm vào giỏ hàng 40](#_Toc167022177)

[Hình 2.3.8. Biểu đồ trình tự use case Đặt hàng 41](#_Toc167022178)

[Hình 2.3.9 Mô hình kết nối với VnPay 42](#_Toc167022179)

[Hình 2.3.10. Sơ đồ tuần tự (Nguồn VnPay) 43](#_Toc167022180)

[Hình 2.3.11 Hình ảnh API Docx Product của Shopify 44](#_Toc167022181)

[Hình 2.3.12 Biểu đồ trình tự Use case tự động đồng bộ sản phẩm từ Shopify về hệ thống 44](#_Toc167022182)

[Hình 2.3.13 Màn hình Private App của ShopBase 45](#_Toc167022183)

[Hình 2.3.14 Biểu đồ trình tự Use case tự động đồng bộ sản phẩm từ ShopBase về hệ thống 46](#_Toc167022184)

[Hình 2.3.15. Biểu đồ trình tự use case Tra cứu đơn hàng 47](#_Toc167022185)

[Hình 2.3.16 Biểu đồ trình tự use case Quản lý giỏ hàng 48](#_Toc167022186)

[Hình 2.3.17. Biểu đồ trình tự use case Bảo trì danh mục 50](#_Toc167022187)

[Hình 2.3.18 Biểu đồ trình tự use case Bảo trì sản phẩm 53](#_Toc167022188)

[Hình 2.3.19 Biểu đồ trình tự use case Quản lý đơn hàng 54](#_Toc167022189)

[Hình 2.3.20 Biểu đồ trìnhh tự use case Xem thống kê 55](#_Toc167022190)

[Hình 2.3.21 Biểu đồ trình tự use case Quản lý người dùng 58](#_Toc167022191)

[Hình 2.4.1. Thiết kế cơ sở dữ liệu 58](#_Toc167022192)

[Hình 2.4.2. Bảng users 59](#_Toc167022193)

[Hình 2.4.3. Bảng orders 60](#_Toc167022194)

[Hình 2.4.4. Bảng order\_details 60](#_Toc167022195)

[Hình 3.1.1 Màn hình trang chủ 63](#_Toc167022196)

[Hình 3.1.2 Màn hình xem sản phẩm theo danh mục 64](#_Toc167022197)

[Hình 3.1.3 Màn hình xem chi tiết sản phẩm 65](#_Toc167022198)

[Hình 3.1.4 Màn hình giỏ hàng 66](#_Toc167022199)

[Hình 3.1.5 Màn hình thông tin thanh toán 67](#_Toc167022200)

[Hình 3.1.6 Màn hình thông tin cá nhân 68](#_Toc167022201)

[Hình 3.1.7 Màn hình đăng ký 69](#_Toc167022202)

[Hình 3.1.8 Màn hình dashboard 69](#_Toc167022203)

[Hình 3.1.9 Màn hình danh sách khách hàng 70](#_Toc167022204)

[Hình 3.1.10 Màn hình danh sách sản phẩm 70](#_Toc167022205)

[Hình 3.1.11 Màn hình danh sách các danh mục 71](#_Toc167022206)

[Hình 3.1.12 Màn hình danh sách các hãng 71](#_Toc167022207)

[Hình 3.1.13 Màn hình quản lý đơn hàng 72](#_Toc167022208)

# DANH MỤC BẢNG BIỂU

[Bảng 2.3.1 Các use case hệ thống 20](#_Toc167022306)

[Bảng 3.2.1 Kiểm thử hệ thống 75](#_Toc167022307)

# LỜI CẢM ƠN

Trước hết, em xin chân thành cảm ơn các thầy cô giáo trong khoa đã tận tình hướng dẫn, giúp đỡ và động viên em trong suốt quá trình thực hiện đồ án này. Các thầy cô không chỉ truyền đạt kiến thức chuyên môn mà còn giúp em hình thành phương pháp nghiên cứu và tinh thần học tập nghiêm túc.

Em cũng xin gửi lời cảm ơn chân thành đến các bạn đã cùng nhau thảo luận, trao đổi và chia sẻ kinh nghiệm trong quá trình làm đồ án.

Đặc biệt, em xin gửi đến thầy Nguyễn Xuân Hoàng người đã luôn tận tình hướng dẫn, giúp đỡ em trong quá trình hoàn thành đồ án tốt nghiệp một lời cảm ơn sâu sắc nhất. Dưới sự hướng dẫn tận tình của thầy Nguyễn Xuân Hoàng em đã hoàn thành báo cáo đồ án.

Cuối cùng, em xin chân thành cảm ơn gia đình, người thân đã luôn bên cạnh, động viên và tạo điều kiện để tôi yên tâm học tập và thực hiện đồ án.

Vì kiến thức bản thân còn hạn chế, trong quá trình thực hiện và hoàn thiện đồ án em không tránh khỏi những sai sót. Em rất mong nhận được sự thông cảm và góp ý của quý thầy cô.

Em xin trân trọng cảm ơn!

# LỜI MỞ ĐẦU

1. **Lý do chọn đề tài**

Hiện nay công nghệ thông tin không ngừng phát triển mạnh mẽ và hiện đại, các nền tảng số ngày càng tiếp cận tới người dùng dễ dàng hơn, trong đó có các hệ thống thương mại điện tử. Nhiều cửa hàng đã dần chuyển từ hình thức bán hàng truyền thống sang thương mại điện tử nhanh chóng với mức chi phí duy trì chỉ bằng một phần nhỏ với việc thuê mặt bằng.

Với mong muốn củng cố và nâng cao kiến thức về lập trình website, áp dụng kiến thức để xây dựng một trang website thương mại điện tử tích hợp đồng bộ sản phẩm giúp người dùng có thể dễ dàng sử dụng, mua sắm, tham khảo thiết bị công nghệ chất lượng cao với mức giá chuẩn xác, sử dụng hệ thống với công nghệ mới nhanh và mượt mà hơn em đã chọn đề tài “Xây dựng trang thương mại điện tử tự động đồng bộ các sản phẩm từ nền tảng thứ ba”.

Quá trình thực hiện đề tài của em có thể còn nhiều thiếu xót. Em rất mong nhận được sự cảm thông và chia sẻ của quý thầy cô để có thể đưa đề tài phát triển thêm mạnh mẽ hơn nữa. Em xin chân thành cảm ơn!

1. **Mục tiêu của đề tài**

Xây dựng được trang thương mại hoàn thiện gồm các module website bán hàng và các module quản trị hệ thống, đồng bộ sản phẩm từ các bên thứ ba. Rèn luyện các kiến thức và kỹ năng phát triển hệ thống phần mềm và thái độ làm việc chuyên nghiệp.

1. **Nội dung nghiên cứu**

* Khảo sát và phân tích yêu cầu quy trình quản lý và mua bán hàng hoá.
* Phân tích thiết kế hệ thống.
* Thiết kế cơ sở dữ liệu.
* Lập trình backend API.
* Lập trình giao diện và kết nối tới API.
* Có kế hoạch và đánh giá kiểm thử hệ thống.

1. **Phạm vi đề tài**

Đề tài mới xây dựng được website bán hàng online mức cơ bản, hỗ trợ thanh toán trực tuyến, đồng bộ sản phẩm từ Shopify, ShopBase, báo cáo thống kê doanh thu theo tháng, năm, đồng

1. **Bố cục đề tài**

Ngoài các phần Mở đầu, Kết luận và Tài liệu tham khảo, báo cáo đồ án được bố cục thành ba chương chính sau.

* Chương 1: Trình bày tổng quan về công nghệ và cơ sở lý thuyết.
* Chương 2: Phân tích thiết kế hệ thống và thiết kế cơ sở dữ liệu.
* Chương 3: Cài đặt phần mềm và kết quả đạt được.

# TỔNG QUAN VỀ CÔNG NGHỆ VÀ CƠ SỞ LÝ THUYẾT

## NestJS

### Giới thiệu về NestJS

Là framework để xây dựng web backend dựa trên Node.js mạnh mẽ, linh hoạt và TypeScript, NestJS sử dụng các khái niệm của Angular để tạo nên một cấu trúc ứng dụng dễ dàng quản lý. NestJS chuyên về việc xây dựng các ứng dụng web server-side hiện đại. NestJS có cấu trúc dự án gọn gàng và tổ chức mã nguồn dễ dàng và hợp lý, giúp việc phát triển và bảo trì ứng dụng trở nên dễ dàng hơn.

NestJS được xây dựng trên TypeScript, giúp tăng tính linh hoạt và sự rõ ràng trong việc phát triển ứng dụng. NestJS hỗ trợ dependency injection, giúp quản lý dependencies một cách dễ dàng và làm cho ứng dụng trở nên dễ mở rộng hơn. NestJS có khả năng sử dụng middleware để xử lý các yêu cầu HTTP, giúp tạo ra các ứng dụng có khả năng linh hoạt và mạnh mẽ. NestJS có một cộng đồng lớn và tích cực, với nhiều tài liệu và hướng dẫn hữu ích, giúp người dùng dễ dàng học và sử dụng NestJS.

### Lịch sử phát triển của NestJS

NestJS là một framework phát triển ứng dụng web server-side được xây dựng trên nền tảng Node.js. Nó ra đời vào năm 2017 và từ đó đã trở thành một trong những lựa chọn phổ biến cho việc xây dựng các ứng dụng web chất lượng cao.

Lịch sử phát triển của NestJS có các cột mốc quan trọng như sau:

Ra mắt lần đầu (2017): NestJS được tạo ra bởi Kamil Myśliwiec, một lập trình viên người Ba Lan. Ông cảm thấy rằng cộng đồng Node.js cần một framework có cấu trúc rõ ràng và quy mô lớn hơn để xây dựng các ứng dụng web phức tạp. NestJS được phát triển dựa trên các nguyên tắc của Angular, một framework phát triển ứng dụng web front-end phổ biến.

Phiên bản 2 (2018): Phiên bản NestJS 2 được phát hành vào năm 2018 với nhiều cải tiến và tính năng mới. Phiên bản này đã cải thiện hiệu suất và tăng tính năng của framework.

NestJS 5 (2018): Phiên bản NestJS 5 tiếp tục nâng cao khả năng và tích hợp của framework. Nó bao gồm sự hỗ trợ tốt hơn cho GraphQL và WebSocket, cũng như các tính năng mới như Middleware, Pipes và Guards.

NestJS 6 (2019): Phiên bản NestJS 6 tiếp tục cải thiện và mở rộng các tính năng của framework. Nó giới thiệu Dependency Injection Container và Caching module, cung cấp khả năng quản lý phụ thuộc tốt hơn và tăng cường hiệu suất.

NestJS 7 (2020): Phiên bản NestJS 7 tập trung vào việc cải thiện trải nghiệm phát triển và tăng tính tương thích. Nó cải thiện hệ thống validation, hỗ trợ TypeScript 3.7 và Node.js 14, và cung cấp các tính năng mới như Code Generation.

NestJS 8 (2021): Phiên bản NestJS 8 tiếp tục mang đến nhiều cải tiến và tính năng mới. Nó hỗ trợ Fastify, một framework web độc lập với hiệu suất cao, cung cấp cơ chế interceptor mới và cải thiện hệ thống ghi nhật ký.

NestJS đã thu hút sự quan tâm và sự ủng hộ từ cộng đồng phát triển Node.js. Nó đạt được sự phát triển nhanh chóng và trở thành một trong những framework phát triển ứng dụng web server-side phổ biến và được ưa chuộng. Với cấu trúc rõ ràng, tích hợp tốt và khả năng mở rộng, NestJS giúp các nhà phát triển xây dựng các ứng dụng web chất lượng cao một cách dễ dàng và hiệu quả.

### Ưu điểm của NestJS

Kiến trúc module và cấu trúc tổ chức rõ ràng: NestJS cung cấp một kiến trúc module để phân chia và tổ chức mã nguồn ứng dụng. Kiến trúc module cho phép bạn xác định rõ ràng các thành phần, phụ thuộc và lưu trữ logic liên quan trong một vùng cụ thể. Điều này giúp tạo ra mã nguồn dễ đọc, dễ bảo trì và dễ mở rộng.

TypeScript và Angular-like: NestJS sử dụng TypeScript làm ngôn ngữ chính, cho phép bạn sử dụng kiểu tĩnh, trình biên dịch và các tính năng của TypeScript. Điều này giúp tăng tính chính xác, khả năng tái sử dụng và sự dễ dàng trong quá trình phát triển. Ngoài ra, NestJS lấy cảm hứng từ cú pháp và các khái niệm của Angular, giúp cho việc chia sẻ kiến thức và mã nguồn giữa phía back-end và front-end trở nên dễ dàng hơn.

Dependency Injection (DI): NestJS sử dụng mô hình Dependency Injection để quản lý các phụ thuộc và tạo ra các thành phần có thể tái sử dụng. DI giúp giảm sự phụ thuộc giữa các thành phần, tạo ra mã nguồn linh hoạt, dễ kiểm thử và dễ bảo trì.

Hỗ trợ tốt cho Microservices: NestJS cung cấp các công cụ và thư viện tích hợp sẵn để phát triển và quản lý kiến trúc Microservices. Việc phát triển các dịch vụ nhỏ, độc lập và có khả năng mở rộng trở nên dễ dàng hơn với NestJS.

### Nhược điểm của NestJS

***Khả năng học:***

NestJS có một độ dốc học đối với những người mới bắt đầu, đặc biệt là đối với những người không quen thuộc với TypeScript hoặc các khái niệm của Angular.

***Tính linh hoạt trong việc lựa chọn công nghệ:***

Mặc dù NestJS hỗ trợ cả GraphQL và RESTful API, nhưng việc lựa chọn công nghệ có thể là một thách thức đối với các dự án lớn hoặc có yêu cầu cụ thể.

***Khả năng cộng đồng:***

Mặc dù cộng đồng NestJS đang phát triển, nhưng nó vẫn chưa bằng phổ biến như các framework khác như Express hoặc Koa, do đó việc tìm kiếm giải pháp cho các vấn đề cụ thể có thể khó khăn hơn.

### So sánh NestJS với công nghệ khác

***Express.js:***

NestJS được xây dựng trên nền tảng của Express.js, vì vậy chúng có nhiều điểm tương đồng. Tuy nhiên, NestJS cung cấp một cấu trúc và các khái niệm tổ chức mã nguồn tốt hơn, như Dependency Injection, Middleware, và Providers, giúp quản lý mã nguồn dễ dàng hơn đặc biệt là cho các dự án lớn.

***Spring Boot (Java):***

So với Java, NestJS cung cấp một cách tiếp cận tương tự với Spring Boot trong việc phát triển các ứng dụng web dựa trên nguyên tắc Dependency Injection và thiết kế module. Tuy nhiên, NestJS sử dụng TypeScript và JavaScript, mang lại sự linh hoạt và tăng tốc độ phát triển so với Java.

***ASP.NET Core (C#):***

So với ASP.NET Core, NestJS được viết bằng TypeScript/JavaScript và sử dụng Node.js, trong khi ASP.NET Core sử dụng C#. Cả hai đều cung cấp các tính năng mạnh mẽ cho việc phát triển ứng dụng web, nhưng lựa chọn giữa chúng phụ thuộc vào kinh nghiệm của bạn và hệ sinh thái công nghệ hiện có.

## PostgreSQL

### Giới thiệu về PostgreQL

PostgreSQL là một hệ thống quản trị cơ sở dữ liệu quan hệ và đối tượng (object-relational database management system) miễn phí và nguồn mở (RDBMS) tiên tiến nhất hiện nay. khả năng mở rộng cao và tuân thủ các tiêu chuẩn kỹ thuật. Nó được thiết kế để xử lý một loạt các khối lượng công việc lớn, từ các máy tính cá nhân đến kho dữ liệu hoặc dịch vụ Web có nhiều người dùng đồng thời.

### Lịch sử phát triển của PostgreQL

Những nguồn gốc (1986-1995): PostgreSQL bắt đầu dưới dạng dự án nghiên cứu tại Đại học California, Berkeley vào năm 1986. Ban đầu, nó được gọi là POSTGRES, một tên viết tắt của "Post Ingres". Sau đó, phiên bản đầu tiên của PostgreSQL đã được phát hành vào năm 1995.

Phát triển mã nguồn mở (1996-2000): Sau khi phát hành lần đầu, dự án PostgreSQL đã chuyển từ một dự án nghiên cứu sang một dự án mã nguồn mở được cộng đồng hỗ trợ rộng rãi. Trong giai đoạn này, sự phát triển chủ yếu dựa vào đóng góp từ cộng đồng.

Phiên bản 7.x và sự phát triển (2000-2005): Trong giai đoạn này, PostgreSQL đã trải qua sự phát triển đáng kể, với sự giới thiệu của các tính năng mới như hỗ trợ OLAP, XML, và các tính năng nâng cao về hiệu suất và an ninh.

Cải tiến về hiệu suất và tính năng (2005-2010): PostgreSQL 8.x đánh dấu giai đoạn mà PostgreSQL trở nên phổ biến hơn, đặc biệt là trong các ứng dụng web và doanh nghiệp. Cải tiến về hiệu suất và tính năng tiếp tục được thực hiện.

Phiên bản 9.x và tính năng đa dạng (2010-2015): PostgreSQL 9.x đưa vào nhiều tính năng mới quan trọng như JSONB, hỗ trợ index GIN, cải tiến về phân vùng và hiệu suất.

Sự phát triển toàn cầu (2015-nay): PostgreSQL ngày càng trở thành một trong những hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến nhất trên toàn thế giới, với sự hỗ trợ từ cộng đồng lớn và các tổ chức công nghệ hàng đầu. Các phiên bản mới tiếp tục mang lại nhiều tính năng mới và cải tiến về hiệu suất, an ninh và tính ổn định.

### Ưu điểm của PostgreQL

Mã nguồn mở: PostgreSQL là một dự án mã nguồn mở, điều này có nghĩa là mã nguồn của nó được công bố công khai và có thể được sửa đổi, phát triển bởi cộng đồng lập trình viên trên khắp thế giới. Điều này tạo điều kiện cho sự phát triển nhanh chóng, sự đa dạng tính năng và sự tin cậy thông qua sự kiểm tra và đóng góp của cộng đồng.

Tính mở rộng và linh hoạt: PostgreSQL hỗ trợ nhiều loại dữ liệu và các tính năng mở rộng, từ dữ liệu cơ bản như số, văn bản đến dữ liệu phức tạp như JSON, XML, và dữ liệu địa lý (GIS). Nó cung cấp khả năng linh hoạt cho việc mở rộng cấu trúc cơ sở dữ liệu để phù hợp với các nhu cầu cụ thể của ứng dụng.

Hiệu suất cao: PostgreSQL được tối ưu hóa để cung cấp hiệu suất cao, đặc biệt là trong các tác vụ xử lý dữ liệu lớn và phức tạp. Nó có các cơ chế như bộ chỉ mục thông minh, tối ưu hóa truy vấn và quản lý bộ nhớ để giúp tối ưu hóa hiệu suất.

An ninh mạnh mẽ: PostgreSQL cung cấp các tính năng an ninh mạnh mẽ để bảo vệ dữ liệu, bao gồm kiểm soát truy cập, mã hóa dữ liệu, xác thực và kiểm tra. Nó hỗ trợ các tiêu chuẩn bảo mật như SSL và GSSAPI, cùng với các tính năng bảo mật mở rộng như mã hóa dữ liệu cột và kiểm soát bởi cơ sở dữ liệu.

Hỗ trợ đa nền tảng: PostgreSQL có sẵn trên nhiều nền tảng hệ điều hành như Linux, Windows, macOS và các hệ điều hành Unix khác. Điều này tạo điều kiện cho tính di động và linh hoạt trong việc triển khai và vận hành hệ thống.

Cộng đồng lớn và hỗ trợ chuyên môn: PostgreSQL có một cộng đồng lớn và đa dạng với các chuyên gia và nhà phát triển trên toàn thế giới. Điều này cung cấp nguồn tài nguyên vô tận cho việc học hỏi, hỗ trợ và giải quyết vấn đề khi sử dụng hoặc triển khai PostgreSQL.

### Nhược điểm của PostgreSQL

***Hiệu suất trong một số trường hợp cụ thể:***

Mặc dù PostgreSQL có hiệu suất tốt, nhưng trong một số trường hợp cụ thể, nhất là khi xử lý các tải lớn hoặc nhiều ghi đồng thời, bạn có thể cần phải tinh chỉnh cấu hình hoặc sử dụng các giải pháp mở rộng.

***Phức tạp cho các ứng dụng nhỏ:***

Đối với các ứng dụng nhỏ hoặc không yêu cầu tính năng phức tạp, PostgreSQL có thể cảm thấy quá mạnh mẽ và phức tạp, với đòi hỏi cấu hình và quản lý cao.

***Đòi hỏi kiến thức chuyên môn:***

Sử dụng PostgreSQL đòi hỏi một số kiến thức về quản trị cơ sở dữ liệu và SQL, đặc biệt là khi cần tối ưu hiệu suất hoặc xử lý các tình huống khó khăn.

### So sánh PostgreSQL với các hệ quản trị cơ sở dữ liệu khác

***MySQL:***

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ phổ biến, giống như PostgreSQL. MySQL thường được lựa chọn cho các dự án với quy mô nhỏ đến trung bình, trong khi PostgreSQL thường được ưa chuộng cho các dự án có tính năng phức tạp và đòi hỏi tính mở rộng cao..

***SQLite:***

SQLite là một hệ quản trị cơ sở dữ liệu nhẹ, được tích hợp trực tiếp vào ứng dụng. Nó thích hợp cho các ứng dụng nhỏ và đơn giản, trong khi PostgreSQL thích hợp cho các ứng dụng với quy mô lớn và yêu cầu tính toàn diện.

## ReactJS

### Giới thiệu ReactJs

ReactJS là thư viện mã nguồn mở của Facebook được viết bằng JavaScript với mục đích xây dựng giao diện người dùng.

Trong ReactJS, giao diện được chia thành các phần nhỏ gọi là component và có thể dễ dàng tái sử dụng.

ReactJS giúp giao diện của ứng dụng web có thể phản ứng nhanh nhẹn và mượt mà hơn.

### Lịch sử phát triển

ReactJS được phát triển bởi Jordan Walke – một kỹ sư phần mềm tại Facebook vào năm 2012. Lúc đầu Facebook chỉ áp dụng vào các sản phẩm của họ nhưng sau đó không lâu, vào tháng 5 năm 2013 ReactJS đã được mã nguồn mở tại JSConf US.

### Ưu điểm của ReactJS

* Tốc độ phát triển ứng dụng nhanh.
* Dễ dàng bảo trì và mở rộng.
* ReactJS được thiết kế cho các ứng dụng yêu cầu hiệu suất cao nhờ việc cung cấp DOM ảo và kết xuất phía máy chủ.
* Dễ dàng sử dụng cho người có kiến thức cơ bản về JavaScript.
* Bên cạnh đó ReactNative là một phiên bản khác của ReactJS giúp xây dựng ứng dụng trên cả nền tảng Android và iOS.

### Nhược điểm của ReactJS

***Độ phức tạp***

React có đường cong học tập khá cao so với các framework khác do những đặc tính độc đáo như JSX, vòng đời component, states, props, v.v. Điều này có thể khiến việc tiếp cận và học ReactJS trở nên khó khăn, đặc biệt đối với những lập trình viên mới bắt đầu.

***Các vấn đề về hiệu suất***

Mặc dù React có tính năng Virtual DOM để cải thiện hiệu suất, nhưng khi ứng dụng trở nên phức tạp, việc quản lý và tối ưu hóa Virtual DOM có thể trở nên khó khăn.

***Những thách thức về SEO***

Ứng dụng React là single-page application, do đó việc index và tùy chỉnh nội dung trang web để tối ưu hóa cho SEO có thể trở nên phức tạp.

***Thiếu cấu trúc định hướng***

React chỉ cung cấp một thư viện để xây dựng giao diện người dùng, do đó nó không có một cấu trúc ứng dụng chuẩn như các framework khác như Angular hay Vue.js. Điều này có thể dẫn đến sự không đồng nhất trong cách tổ chức code.

***Kích thước bundle lớn***

Khi sử dụng React, kích thước bundle của ứng dụng có thể khá lớn, đặc biệt khi sử dụng nhiều thư viện bên thứ ba. Điều này có thể ảnh hưởng đến thời gian tải trang.

### So sánh ReactJS với các công nghệ khác

***Angular:***

* React là một thư viện JavaScript, trong khi Angular là một framework JavaScript hoàn chỉnh.
* React có cách tiếp cận "component-based", Angular sử dụng template HTML.
* React sử dụng JSX (kết hợp JavaScript và HTML), Angular sử dụng HTML và TypeScript.
* React có đường cong học tập thấp hơn Angular.
* React có tốc độ nhanh hơn và nhẹ hơn Angular.
* Angular cung cấp nhiều tính năng tích hợp sẵn, React cần phải tích hợp thêm các thư viện bên thứ ba.

***Vue.js***

* Cả hai đều là các thư viện JavaScript component-based.
* Cú pháp của Vue.js gần với HTML truyền thống hơn, trong khi React sử dụng JSX.
* Vue.js có đường cong học tập thấp hơn React.
* Vue.js có tốc độ nhanh hơn React.
* Vue.js được coi là dễ học và sử dụng hơn React.
* React có cộng đồng và tài liệu lớn hơn Vue.js.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU

## Phân tích vấn đề đồng bộ

### Tại sao cần đồng bộ sản phẩm từ bên thứ ba về hệ thống

***Tiết Kiệm Thời Gian và Nguồn Lực***

Việc tự động đồng bộ hóa giúp giảm bớt công việc thủ công, tiết kiệm thời gian và công sức so với việc nhập liệu thủ công.

***Tiềm Năng Mở Rộng***

Giải pháp có thể mở rộng để tích hợp thêm nhiều nền tảng khác trong tương lai, tạo ra một hệ thống quản lý sản phẩm mạnh mẽ và linh hoạt.

***Tích Hợp Đa Kênh Bán Hàng***

Đồng bộ sản phẩm từ nhiều nền tảng khác nhau giúp mở rộng kênh bán hàng, từ đó tiếp cận được nhiều khách hàng hơn.

Quản lý sản phẩm từ nhiều kênh bán hàng chỉ trong một hệ thống duy nhất, giúp giảm tải công việc và nâng cao hiệu quả quản lý.

***Đảm Bảo Tính Chính Xác***

Tự động đồng bộ giúp giảm thiểu sai sót do nhập liệu thủ công, đảm bảo dữ liệu luôn chính xác và nhất quán.

***Cập Nhật Liên Tục***

Dữ liệu sản phẩm từ các nền tảng thứ ba sẽ được cập nhật liên tục, đảm bảo rằng thông tin sản phẩm trên website của bạn luôn mới nhất.

***Quản Lý Tập Trung***

Tất cả sản phẩm từ các nền tảng khác nhau được tập trung quản lý tại một nơi, giúp việc theo dõi và quản lý trở nên dễ dàng hơn.

***Cải Thiện Trải Nghiệm Khách Hàng***

Với dữ liệu sản phẩm chính xác và cập nhật liên tục, khách hàng có trải nghiệm mua sắm tốt hơn, tăng khả năng giữ chân khách hàng và thúc đẩy doanh số.

### Khó khăn khi tích hợp đồng bộ sản phẩm

***Khó Khăn Khi Tích Hợp API***

* Đa Dạng API: Mỗi nền tảng như Shopify và ShopBase có cách triển khai API khác nhau, yêu cầu phải hiểu rõ từng API để tích hợp chính xác.
* Thay Đổi API: Các nền tảng thường cập nhật và thay đổi API, yêu cầu hệ thống phải liên tục cập nhật để đảm bảo tương thích.
* Hạn Chế API Rate Limit: Các API thường có giới hạn về số lượng yêu cầu có thể gửi trong một khoảng thời gian, cần quản lý tốt để tránh bị giới hạn.

***Xử Lý Dữ Liệu Phức Tạp***

* Định Dạng Dữ Liệu Khác Nhau: Dữ liệu từ các nền tảng có thể có cấu trúc và định dạng khác nhau, đòi hỏi chuyển đổi dữ liệu phức tạp.
* Dữ Liệu Lớn: Xử lý và đồng bộ hóa số lượng lớn sản phẩm đòi hỏi hệ thống phải có khả năng xử lý và lưu trữ dữ liệu lớn.
* Dữ Liệu Không Đồng Nhất: Sự không đồng nhất trong dữ liệu sản phẩm (ví dụ: tên, mô tả, giá cả) giữa các nền tảng cần được xử lý để tránh xung đột.

***Quản Lý Xung Đột Dữ Liệu***

* Cập Nhật Đồng Thời: Khi sản phẩm được cập nhật trên nhiều nền tảng cùng một lúc, cần có cơ chế để giải quyết xung đột dữ liệu.
* Đồng Bộ Thời Gian Thực: Đảm bảo rằng dữ liệu đồng bộ hóa được thực hiện thời gian thực hoặc gần thời gian thực để tránh sai lệch.

***Kiểm Thử Hệ Thống***

* Độ Phức Tạp: Kiểm thử tích hợp các API từ nhiều nền tảng yêu cầu một môi trường kiểm thử phức tạp để mô phỏng các tình huống thực tế.
* Đảm Bảo Tính Nhất Quán: Kiểm thử phải đảm bảo rằng dữ liệu sản phẩm được đồng bộ hóa đúng và nhất quán trên tất cả các nền tảng và hệ thống.

***Hiệu Năng Hệ Thống***

* Tải Cao: Đảm bảo rằng hệ thống có thể xử lý tải cao khi đồng bộ hóa nhiều sản phẩm từ nhiều nền tảng cùng một lúc.
* Thời Gian Phản Hồi: Duy trì thời gian phản hồi nhanh cho người dùng ngay cả khi hệ thống đang thực hiện đồng bộ hóa dữ liệu lớn.

***Đồng Bộ Hóa Hai Chiều***

* Tính Phức Tạp: Đồng bộ hóa hai chiều (từ nền tảng về hệ thống và ngược lại) thêm phức tạp và đòi hỏi quản lý cẩn thận để tránh xung đột và mất dữ liệu.
* Kiểm Soát Phiên Bản: Cần có cơ chế kiểm soát phiên bản để đảm bảo rằng thay đổi từ cả hai phía không gây xung đột dữ liệu.

## Xác định các yêu cầu chính của hệ thống

### Phân tích các yêu cầu

**Những quyền lợi đối với khách hàng**

* Xem trang tĩnh: Khách hàng có quyền xem nội dung trang tĩnh trên website.
* Đăng nhập: Khách hàng có thể đăng nhập hệ thống.
* Đăng ký: Khách hàng đăng kí vào hệ thống.
* Cập nhật thông tin: Khách hàng có quyền cập nhật thông tin cá nhân của bản thân.
* Xem sản phẩm: Khách hàng có quyền xem danh sách sản phẩm và chi tiết sản phẩm.
* Tìm kiếm sản phẩm: Khách hàng có thể tìm kiếm sản phẩm theo từ khoá là tên sản phẩm.
* Thêm vào giỏ hàng: Khách hàng có thể thêm bất kỳ sản phẩm nào vào giỏ hàng.
* Đặt hàng: Khách hàng có quyền đặt hàng trên website khi đăng nhập hoặc không cần đăng nhập.
* Tra cứu đơn hàng: Khách hàng xem được những đơn hàng mà mình đã đặt trước đó.
* Quản lý giỏ hàng: Khách hàng có quyền xem, thêm, sửa, xoá các sản phẩm trong giỏ hàng.

**Những quyền lợi đối với quản trị viên**

* Đăng nhập: Quản trị viên cần đăng nhập vào trang quản lý để thực hiện các chứ năng quản trị hệ thống.
* Đồng bộ sản phẩm từ Shopify, ShopBase: hệ thống tự đồng đồng bộ hoặc có thể tự đồng bộ bằng tay.
* Bảo trì danh mục: xem, thêm, sửa, xoá danh mục.
* Bảo trì sản phẩm: xem, thêm sửa xoá sản phẩm.
* Quản lý người dùng: xem, cập nhật quyền, cập nhật trạng thái tài khoản.
* Quản lý đơn hàng: xem, cập nhật trạng thái đơn hàng.
* Xem thống kê

### Các yêu cầu phi chức năng

* Hiệu năng cao, trải nghiệm mượt mà, ổn định.
* Giao diện thân thiện.
* Đáp ứng được lượng lớn truy cập đồng thời.
* Tối ưu hoá SEO.
* Dễ dàng bảo trì và mở rộng.

## Xây dựng biểu đồ use case

### Các tác nhân

* Người quản trị: Quản lý nội dung website, xử lý các yêu cầu từ khách hàng.
* Khách hàng: Tìm kiếm, xem thông sản phẩm, đặt hàng.

### Các use case của hệ thống

|  |  |
| --- | --- |
| **Tác nhân** | **Các ca sử dụng** |
| Khách hàng | Đăng nhập  Đăng ký  Cập nhật thông tin  Xem sản phẩm  Tìm kiếm sản phẩm  Thêm vào giỏ hàng  Đặt hàng  Quản lý giỏ hàng  Tra cứu đơn hàng |
| Người quản trị | Bảo trì danh mục  Bảo trì sản phẩm  Quản lý đơn hàng  Quản lý người dung  Đồng bộ sản phẩm  Xem thống kê |

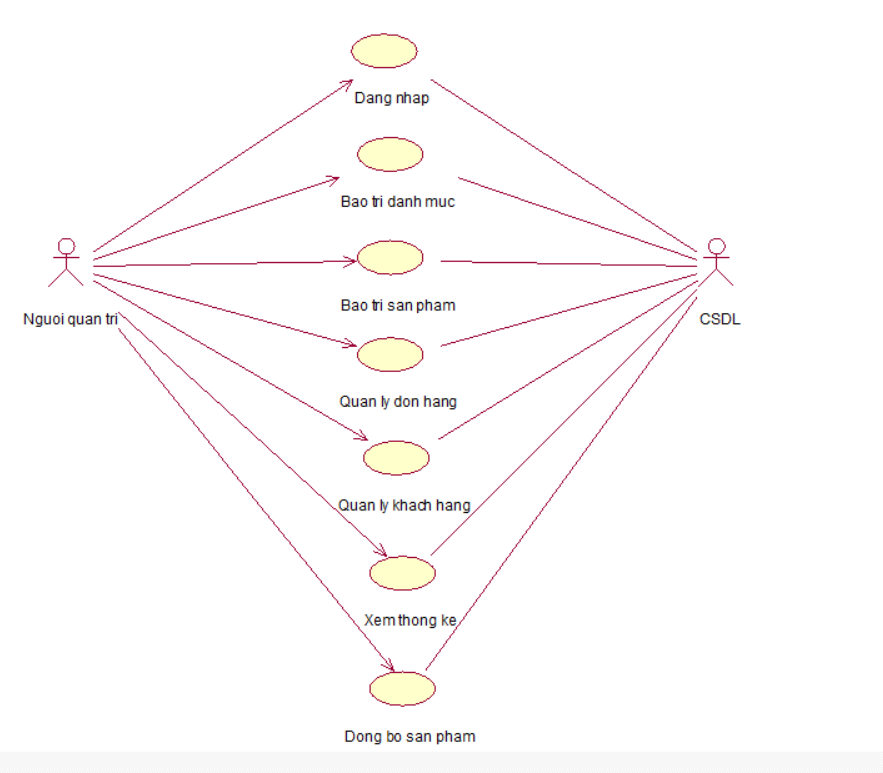
Bảng 2.3.1 Các use case hệ thống

### Biểu đồ use case

A diagram of a network

Description automatically generated

Hình 2.3.1. Các use case phía khách hàng



Hình 2.3.2. Các use case phía người quản trị

### Mô tả chi tiết các use case

#### Use case Đăng nhập

* **Mô tả vắn tắt**: Use case này cho phép khách hàng đăng nhập vào hệ thống.
* Luồng sự kiện:
  + Luồng cơ bản:

1. Use ase này bắt đầu khi khách hàng nhấn vào nút đăng nhập. Hệ thống sẽ yêu cầu khách hàng nhập email và mật khẩu.
2. Khách hàng nhập email và mật khẩu sau đó nhấn vào nút “Đăng nhập”. Hệ thống sẽ kiểm tra email, mật khẩu của khách hàng trong bảng USERS, nếu có hệ thống sẽ thông báo đăng nhập thành công và chuyển hướng đến trang chủ.

Use case kết thúc.

* + Luồng rẽ nhánh

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 2 trong luồng cơ bản, khi người dùng nhập thiếu thông tin hoặc thông tin không hợp lệ hoặc không tìm thấy bản ghi trong bảng USERS. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

* Các yêu cầu đặc biệt:

Khách hàng cần nhập đúng thông tin để yêu cầu đăng nhập.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

#### Use case Đăng ký

* Mô tả vắn tắt: Use case này cho phép người dùng đăng ký tài khoản trên hệ thống.
* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích chuột vào nút “Đăng nhập”. Hệ thống điều hướng sang màn hình chứa form “Đăng nhập”. Hệ thống yêu cầu nhập thông tin đăng ký: name, email, password.
2. Khách hàng nhập name, email và password sau đó nhấn vào nút “Create an Accounts”. Hệ thống sẽ kiểm tra email, password của khách hàng trong bảng USERS, nếu chưa có hệ thống sẽ gửi mail xác thực tài khoản đến tài khoản vừa đăng ký. Sau khi người dùng xác thực tài khoản thành công hệ thống sẽ thông báo “Đăng ký thành công” và chuyển hướng đến trang chủ.

Use case kết thúc.

* + Luồng rẽ nhánh

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 2 trong luồng cơ bản, khi người dùng nhập thiếu thông tin, thông tin không hợp lệ hoặc email đã được đăng ký trong bảng USERS. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

* Các yêu cầu đặc biệt:

Khách hàng cần nhập đúng thông tin để yêu cầu đăng nhập.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

#### Use case Cập nhật thông tin

* Mô tả vắn tắt:

Use case này cho phép người dùng thay đổi thông tin về họ tên, ngày sinh, giới tính, địa chỉ, số điện thoại.

* Luồng sự kiện
  + Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích chuột vào trang cá nhân và nhấn vào nút “Thông tin cá nhân” Trang sẽ điều hướng sang màn hình “Thông tin cá nhân” và hiển thị thông tin cá nhân của khách hàng bao gồm: username, email từ bảng USERS sau đó hiển thị lên màn hình.
2. Sửa thông tin cá nhân: User nhập thông tin muốn thay đổi và nhấn vào nút “Cập nhật”. Hệ thống cập nhật thông tin vừa thay đổi vào bảng USERS, sau đó lấy thông tin cá nhân của người dùng đang đăng nhập và hiển thị lên màn hình.
   * Luồng rẽ nhánh:
3. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi.

Use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

* Hậu điều kiện:

Sau khi use case kết thúc thành công thì các thông tin cá nhân của người dùng sẽ được cập nhật ở bảng USERS.

* Điểm mở rộng:

Không có.

#### Use case Xem sản phẩm

* Mô tả vắn tắt: Use case này cho phép khách hàng xem thông tin sản phẩm.
* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi người dùng nhấn vào một sản phẩm. Hệ thống sẽ lấy danh sách sản phẩm, ảnh, mô tả, đánh giá, của sản phẩm từ bảng PRODUCTS, PRODUCT\_IMAGES để hiển thị lên màn hình.

Use case kết thúc.

* + Luồng rẽ nhánh

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị lên một thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt

Không có

* Tiền điều kiện

Không có

* **Hậu điều kiện**

Không có

* Điểm mở rộng:

Không có

#### Use case Tìm kiếm sản phẩm

* Mô tả vắn tắt: Use này cho phép khách hàng tìm kiếm sản phẩm theo từ khoá là tên sản phẩm.
* Luồng sự kiện:
  + **Luồng cơ bản**:

1. Use case này bắt đầu khi khách hàng nhập tên sản phẩm vào textbox tìm kiếm trên thanh input tìm kiếm. Sau 0,8s hệ thống sẽ hiển thị ra các sản phẩm có tên chứa những từ mà người dùng đã nhập và hiển thị thông tin sản phẩm ra màn hình (tên, hình ảnh, mô tả, tồn kho, giá tiền) từ bảng PRODUCTS, PRODUCT\_IMAGES.

Use case kết thúc.

* **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị lên thông báo lỗi.
2. Tại bước 1 trong luồng cơ bản, nếu hệ thống không tìm được sản phẩm nào phù hợp với text đã nhập thì hệ thống sẽ hiển thị kết là “Không tìm thấy sản phẩm nào”. Use case kết thúc.

* Các yêu cầu đặc biệt

Không có.

* Tiền điều kiện

Không có.

* Hậu điều kiện

Không có.

* Điểm mở rộng

Không có.

#### Use case Thêm vào giỏ hàng

* Mô tả vắn tắt: Use case này cho phép khách hàng thêm bất kỳ sản phẩm nào vào giỏ hàng
* Luồng sự kiện
  + **Luồng cơ bản:**

1. Người dùng nhấn vào nút “Thêm vào giỏ hàng” trong chi tiết sản phẩm sau khi chọn số lượng cần mua hoặc icon giỏ hang ngay Item.
2. Hệ thống sẽ lấy ra thông tin các sản phẩm người dùng đã thêm vào giỏ hàng của mình từ bảng “ORDER\_DETALS”, nếu sản phẩm muốn thêm đã tồn tại trong giỏ hàng hệ thống sẽ cộng số lượng sản phẩm muốn thêm vào sản phẩm trong giỏ hàng, nếu sản phẩm muốn thêm chưa tồn tại hệ thống sẽ tạo mới một bản ghi và lưu vào bảng “ORDER \_DETAILS”

* **Luồng rẽ nhánh**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi.
2. Tại bước 1 trong luồng cơ bản, nếu người dùng chưa chọn số lượng sản phẩm muốn thêm vào giỏ hàng. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

Use case kết thúc.

* Các yêu cầu đặc biệt

Không có

* Tiền điều kiện

Không có

* Hậu điều kiện

Không có

* Điểm mở rộng:

Không có

#### Use case Đặt hàng

* Mô tả vắn tắt: Use case này cho phép khách hàng đặt những sản phẩm trên trang web.
* Luồng sự kiện:
  + **Luồng cơ bản:**

1. Tại màn hình giỏ hàng, người dùng bấm vào nút “Thanh toán”, hệ thống sẽ chuyển sang màn hình thông tin khách hàng và thông tin của đơn đặt hàng.
2. Người dùng tiến hành nhập đầy đủ thông tin: Người nhận, Số điện thoại, Địa chỉ nhận hàng, Ghi chú.
3. Người dùng nhấn nút “Thanh toán” hệ thống sẽ thực hiện tạo thêm bản ghi mới vào bảng ORDERS, ORDER\_DETAILS và cập nhật lại thông tin bảng PRODUCTS sau đó hiển thị “Đặt hàng thành công”.
4. Hệ thống gửi mail đến người dùng thông tin đơn hàng vừa được người dùng đặt.

Use case kết thúc.

* **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống hiển thị thông báo lỗi.
2. Tại bước 2 trong luồng cơ bản, nếu người dùng đã đăng nhập thì sẽ hiển thị thông tin của người dùng đã đăng nhập lên màn hình. Nếu người dùng không nhập đủ thông tin thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.
3. Tại bước 3 trong luồng cơ bản, nếu số lượng sản phẩm tồn kho nhỏ hơn số lượng đặt hàng thì sẽ hiển thị thông báo lỗi “Số lượng không hợp lệ” và use case kết thúc.

* Các yêu cầu đặc biệt

Không có

* Tiền điều kiện

Khách hàng cần cung cấp đầy đủ thông tin để có thể đặt hàng.

* Hậu điều kiện

Khi use case kết thúc thành công, các thông tin sẽ được lưu trữ vào bảng ORDERS, ORDER\_DETAILS

* Điểm mở rộng:

Không có

#### Use case Đặt hàng bằng VnPay

* + **Luồng cơ bản:**

1. Tại màn hình giỏ hàng, người dùng bấm vào nút “Thanh toán”, hệ thống sẽ chuyển sang màn hình thông tin khách hàng và thông tin của đơn đặt hàng.
2. Người dùng tiến hành nhập đầy đủ thông tin: Người nhận, Số điện thoại, Địa chỉ nhận hàng, Ghi chú.
3. Ở mục “Phương thức thanh toán”, người dùng ấn thanh toán bằng VnPay hệ thống sẽ redirect tới phần giao diện thanh toán của VnPay.
4. Người dùng nhập thông tin như Số thẻ, chủ thẻ, ngày phát hành, xong đó sẽ nhập thêm mã OTP được VnPay gửi về.
5. Người dùng nhấn nút “Thanh toán” hệ thống VnPay sẽ trả về một thông báo thành công tới hệ thống, hệ thống sẽ thực hiện tạo thêm bản ghi mới vào bảng ORDERS, ORDER\_DETAILS và cập nhật lại thông tin bảng PRODUCTS sau đó hiển thị “Đặt hàng thành công”.
6. Hệ thống gửi mail đến người dùng thông tin đơn hàng vừa được người dùng đặt.

Use case kết thúc.

* **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống hiển thị thông báo lỗi.
2. Tại bước 2 trong luồng cơ bản, nếu người dùng đã đăng nhập thì sẽ hiển thị thông tin của người dùng đã đăng nhập lên màn hình. Nếu người dùng không nhập đủ thông tin thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.
3. Tại bước 3 trong luồng cơ bản, nếu số lượng sản phẩm tồn kho nhỏ hơn số lượng đặt hàng thì sẽ hiển thị thông báo lỗi “Số lượng không hợp lệ” và use case kết thúc.

* Các yêu cầu đặc biệt

Không có

* Tiền điều kiện

Khách hàng cần cung cấp đầy đủ thông tin để có thể đặt hàng.

* Hậu điều kiện

Khi use case kết thúc thành công, các thông tin sẽ được lưu trữ vào bảng ORDERS, ORDER\_DETAILS

* Điểm mở rộng:

Không có

#### Use case Tra cứu đơn hàng

* Mô tả vắn tắt: Use case này cho phép người dùng xem được những đơn hàng mà mình đã đặt trước đó.
* Luồng sự kiện:
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng nhấn vào nút “Trang cá nhân” trên thanh Navigation. Hệ thống sẽ thực hiện lấy thông tin về các đơn hàng của người dùng đang đăng nhập từ bảng ORDERS và ORDER\_DETAILS sau đó hiển thị lên màn hình.
2. Người dùng nhấn vào đơn hàng, hệ thống sẽ hiển thị thông tin chi tiết của đơn hàng đó bao gồm thông tin người mua, danh sách sản phẩm trong đơn hàng từ bảng ORDERS và ORDER\_DETAILS.

* **Luồng rẽ nhánh**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được tới cơ sở dữ liệu thì sẽ hiển thị thông báo lỗi.
2. Tại bước 1 trong luồng rẽ nhánh, nếu người dùng chưa đăng nhập hệ thống sẽ chuyển hướng sang trang đăng nhập.

Use case kết thúc.

* Các yêu cầu đặc biệt

Không có

* Tiền điều kiện

Người dùng đăng nhập hệ thống

* Hậu điều kiện

Không có

* Điểm mở rộng:

Không có

#### Use case Quản lý giỏ hàng

* **Mô tả vắn tắt:** Use case này cho phép khách hàng xem, thêm, sửa, xoá sản phẩm ra khỏi giỏ hàng.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích vào nút “Giỏ hàng” trên menu. Hệ thống sẽ  lấy thông tin chi tiết giỏ hàng của người dùng đó gồm: mã giỏ hàng từ bảng “ORDER” và danh sách các sản phẩm thuộc giỏ hàng này từ bảng “ORDER\_DETAILS” và bảng “PRODUCTS”, hiển thị danh sách các sản phẩm lên màn hình.
2. Xoá sản phẩm trong giỏ hàng: Người dùng nhấn vào biểu tượng xoá trên màn hình, hệ thống sẽ thực hiện loại bỏ sản phẩm đó ra khỏi giỏ hàng.

* **Luồng rẽ nhánh**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi.
2. Tại bước 1 trong luồng cơ bản, nếu chưa có sản phẩm nào trong giỏ hàng sẽ hiển thị “Giỏ hàng trống”.

Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Không có

* **Hậu điều kiện**

Không có

* **Điểm mở rộng:**

Không có

#### Use case Bảo trì danh mục

* **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem, thêm, sửa, xoá danh mục.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng nhấn vào nút “Danh mục” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng CATEGORIES gồm các thông tin: id, tên, mô tả và hiển thị lên màn hình.
2. Thêm danh mục mới:
   1. Người quản trị nhấn vào nút “Thêm danh mục” trong màn hình danh sách danh mục. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho danh mục gồm tên danh mục và mô tả cho danh mục.
   2. Người quản trị nhập thông tin cho danh mục mới và nhấn vào nút “Gửi”. Hệ thống sẽ thêm một bản ghi mới vào bảng CATEGORIES sau đó lấy thông tin về danh sách danh mục mới gồm id, tên và hiển thị lên màn hình.
3. Sửa danh mục:
   1. Người quản trị nhấn vào biểu tưởng chỉnh sửa trên một dòng danh mục. Hệ thống sẽ lấy ảnh, tên, mô tả của danh mục được chọn từ bảng CATEGORIES.
   2. Người quản trị sẽ sửa tên, mô tả của danh mục sau đó nhấn vào nút “Cập nhật”. Hệ thống cập nhật thông tin danh mục vào bảng CATEGORIES sau đó lấy thông tin danh sách danh mục gồm id, tên và hiển thị lên màn hình.
4. Xoá danh mục:
   1. Người quản trị nhấn vào biểu tượng xoá trên một dòng danh mục. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.
   2. Người quản trị nhấn vào nút “Gửi”, hệ thống sẽ thực hiện xoá danh mục đã chọn khỏi bảng CATEGORIES sau đó lấy thông tin danh sách danh mục gồm id, tên và hiển thị lên màn hình.

Use case kết thúc.

* **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng CATEGORIES thì hệ thống sẽ hiển thị “Không có danh mục sản phẩm” và use case kết thúc.
3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin danh mục không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các danh mục trong bảng CATEGORIES lên màn hình và use case kết thúc.

* **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

* **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

* **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng CATEGORIES

* **Điểm mở rộng**

Không có

#### Use case Bảo trì sản phẩm

* **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem, thêm, sửa, xoá sản phẩm.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng nhấn vào nút “Sản phẩm” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng PRODUCTS, CATEGORIES gồm các thông tin: id, name, stock, price, brand, category, thumbnail và hiển thị lên màn hình.
2. Thêm sản phẩm mới:
   1. Người quản trị nhấn vào nút “Add product” trong màn hình danh sách sản phẩm. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho sản phẩm gồm tên sản phẩm, tồn kho, giá, hãng, danh mục, ảnh.
   2. Người quản trị nhập thông tin cho sản phẩm mới và nhấn vào nút “Gửi”. Hệ thống sẽ thêm một bản ghi mới vào bảng PRODUCTS sau đó lấy thông tin về danh sách sản phẩm mới gồm id, name, stock, price, brand, category, thumbnail và hiển thị lên màn hình.
3. Sửa sản phẩm:
   1. Người quản trị nhấn vào icon “Sửa” trên một dòng sản phẩm. Hệ thống sẽ tên sản phẩm, tồn kho, giá, hãng, danh mục, ảnh từ bảng PRODUCTS.
   2. Người quản trị sẽ sửa các thông tin của sản phẩm sau đó nhấn vào nút “Gửi”. Hệ thống cập nhật thông tin sản phẩm vào bảng PRODUCTS sau đó lấy thông tin danh sách sản phẩm gồm tên sản phẩm, tồn kho, giá, hãng, danh mục, ảnh và hiển thị lên màn hình.
4. Xoá sản phẩm:
   1. Người quản trị nhấn vào biểu tượng xoá trên một dòng sản phẩm. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.
   2. Người quản trị nhấn vào nút “Đồng ý”, hệ thống sẽ thực hiện xoá thương hiệu đã chọn khỏi bảng PRODUCTS sau đó lấy thông tin danh sách sản phẩm gồm tên sản phẩm, tồn kho, giá, hãng, danh mục, ảnh và hiển thị lên màn hình.

Use case kết thúc.

* **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng PRODUCTS thì hệ thống sẽ hiển thị “Không có sản phẩm” và use case kết thúc.
3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin sản phẩm không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các sản phẩm trong bảng PRODUCTS lên màn hình và use case kết thúc.

* **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

* **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

* **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng PRODUCTS.

* **Điểm mở rộng**

Không có

#### Use case Quản lý đơn hàng

* **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem, thay đổi trạng thái đơn hàng.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng nhấn vào nút “Quản lý đơn hàng” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng ORDERS, ORDER\_DETAILS gồm các thông tin: id, người mua, số điện thoại, địa chỉ, trạng thái, tổng giá trị, tổng số lượng sản phẩm, ngày tạo và hiển thị lên màn hình.
2. Xem chi tiết đơn hàng:
   1. Người quản trị nhấn vào biểu tượng chi tiết trên một dòng của đơn hàng. Hệ thống sẽ lấy ra thông tin đơn hàng gồm mã, trạng thái, người mua, số điện thoại, địa chỉ nhận hàng, danh sách các sản phẩm trong giỏ hàng từ bảng ORDER, ORDER\_DETAIL, PRODUCTS và hiển thị lên màn hình.
3. Cập nhật trạng thái đơn hàng:
   1. Người quản trị nhấn vào biểu tượng cập nhật trên một dòng đơn đặt hàng. Hệ thống sẽ thay đổi trạng thái của đơn hàng theo lựa chọn của người quản trị. Các trạng thái là:
4. Huỷ đơn hàng:
   1. Người quản trị nhấn vào biểu tượng huỷ trên một dòng thương hiệu. Hệ thống sẽ thay đổi trạng thái của đơn hàng sang đã huỷ và hiển thị lên màn hình.

Use case kết thúc.

* **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

* **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

* **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng ORDERS.

* **Điểm mở rộng**

Không có

#### Use case Xem thống kê

* **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem thống kê về sản phẩm đang thịnh hành theo thời gian, theo độ tuổi.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người Quản trị nhấn vào nút “Xem thống kê” trên thanh menu. Hệ thống sẽ hiển thị ra form nhập mức tuổi, khoảng thời gian để lọc ra những sản phẩm đang được quan tâm nhất.

Use case kết thúc.

* **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

* **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

* **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

* **Hậu điều kiện**

Không có

* **Điểm mở rộng**

Không có

#### Use case Quản lý người dùng

* **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem, khóa người dùng.
* **Luồng sự kiện**
  + **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng nhấn vào nút “Quản lý người dùng” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng USERS gồm các thông tin: id, số điện thoại, email, họ tên, giới tính, quyền và hiển thị lên màn hình.
2. Khóa người dùng:
   1. Người quản trị nhấn vào nút “Khóa” trong màn hình danh sách người dùng. Hệ thống sẽ khóa thông tin người dùng đó
3. Khóa người dùng:
   1. Người quản trị nhấn vào biểu tượng khóa trên một dòng người dùng. Hệ thống sẽ hiển thị lên popup để xác nhận kh.
   2. Người quản trị nhấn vào nút “Đồng ý”, hệ thống sẽ thực hiện xoá thương hiệu đã chọn khỏi bảng USERS sau đó lấy thông tin danh sách sản phẩm gồm họ tên, email, ngày sinh, số điện thoại, địa chỉ, giới tính và hiển thị lên màn hình.

Use case kết thúc.

* **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng USERS thì hệ thống sẽ hiển thị “Không có người dùng” và use case kết thúc.
3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin người dùng không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các sản phẩm trong bảng USERS lên màn hình và use case kết thúc.

* **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

* **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

* **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng USERS

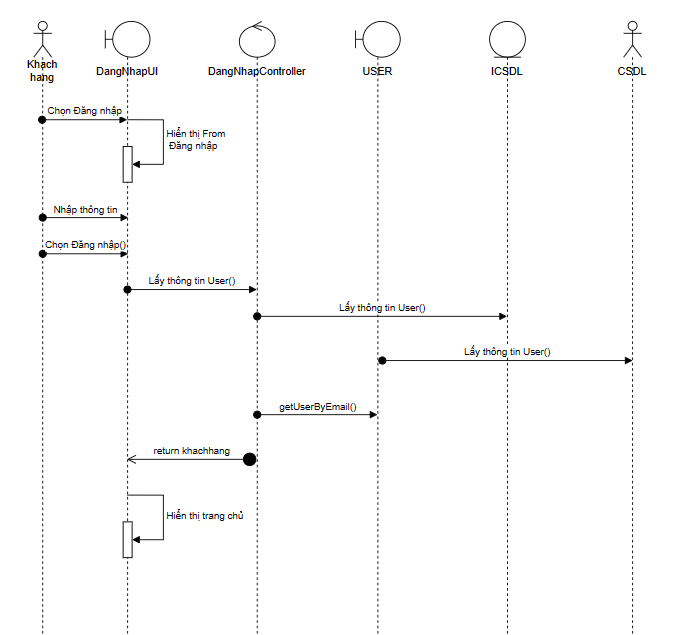
* **Điểm mở rộng**

Không có

### Phân tích các use case

#### Use case Đăng nhập

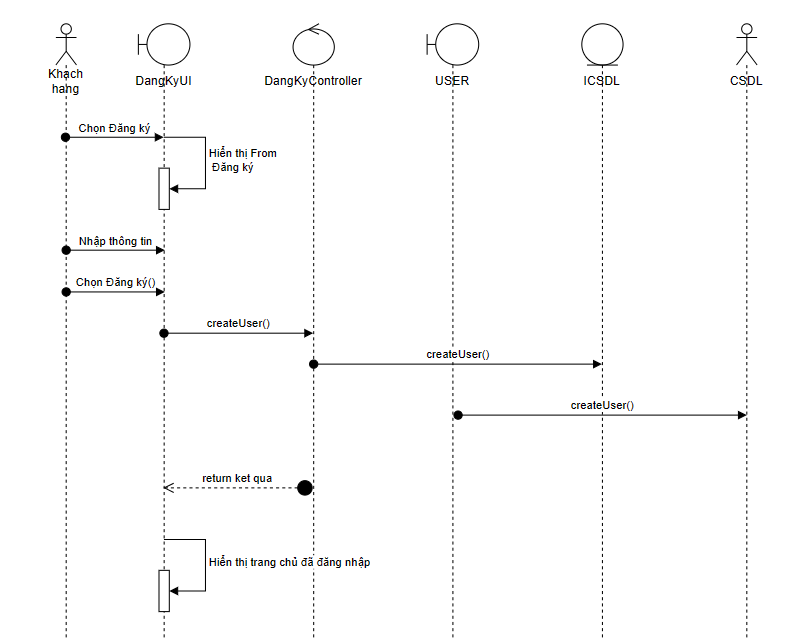
* **Biểu đồ trình tự**



Hình 2.3.3. Biểu đồ trình tự use case Đăng nhập

#### Use case Đăng ký

* **Biểu đồ trình tự**



Hình 2.3.4. Biểu đồ trình tự use case Quản lý địa chỉ

#### Use case Xem sản phẩm

* **Biểu đồ trình tự**

A black background with white text

Description automatically generated

Hình 2.3.5. Biểu đồ trình tự use case Xem sản phẩm

#### Use case Tìm kiếm sản phẩm

* **Biểu đồ trình tự**

A black screen with white text

Description automatically generated

Hình 2.3.6. Biểu đồ trình tự use case Tìm kiếm sản phẩm

#### Use case Thêm vào giỏ hàng

Biểu đồ trình tự

A black screen with white text

Description automatically generated

Hình 2.3.7 Biểu đồ trình tự use case Thêm vào giỏ hàng

#### Use case Đặt hàng

* **Biểu đồ trình tự**

A black background with white text

Description automatically generated

Hình 2.3.8. Biểu đồ trình tự use case Đặt hàng

#### Use case Đặt hàng VnPay

Mô hình kết nối

A diagram of a shopping cart

Description automatically generated

Hình 2.3.9 Mô hình kết nối với VnPay

Bước 1: Khách hàng thực hiện mua hàng trên Website - ứng dụng TMĐT và tiến hành thanh toán trực tuyến cho đơn hàng.

Bước 2: Website - ứng dụng TMĐT thành lập yêu cầu thanh toán dưới dạng URL mang thông tin thanh toán và chuyển hướng khách hàng sang Cổng thanh toán VNPAY bằng URL đó.

Cổng thanh toán VNPAY xử lý yêu cầu thanh toán mà Website - ứng dụng TMĐT gửi sang. Khách hàng tiến hành nhập hoặc xử lý xác thực các thông tin được yêu cầu Thanh toán.

Bước 3,4: Khách hàng nhập thông tin để xác minh tài khoản Ngân hàng của khách hàng và xác thực giao dịch (Nhập thông tin tài khoản, thẻ hoặc quét mã VNPAY-QR).

Bước 5: Giao dịch thành công tại Ngân hàng, VNPAY tiến hành:

Chuyển hướng khách hàng về Website - ứng dụng TMĐT (vnp\_ReturnUrl)

Thông báo cho Website - ứng dụng TMĐT kết quả thanh toán của khách hàng thông qua IPN URL. Merchant cập nhật kết quả thanh toán VNPAY gửi tại URL này.

Bước 6: Merchant hiển thị kết quả giao dịch tới khách hàng (vnp\_ReturnUrl).

Sơ đồ tuần tự   
A diagram of a computer program

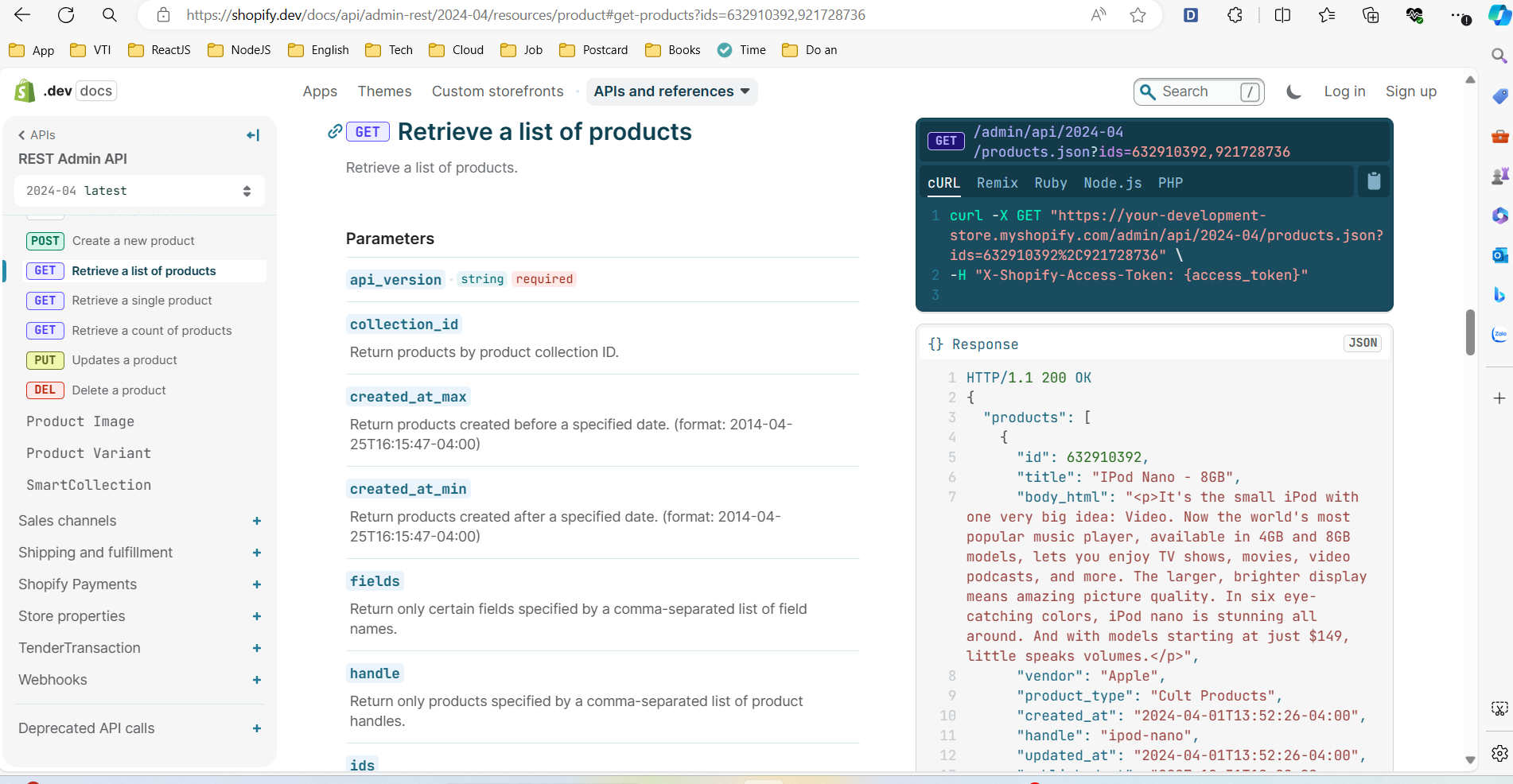
Description automatically generated with medium confidence

Hình 2.3.10. Sơ đồ tuần tự (Nguồn VnPay)

#### Use case Đồng bộ sản phẩm từ Shopify

***Các bước tiến hành đồng bộ***

* Đầu tiên chúng ta phải có 1 tài khoản Shopify
* Sau đó chúng ta sẽ tạo Store trong Shopify
* Tiếp theo sẽ lấy “X-Shopify-Access-Token” của Store đó (tham khảo tài liệu [https://shopify.dev/docs/api/admin-rest#authentication](https://shopify.dev/docs/api/admin-rest%23authentication) <https://help.shopify.com/en/manual/apps/app-types/custom-apps>)
* Sau đó chúng ta sẽ phân quyền cho “X-Shopify-Access-Token” được quyền gì trong Shopify ở đây chỉ cần quyền “read\_products” là đủ

Sau khi lấy được “X-Shopify-Access-Token”, vào docx development của Shopify để xem API lấy ra tất cả sản phẩm và phân tích để cho việc đồng bộ về hệ thống được thành công (tham khảo tài liệu <https://shopify.dev/docs/api/admin-rest/2024-04/resources/product>)  


Hình 2.3.11 Hình ảnh API Docx Product của Shopify

Biểu đồ trình tự

A diagram of a shopify process

Description automatically generated with medium confidence

Hình 2.3.12 Biểu đồ trình tự Use case tự động đồng bộ sản phẩm từ Shopify về hệ thống

#### Use case Đồng bộ sản phẩm từ ShopBase

***Các bước tiến hành đồng bộ***

* Đầu tiên chúng ta phải có 1 tài khoản ShopBase
* Sau đó chúng ta sẽ tạo App Private trong ShopBase (tham khảo tài liệu: <https://developers.shopbase.com/build-an-app/making-your-first-request/authentication/private-apps>)
* Sau đó chúng ta sẽ phân quyền cho App được quyền gì trong Shopify ở đây chỉ cần quyền “read\_products” là đủ
* Ở màn hình Private App sau khi được tạo, mình sẽ có được “API key” và “Password”, mình sẽ lấy 2 thông tin này để truy cập tới API mà ShopBase cho



Hình 2.3.13 Màn hình Private App của ShopBase

* Sau khi lấy được “API key” và “Password” của Private App ta sẽ dùng chúng làm Access Token, vào docx development của ShopBase để xem API lấy ra tất cả sản phẩm và phân tích để cho việc đồng bộ về hệ thống được thành công

(tham khảo tài liệu <https://api-doc.shopbase.com/#tag/Product/operation/get-all-products>)

Biểu

Biểu đồ trình tự

A diagram with arrows and text

Description automatically generated with medium confidence

Hình 2.3.14 Biểu đồ trình tự Use case tự động đồng bộ sản phẩm từ ShopBase về hệ thống

#### Use case Tra cứu đơn hàng

* **Biểu đồ trình tự**

A black background with white text

Description automatically generated

Hình 2.3.15. Biểu đồ trình tự use case Tra cứu đơn hàng

#### Use case Quản lý giỏ hàng

* **Biểu đồ trình tự**

A black background with white text

Description automatically generated

Hình 2.3.16 Biểu đồ trình tự use case Quản lý giỏ hàng

#### Use case Bảo trì danh mục

* **Biểu đồ trình tự**

A black background with white lines

Description automatically generated

Hình 2.3.17. Biểu đồ trình tự use case Bảo trì danh mục

#### Use case Bảo trì sản phẩm

* **Biểu đồ trình tự**

A diagram of a product

Description automatically generated

A diagram of a product

Description automatically generated

A diagram with lines and dots

Description automatically generated

A diagram of a product

Description automatically generated

A close-up of a diagram

Description automatically generated

Hình 2.3.18 Biểu đồ trình tự use case Bảo trì sản phẩm

#### Use case Quản lý đơn hàng

* **Biểu đồ trình tự**

A diagram of a product

Description automatically generated

A white sheet with black lines and black dots

Description automatically generated with medium confidence

A white sheet of paper with a grid

Description automatically generated

Hình 2.3.19 Biểu đồ trình tự use case Quản lý đơn hàng

#### Use case Xem thống kê

* **Biểu đồ trình tự**

A diagram of a diagram

Description automatically generated

Hình 2.3.20 Biểu đồ trìnhh tự use case Xem thống kê

#### Use case Quản lý người dùng

* **Biểu đồ trình tự**

A black background with white text

Description automatically generated

Hình 2.3.21 Biểu đồ trình tự use case Quản lý người dùng

## Xây dựng cơ cở dữ liệu

### Biểu đồ Entity Relationship Diagram

A screenshot of a computer

Description automatically generated

Hình 2.4.1. Thiết kế cơ sở dữ liệu

### Các bảng trong cơ sở dữ liệu

**Bảng USERS** (Lưu tài khoản)

A screenshot of a computer

Description automatically generated

Hình 2.4.2. Bảng users

**Bảng CATEGORIES** (Lưu danh mục sản phẩm)

A screenshot of a computer

Description automatically generated

Hình 2.44. Bảng categories

**Bảng BRANCHES** (Lưu hãng sản phẩm)

A screenshot of a computer

Description automatically generated

Hình 2.45. Bảng branhes

**Bảng ORDERS** (Thông tin giỏ hàng và đơn hàng)

A screenshot of a computer

Description automatically generated

Hình 2.4.3. Bảng orders

**Bảng ORDER\_DETAILS** (Lưu thông tin chi tiết đơn hàng)

A screenshot of a computer

Description automatically generated

Hình 2.4.4. Bảng order\_details

**Bảng PRODUCTS** (Lưu thông tin sản phẩm)

A screenshot of a computer

Description automatically generated

Hình 2.48. Bảng products

**Bảng PRODUCT\_IMAGES** (Lưu các hình ảnh của sản phẩm)

A screenshot of a computer

Description automatically generated

Hình 2.49. Bảng product\_images

# KẾT QUẢ ĐẠT ĐƯỢC

## Giao diện chương trình

### Giao diện phía khách hàng

#### Giao diện trang chủ

A screenshot of a smart watch

Description automatically generated

A screenshot of a smart watch

Description automatically generated

A screenshot of a computer

Description automatically generated

Hình 3.1.1 Màn hình trang chủ

#### Màn hình xem sản phẩm theo danh mục

A screenshot of a computer

Description automatically generated

Hình 3.1.2 Màn hình xem sản phẩm theo danh mục

#### Màn hình xem chi tiết sản phẩm

A screenshot of a smart watch

Description automatically generated

Hình 3.1.3 Màn hình xem chi tiết sản phẩm

#### Màn hình giỏ hàng

A screenshot of a computer

Description automatically generated

Hình 3.1.4 Màn hình giỏ hàng

#### Màn hình thông tin thanh toán

A screenshot of a computer

Description automatically generated

Hình 3.1.5 Màn hình thông tin thanh toán

#### Màn hình thông tin cá nhân

A screenshot of a computer

Description automatically generated

Hình 3.1.6 Màn hình thông tin cá nhân

3.1.1.7. Màn hình đăng ký

A screenshot of a login form

Description automatically generated

Hình 3.1.7 Màn hình đăng ký

### Giao diện phía quản lí

#### Màn hình dashboard

A screenshot of a computer

Description automatically generated

Hình 3.1.8 Màn hình dashboard

#### Màn hình danh sách khách hàng

A screenshot of a computer

Description automatically generated

Hình 3.1.9 Màn hình danh sách khách hàng

#### Màn hình danh sách sản phẩm

A screenshot of a computer

Description automatically generated

Hình 3.1.10 Màn hình danh sách sản phẩm

#### Màn hình danh sách các danh mục

A screenshot of a computer

Description automatically generated

Hình 3.1.11 Màn hình danh sách các danh mục

#### Màn hình danh sách các hãng

A screenshot of a computer

Description automatically generated

Hình 3.1.12 Màn hình danh sách các hãng

#### Màn hình quản lý đơn hàng

A screenshot of a computer

Description automatically generated

Hình 3.1.13 Màn hình quản lý đơn hàng

## Kiểm thử hệ thống

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| STT | | Chức năng | Trường hợp kiểm thử | Đầu vào | Đầu ra thực tế | Đầu ra mong đợi | Kết quả |
| 1 | Xác thực người dùng | | Đăng nhập tài khoản | Nhập tài khoản | Chuyển hướng sang trang chủ |  | Pass |
| Nhập dữ liệu không hợp lệ | Hiển thị thông báo lỗi |  | Pass |
| Đăng xuất tài khoản | Click nút “Đăng xuất” | Chuyển hướng trang đăng nhập |  | Pass |
| Nhập dữ liệu không hợp lệ | Hiển thị thông báo lỗi |  | Pass |
| 2 | Đăng ký người dùng | | Đăng ký tài khoản | Nhập dữ liệu hợp lệ | Chuyển hướng sang trang chủ |  | Pass |
| Nhập dữ liệu không hợp lệ | Hiển thị thông báo lỗi |  | Pass |
| 3 | Xem sản phẩm | | Xem danh sách sản phẩm | Vào màn danh sách sản phẩm | Hiển thị danh sách sản phẩm |  | Pass |
| Nhập dữ liệu không hợp lệ | Hiển thị thông báo lỗi |  | Pass |
| Xem chi tiết sản phẩm | Nhấn vào sản phẩm | Hiển thị thông tin chi tiết của từng sản phẩm. |  | Pass |
| 4 | Thêm vào giỏ hàng | | Thêm vào giỏ hàng | Nhấn thêm sản phẩm vào giỏ hàng | Hiển thị thêm giỏ hàng thành công |  | Pass |
|  | | Xem chi tiết giỏ hàng | Vào màn giỏ hàng | Hiển thị chi tiết giỏ hàng của bạn |  | Pass |
| 5 | Đặt hàng | | Đặt hàng COD | Nhập thông tin, ấn đặt hàng | Hiển thị đặt hàng thành công |  | Pass |
| Sản phẩm hết tồn kho | Hiển thị thông báo lỗi |  | Pass |
| Đặt hàng VnPay | Nhập thông tin, ấn đặt hàng | Hiển thị đặt hàng thành công |  | Pass |
|  |  | | Sản phẩm hết tồn kho | Hiển thị thông báo lỗi |  | Pass |
| 6 | Đồng bộ sản phẩm | | Đồng bộ từ Shopify | Nhập sản phẩm mới bên Shopify | Hiển thị thông tin sản phẩm mới bên hệ thống |  | Pass |
| Sửa đổi sản phẩm bên Shopify | Hiển thị thông tin sản phẩm được sửa bên hệ thống |  | Pass |
| Đồng bộ từ ShopBase | Nhập sản phẩm mới bên Shopify | Hiển thị thông tin sản phẩm mới bên hệ thống |  | Pass |
| Sửa đổi sản phẩm bên Shopify | Hiển thị thông tin sản phẩm được sửa bên hệ thống |  | Pass |
| 7 | Quản lý sản phẩm | | Thêm sản phẩm | Nhập thông tin sản phẩm | Thông báo thành công và hiển thị danh sách sản phẩm |  | Pass |
| Sửa | Nhập thông tin sản phẩm | Thông báo thành công và hiển thị danh sách sản phẩm |  | Pass |
| Xóa | Chọn sản phẩm cần xóa , hiển thị Popup click Xóa | Thông báo thành công và hiển thị danh sách sản phẩm |  | Pass |
| 8 | Quản lý danh mục | | Thêm danh mục | Nhập thông tin danh mục | Thông báo thành công và hiển thị danh sách danh mục |  | Pass |
| Sửa | Nhập thông tin danh mục | Thông báo thành công và hiển thị danh sách danh mục |  | Pass |
| Xóa | Chọn sản phẩm cần xóa , hiển thị Popup click Xóa | Thông báo thành công và hiển thị danh sách danh mục |  | Pass |
| 9 | Quản lý hãng | | Thêm hãng | Nhập thông tin hãng | Thông báo thành công và hiển thị danh sách hãng |  | Pass |
| Sửa | Nhập thông tin danh mục | Thông báo thành công và hiển thị danh sách hãng |  | Pass |
| Xóa | Chọn sản phẩm cần xóa , hiển thị Popup click Xóa | Thông báo thành công và hiển thị danh sách hãng |  | Pass |
| 10 | Quản lý người dùng | | Khóa tài khoản | Chọn người dùng cần khóa | Thông báo thành công và hiển thị danh sách người dùng |  | Pass |
|  |  | | Mở khóa tài khoản | Chọn người dùng cần mở kjoas | Thông báo thành công và hiển thị danh sách người dùng |  | Pass |

Bảng 3.2.1 Kiểm thử hệ thống

# TÀI LIỆU THAM KHẢO

[1] Giáo trình môn học Phân tích thiết kế hệ thống.

[2] Giáo trình môn học Thiết kế cơ sở dữ liệu.

[3] <https://www.postgresql.org/docs> : Tài liệu hướng dẫn PostgreSQL.

[4] <https://typeorm.io> : Tài liệu hướng dẫn thư viện TypeORM.

[5] <https://docs.nestjs.com> : Tài liệu framework NestJS.

[6] [https://react.dev/](https://react.dev/%20) : Tài liệu tham khảo ReactJS.

[7] <https://sandbox.vnpayment.vn/apis/docs/thanh-toan-pay/pay.html>: Tài liệu về tích hợp VnPay

[8] <https://shopify.dev/docs/api/admin-rest/2024-04/resources/product>: Tài liệu về api docs của Shopify

[9] [https://api-doc.shopbase.com/#tag/Product/operation/get-all-products](https://api-doc.shopbase.com/%23tag/Product/operation/get-all-products): : Tài liệu về api docs của ShopBase