**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

A blue and white logo

AI-generated content may be incorrect.

**ISO 9001:2015**

**TRẦN QUỐC NINH**

**PHÁT TRIỂN WEBSITE**

**BÁN ĐIỆN THOẠI DI ĐỘNG**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**VĨNH LONG, NĂM 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

**PHÁT TRIỂN WEBSITE**

**BÁN ĐIỆN THOẠI DI ĐỘNG**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Sinh viên thực hiện: **TRẦN QUỐC NINH**

Mã số sinh viên: **110121079**

Lớp: **DA21TTA**

GVHD: **PHẠM MINH ĐƯƠNG**

**VĨNH LONG, NĂM 2025**

LỜI CAM ĐOAN

Tôi xin cam đoan rằng toàn bộ nội dung báo cáo khóa luận tốt nghiệp với đề tài “Phát triển website bán điện thoại di động” là kết quả nghiên cứu và thực hiện của chính tôi. Các số liệu, kết quả nêu trong báo cáo là trung thực, chưa từng được sử dụng để bảo vệ một học vị nào trước đây.

Trong quá trình thực hiện, tôi đã tham khảo, trích dẫn đầy đủ các nguồn tài liệu, các ý kiến đóng góp của thầy cô, bạn bè, các tác giả, cũng như các tài liệu tham khảo khác để phục vụ cho việc hoàn thành khóa luận này. Nếu phát hiện có sự gian lận hoặc sao chép, tôi xin hoàn toàn chịu trách nhiệm trước quy định của nhà trường.

|  |  |
| --- | --- |
|  | Vĩnh Long, ngày tháng năm 2025  **Người cam đoan**  **Trần Quốc Ninh** |

LỜI CẢM ƠN

Lời đầu tiên, tôi xin bày tỏ lòng biết ơn của mình tới thầy Phạm Minh Đương. Người đã luôn giúp đỡ, hướng dẫn, động viên và đồng hành với tôi trong quá trình học tập và thời gian thực hiện đề tài.

Tôi cũng muốn gửi lời cảm ơn đến các giảng viên Trường Kỹ thuật và Công nghệ, những người đã chia sẽ kiến thức quý báo và kinh nghiệm thực tế cho tôi trong suốt thời gian học tập vừa qua. Nhờ sự chỉ dạy tận tình của quý thầy (cô) mà tôi có được một nền tảng vững chắc để sẵn sàng bước về phía trước với một chút niềm tin và sự tự tin.

Cuối cùng tôi xin bày tỏ lòng biết ơn đến gia đình, người thân, bạn bè đã luôn giúp đỡ và động viên em hoàn thành bài báo cáo này.

Do kiến thức còn nhiều hạn chế, thời gian có hạn nên đồ án không tránh khỏi những sai sót. Vì vậy, rất mong nhận được những góp ý của thầy (cô) để tôi có thêm kinh nghiệm để khắc phục những thiếu sót.

Một lần nữa tôi xin chân thành cảm ơn.

*Vĩnh Long, ngày tháng năm 2025*

**Sinh viên thực hiện**

**Trần Quốc Ninh**

MỤC LỤC

[LỜI CAM ĐOAN i](#_Toc206407141)

[LỜI CẢM ƠN ii](#_Toc206407142)

[MỤC LỤC iii](#_Toc206407143)

[DANH MỤC CÁC BẢNG x](#_Toc206407144)

[DANH MỤC HÌNH ẢNH xi](#_Toc206407145)

[DANH MỤC TỪ VIẾT TẮT xiii](#_Toc206407146)

[CHƯƠNG 1. ĐẶT VẤN ĐỀ 1](#_Toc206407147)

[1.1. Lý do chọn đề tài 1](#_Toc206407148)

[1.2. Mục tiêu 1](#_Toc206407149)

[1.3. Đối tượng nghiên cứu 2](#_Toc206407150)

[1.4. Phạm vi nghiên cứu 3](#_Toc206407151)

[1.5. Phương pháp nghiên cứu 3](#_Toc206407152)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 4](#_Toc206407153)

[2.1. Tìm hiểu về API, Restful APIs 4](#_Toc206407154)

[2.1.1. Khái niệm API 4](#_Toc206407155)

[2.1.2. Restful API 4](#_Toc206407156)

[2.2. Giới thiệu về SPA 4](#_Toc206407157)

[2.2.1. Khái niệm SPA 4](#_Toc206407158)

[2.2.2. Một số thành phần cơ bản của SPA 5](#_Toc206407159)

[2.3. Kiến trúc Clean Architecture 5](#_Toc206407160)

[2.3.1. Kiến trúc Clean Architecture 5](#_Toc206407161)

[2.3.2. Nguyên tắc cốt lỗi của Clean Architecture 5](#_Toc206407162)

[2.3.3. Kiến trúc lớp và chiều của sự phụ thuộc 6](#_Toc206407163)

[2.3.4. Lợi ích và hạn chế trong phát triển phần mềm 6](#_Toc206407164)

[2.4. Tìm hiểu ASP.NET Core 7](#_Toc206407165)

[2.4.1. Tổng quan về .NET Core 7](#_Toc206407166)

[2.4.2. Đặc điểm của .NET Core 7](#_Toc206407167)

[2.4.3. Thành phần của .NET Core 8](#_Toc206407168)

[2.4.4. ASP .NET Core 8](#_Toc206407169)

[2.5. Tìm hiểu VueJS Framwork 9](#_Toc206407170)

[2.5.1. Khái niệm 9](#_Toc206407171)

[2.5.2. Ưu điểm của VueJS so với các Framework khác 9](#_Toc206407172)

[2.5.3. Một số tính năng nổi bật của VueJS Framework 9](#_Toc206407173)

[2.5.4. Cấu trúc cơ bản của một component 10](#_Toc206407174)

[2.5.5. Một số phong cách viết trong VueJS 11](#_Toc206407175)

[2.5.6. Các thư viện hỗ trợ 12](#_Toc206407176)

[2.6. Tìm hiểu ML.NET 14](#_Toc206407177)

[2.6.1. Khái niệm 14](#_Toc206407178)

[2.6.2. Một số bài toán mà ML.NET hỗ trợ 15](#_Toc206407179)

[2.6.3. Cơ chế hoạt động 17](#_Toc206407180)

[2.7. JWT 18](#_Toc206407181)

[2.7.1. Khái niệm 18](#_Toc206407182)

[2.7.2. Cấu trúc 18](#_Toc206407183)

[2.7.3. Cách hoạt động 20](#_Toc206407184)

[2.8. Sepay 20](#_Toc206407185)

[2.8.1. Giới thiệu về Sepay 20](#_Toc206407186)

[2.8.2. Quy trình thanh toán qua Sepay 21](#_Toc206407187)

[2.8.3. Lợi ích khi tích hợp Sepay 22](#_Toc206407188)

[2.9. Các nghiệp vụ liên quan đến đề tài 22](#_Toc206407189)

[2.9.1. Khái niệm website thương mại điện tử bán hàng 22](#_Toc206407190)

[2.9.2. Một số chức năng chính của một website thương mại điện tử bán hàng 22](#_Toc206407191)

[2.10. Các công trình nghiên cứu liên quan 24](#_Toc206407192)

[CHƯƠNG 3. THỰC HIỆN HÓA NGHIÊN CỨU 26](#_Toc206407193)

[3.1. Mô tả hệ thống 26](#_Toc206407194)

[3.2. Các yêu cầu hệ thống 27](#_Toc206407195)

[3.2.1. Đối với người quản trị 27](#_Toc206407196)

[3.2.2. Đối với khách hàng 27](#_Toc206407197)

[3.2.3. Đối với khách vãng lai 27](#_Toc206407198)

[3.3. Thiết kế dữ liệu 28](#_Toc206407199)

[3.3.1. Mô hình ERD 28](#_Toc206407200)

[3.3.2. Mô tả thực thể 28](#_Toc206407201)

[3.4. Thiết kế xử lý 35](#_Toc206407202)

[3.4.1. Sơ đồ use case 35](#_Toc206407203)

[3.4.2. Mô hình hoạt động hệ thống 37](#_Toc206407204)

[CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU 38](#_Toc206407205)

[4.1. Giao diện khách hàng 38](#_Toc206407206)

[4.1.1. Giao diện trang chủ 38](#_Toc206407207)

[4.1.2. Giao diện trang sản phẩm 38](#_Toc206407208)

[4.1.3. Giao diện chi tiết sản phẩm 39](#_Toc206407209)

[4.1.4. Giao diện đăng ký, đăng nhập 41](#_Toc206407210)

[4.1.5. Giao diện giỏ hàng 43](#_Toc206407211)

[4.1.6. Giao diện thông tin người dùng 43](#_Toc206407212)

[4.1.7. Giao diện danh sách đơn hàng 44](#_Toc206407213)

[4.1.8. Giao diện khi mua sản phẩm 45](#_Toc206407214)

[4.1.9. Giao diện khi đặt hàng thành công 46](#_Toc206407215)

[4.1.10. Giao diện thanh toán online 47](#_Toc206407216)

[4.2. Giao diện quản trị 48](#_Toc206407217)

[4.2.1. Giao diện trang dashboard 48](#_Toc206407218)

[4.2.2. Giao diện trang quản lý sản phẩm 49](#_Toc206407219)

[4.2.3. Giao diện trang quản lý danh mục 49](#_Toc206407220)

[4.2.4. Giao diện trang quản lý người dùng 50](#_Toc206407221)

[4.2.5. Giao diện trang quản lý đơn hàng 50](#_Toc206407222)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 52](#_Toc206407223)

[5.1. Kết luận 52](#_Toc206407224)

[5.2. Hướng phát triển 52](#_Toc206407225)

[PHỤ LỤC 54](#_Toc206407226)

[DANH MỤC TÀI LIỆU THAM KHẢO 58](#_Toc206407227)

UBND TỈNH VĨNH LONG **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập - Tự do - Hạnh phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

*(Của giảng viên hướng dẫn)*

Họ và tên sinh viên: Trần Quốc Ninh MSSV: 110121079

Ngành: Công nghệ Thông tin Khóa: 2021

Tên đề tài: Phát triển website bán điện thoại di động.

Họ và tên Giáo viên hướng dẫn: Phạm Minh Đương

Chức danh: Giảng viên Học vị: Thạc sĩ

**NHẬN XÉT**

1. Nội dung đề tài:

1. Ưu điểm:

1. Khuyết điểm:

1. Điểm mới đề tài:

1. Giá trị thực trên đề tài:

7. Đề nghị sửa chữa bổ sung:

8. Đánh giá:

………..*, ngày tháng năm 20…*

Giảng viên hướng dẫn

*(Ký & ghi rõ họ tên)*

UBND TỈNH VĨNH LONG **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập - Tự do - Hạnh phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

*(Của cán bộ chấm đồ án, khóa luận)*

Họ và tên người nhận xét:

Chức danh: Học vị:

Chuyên ngành:

Cơ quan công tác:

Tên sinh viên:

Tên đề tài đồ án, khóa luận tốt nghiệp:

**I. Ý KIẾN NHẬN XÉT**

1. Nội dung:

2. Điểm mới các kết quả của đồ án, khóa luận:

3. Ứng dụng thực tế:

**II. CÁC VẤN ĐỀ CẦN LÀM RÕ**

(Các câu hỏi của giáo viên phản biện)

**III. KẾT LUẬN**

(Ghi rõ đồng ý hay không đồng ý cho bảo vệ đồ án khóa luận tốt nghiệp)

*……………, ngày …… tháng …… năm 20…*

Người nhận xét

*(Ký & ghi rõ họ tên)*

DANH MỤC CÁC BẢNG

[Bảng 3.1: Bảng Users 28](#_Toc206403045)

[Bảng 3.2: Bảng OTP 29](#_Toc206403046)

[Bảng 3.3: Bảng Brands 29](#_Toc206403047)

[Bảng 3.4: Bảng Phones 30](#_Toc206403048)

[Bảng 3.5: Bảng PhoneImages 30](#_Toc206403049)

[Bảng 3.6: Bảng Colors 31](#_Toc206403050)

[Bảng 3.7: Bảng Rams 31](#_Toc206403051)

[Bảng 3.8: Bảng Storages 31](#_Toc206403052)

[Bảng 3.9: Bảng PhoneVariants 31](#_Toc206403053)

[Bảng 3.10: Bảng Carts 32](#_Toc206403054)

[Bảng 3.11: Bảng PaymentMethods 32](#_Toc206403055)

[Bảng 3.12: Bảng Shippers 33](#_Toc206403056)

[Bảng 3.13: Bảng Orders 33](#_Toc206403057)

[Bảng 3.14: Bảng OrderDetails 34](#_Toc206403058)

[Bảng 3.15: Bảng Reviews 34](#_Toc206403059)

DANH MỤC HÌNH ẢNH

[Hình 2.1: Cơ chế hoạt động của ML.NET 17](#_Toc206403123)

[Hình 2.2: Cấu trúc cơ bản của JWT 18](#_Toc206403124)

[Hình 2.3: Json phần tiêu đề của JWT 18](#_Toc206403125)

[Hình 2.4: Json phần tải của JWT 19](#_Toc206403126)

[Hình 2.5: JWT hoàn chỉnh 19](#_Toc206403127)

[Hình 2.6: Truy cập API với JWT 20](#_Toc206403128)

[Hình 2.7: Nội dung phần tiêu đề 20](#_Toc206403129)

[Hình 3.1: Mô hình ERD 28](#_Toc206403130)

[Hình 3.2: Sơ đồ use case người quản trị 35](#_Toc206403131)

[Hình 3.3: Sơ đồ use case khách hàng 36](#_Toc206403132)

[Hình 3.4: Sơ đồ use case khách vãng lai 36](#_Toc206403133)

[Hình 3.5: Mô hình hoạt động hệ thống 37](#_Toc206403134)

[Hình 4.1: Giao diện trang chủ 38](#_Toc206403135)

[Hình 4.2: Giao diện trang sản phẩm 39](#_Toc206403136)

[Hình 4.3: Giao diện chi tiết sản phẩm 40](#_Toc206403137)

[Hình 4.4 Giao diện chi tiết sản phẩm – thông số 40](#_Toc206403138)

[Hình 4.5 Giao diện chi tiết sản phẩm – đánh giá 41](#_Toc206403139)

[Hình 4.6 Giao diện đăng nhập tài khoản 41](#_Toc206403140)

[Hình 4.7 Giao diện đăng ký tài khoản 42](#_Toc206403141)

[Hình 4.8 Giao diện xác thực OTP 43](#_Toc206403142)

[Hình 4.9 Giao diện giỏ hàng 43](#_Toc206403143)

[Hình 4.10: Giao diện thông tin người dùng 44](#_Toc206403144)

[Hình 4.11: Giao diện danh sách đơn hàng 44](#_Toc206403145)

[Hình 4.12: Giao diện mua sản phẩm - đặt hàng 45](#_Toc206403146)

[Hình 4.13: Giao diện mua sản phẩm - thông tin giao hàng 46](#_Toc206403147)

[Hình 4.14: Giao diện mua sản phẩm - phương thức thanh toán 46](#_Toc206403148)

[Hình 4.15: Giao diện đặt hàng thành công 47](#_Toc206403149)

[Hình 4.16: Giao diện thanh toán QR Code 48](#_Toc206403150)

[Hình 4.17: Giao diện trang dashboard 48](#_Toc206403151)

[Hình 4.18: Giao diện trang quản lý sản phẩm 49](#_Toc206403152)

[Hình 4.19: Giao diện trang quản lý danh mục 50](#_Toc206403153)

[Hình 4.20: Giao diện trang quản lý người dùng 50](#_Toc206403154)

[Hình 4.21: Giao diện trang quản lý đơn hàng 51](#_Toc206403155)

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| Từ viết tắt | Ý nghĩa |
| AJAX | Asynchronous JavaScript and XML |
| API | Application Programming Interface |
| ARM | Advanced RISC Machine |
| ASP.NET | Active Server Pages .NET |
| ATM | Automated Teller Machine |
| CLI | Command Line Interface |
| CSS | Cascading Style Sheets |
| CSV | Comma Separated Values |
| CSDL | Cơ sở dữ liệu |
| CSRF | Cross-Site Request Forgery |
| DOM | Document Object Model |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| ERD | Entity Relationship Diagram |
| FK | Foreign Key |
| HMAC | Hash-based Message Authentication Code |
| HMR | Hot Module Replacement |
| HTML | Hypertext Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IoT | Internet of Things |
| JSON | JavaScript Object Notation |
| JWT | Json Web Token |
| LightGBM | Light Gradient Boosting Machine |
| MAE | Mean Absolute Error |
| MIT | Massachusetts Institute of Technology |
| ML | Machine Learning |
| MSE | Mean Squared Error |
| MVC | Model-View-Controller |
| OTP | One Time Password |
| PCA | Principal Component Analysis |
| PK | Primary Key |
| QR | Quick Response code |
| REST | Representational State Transfer |
| RSA | Rivest-Shamir-Adleman |
| SAML | Security Assertion Markup Language |
| SDK | Software Development Kit |
| SEO | Search Engine Optimization |
| SPA | Single Page Application |
| SSA | Singular Spectrum Analysis |
| SSR | Server-Side Rendering |
| UI | User Interface |
| URL | Uniform Resource Locator |
| URI | Uniform Resource Identifier |
| VA | Virtual Account |
| VietQR | Vietnamese Quick Response |
| Web | World Wide Web |
| XML | Extensible Markup Language |

# ĐẶT VẤN ĐỀ

## Lý do chọn đề tài

Trong bối cảnh công nghệ phát triển không ngừng, các hệ thống thương mại điện tử đang trở thành một phần không thể thiếu trong đời sống hiện đại. Với sự gia tăng của các doanh nghiệp chuyển đổi số, yêu cầu về việc xây dựng những hệ thống web vừa linh hoạt, dễ mở rộng, vừa đảm bảo hiệu năng và khả năng bảo trì là rất cấp thiết. Tuy nhiên, việc thiết kế và phát triển một hệ thống đáp ứng những yêu cầu này không hề đơn giản, đòi hỏi phải lựa chọn kiến trúc và công nghệ phù hợp.

**Clean Architecture** được đánh giá là một trong những kiến trúc hiện đại và mạnh mẽ nhất để giải quyết vấn đề trên. Clean Architecture tập trung vào việc tổ chức mã nguồn theo nguyên tắc tách biệt các thành phần logic, giảm thiểu sự phụ thuộc giữa các lớp trong ứng dụng. Điều này giúp đảm bảo rằng hệ thống dễ bảo trì, dễ kiểm thử và có khả năng mở rộng linh hoạt trong tương lai.

Trong khi đó, **.NET Core** là một nền tảng phát triển backend mạnh mẽ, hỗ trợ đa nền tảng, cung cấp hiệu suất cao và các công cụ tối ưu để xây dựng các dịch vụ web hiện đại. .NET Core được phát triển bởi Microsoft, có cộng đồng người dùng rộng lớn và được tích hợp sẵn nhiều tính năng như bảo mật, quản lý tài nguyên và hỗ trợ kết nối cơ sở dữ liệu. Việc lựa chọn .NET Core làm nền tảng phát triển backend sẽ giúp khai thác tối đa những ưu điểm trên để xây dựng hệ thống hiệu quả và đáng tin cậy.

Đồng thời, **VueJS** được chọn làm framework phát triển frontend nhờ tính linh hoạt và khả năng tùy biến cao. **VueJS** nổi bật với cú pháp đơn giản, dễ học nhưng vẫn mạnh mẽ khi xây dựng các ứng dụng phức tạp. Đặc biệt, VueJS cung cấp nhiều công cụ hỗ trợ như Vue Router, Pinia và khả năng tích hợp TypeScript giúp tối ưu hóa quy trình phát triển giao diện người dùng. Với mục tiêu tạo ra một website có trải nghiệm người dùng tốt và giao diện thân thiện, VueJS là lựa chọn phù hợp để hiện thực hóa điều đó.

## Mục tiêu

Xây dựng một website bán điện thoại dựa trên kiến trúc Clean Architecture nhằm đảm bảo hệ thống dễ bảo trì, mở rộng và đáp ứng tốt yêu cầu về hiệu năng.

Ứng dụng .NET Core để phát triển backend, cung cấp dịch vụ và xử lý dữ liệu hiệu quả.

Sử dụng VueJS để phát triển frontend, tối ưu hóa giao diện người dùng và trải nghiệm người dùng.

## Đối tượng nghiên cứu

Kiến trúc Clean Architecture: Một mô hình kiến trúc phần mềm hiện đại, giúp phân tách ứng dụng thành các lớp riêng biệt như: Domain (các quy tắc nghiệp vụ cốt lõi và thực thể), Application (các trường hợp sử dụng và xử lý nghiệp vụ), Interface Adapter (chuyển đổi dữ liệu giữa các lớp bên trong và giao diện bên ngoài như giao diện người dùng hoặc API) và Frameworks and Drivers (thành phần liên quan đến cơ sở dữ liệu, thư viện và framework cụ thể). Nghiên cứu sẽ tập trung vào cách áp dụng Clean Architecture để xây dựng hệ thống phần mềm linh hoạt, dễ bảo trì và dễ kiểm thử.

Nguyên tắc thiết kế RESTful API: RESTful API là một phong cách kiến trúc cho phép các hệ thống giao tiếp với nhau thông qua giao thức HTTP. Nghiên cứu sẽ khám phá các nguyên tắc thiết kế RESTful API như sử dụng các phương thức HTTP (GET, POST, PUT, DELETE), định danh tài nguyên thông qua URI và đảm bảo tính stateless của các request.

Công nghệ .NET Core trong phát triển backend: .NET Core là một nền tảng mã nguồn mở, đa nền tảng, hiệu suất cao, được sử dụng để xây dựng các ứng dụng web hiện đại. Nghiên cứu sẽ tập trung vào cách sử dụng ASP.NET Core để xây dựng các API backend, bao gồm việc sử dụng Minimal APIs, Entity Framework Core và các công cụ hỗ trợ khác.

Framework VueJS và các công cụ liên quan: VueJS là một framework JavaScript tiến bộ, được sử dụng để xây dựng giao diện người dùng. Nghiên cứu sẽ tập trung vào việc sử dụng VueJS kết hợp với các công cụ như Pinia (quản lý trạng thái), Vue Router (định tuyến) và Axios (giao tiếp với API) để phát triển frontend hiệu quả và tương tác mượt mà với backend.

## Phạm vi nghiên cứu

**Về mặt công nghệ:** Tập trung vào việc kết hợp Clean Architecture với .NET Core API và VueJS để phát triển hệ thống. Các công nghệ khác chỉ được đề cập nếu cần thiết để hỗ trợ hoặc tích hợp với hệ thống chính.

**Về chức năng:** Hệ thống sẽ được phát triển với các chức năng cơ bản của một website bán hàng như quản lý sản phẩm, giỏ hàng, thanh toán và quản lý đơn hàng.

**Về đối tượng áp dụng:** Đề tài hướng tới các hệ thống thương mại điện tử vừa và nhỏ, phù hợp với các doanh nghiệp khởi nghiệp hoặc có quy mô hạn chế.

## Phương pháp nghiên cứu

Phương pháp nghiên cứu lý thuyết: Tìm hiểu và phân tích các nguyên lý của Clean Architecture, các nguyên tắc thiết kế RESTful API và công nghệ .NET Core thông qua sách chuyên ngành, tài liệu học thuật, và nguồn tài liệu trực tuyến đáng tin cậy như Microsoft Learn và các blog kỹ thuật uy tín. Đồng thời, nghiên cứu cách VueJS và các công cụ liên quan như Pinia, Vue Router, Axios hỗ trợ phát triển frontend và giao tiếp với API.

Phương pháp khảo sát: Tìm hiểu về nhu cầu của thị trường và người dùng liên quan đến việc mua sắm điện thoại di động trực tuyến.

Phương pháp thực nghiệm: Xây dựng một ứng dụng website bán điện thoại di động minh họa cho vấn đề nghiên cứu.

# CƠ SỞ LÝ THUYẾT

## Tìm hiểu về API, Restful APIs

### Khái niệm API

**API** 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 [18].

### Restful API

**REST** 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. 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… 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 (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 [18]**.

## Giới thiệu về SPA

### Khái niệm SPA

SPA là một loại ứng dụng web sử dụng một trang duy nhất, tức là nội dung của trang được tải một lần duy nhất trong quá trình sử dụng. Thay vì tải lại toàn bộ trang khi người dùng di chuyển qua các trang khác nhau, SPA tải các phần của trang cần thiết và thực hiện các thay đổi trạng thái mà không cần tải lại trang. Điều này tạo ra trải nghiệm người dùng mượt mà và nhanh chóng.

### Một số thành phần cơ bản của SPA

UI Component: Là thành phần giao diện hiển thị cho người dùng xem và tương tác. Nó được tạo ra để tái sử dụng, có thể chứa các logic.

Routing: Thành phần quản lý và điều hướng giữa các trang hoặc các chế độ xem khác nhau trong ứng dụng mà không cần phải tải lại trang.

State Management: Thành phần quản lý trạng thái toàn cục cuả ứng dụng, giúp theo dõi và quản lý trạng thái của các thành phần khác nhau mà không cần truyền trạng thái qua nhiều cấp độ.

AJAX và API Communication: Thành phần giúp tương tác với máy chủ để lấy và gửi dữ liệu thông qua API, thường được sử dụng để cập nhật nội dung của trang mà không cần tải lại toàn bộ trang.

Views hoặc Pages: Đây là thành phần chính của trang mà người dùng thấy và tương tác. Trong SPA, chúng có thể được cập nhật và hiển thị mà không làm tải lại toàn bộ trang.

Navigation Bar: Thành phần giúp người dùng di chuyển qua lại giữa các chế độ xem hoặc giữa các trang khác nhau trong website.

## Kiến trúc Clean Architecture

### Kiến trúc Clean Architecture

Clean Architecture là một mô hình thiết kế phần mềm được giới thiệu bởi Robert C. Martin (Uncle Bob). Mục tiêu của Clean Architecture là tạo ra các ứng dụng dễ bảo trì, mở rộng và có thể tái sử dụng cao. Kiến trúc này phân chia hệ thống thành các tầng (layers) có nhiệm vụ rõ ràng, độc lập và có thể dễ dàng thay đổi mà không ảnh hưởng đến các phần khác trong hệ thống.

### Nguyên tắc cốt lỗi của Clean Architecture

Độc lập với công nghệ: Các thành phần của hệ thống không nên phụ thuộc vào các công nghệ cụ thể. Điều này đảm bảo rằng chúng có thể được thay đổi mà không ảnh hưởng đến các thành phần khác.

Tách biệt logic kinh doanh: Logic kinh doanh nên được tách biệt và không bị gắn kết với các yếu tố kỹ thuật khác trong hệ thống.

Đảm bảo kiểm soát đầu vào và đầu ra: Các thành phần trong Clean Architecture nên có khả năng kiểm soát và xử lý dữ liệu đầu vào và đầu ra của chúng.

Đảm bảo khả năng thay đổi: Kiến trúc Clean Architecture phải cho phép các thành phần có khả năng mở rộng, thay đổi và tái sử dụng một cách dễ dàng.

### Kiến trúc lớp và chiều của sự phụ thuộc

Trong Clean Architecture, các layer (tầng) được tách biệt rõ ràng nhằm đảm bảo sự rõ ràng và độc lập giữa chúng. Các layer chính bao gồm Presentation (giao diện người dùng), Application (quy tắc kinh doanh), Domain (logic kinh doanh cốt lõi) và Infrastructure (cơ sở hạ tầng). Mỗi layer có nhiệm vụ và trách nhiệm riêng biệt, tương tác với nhau thông qua các cổng (ports) và đối tượng chuyển giao (adapters).

Trong Clean Architecture, sự phụ thuộc luôn diễn ra từ các layer cao hơn đến các layer thấp hơn. Các layer cao hơn không phụ thuộc vào các layer thấp hơn, đảm bảo tính độc lập và khả năng thay đổi. Sự phụ thuộc theo chiều này giúp đảm bảo rằng các quy tắc kinh doanh và logic cốt lõi không bị ảnh hưởng bởi các chi tiết kỹ thuật và cơ sở hạ tầng. Thay vào đó, các layer thấp hơn sẽ phụ thuộc vào các layer cao hơn để nhận các yêu cầu và truyền dữ liệu giữa các layer.

### Lợi ích và hạn chế trong phát triển phần mềm

Clean Architecture mang lại nhiều lợi ích đáng kể trong phát triển phần mềm. Việc tách biệt rõ ràng giữa các thành phần bên trong hệ thống giúp quá trình bảo trì và mở rộng trở nên dễ dàng hơn. Sự giảm thiểu phụ thuộc lẫn nhau giữa các lớp cho phép kiểm soát hệ thống một cách hiệu quả, đồng thời tạo điều kiện tự do thay đổi công nghệ mà không ảnh hưởng đến logic nghiệp vụ. Mô hình này cũng rất phù hợp với kiến trúc microservices nhờ khả năng tổ chức và phân chia rõ ràng. Tuy nhiên, Clean Architecture cũng tồn tại một số hạn chế. Việc triển khai ban đầu đòi hỏi nhiều công sức do cần thiết kế các lớp và cấu trúc hệ thống phức tạp, điều này có thể không phù hợp với các dự án quy mô vừa và nhỏ. Bên cạnh đó, để áp dụng hiệu quả mô hình này, lập trình viên cần có nền tảng vững chắc về các nguyên lý thiết kế phần mềm như SOLID cũng như nắm vững nhiều mẫu thiết kế (design pattern). Mã nguồn có thể trở nên khó hiểu do số lượng lớn các lớp và interface, đồng thời việc giao tiếp giữa các tầng trong hệ thống đôi khi cũng có thể gặp khó khăn.

## Tìm hiểu ASP.NET Core

### Tổng quan về .NET Core

Phiên bản đầu tiên là .NET Core 1.0 phát hành vào ngày 27 tháng 6 năm 2016. Sau phiên bản .NET Core 3.1, ngày 10 tháng 11 năm 2020 Microsoft phát hành phiên bản mới có tên .NET 5, là phiên bản hợp nhất các nền tảng .NET Core, .NET Framework và Xamarin riêng biệt trước đây giúp các nhà phát triển dễ dàng sử dụng một nền tảng duy nhất cho nhiều loại ứng dụng khác nhau gọi chung là .NET.

**.NET Core là phiên bản cải tiến của .NET Framework**, là nền tảng phát triển đa năng, mã nguồn mở, miễn phí được duy trì bởi Microsoft, một framework đa nền tảng chạy trên các hệ điều hành Windows, macOS và Linux.

.NET Core có thể được sử dụng để xây dựng các loại ứng dụng khác nhau như thiết bị di động, máy tính để bàn, web, đám mây, IoT, máy học, microservice, trò chơi…[15].

### Đặc điểm của .NET Core

Đa nền tảng: Chạy trên các hệ điều hành Windows, macOS và Linux.

Nhất quán trên các kiến ​​trúc: Có thể chạy mã nguồn của bạn với cùng một hành vi trên nhiều kiến ​​trúc hệ thống bao gồm x64, x86 và ARM.

Các công cụ dòng lệnh: Bao gồm các công cụ dòng lệnh dễ sử dụng, có thể được sử dụng để phát triển cục bộ và trong các tình huống tích hợp liên tục.

Triển khai linh hoạt: Có thể cài đặt song song (cài đặt toàn người dùng hoặc toàn hệ thống). Có thể được sử dụng với các container Docker.

Tương thích: .NET Core tương thích với .NET Framework, Xamarin và Mono, thông qua .NET Standard.

Nguồn mở: Nền tảng .NET Core là nguồn mở, sử dụng giấy phép MIT và Apache 2. .NET Core là một dự án .NET Foundation.

Được hỗ trợ bởi Microsoft: .NET Core được Microsoft hỗ trợ theo Hỗ trợ .NET Core [15].

### Thành phần của .NET Core

.NET Core runtime: Cung cấp một hệ thống kiểu, tải lắp ráp, trình thu gom rác, interop gốc và các dịch vụ cơ bản khác. Các thư viện khung .NET Core cung cấp các kiểu dữ liệu nguyên thủy, các kiểu thành phần ứng dụng và các tiện ích cơ bản.

ASP.NET Core runtime: Cung cấp khung để xây dựng các ứng dụng kết nối internet , điện toán đám mây hiện đại, chẳng hạn như ứng dụng web, ứng dụng IoT và phụ trợ di động.

.NET Core SDK và trình biên dịch ngôn ngữ cho phép trải nghiệm nhà phát triển .NET Core.

Dotnet command, được sử dụng để khởi chạy các ứng dụng .NET Core và các lệnh CLI. Nó chọn thời gian chạy và lưu trữ thời gian chạy, cung cấp chính sách tải lắp ráp và khởi chạy các ứng dụng và công cụ.

### ASP .NET Core

Năm 2002, Microsoft giới thiệu ASP.NET, chạy trên nền tảng .NET Framework, chủ yếu hoạt động trên Windows. ASP.NET được sử dụng rộng rãi để xây dựng các ứng dụng web và dịch vụ.

Đến năm 2016, ASP.NET Core ra đời như một nền tảng mới, độc lập với .NET Framework. Đây là bước chuyển lớn của Microsoft để hỗ trợ phát triển ứng dụng đa nền tảng và mã nguồn mở. ASP.NET Core 1.0 là phiên bản đầu tiên.

ASP.NET Core 2.1 được giới thiệu năm 2018, cùng với các cải tiến như Razor Pages và SignalR. Vào 2 năm sau đó, với sự ra mắt của .NET 5, ASP.NET Core được hợp nhất vào nền tảng .NET thống nhất.

ASP.NET Core bao gồm các thành phần theo hướng module, giúp tối ưu hóa tài nguyên và giảm chi phí phát triển, đồng thời giữ được tính linh hoạt trong quá trình xây dựng giải pháp. Các ứng dụng ASP.NET Core có thể được phát triển và chạy trên nhiều nền tảng khác nhau như Windows, macOS và Linux.

## Tìm hiểu VueJS Framwork

### Khái niệm

Vuejs Framework là một framework Javascript chuyên về thiết kế giao diện người dùng (UI) tương tác trên các website và ứng dụng đơn trang (SPA).

### Ưu điểm của VueJS so với các Framework khác

Kích thước nhỏ gọn: VueJS có kích thước nhỏ so với một số framework khác như Angular, React. Người dùng sẽ không mất nhiều thời gian để tải xuống và sử dụng.

Dễ tìm hiểu và sử dụng: Được khyến khích sử dụng cho người mới bắt đầu bởi cú pháp khá đơn giản, dễ học cùng với cộng đồng đông đảo trên github khiến cho việc tìm giải pháp giải quyết các vấn đề trở nên dễ dàng.

Tốc độ nhanh: Ngoài việc có kích thước nhỏ gọn VueJS còn được thiết kế với nhiều tính năng nhằm tối ưu hóa hiệu suất giúp giảm thời gian tải trang web và cải thiện trải nghiệm người dùng.

### Một số tính năng nổi bật của VueJS Framework

VueJS sử dụng hệ thống ràng buộc dữ liệu hai chiều (two-way data binding) để tự động đồng bộ hóa dữ liệu giữa DOM và dữ liệu ứng dụng hay nói cách khác là giữa giao diện người dùng (UI layer) và lớp dữ liệu (data layer). Khi có sự thay đổi ở một trong hai, sự thay đổi đó sẽ tự động ảnh hưởng đến bên còn lại giúp cho dữ liệu bên trong và giao diện người dùng luôn đồng bộ.

Sử dụng Vue Directives. Vue Directives là một thuộc tính riêng do VueJS cung cấp được bắt đầu với “v-“, thuộc tính này sẽ áp dụng một hành vi (behavior) đặc biệt lên kết quả DOM được render ra. Ngoài ra VueJS còn cung cấp tính năng Custom Directives cho phép mở rộng cú pháp HTML của VueJS bằng cách tự định nghĩa các directives của riêng bạn, có thể thực hiện một số hành động khác nhau trong quá trình render hoặc tương tác với các phần tử của DOM.

VueJS cung cấp tính năngMixins cho phép tái sử dụng logic giữa các component, giúp truy cập dữ liệu và phương thức chung mà không cần sử dụng kế thừa, đồng thời có thể chia sẻ với component mong muốn.

VueJS sử dụng một cơ chế DOM ảo (Virtual DOM) để tối ưu hóa hiệu suất mỗi khi cập nhật DOM. Thay vì phải cập nhật toàn bộ DOM, VueJS chỉ cập nhật những phần cần thiết.

VueJS cung cấp một số Lifecycle hooks cho phép thực hiện các hoạt động tại một giai đoạn nào đó trong vòng đời của một component. Ví dụ: *create()* được gọi sau khi một Vue instance của component được tạo ra, *beforeMount()* được gọi trước khi một Vue instance được liên kết (mount) với một phần tử trong DOM…

Cho phép thêm các plugin cần thiết vào dự án một cách đơn giản và dễ dàng.

### Cấu trúc cơ bản của một component

VueJS sử dụng mô hình kiến trúc dựa trên thành phần hay nói cách khác là mô hình kiến trúc được xây dựng dựa trên nhiều components. Component là đơn vị quan trọng nhất để xây dựng giao diện và tái sử dụng mã nguồn. Mỗi component bao gồm ba thành phần chính là Template, Script và Style.

Đầu tiên là phần Template được định nghĩa bên trong cặp thẻ “<template>…</template>”, giống như một file **\*.**html thu nhỏ – đây là nơi xác định cấu trúc HTML của component. Ngôn ngữ bên trong Template tương tự như HTML nhưng có thể sử dụng một số cú pháp đặc biệt do Vue cung cấp để liên kết dữ liệu và thực hiện logic. Ngoài ra, cũng có thể viết CSS trực tiếp trong các thuộc tính của thẻ HTML tại đây.

Phần Script được định nghĩa bên trong cặp thẻ “<script>…</script>”, bao gồm toàn bộ logic của component. Trước tiên cần định nghĩa giá trị mặc định cho component này thông qua “export default{…}” và đặt tên cho component với tùy chọn “name: ‘tên\_component’”, sử dụng tùy chọn “components:{tên\_component1, tên\_component2 …}” để sử dụng một số component bên trong component hiện tại đồng thời cần import component muốn sử dụng theo cú pháp “import tên\_componennt from ‘địa\_chỉ\_component trong project’;” (lưu ý câu lệnh import phải được đặt bên ngoài export default).

Phần Style được định nghĩa bên trong cặp thẻ “<style>…</style>”, là nơi đặt các quy tắc CSS để định dạng và style cho phần giao diện bên trong temlpate của component.

### Một số phong cách viết trong VueJS

#### Options API

Options API là cách viết truyền thống và quen thuộc nhất trong VueJS. Phong cách này tổ chức code bằng cách khai báo các thuộc tính như data, method, computed và watch trong một đối tượng. Mỗi thuộc tính phục vụ một mục đích cụ thể, giúp lập trình viên dễ dàng hiểu cấu trúc của component.

Ưu điểm của Options API là tính dễ tiếp cận phù hợp với người mới học. Ngược lại, khi áp dụng Options API cho hệ thống lớn với nhiều logic phức tạp thì việc quản lý các thành phần liên quan đến một tính năng trở nên khó khăn vì chúng bị tách rời nằm trong nhiều thuộc tính khác nhau.

#### Composition API

Composition API được giới thiệu trong Vue 3, linh hoạt hơn trong việc tổ chức logic. Thay vì tổ chức logic theo các thuộc tính cố định, Composition API cho phép tổ chức các thành phần liên quan thành từng khối chức năng sử dụng bên trong hàm setup.

Composition API không phù hợp với người mới bắt đầu do cú pháp và khái niệm phức tạp hơn Options API.

#### Script Setup

Script Setup là cú pháp rút gọn được giới thiệu trong Vue 3.2, giúp việc sử dụng Composition API trở nên đơn giản hơn.

Đối với Script Setup, không cần khai báo hàm setup, các biến và hàm được định nghĩa trực tiếp trong thẻ <script setup>. Giúp lập trình viên viết code nhanh và dễ dàng đọc code hơn.

Script Setup là phong cách viết hiện đại và được sử dụng nhiều trong các dự án lớn.

#### Multiple Script

Multiple Script là việc kết hợp sử dụng nhiều thẻ <script> trong cùng một component, cho phép tách biệt các thành phần logic khác nhau trong ứng dụng. Giúp tăng khả năng tổ chức mã nguồn trong các dự án lớn.

Không giống như Script Setup, Multiple Script ít phổ biến hơn, chỉ phù hợp trong một số trường hợp cụ thể khi cần phân tách rõ ràng logic.

#### TypeScript

VueJS tích hợp tốt với TypeScript, chỉ cần thêm thuộc tính lang=“ts” vào thẻ <script>. Có thể sử dụng TypeScript cho Options API, Composition API và Script Setup.

TypeScript phù hợp với các dự án lớn, các dự án ưu tiên tính nhất quán và an toàn về kiểu dữ liệu. Đồng thời, việc sử dụng TypeScript đòi hỏi kiến thức cơ bản về kiểu dữ liệu và có thể làm tăng độ phức tạp cho dự án.

### Các thư viện hỗ trợ

#### Vue Route

Vue Router là router chính thức của VueJS. Nó tích hợp sâu với lõi VueJS để giúp việc xây dựng SPA trở nên dễ dàng. Một số tính năng quan trọng:

Chức năng định hướng (Routing): Vue Router cho phép di chuyển qua lại giữa các trang trong website mà không cần phải tải lại toàn bộ trang.

Khả năng thiết lập layouts: cho phép định nghĩa các layout sử dụng chung cho từng nhóm các trang có liên quan, Vue Router cũng cho phép xác định Component hiển thị cho từng route.

Cho phép định nghĩa các route cho website, các route lồng nhau, các route động thông qua tham số.

Hỗ trợ sử dụng các hiệu ứng chuyển đổi (transition) khi di chuyển đến hoặc rời khỏi một route, giúp tối ưu hóa trãi nghiệm người dùng.

Chỉ tải các thành phần cần được hiển thị, giúp giảm thời gian tải trang.

#### Vuetify

Vuetify là một thư viện giao diện người dùng (UI) dựa trên Vue.js, hỗ trợ xây dựng các giao diện đẹp mắt và linh hoạt. Được phát triển theo tiêu chuẩn Material Design, Vuetify cung cấp tập hợp các thành phần (components) và chỉ thị (directives) được thiết kế để tương tác và hiển thị theo phong cách của Material Design.

Vuetify là một bộ sưu tập các component được tạo sẵn, kết hợp với nhiều tính năng mạnh mẽ như chủ đề động, giá trị mặc định toàn cục, bố cục ứng dụng và nhiều tiện ích khác. Khi sử dụng thành thạo Vuetify, có thể cấu hình lại toàn bộ các component dựng sẵn để đáp ứng các yêu cầu cụ thể. Ngoài ra, cũng có thể tự xây dựng các component tùy chỉnh dựa trên các thành phần đã được định nghĩa sẵn từ API của Vuetify.

#### Axios

Axios là một thư viện HTTP Client dựa trên Promise dành cho node.js và trình duyệt. Nó có tính isomorphic (tức là cùng codebase có thể chạy trong cả trình duyệt và node.js). Ở phía server thì nó sử dụng native module http trong node.js, còn ở phía client (trình duyệt) thì nó sử dụng XMLHttpRequest.

Các tính năng của Axios:

Tạo request từ trình duyệt bằng XMLHttpRequest.

Tạo request từ node.js bằng http.

Hỗ trợ Promise API.

Đón chặn request và response.

Biến đổi dữ liệu request và response.

Bãi bỏ request.

Tự động chuyển đổi cho dữ liệu JSON.

Hỗ trợ phía client bảo vệ chống lại CSRF.

#### Pinia

Pinia được bắt đầu như một thử nghiệm để thiết kế lại cách một Store (kho lưu trữ trạng thái) hoạt động cho Vue, sử dụng **Composition API** vào khoảng tháng 11 năm 2019. Từ đó đến nay, các nguyên tắc ban đầu vẫn được giữ nguyên, nhưng Pinia hiện hỗ trợ cả Vue 2 và Vue 3 mà không yêu cầu phải sử dụng Composition API. API của Pinia là tương tự cho cả hai phiên bản, ngoại trừ cách cài đặt và hỗ trợ SSR. Tài liệu chính thức của Pinia được viết cho Vue 3, với các ghi chú về Vue 2 khi cần thiết, để có thể sử dụng cho cả người dùng Vue 2 và Vue 3.

**Pinia** là một thư viện quản lý trạng thái chính thức cho Vue, cho phép chia sẻ trạng thái giữa các thành phần (components) và trang (pages). Nó được thiết kế để đơn giản hóa việc quản lý trạng thái trong ứng dụng Vue và thay thế Vuex trở thành giải pháp được khuyến nghị.

Một số tính năng nổi bật của Pinia:

**Hỗ trợ TypeScript:** Pinia cung cấp khả năng hỗ trợ TypeScript tốt, giúp tăng độ an toàn và tính chính xác trong mã nguồn.

**Tích hợp DevTools:** Pinia tích hợp hoàn hảo với Vue DevTools, cho phép theo dõi và debug trạng thái dễ dàng.

**Hỗ trợ SSR:** Pinia tương thích tốt với các ứng dụng cần SSR, đảm bảo tính bảo mật và hiệu suất.

**Timeline theo dõi actions và mutations:** Dễ dàng kiểm tra các thao tác đã diễn ra trong ứng dụng.

**Thay đổi trạng thái mà không cần reload:** Pinia hỗ trợ tính năng HMR, giúp phát triển ứng dụng nhanh hơn mà không làm mất trạng thái hiện tại.

**Quản lý trạng thái động:** Các stores trong Pinia được tạo động theo mặc định, không yêu cầu thêm cấu hình phức tạp.

**API đơn giản và thân thiện:** Cú pháp ngắn gọn, ít rườm rà hơn so với Vuex.

**Tích hợp các công cụ kiểm thử:** Hỗ trợ tốt hơn khi viết test cho các ứng dụng Vue.

**Plugin:** Dễ dàng mở rộng tính năng của Pinia thông qua các plugin.

## Tìm hiểu ML.NET

### Khái niệm

ML.NET là thư viện máy học mã nguồn mở và chạy đa nền tảng của Microsoft. Được đánh giá là một thư viện mạnh mẽ không thua kém các thư viện viết bằng các ngôn ngữ khác.

Thành phần cốt lõi của ML.NET là một mô hình học máy, định nghĩa quy trình chuyển đổi dữ liệu đầu vào thành kết quả dự đoán dựa trên thuật toán được lựa chọn. Với ML.NET, người dùng có thể tùy chỉnh mô hình bằng cách lựa chọn các thuật toán học máy tích hợp sẵn tương ứng với yêu cầu của bài toán hoặc tích hợp các mô hình đã được huấn luyện trước từ các thư viện học máy khác để sử dụng. Ngoài ra, ML.NET còn cung cấp các API cho phép lưu trữ và nạp lại mô hình, giúp tái sử dụng và chia sẻ kết quả nghiên cứu một cách thuận tiện [16].

### Một số bài toán mà ML.NET hỗ trợ

ML.NET được thiết kế như một danh mục các bài toán học máy, cho phép xây dựng và huấn luyện mô hình dựa trên từng loại bài toán cụ thể. Một số bài toán mà ML.NET hỗ trợ:

**Phân loại (Classification)**: ML.NET cung cấp các thư viện để xử lý phân loại nhị phân (binary classification), phân loại đa lớp (multiclass classification) và phân loại ảnh (image classification). Trong phân loại nhị phân, mô hình học máy được huấn luyện để phân tách dữ liệu vào hai nhóm, ví dụ: phân tích cảm xúc tweet thành tích cực/tiêu cực hoặc nhận diện email là spam hoặc không spam. Phân loại đa lớp mở rộng phân loại nhị phân sang nhiều nhãn hơn, chẳng hạn phân loại trạng thái chuyến bay thành “đúng giờ”, “trễ” hoặc “hủy”. Đối với phân loại ảnh, mô hình không chỉ phân loại hình ảnh mà còn có thể tinh chỉnh với các kiến trúc mạng sâu để nhận diện giống chó, phát hiện sản phẩm lỗi…

**Hồi quy (Regression)**: Trong bài toán hồi quy, ML.NET hỗ trợ dự đoán các giá trị liên tục. Mô hình hồi quy được sử dụng để dự báo giá nhà, ước tính doanh số bán hàng hay thậm chí dự đoán giá cổ phiếu dựa trên các đặc trưng đầu vào như diện tích, vị trí hoặc dữ liệu lịch sử thị trường. Sau khi huấn luyện, mô hình hồi quy trả về một hàm số để dự đoán giá trị nhãn cho các bộ đặc trưng mới.

**Phân cụm (Clustering)**: Phân cụm là bài toán học không giám sát, dùng để nhóm các bản ghi có đặc tính tương đồng. ML.NET hỗ trợ thuật toán K-Means để phân đoạn khách hàng theo hành vi hoặc nhóm sản phẩm theo chỉ số chất lượng. Phân cụm giúp khám phá các mối quan hệ tiềm ẩn và phân khúc dữ liệu mà mắt thường khó nhận biết.

**Phát hiện bất thường (Anomaly Detection)**: Đối với dữ liệu có tính mất cân bằng (giao dịch hợp lệ nhiều, giao dịch gian lận ít), ML.NET cung cấp thuật toán phát hiện bất thường dựa trên phân tích thành phần chính (PCA) để nhận diện các điểm dữ liệu có hành vi khác biệt. Bài toán này áp dụng trong phát hiện gian lận tài chính, giám sát an ninh mạng hoặc kiểm tra lỗi trên dây chuyền sản xuất.

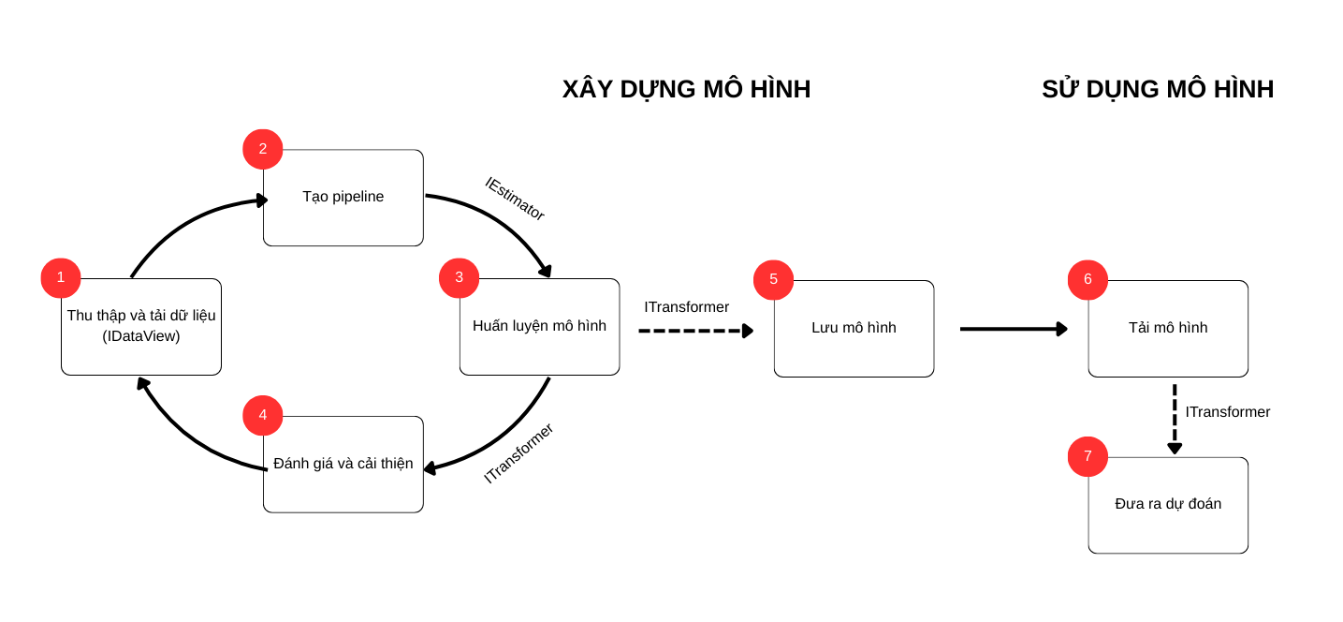
Xếp hạng (Ranking)**:** Bài toán xây dựng mô hình để sắp xếp thứ tự các đối tượng theo tiêu chí cho trước. ML.NET hỗ trợ các thuật toán như LightGBM và FastTree cho nhiệm vụ này, thường áp dụng trong hệ thống tìm kiếm, gợi ý nội dung hoặc tối ưu thứ tự kết quả hiển thị dựa trên mức độ liên quan.

Hệ gợi ý (Recommendation): ML.NET sử dụng thuật toán phân rã ma trận (Matrix Factorization) để xây dựng hệ gợi ý. Dựa trên dữ liệu đánh giá và hành vi chung của người dùng, mô hình có thể đề xuất phim, sản phẩm hoặc dịch vụ phù hợp thông qua phương pháp lọc cộng tác, tức gợi ý dựa trên sở thích tương đồng trong cộng đồng người dùng.

**Dự báo (Forecasting)**: Với dữ liệu chuỗi thời gian, ML.NET hỗ trợ bài toán dự báo (forecasting) bằng các thuật toán như SSA, giúp dự đoán doanh số theo mùa, nhu cầu tiêu thụ hay điều kiện thời tiết trong tương lai.

**Phát hiện đối tượng (Object Detection)**: ML.NET tích hợp khả năng phát hiện đối tượng trong ảnh - không chỉ phân loại ảnh mà còn xác định vùng chứa (bounding box) của nhiều đối tượng khác nhau. Thực tế, tính năng này hiện được triển khai qua Model Builder và Azure ML, áp dụng cho phát hiện xe cộ, biển báo giao thông hoặc bất thường trong ảnh X-ray [16].

### Cơ chế hoạt động



Hình 2.1: Cơ chế hoạt động của ML.NET

Trong ML.NET, quá trình xây dựng và ứng dụng mô hình học máy được tổ chức thành hai giai đoạn lớn: **Xây dựng mô hình** và **sử dụng mô hình**. Quy trình này giúp tách biệt rõ ràng các bước, từ việc chuẩn bị dữ liệu, huấn luyện cho đến triển khai dự đoán thực tế.

**Ở giai đoạn xây dựng mô hình**, đầu tiên là bước **thu thập và tải dữ liệu** (1). Dữ liệu từ các nguồn như file, cơ sở dữ liệu sẽ được làm sạch và chuẩn hóa, sau đó chuyển thành đối tượng IDataView. Tiếp theo, nhà phát triển sẽ **tạo pipeline** (2) bằng cách sử dụng IEstimator, định nghĩa các bước tiền xử lý dữ liệu và lựa chọn thuật toán học máy phù hợp. Pipeline này đóng vai trò xác định cách dữ liệu được biến đổi và huấn luyện.

Sau khi pipeline hoàn tất, mô hình sẽ được **huấn luyện** (3) bằng phương thức Fit(), tạo ra một đối tượng ITransformer. Mô hình sau huấn luyện được **đánh giá và cải thiện** (4) thông qua hàm Evaluate(), sử dụng tập kiểm thử để đo lường chất lượng dự đoán. Nếu kết quả chưa đạt yêu cầu, nhà phát triển có thể lặp lại quy trình để tối ưu hóa mô hình. Khi đã đạt chất lượng mong muốn, mô hình được **lưu lại** (5) thành file để chuẩn bị cho giai đoạn triển khai.

**Giai đoạn sử dụng mô hình** bắt đầu khi mô hình đã huấn luyện được **tải lên** (6) từ file lưu trữ. Cuối cùng, nhà phát triển chỉ cần sử dụng mô hình này để **đưa ra dự đoán** (7) cho dữ liệu mới bằng cách tạo prediction engine và gọi hàm Predict(). Quy trình này giúp dễ dàng tích hợp mô hình vào các ứng dụng thực tế, đảm bảo tính linh hoạt và tái sử dụng cao [16].

## JWT

### Khái niệm

JWT là một tiêu chuẩn mở định nghĩa một phương thức gọn nhẹ và khép kín để truyền tải thông tin an toàn giữa các bên dưới dạng một đối tượng JSON. Thông tin này có thể được xác minh và tin cậy vì nó được ký số. JWT có thể được ký bằng một khóa bí mật (với thuật toán HMAC ) hoặc một cặp khóa công khai/riêng tư bằng RSA hoặc ECDSA .

### Cấu trúc

JWT ở dạng gọn gồm ba thành phần chính, phân tách bởi dấu chấm (.): Phần header (đầu tin), phần payload (phần tải) và phần signature (chữ ký).



Hình 2.2: Cấu trúc cơ bản của JWT

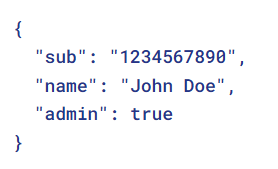
Phần đầu tiên là phần tiêu đề thường bao gồm hai phần: Loại mã thông báo và thuật toán chữ ký được sử dụng.



Hình 2.3: Json phần tiêu đề của JWT

JSON này được mã hóa theo Base64Url để tạo thành phần đầu tiên của JWT.

Phần thứ hai là phần tải chứa các “claims” - tuyên bố về một thực thể (thường là người dùng) và dữ liệu liên quan. Có ba loại claim: Đã đăng ký (registered*)*, công khai (public) và riêng tư (private).

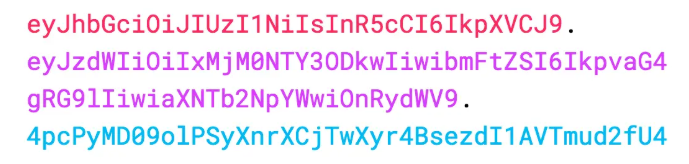


Hình 2.4: Json phần tải của JWT

JSON chứa dữ liệu được mã hóa theo Base64Url để tạo thành phần thứ hai của JWT.

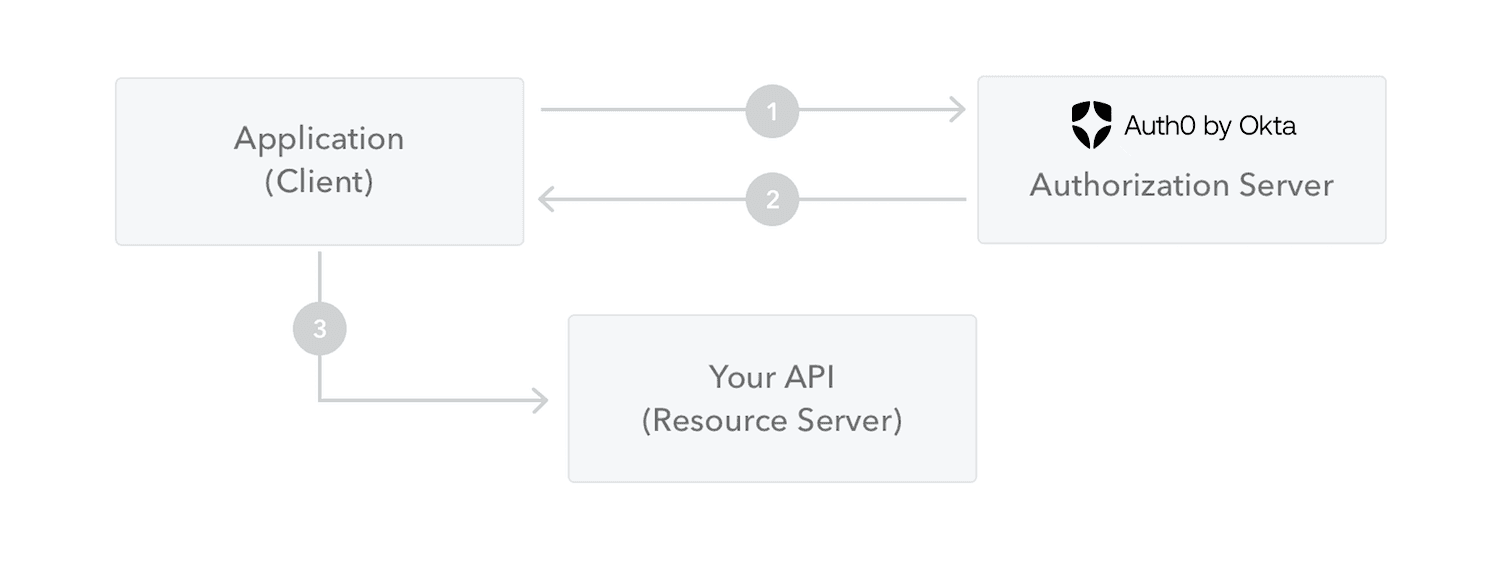
Phần thứ ba là phần chữ ký, để tạo chữ ký, hệ thống kết hợp phần tiêu đề đã mã hóa và phần tải đã mã hóa thành một chuỗi. Sau đó áp dụng thuật toán được chỉ định trong tiêu đề với một khóa bí mật hoặc cặp khóa công khai - riêng tư để sinh ra chữ ký.

Khi ghép ba thành phần lại với nhau, ta có một JWT hoàn chỉnh, dễ dàng truyền qua URL, HTTP header hoặc HTML form và vẫn đảm bảo tính bảo mật cũng như độ gọn nhẹ so với các chuẩn XML khác như SAML.



Hình 2.5: JWT hoàn chỉnh

### Cách hoạt động



Hình 2.6: Truy cập API với JWT

Trong xác thực (Authentication), khi người dùng đăng nhập thành công bằng thông tin đăng nhập của họ, một mã JWT sẽ được trả về. Vì Token chứa thông tin đăng nhập, cần hết sức cẩn thận để ngăn ngừa các vấn đề bảo mật. Nhìn chung, bạn không nên giữ Token lâu hơn mức cần thiết.

Bất cứ khi nào người dùng muốn truy cập một tài nguyên được bảo vệ, người dùng phải gửi JWT, thường là trong tiêu đề để phân quyền (Authorization) bằng cách sử dụng Bearer schema.



Hình 2.7: Nội dung phần tiêu đề

## Sepay

### Giới thiệu về Sepay

SePay là một nền tảng trung gian thanh toán tại Việt Nam, cung cấp giải pháp thanh toán điện tử nhanh chóng và an toàn giữa người mua và người bán. Với khả năng tích hợp đa dạng cổng thanh toán ngân hàng, SePay giúp các website thương mại điện tử dễ dàng tiếp cận người dùng có tài khoản tại nhiều ngân hàng khác nhau mà không cần xây dựng từng hệ thống thanh toán riêng lẻ.

SePay cung cấp giao diện API đơn giản, hỗ trợ cả phương thức thanh toán thủ công (chuyển khoản) và tự động (QR code, liên kết ngân hàng). Việc tích hợp SePay giúp doanh nghiệp tối ưu quy trình thanh toán, tăng tỷ lệ chuyển đổi đơn hàng và cải thiện trải nghiệm người dùng.

### Quy trình thanh toán qua Sepay

Thanh toán qua Sepay sẽ được thực hiện theo các bước sau:

**Bước 1. Giao diện thanh toán:** Sau khi khách hàng hoàn tất việc lựa chọn sản phẩm và chuyển sang trang thanh toán (Checkout), hệ thống sẽ tạo một mã QR chuyển khoản theo chuẩn VietQR cho từng đơn hàng. Mã QR này chứa thông tin tài khoản ngân hàng nhận tiền và nội dung chuyển khoản định danh, trong đó **bắt buộc phải có mã đơn hàng** (ví dụ: “DH182901”). Việc sử dụng VietQR giúp khách hàng thao tác nhanh chóng và tránh nhập sai thông tin.

**Bước 2. Khách hàng quét QR để chuyển khoản:** Khách hàng sử dụng ứng dụng ngân hàng trên điện thoại (Mobile Banking) để quét mã QR vừa hiển thị và tiến hành chuyển khoản đúng theo thông tin đã được hệ thống mã hóa. Nội dung chuyển khoản phải **giữ nguyên đúng mã đơn hàng** được tạo sẵn. Việc chuyển tiền hoàn toàn diễn ra bên ngoài hệ thống website, thông qua kênh ngân hàng của khách hàng.

**Bước 3. Ngân hàng xử lý giao dịch:** Sau khi khách hàng chuyển tiền thành công, ngân hàng của người gửi xử lý giao dịch và chuyển khoản đến tài khoản nhận. Khi giao dịch hoàn tất, thông tin chuyển tiền sẽ được ghi nhận và đồng bộ đến hệ thống của SePay - đơn vị trung gian theo dõi luồng giao dịch.

**Bước 4. SePay nhận thông tin giao dịch:** Ngay sau khi ngân hàng xử lý xong giao dịch, SePay sẽ nhận được thông báo giao dịch từ phía ngân hàng. Các thông tin này bao gồm thời gian giao dịch, số tiền, số tài khoản gửi, nội dung chuyển khoản, mã tham chiếu giao dịch… Đây là cơ sở để SePay xác thực và gửi thông báo đến hệ thống của bên bán.

**Bước 5. SePay gửi Webhook về hệ thống website:** SePay sẽ gửi một **yêu cầu HTTP POST (Webhook)** đến một đường dẫn đã được cấu hình trước từ phía hệ thống website. Nội dung của webhook bao gồm đầy đủ dữ liệu giao dịch đã nêu ở bước 4. Website cần có cơ chế tiếp nhận và xử lý dữ liệu này một cách bảo mật và chính xác.

**Bước 6. Website xử lý và cập nhật đơn hàng:** Sau khi nhận được Webhook từ SePay, hệ thống website sẽ tiến hành đối chiếu nội dung chuyển khoản để xác định mã đơn hàng, kiểm tra số tiền và trạng thái thanh toán. Nếu mọi thông tin hợp lệ, hệ thống sẽ cập nhật trạng thái đơn hàng và lưu giao dịch vào cơ sở dữ liệu. Khách hàng có thể kiểm tra để lấy kết quả thanh toán thành công.

### Lợi ích khi tích hợp Sepay

**Tiện lợi cho người dùng:** Thanh toán nhanh chóng mà không cần tài khoản thanh toán quốc tế.

**Giảm chi phí vận hành:** Không cần xây dựng hệ thống thanh toán riêng cho từng ngân hàng.

**Tự động hóa quy trình xác nhận đơn hàng:** Cập nhật trạng thái đơn hàng tức thời sau khi nhận thanh toán.

**Bảo mật cao:** Cơ chế mã hóa, xác thực và xác minh giúp ngăn chặn các hành vi gian lận.

## Các nghiệp vụ liên quan đến đề tài

### Khái niệm website thương mại điện tử bán hàng

Bán hàng online (bán hàng trực tuyến) là hình thức kinh doanh các sản phẩm, dịch vụ trên môi trường internet. Người bán và người mua không nhất thiết phải gặp mặt nhau trực tiếp mà có thể thông qua các nền tảng công nghệ số như [website](https://ghn.vn/blogs/tip-ban-hang/cac-trang-web-ban-hang-online-hieu-qua-uy-tin-nhat-shop-nen-biet), [mạng xã hội](https://ghn.vn/blogs/tip-ban-hang/cac-mang-xa-hoi-pho-bien-hien-nay) (Facebook, Tiktok, Instagram, Zalo, Youtube, blog, forum...) hoặc các [sàn thương mại điện tử](https://ghn.vn/blogs/tip-ban-hang/san-thuong-mai-dien-tu-co-luot-truy-cap-khung-ban-hang-hieu-qua) ([Shopee](https://ghn.vn/blogs/tip-ban-hang/shopee-la-gi-dieu-nen-biet-ve-mo-hinh-kinh-doanh-shopee), Lazada, Tiki...). Chỉ với vài thao tác đơn giản trên thiết bị điện thoại thông minh hay máy tính, hàng hóa sẽ được vận chuyển từ người bán đến tận nơi người nhận mà không cần mất thời gian, công sức di chuyển [17].

Website thương mại điện tử bán hàng là website thương mại điện tử do các thương nhân, tổ chức, cá nhân thiết lập để phục vụ hoạt động xúc tiến thương mại, bán hàng hóa hoặc cung ứng dịch vụ của mình [16].

### Một số chức năng chính của một website thương mại điện tử bán hàng

Cung cấp thông tin: Hầu hết khách hàng hiện nay đều muốn tham khảo trước các thông tin liên quan đến sản phẩm, dịch vụ như: Nguồn gốc, mẫu mã, giá bán, chế độ bảo hành, chính sách giao hàng, địa chỉ nhà cung cấp… rồi mới liên hệ đặt hàng. Chính vì vậy, một trang web bán hàng chuyên nghiệp thì chức năng đầu tiên phải có đó là đăng tải và cung cấp đầy đủ, rõ ràng các thông tin về sản phẩm, dịch vụ cần bán.

Chức năng tìm kiếm: Đến với website là đến với một cửa hàng ảo, nơi công ty giới thiệu dịch vụ sản phẩm tới khách hàng. Tuy nhiên tại một số công ty lớn hay các công ty thời trang có quá nhiều sản phẩm dịch vụ khiến khách hàng cảm thấy bối rối. Để giải quyết vấn đề này thì trong quá trình thiết kế nên có chức năng tìm kiếm. Nhiệm vụ chính của module tìm kiếm là giúp khách hàng có thể dễ dàng kiếm được món hàng hay dịch vụ cần thiết trong thời gian ngắn nhất dựa trên các từ khóa có dấu hoặc không dấu khi nhập vào.

Chức năng giỏ hàng: Bạn có thể dễ dàng nhận ra chức năng giỏ hàng trên các website bán hàng chuyên nghiệp, website thương mại điện tử có dịch vụ giao hàng tận nơi. Đặc điểm chính của chức năng giỏ hàng là giúp người mua dễ dàng đặt món hàng yêu thích bằng cách nhập thông tin cá nhân bao gồm họ tên, địa chỉ, số điện thoại và một số yêu cầu khác liên quan đến việc mua hàng. Sau đó thông tin sẽ được chuyển đến cho người bán và các bước kiểm tra đơn hàng, giao hàng, thanh toán tiền sẽ được thực hiện.

Thanh toán trực tuyến: Thanh toán trực tuyến hay còn có tên gọi khác là thanh toán online thông qua tài khoản ngân hàng hoặc các ví tiền ảo tạo trên các website thương mại do chính Nhà nước cấp quyền. Có thể nói đây là chức năng cơ bản cần thiết và không thể thiếu đối với các website bán hàng chuyên nghiệp hiện nay, đặc biệt là những trang web bán hàng lớn. Đa phần các công ty trả lương nhân viên bằng hình thức chuyển khoản thông qua ngân hàng. Do đó, khách hàng dễ dàng thanh toán mà không cần xếp hàng chờ đợi trước cây ATM hoặc quầy giao dịch để rút tiền và thanh toán tiền hàng. Tuy nhiên trên thực tế thì chức năng thanh toán online thông qua các trung gian như Ngân Lượng, Bảo Kim... cũng sẽ gây ra một chút phiền phức cho người dùng. Vậy nên các website bán hàng có thể có chức năng thanh toán online hoặc không tùy thuộc vào quy mô kinh doanh và chiến lược bán hàng của từng doanh nghiệp.

Hỗ trợ trực tuyến: Hiện nay có rất nhiều website công ty có lượng truy cập tương đối lớn và đều đặn. Mặc dù vậy, nhiều khách hàng khi tìm đến chưa hiểu rõ nhu cầu cũng như thông tin sản phẩm mà họ đang cần mua. Lúc này, phần mềm tư vấn trực tuyến (live chat) sẽ giúp bạn. Phần mềm tư vấn trực tuyến giống như một chiếc cầu nối cho phép nhân viên trực tiếp nói chuyện giải đáp thắc mắc cũng như giới thiệu dòng sản phẩm, dịch vụ phù hợp nhu cầu ngay tại văn phòng làm việc. Tư vấn thông tin kịp thời, đúng lúc, song với việc đáp ứng yêu cầu một cách nhanh nhất sẽ mang lại tỉ lệ thành công lớn trong việc bán hàng. Giao dịch tăng lên đáng kể góp phần không nhỏ trên con đường phát triển về lâu dài của doanh nghiệp. Hoặc nếu muốn đơn giản hơn và đỡ tốn kém chi phí mà vẫn tương tác trực tuyến được với khách hàng thì bạn cũng có thể tích hợp Messenger, Skype, Zalo chat cũng là một ý tưởng không tồi.

Hỗ trợ quảng cáo: Để trang web bán hàng có thể phát huy tối đa lợi ích vốn có, mang về nhiều khách hàng tiềm năng thì bạn cần thực hiện các chiến dịch quảng cáo trực tuyến như: Google Adwords, SEO, Facebook Ads... Một website bán hàng được thiết kế chuẩn SEO, tích hợp đầy đủ các chức năng như like, share mạng xã hội sẽ hỗ trợ đắc lực cho bạn làm SEO hoặc quảng cáo online nhằm thu hút khách hàng truy cập vào website [5].

## Các công trình nghiên cứu liên quan

Trong quá trình nghiên cứu và phát triển các hệ thống website bán hàng hiện đại, đã có nhiều đề tài, báo cáo và khóa luận được thực hiện tại các trường đại học lớn trong nước. Những công trình này không chỉ tập trung khai thác các nền tảng phổ biến như ASP.NET mà còn tiếp cận các framework mới như VueJS nhằm nâng cao trải nghiệm người dùng và tối ưu hóa hiệu quả quản lý hệ thống. Dưới đây là một số đề tài tiêu biểu liên quan trực tiếp đến các chủ đề này, góp phần cung cấp cái nhìn tổng quan về các hướng tiếp cận công nghệ, quy trình phân tích, thiết kế và triển khai hệ thống website bán hàng, cũng như các giải pháp kỹ thuật và chức năng nổi bật được áp dụng.

Đầu tiên là đề tài “Xây dựng website bán hàng với công nghệ ASP.NET MVC và Web API” vào năm 2016 của sinh viên Nguyễn Hồng Mai Thảo trường Đại học Sư phạm Kỹ thuật TP.HCM. Đề tài nhằm mục tiêu tìm hiểu và triển khai một website bán văn phòng phẩm online sử dụng công nghệ ASP.NET MVC5 kết hợp Web API và Bootstrap. Đề tài này trình bày chi tiết các bước phân tích và thiết kế hệ thống, gồm sơ đồ use case, thiết kế giao diện người dùng và quản trị, cùng mô hình cơ sở dữ liệu. Kết quả đạt được là website bán hàng công nghệ với đầy đủ các chức năng chính như đăng ký, đăng nhập, tìm kiếm mặt hàng, quản lý đơn hàng và các nghiệp vụ quản trị liên quan đến sản phẩm, cung cấp, vận chuyển [6].

Tiếp theo, là đề tài “Lập trình web bán hàng công nghệ” của nhóm sinh viên Đại học Công nghệ TP.HCM: Nguyễn Xuân Phúc, Hồ Văn Quang, Nguyễn Trọng Quý vào năm 2017. Trong báo cáo này, nhóm đã xây dựng thành công một website bán hàng sản phẩm công nghệ sử dụng nền tảng ASP.NET MVC5. Đề tài trình bày chi tiết các bước phân tích, thiết kế hệ thống với các sơ đồ use case, activity, class diagram và mô hình cơ sở dữ liệu. Ngoài ra, nhóm này cũng triển khai đầy đủ các chức năng như quản lý sản phẩm, đơn hàng, tài khoản người dùng, đồng thời xây dựng giao diện quản trị viên với các thao tác thêm, sửa, xóa và thống kê [8].

Một đề tài tiêu biểu khác ‘‘Giới thiệu framework VueJS’’ của nhóm sinh viên trường Đại học Công nghệ Thông tin thuộc Đại học Quốc gia Thành phố Hồ Chí Minh: Nguyễn Hữu Tiến, Nguyễn Lương Toàn, Đặng Ngọc Liêm, Trương Quốc Bảo vào năm 2020. Trong đề tài này các bạn đã giới thiệu được tổng quan về VueJS đồng thời cũng trình bài được một số nội dung quan trọng của framework này có thể kể đến như directive, xử lý event, component, vòng đời của một Vue instance ngoài ra trong đề tài còn nói thêm về các thư viện của VueJS như Vuex và Vue Router [7].

Cũng vào năm 2020, hai sinh viên của trường Đại học Giao Thông Vận Tải TP.Hồ Chí Minh bao gồm Bùi Văn Quang và Lê Tuấn Khải đã bắt tay vào nghiên cứu một số framework để xây dựng một website hoàn chỉnh, trong đó có VueJS với đề tài ‘‘Tìm hiểu framwork VueJS, Java Spring Boot và xây dựng website xem phim online’’. Trong đề tài này hai bạn cũng trình bài được tổng quan cũng như các đặc trưng quan trọng của VueJS [1].

Ngay sau đó, vào năm 2021 tại trường Đại học Công nghệ thông tin và truyền thông Việt Hàn, một bạn sinh viên có tên là Phan Thanh Nguyên thực hiện đề tài ‘‘Xây dựng ứng dụng chat với VueJS và Firebase’’. Tuy không đi sâu vào tìm hiểu VueJS nhưng đề tài cũng cho chúng ta biết được khái quát về VueJS cũng như lịch sử hình thành của framework này [10].

# THỰC HIỆN HÓA NGHIÊN CỨU

## Mô tả hệ thống

Hệ thống được thiết kế nhằm phục vụ ba nhóm đối tượng chính: Khách vãng lai, khách hàng (đã đăng nhập) và người quản trị. Mỗi nhóm người dùng sẽ có những chức năng và quyền truy cập riêng biệt, đảm bảo sự linh hoạt, bảo mật và hiệu quả trong quản lý và vận hành hệ thống thương mại điện tử.

Khách vãng lai là những người truy cập hệ thống mà chưa đăng ký hoặc đăng nhập tài khoản. Họ có thể xem danh sách sản phẩm và truy cập vào trang chi tiết để tìm hiểu thông tin về từng sản phẩm như hình ảnh, giá bán, mô tả, đánh giá… Ngoài ra, hệ thống hỗ trợ tính năng tìm kiếm và lọc sản phẩm theo tên, danh mục, mức giá hoặc các tiêu chí khác, giúp khách hàng dễ dàng tìm thấy sản phẩm mong muốn. Tuy nhiên, khách vãng lai không thể sử dụng các tính năng giỏ hàng, đặt hàng hay quản lý đơn hàng, trừ khi họ đăng ký và đăng nhập tài khoản.

Khách hàng sau khi đăng ký tài khoản có thể đăng nhập vào hệ thống để sử dụng đầy đủ các chức năng dành cho người mua. Họ có quyền cập nhật thông tin cá nhân, quản lý giỏ hàng như thêm sản phẩm, điều chỉnh số lượng hoặc xóa sản phẩm. Sau khi hoàn tất việc chọn sản phẩm, khách hàng có thể thực hiện đặt hàng và theo dõi danh sách các đơn hàng đã mua, bao gồm trạng thái xử lý (chờ xác nhận, đang giao, đã giao...) đồng thời khách hàng có thể thêm đánh giá cho sản phẩm đã mua. Tài khoản người dùng được bảo vệ bằng cơ chế mã hóa mật khẩu và xác thực OTP, đồng thời sử dụng JWT để đảm bảo an toàn khi truy cập hệ thống.

Về phía người quản trị, hệ thống hỗ trợ các chức năng quản lý toàn diện bao gồm: Danh mục sản phẩm, sản phẩm, người dùng và đơn hàng. Người quản trị có thể dễ dàng thêm mới, chỉnh sửa hoặc xóa thông tin danh mục và sản phẩm để cập nhật kịp thời với thị trường. Họ cũng có thể truy cập vào danh sách người dùng để xem một số thông tin phục vụ cho mục đích hỗ trợ hoặc kiểm duyệt. Đặc biệt, việc cập nhật trạng thái đơn hàng và theo dõi doanh thu là những tính năng quan trọng giúp người quản trị đánh giá hiệu quả hoạt động kinh doanh theo từng thời kỳ. Người quản trị còn có thể xem dự đoán doanh thu và các sản phẩm có thể sẽ bán chạy của tháng tiếp theo từ dữ liệu kinh doanh của ba tháng gần nhất, giúp người quản trịđưa ra quyết định kinh doanh kịp thời và hiệu quả hơn. Nhờ khả năng khai thác dữ liệu lịch sử để đưa ra dự đoán, hệ thống không chỉ hỗ trợ vận hành mà còn đóng vai trò tư vấn, định hướng cho hoạt động kinh doanh trong tương lai. Đây là điểm nổi bật giúp hệ thống trở nên thông minh và thích ứng nhanh với thị trường cạnh tranh.

## Các yêu cầu hệ thống

### Đối với người quản trị

Quản lý danh mục: Quản lý danh sách danh mục, thêm, chỉnh sửa thông tin danh mục.

Quản lý sản phẩm: Quản lý danh sách sản phẩm, thêm, chỉnh sửa và xóa thông tin sản phẩm.

Quản lý người dùng: Xem danh sách và thông tin người dùng.

Quản lý đơn hàng: Xem danh sách đơn hàng, cập nhật trạng thái đơn hàng.

Quản lý doanh thu: Xem thông tin về doanh thu.

### Đối với khách hàng

Quản lý tài khoản: Đăng ký, đăng nhập, đăng xuất và cập nhật thông tin cá nhân.

Xem sản phẩm: Xem danh sách sản phẩm, xem thông tin chi tiết sản phẩm.

Tìm kiếm và lọc sản phẩm: Tìm kiếm sản phẩm theo tên và lọc theo một số tiêu chí.

Quản lý giỏ hàng: Thêm sản phẩm vào giỏ hàng, cập nhật số lượng và xóa sản phẩm khỏi giỏ hàng.

Quản lý đơn hàng: Xem danh sách đơn hàng, xem thông tin và trạng thái đơn hàng.

### Đối với khách vãng lai

Xem sản phẩm: Xem danh sách sản phẩm, xem thông tin chi tiết sản phẩm.

Tìm kiếm và lọc sản phẩm: Tìm kiếm sản phẩm theo tên và lọc theo một số tiêu chí.

## Thiết kế dữ liệu

### Mô hình ERD



Hình 3.1: Mô hình ERD

### Mô tả thực thể

Bảng Users lưu trữ thông tin người dùng hệ thống, bao gồm thông tin cá nhân, xác thực và vai trò.

Bảng 3.1: Bảng Users

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| user\_id | Mã người dùng | int | PK |
| full\_name | Tên đầy đủ | nvarchar(100) |  |
| user\_name | Tên đăng nhập | varchar(100) |  |
| pass\_word | Mật khẩu | varchar(100) |  |
| create\_date | Ngày tạo tài khoản | datetime |  |
| email | Email xác thực | varchar(50) |  |
| user\_phone\_number | Số điện thoại | varchar(10) |  |
| user\_address | Địa chỉ | nvarchar(200) |  |
| user\_image\_url | Ảnh người dùng | varchar(100) |  |
| is\_verify | Trạng thái xác thực | bit |  |
| user\_role | Vai trò | int |  |
| refresh\_token |  | varchar(200) |  |

Bảng OTP chứa thông tin để xác thực tài khoản thông qua email.

Bảng 3.2: Bảng OTP

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| otp\_id | Mã OTP | int | PK |
| email | Email xác thực | varchar(50) |  |
| otp\_code | Mã code OTP | varchar(6) |  |
| otp\_expired | Hạn sử dụng OTP | datetime |  |

Bảng Brands dùng để lưu trữ các danh mục sản phẩm.

Bảng 3.3: Bảng Brands

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| brand\_id | Mã danh mục | int | PK |
| brand\_name | Tên danh mục | Nvarchar(100) |  |
| brand\_slug | Định danh Url | varchar(150) |  |
| brand\_image\_url | Đường dẫn đến ảnh danh mục | varchar(100) |  |

Bảng Phones dùng để lưu trữ thông tin cơ bản của sản phẩm như tên, mô tả, ảnh, màn hình, bộ vi xử lý, dung lượng pin, slug và liên kết với bảng Brand.

Bảng 3.4: Bảng Phones

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| phone\_id | Mã sản phẩm | int | PK |
| phone\_name | Tên sản phẩm | nvarchar(100) |  |
| phone\_slug | Định danh Url | varchar(150) |  |
| description | Mô tả sản phẩm | nvarchar(1000) |  |
| phone\_image\_url | Đường dẫn đến ảnh sản phẩm | varchar(100) |  |
| phone\_screen | Màn hình | nvarchar(100) |  |
| phone\_chip | Bộ vi xử lý | nvarchar(100) |  |
| phone\_battery | Dung lượng pin | nvarchar(50) |  |
| is\_active | Trạng thái hoạt động | bit |  |
| brand\_id | Mã danh mục | int | FK |

Bảng PhoneImages liên kết với bảng Phones, chứa danh sách đường dẫn đến ảnh sản phẩm.

Bảng 3.5: Bảng PhoneImages

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| img\_id | Mã hình ảnh | int | PK |
| phone\_id | Mã sản phẩm | int | FK |
| img\_image\_url | Đường dẫn đến ảnh sản phẩm | varchar(100) |  |

Bảng Colors chứa các màu sắc của sản phẩm.

Bảng 3.6: Bảng Colors

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| color\_id | Mã màu sắc | int | PK |
| color\_name | Tên màu sắc | varchar(50) |  |

Bảng Rams chứa các lựa chọn về ram của sản phẩm.

Bảng 3.7: Bảng Rams

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| ram\_id | Mã ram | int | PK |
| ram\_size | Dung lượng ram | varchar(50) |  |

Bảng Storages chứa các lựa chọn về bộ nhớ trong của sản phẩm.

Bảng 3.8: Bảng Storages

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| storage\_id | Mã storage | int | PK |
| storage\_size | Dung lượng bộ nhớ trong | varchar(50) |  |

Bảng PhoneVariants liên kết với các bảng Phones, Colors, Rams và Storages, dùng để lưu trữ các biến thể của sản phẩm, mỗi biến thể có giá, số lượng tồn kho riêng.

Bảng 3.9: Bảng PhoneVariants

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| var\_id | Mã biến thể | int | PK |
| phone\_id | Mã sản phẩm | int | FK |
| color\_id | Mã màu sắc | int | FK |
| ram\_id | Mã ram | int | FK |
| storage\_id | Mã storage | int | FK |
| var\_price | Giá biến thể | int |  |
| stock\_quantity | Số lượng còn lại | int |  |

Bảng Carts lưu thông tin giỏ hàng của người dùng hệ thống, liên kết với bảng Users để lấy thông tin người dùng và liên kết với bảng PhoneVariants để lấy thông tin sản phẩm cùng biến thể.

Bảng 3.10: Bảng Carts

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| cart\_id | Mã giỏ hàng | int | PK |
| user\_id | Mã người dùng | int | FK |
| var\_id | Mã biến thể | int | FK |
| quantity | Số lượng | int |  |
| added\_date | Ngày tạo | datetime |  |

Bảng PaymentMethods chứa các lựa chọn về phương thức thanh toán.

Bảng 3.11: Bảng PaymentMethods

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| method\_id | Mã phương thức thanh toán | int | PK |
| method\_name | Tên phương thức | nvarchar(100) |  |

Bảng Shippers chứa các lựa chọn về đơn vị vận chuyển cùng chi phí.

Bảng 3.12: Bảng Shippers

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| ship\_id | Mã vận chuyển | int | PK |
| ship\_name | Tên đơn vị | nvarchar(100) |  |
| ship\_cost | Chi phí vận chuyển | int |  |

Bảng Orders dùng để lưu thông tin đơn hàng khi người dùng đặt hàng, liên kết với các bảng Users, PaymentMethods và Shippers.

Bảng 3.13: Bảng Orders

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| order\_id | Mã đơn hàng | int | PK |
| user\_id | Mã người dùng | int | FK |
| order\_date | Ngày tạo đơn hàng | datetime |  |
| status | Trạng thái đơn hàng | int |  |
| method\_id | Mã phương thức thanh toán | int | FK |
| shipper\_id | Mã vận chuyển | int | FK |
| shipping\_address | Địa chỉ giao hàng | nvarchar(200) |  |
| order\_code | Mã code đơn hàng | varchar(20) |  |
| total\_price | Tổng giá trị đơn hàng | int |  |
| is\_prepaid | Trạng thái thanh toán | bit |  |

Bảng OrderDetails lưu trữ thông tin từng sản phẩm được mua trong đơn hàng, liên kết với bảng Orders và PhoneVariants.

Bảng 3.14: Bảng OrderDetails

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| order\_detail\_id | Mã chi tiết đơn hàng | int | PK |
| order\_id | Mã đơn hàng | int | FK |
| var\_id | Mã biến thể | int | FK |
| quantity | Số lượng mua | int |  |
| price\_at\_order | Tổng giá | int |  |
| is\_review | Trạng thái đánh giá | bit |  |

Bảng Reviews lưu trữ các đánh giá của người dùng về sản phẩm đã mua, bao gồm thông tin của người đánh giá, sản phẩm và biến thể được đánh giá, số điểm, hình ảnh, thời gian và nội dung đánh giá.

Bảng 3.15: Bảng Reviews

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Mô tả** | **Kiểu dữ liệu** | **Ràng buộc** |
| review\_id | Mã đánh giá | int | PK |
| user\_id | Mã người dùng | int | FK |
| phone\_id | Mã sản phẩm | int | FK |
| var\_id | Mã biến thể | int | FK |
| rating | Số điểm đánh giá | int |  |
| review\_text | Nội dung | nvarchar(1000) |  |
| review\_date | Thời gian đánh giá | datetime |  |
| review\_image\_url | Hình ảnh minh họa | varchar(100) |  |

## Thiết kế xử lý

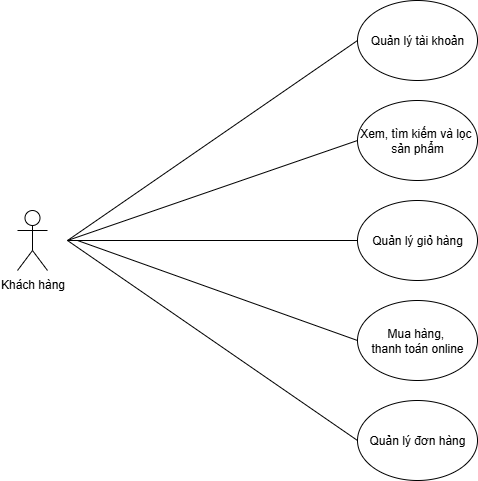
### Sơ đồ use case

Sơ đồ use case cho người quản trị trong hệ thống quản lý bán hàng. Sơ đồ mô tả các chức năng chính mà Admin có thể thực hiện bao gồm: Quản lý danh mục, quản lý sản phẩm, quản lý đơn hàng, quản lý khách hàng và quản lý doanh thu. Mỗi chức năng là một use case độc lập, thể hiện quyền truy cập và tác vụ của người quản trị đối với từng mảng nghiệp vụ trong hệ thống.



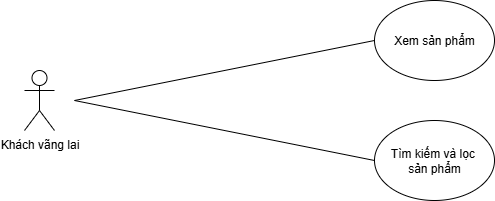
Hình 3.2: Sơ đồ use case người quản trị

Sơ đồ use case dành cho người dùngtrong hệ thống bán hàng. Sơ đồ mô tả các chức năng chính mà người dùng có thể thao tác, bao gồm: Quản lý tài khoản, xem và tìm kiếm sản phẩm, quản lý giỏ hàng, mua hàng và thanh toán online, quản lý đơn hàng. Mỗi chức năng thể hiện một nghiệp vụ quan trọng mà người dùng sẽ trải nghiệm trong quá trình sử dụng hệ thống.



Hình 3.3: Sơ đồ use case khách hàng

Sơ đồ use case cho khách vãng lai (người dùng không đăng nhập). Sơ đồ thể hiện hai chức năng cơ bản mà khách vãng lai có thể sử dụng: Xem sản phẩm và tìm kiếm, lọc sản phẩm. Điều này phản ánh đúng thực tế, khi chưa đăng nhập, người dùng vẫn có thể tham khảo thông tin và tìm kiếm các sản phẩm trên hệ thống.



Hình 3.4: Sơ đồ use case khách vãng lai

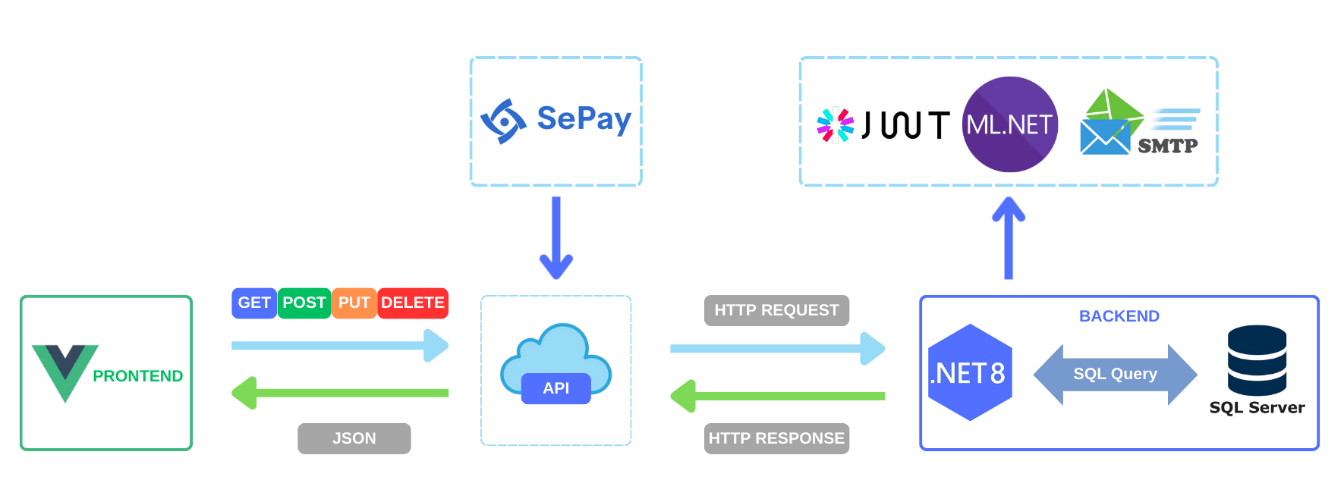
### Mô hình hoạt động hệ thống

Kiến trúc hệ thống website bán hàng được thiết kế theo mô hình hiện đại, đảm bảo tách biệt rõ ràng giữa frontend và backend, đồng thời hỗ trợ tích hợp nhiều dịch vụ tiện ích.

Ở phía người dùng, frontend sử dụng VueJS để xây dựng giao diện trực quan, đảm nhận các thao tác như xem sản phẩm, đặt hàng, quản lý tài khoản cá nhân… Tất cả yêu cầu từ người dùng sẽ được gửi tới tầng API trung gian thông qua các phương thức GET, POST, PUT, DELETE với dữ liệu truyền dưới dạng JSON.

Tầng API đóng vai trò trung gian tiếp nhận, xử lý và chuyển tiếp các request tới backend. Backend được xây dựng trên nền tảng .NET 8, chịu trách nhiệm xử lý toàn bộ logic nghiệp vụ và tương tác với cơ sở dữ liệu SQL Server. Kết quả xử lý sẽ được trả về frontend thông qua API dưới dạng phản hồi JSON.

Ngoài ra, hệ thống còn tích hợp các dịch vụ bên ngoài như SePay để thanh toán trực tuyến, JWT để xác thực và phân quyền người dùng, ML.NET cho các tính năng trí tuệ nhân tạo như gợi ý sản phẩm và SMTP để tự động gửi thông tin đơn hàng qua email, xác thực người dùng khi đăng ký. Nhờ đó, hệ thống có khả năng vận hành linh hoạt, mở rộng dễ dàng, đáp ứng hiệu quả các nhu cầu của một nền tảng thương mại điện tử hiện đại.



Hình 3.5: Mô hình hoạt động hệ thống

# KẾT QUẢ NGHIÊN CỨU

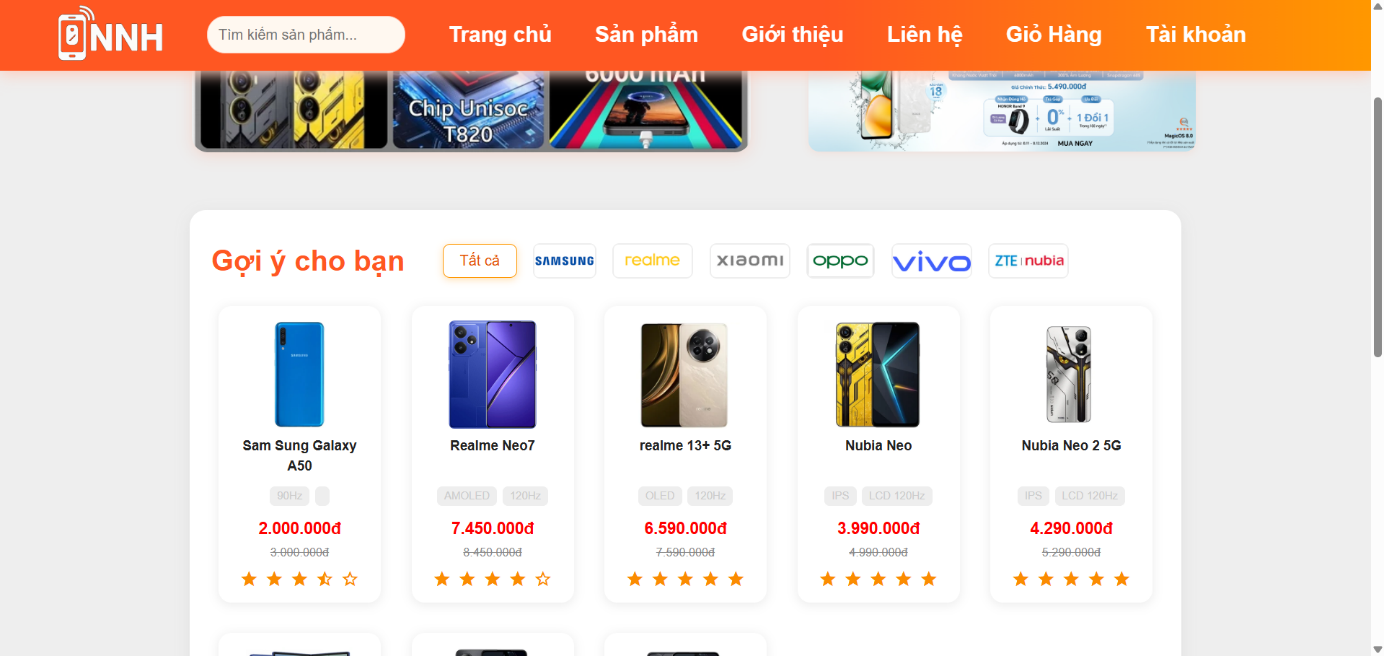
## Giao diện khách hàng

### Giao diện trang chủ

Trang chủ của hệ thống chú trọng sự trực quan, hiện đại và thân thiện, tạo trải nghiệm thuận tiện ngay từ lần truy cập đầu tiên.

Phía trên là thanh tìm kiếm và menu chính giúp người dùng dễ dàng điều hướng sang các trang khác như sản phẩm, giới thiệu, liên hệ, giỏ hàng và tài khoản. Bên dưới có các hình ảnh quảng cáo, dưới quảng cáo là nơi hiển thị một số sản phẩm điện thoại được sắp xếp theo từng thương hiệu như Samsung, Realme, Xiaomi, Oppo, Vivo, ZTE, Nubia… Người dùng có thể lọc nhanh sản phẩm theo thương hiệu bằng các nút lựa chọn phía trên danh sách.

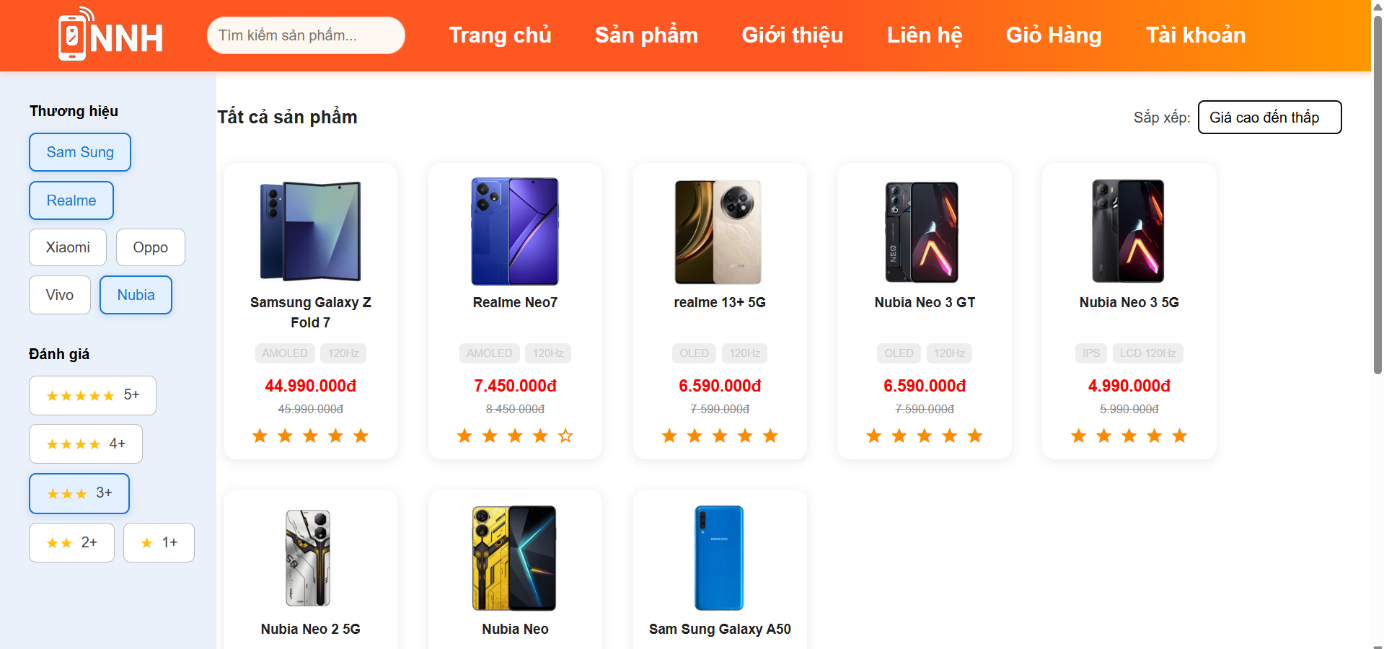
Mỗi sản phẩm hiển thị một số thông tin bao gồm hình ảnh, tên, các thông số nổi bật, giá bán hiện tại và xếp hạng sao đánh giá. Thiết kế này giúp khách hàng dễ dàng xem và so sánh giữa các sản phẩm, đồng thời tăng khả năng tương tác và mua hàng ngay từ trang chủ.



Hình 4.1: Giao diện trang chủ

### Giao diện trang sản phẩm

Trang sản phẩm hiển thị toàn bộ các điện thoại có trên website, cho phép người dùng lọc nhanh theo thương hiệu hoặc đánh giá sao, sắp xếp sản phẩm theo giá và dễ dàng xem thông tin, giá bán, đánh giá của từng sản phẩm trong giao diện trực quan, dễ sử dụng.

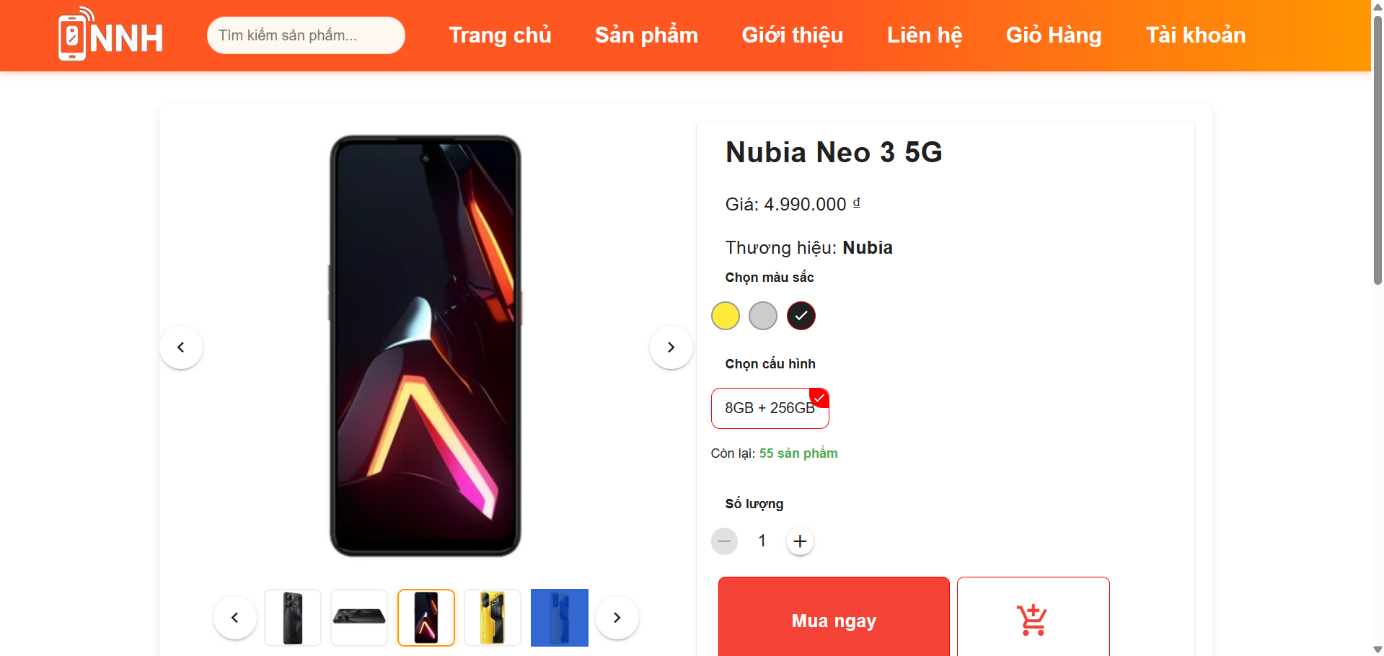


Hình 4.2: Giao diện trang sản phẩm

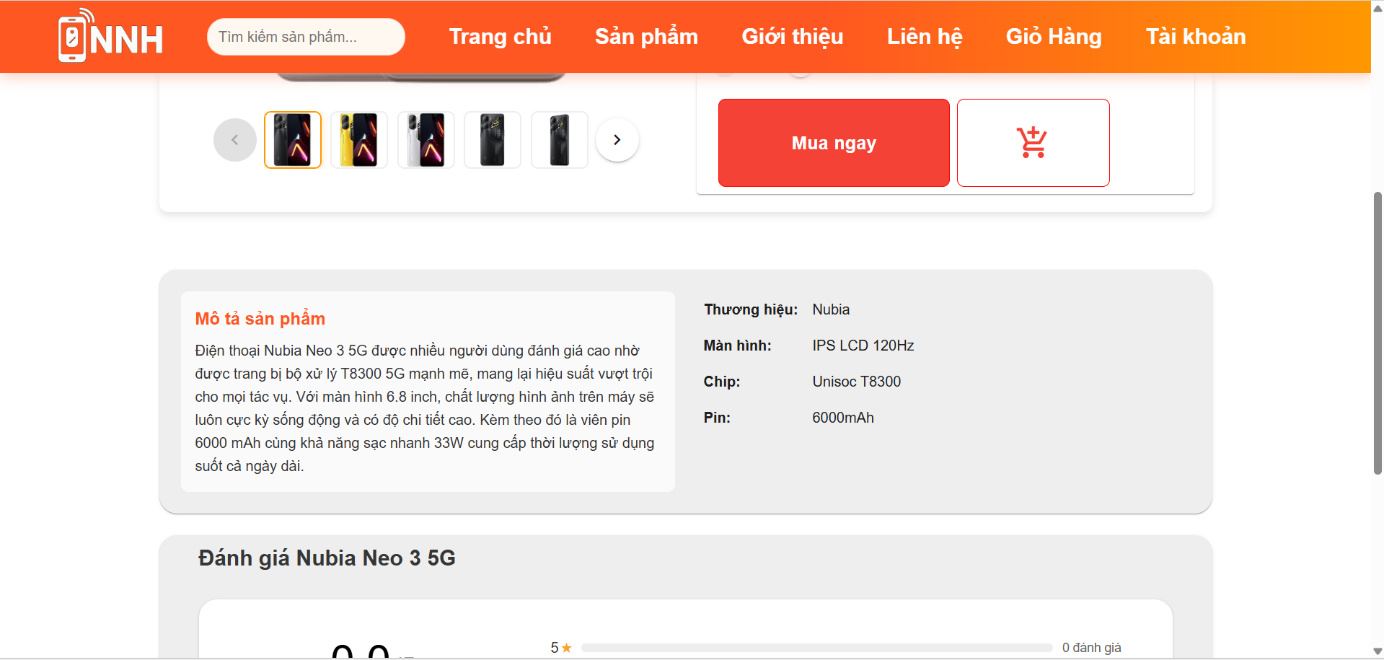
### Giao diện chi tiết sản phẩm

Trang **chi tiết sản phẩm** cung cấp đầy đủ thông tin về điện thoại mà người dùng đang quan tâm. Giao diện gồm hình ảnh sản phẩm ở bên trái, cho phép xem nhiều góc độ khác nhau, và các tùy chọn về màu sắc, cấu hình, số lượng mua ở bên phải. Thông tin về giá bán, thương hiệu, tình trạng kho hàng, cùng các nút mua nhanh và thêm vào giỏ hàng được bố trí rõ ràng, thuận tiện cho thao tác.

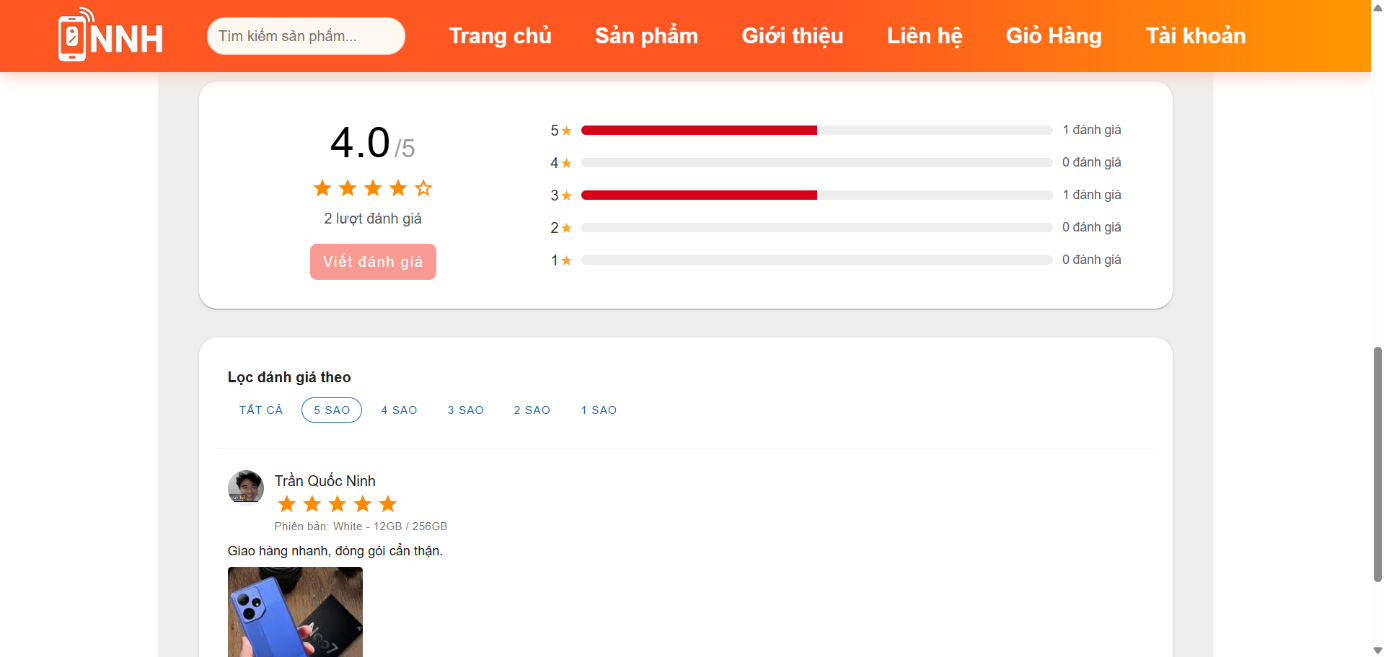
Bên dưới là phần mô tả chi tiết sản phẩm, thông số kỹ thuật như màn hình, chip, dung lượng pin… giúp người dùng dễ dàng nắm bắt các đặc điểm nổi bật. Cuối trang là khu vực đánh giá và nhận xét, hiển thị điểm đánh giá trung bình, phân bố sao và các bình luận thực tế từ khách hàng đã mua hàng, góp phần tăng độ tin cậy và hỗ trợ quyết định mua sắm.



Hình 4.3: Giao diện chi tiết sản phẩm



Hình 4.4 Giao diện chi tiết sản phẩm – thông số



Hình 4.5 Giao diện chi tiết sản phẩm – đánh giá

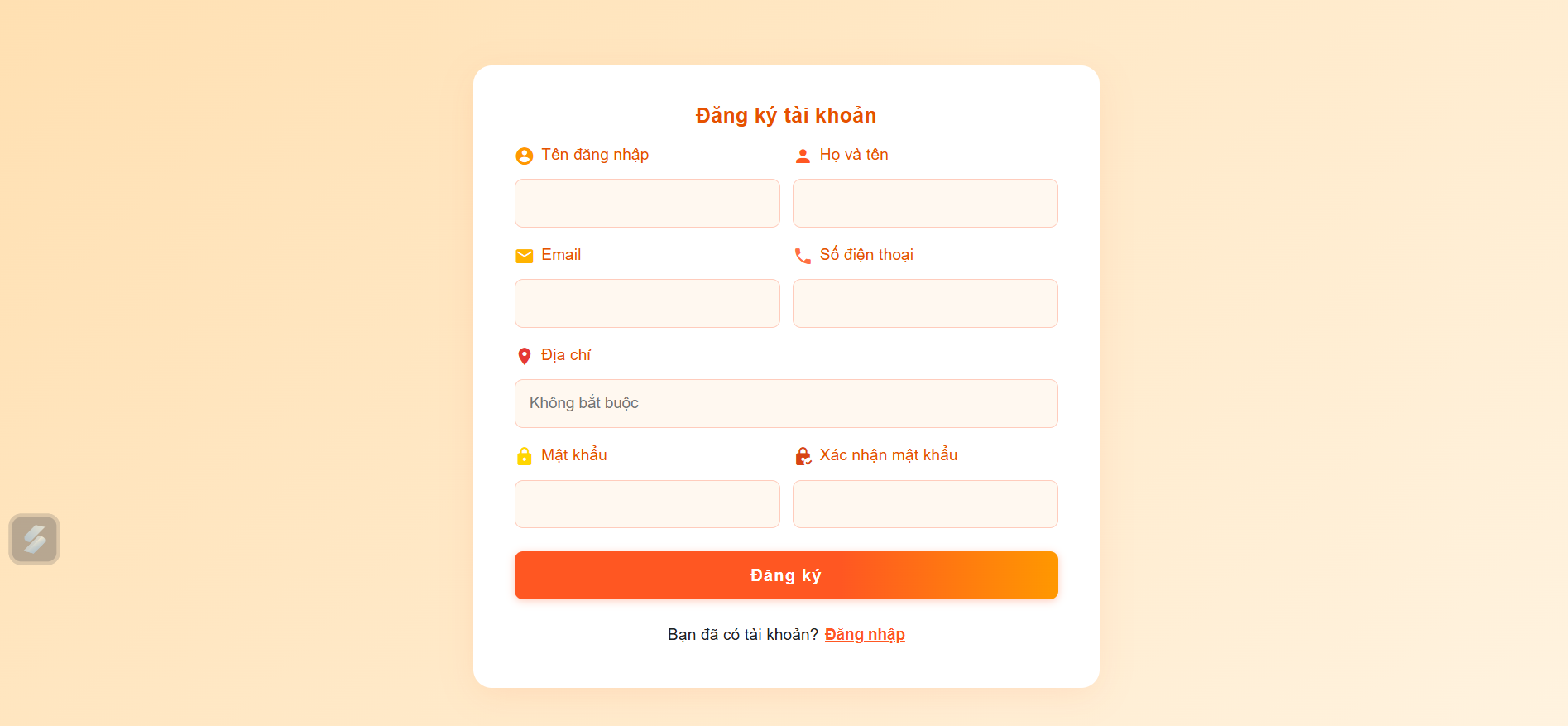
### Giao diện đăng ký, đăng nhập

Trang đăng nhập của hệ thống được xây dựng với bố cục gọn gàng, tập trung vào những chức năng chính. Người dùng cần cung cấp tên đăng nhập và mật khẩu sau đó dùng nút “Đăng nhập” để thực hiện quá trình đăng nhập. Ngoài ra, giao diện còn hỗ trợ tùy chọn ghi nhớ thông tin đăng nhập, giúp thuận tiện hơn cho những lần truy cập sau. Trong trường hợp “quên mật khẩu”, hệ thống cho phép chuyển sang giao diện cung cấp chức năng thay đổi mật khẩu. Khi người dùng chưa có tài khoản, có thể chuyển sang trang đăng ký tài khoản với liên kết “Đăng ký” để tạo một tài khoản mới.



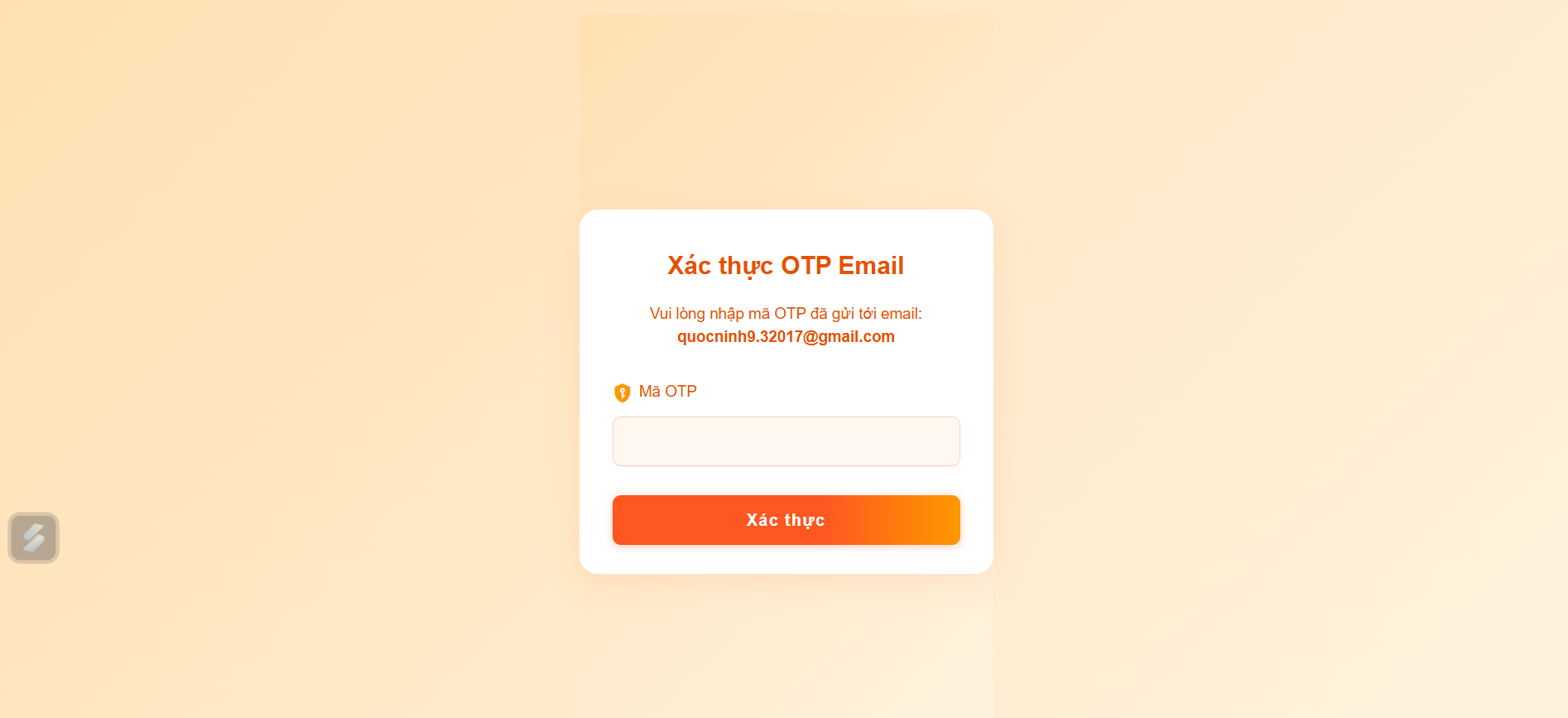
Hình 4.6 Giao diện đăng nhập tài khoản

Trang đăng ký tài khoản được thiết kế nhằm thu thập đầy đủ thông tin cần thiết từ người dùng để khởi tạo tài khoản mới. Các trường bắt buộc bao gồm tên đăng nhập, họ và tên, email, số điện thoại và mật khẩu. Hệ thống cũng cung cấp trường địa chỉ ở dạng tùy chọn, cho phép người dùng bổ sung khi cần. Ngoài ra, để đảm bảo tính chính xác và bảo mật, giao diện yêu cầu nhập lại mật khẩu nhằm xác nhận thông tin. Sau khi điền đầy đủ, người dùng nhấn nút “Đăng ký” để hoàn tất. Đối với những người đã có tài khoản, liên kết “Đăng nhập” ở phía dưới giúp họ chuyển nhanh sang trang đăng nhập.



Hình 4.7 Giao diện đăng ký tài khoản

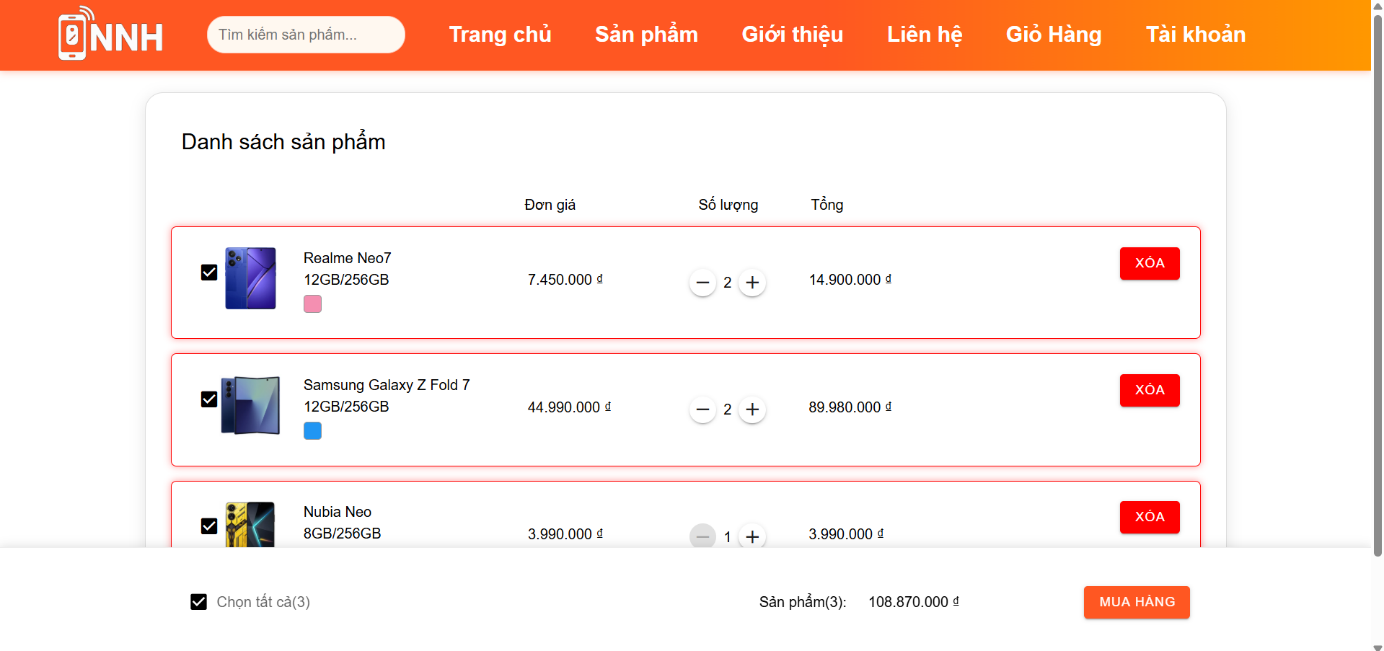
Tiếp theo, trang xác thực OTP Email đóng vai trò quan trọng trong việc bảo đảm an toàn tài khoản. Sau khi đăng ký, hệ thống gửi mã OTP đến địa chỉ email đã cung cấp. Người dùng cần nhập mã này vào ô “Mã OTP” và nhấn nút “Xác thực” để hoàn tất quy trình kích hoạt tài khoản. Cơ chế này giúp xác minh email là hợp lệ, đồng thời ngăn chặn tình trạng đăng ký giả mạo, góp phần nâng cao độ tin cậy và bảo mật của hệ thống.



Hình 4.8 Giao diện xác thực OTP

### Giao diện giỏ hàng

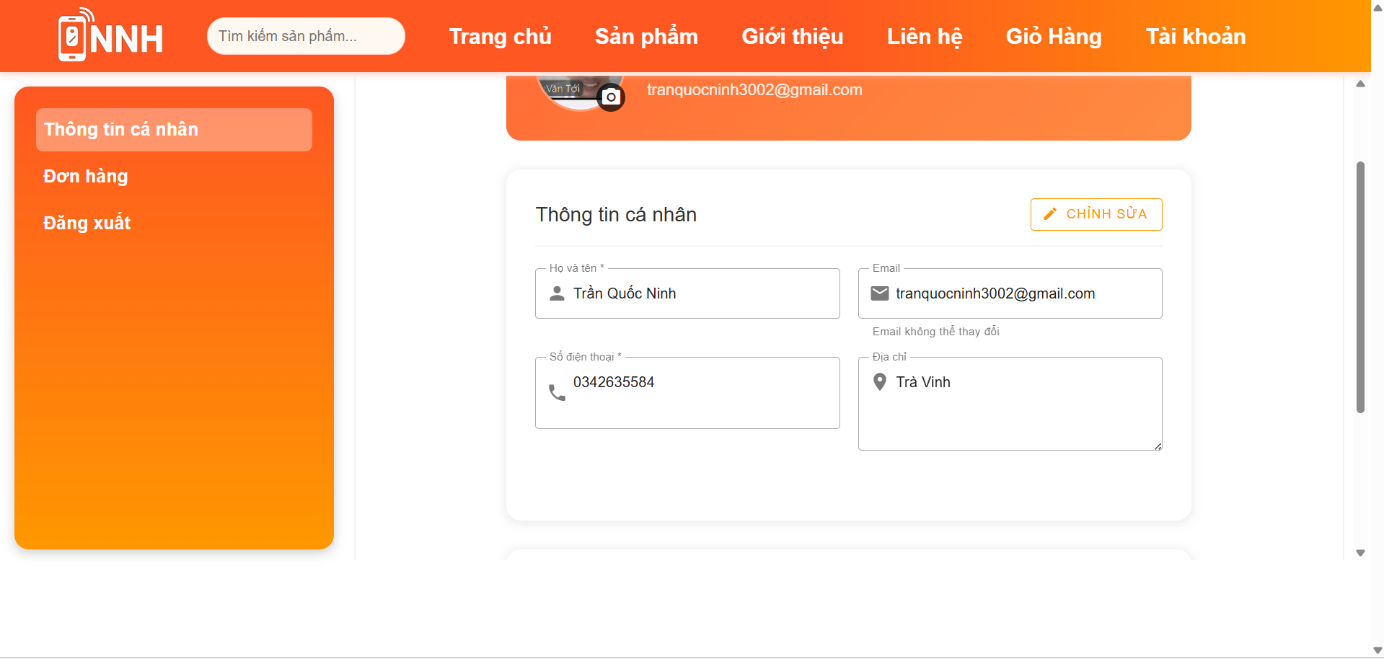
Trang giỏ hàng hiển thị danh sách các sản phẩm mà người dùng đã thêm vào giỏ hàng. Tại đây, người dùng có thể xem tên, hình ảnh, cấu hình từng sản phẩm, điều chỉnh số lượng, xem đơn giá, tổng giá tiền cho từng mặt hàng hoặc xóa sản phẩm khỏi giỏ nếu không muốn mua nữa. Phía dưới là tổng số lượng sản phẩm đã chọn, tổng giá trị đơn hàng và nút mua hàng để tiến hành quá trình mua hàng.



Hình 4.9 Giao diện giỏ hàng

### Giao diện thông tin người dùng

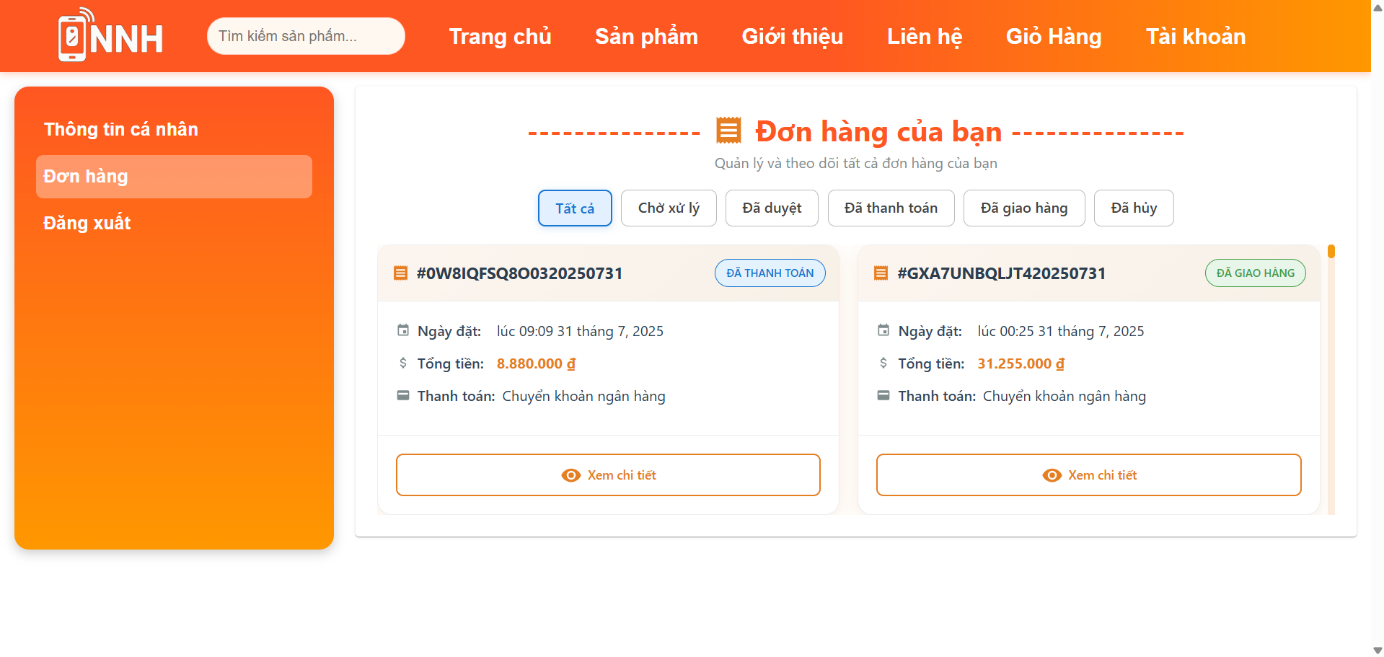
Trang thông tin người dùng hiển thị các thông tin cá nhân cơ bản của người dùng như họ tên, ảnh đại diện, email, số điện thoại và địa chỉ. Giao diện cho phép người dùng dễ dàng xem và cập nhật lại thông tin cá nhân. Người dùng cũng có thể thay đổi thông tin nhạy cảm như mật khẩu bằng cách cung cấp mật khẩu cũ.



Hình 4.10: Giao diện thông tin người dùng

### Giao diện danh sách đơn hàng

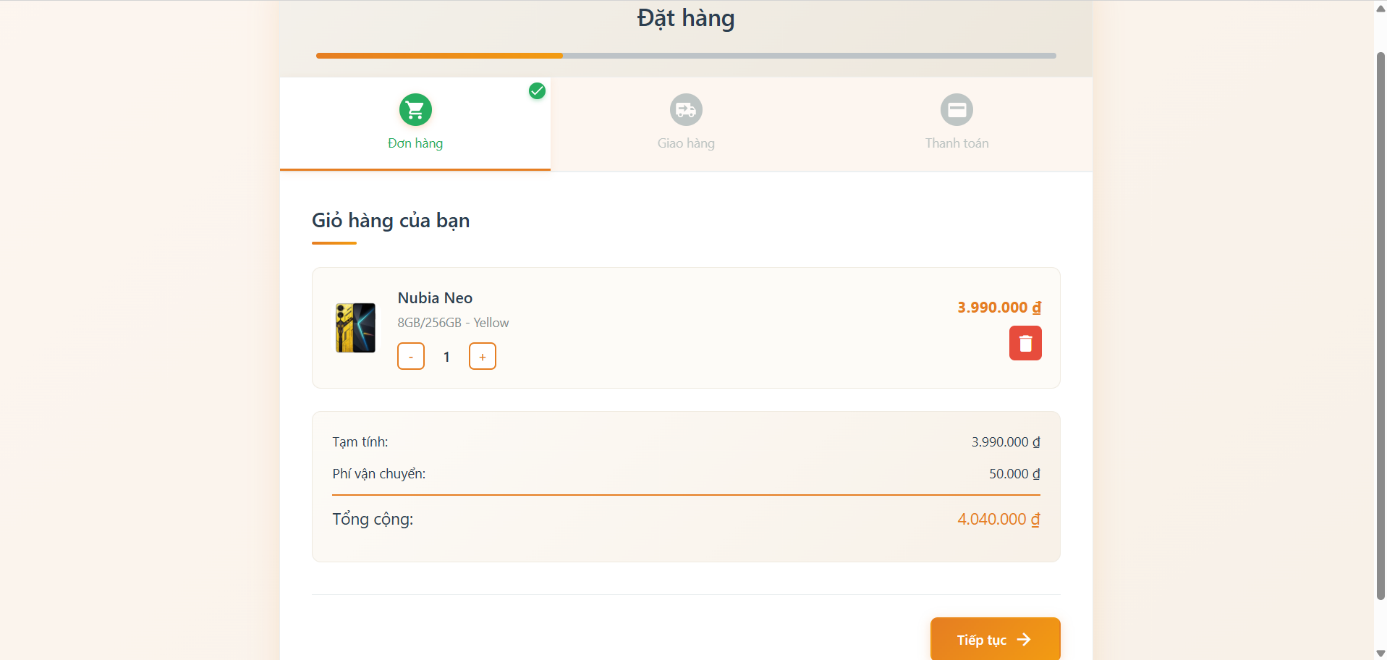
Trang **danh sách đơn hàng** cho phép người dùng quản lý và theo dõi tất cả các đơn hàng của mình. Giao diện hiển thị thông tin cơ bản của từng đơn hàng như mã đơn, ngày đặt, tổng tiền, phương thức thanh toán và trạng thái đơn hàng. Người dùng có thể lọc nhanh các đơn hàng theo trạng thái (đã xử lý, đã duyệt, đã thanh toán, đã giao hàng, đã hủy) và xem chi tiết từng đơn hàng khi cần.



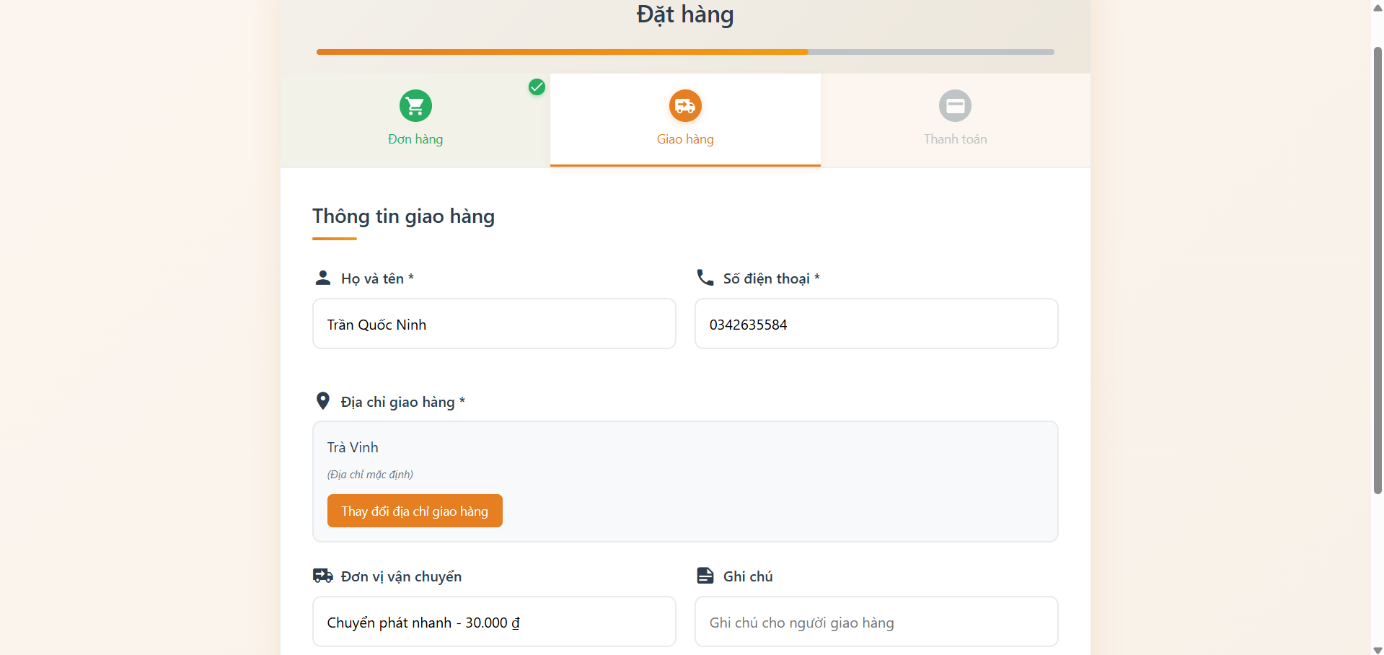
Hình 4.11: Giao diện danh sách đơn hàng

### Giao diện khi mua sản phẩm

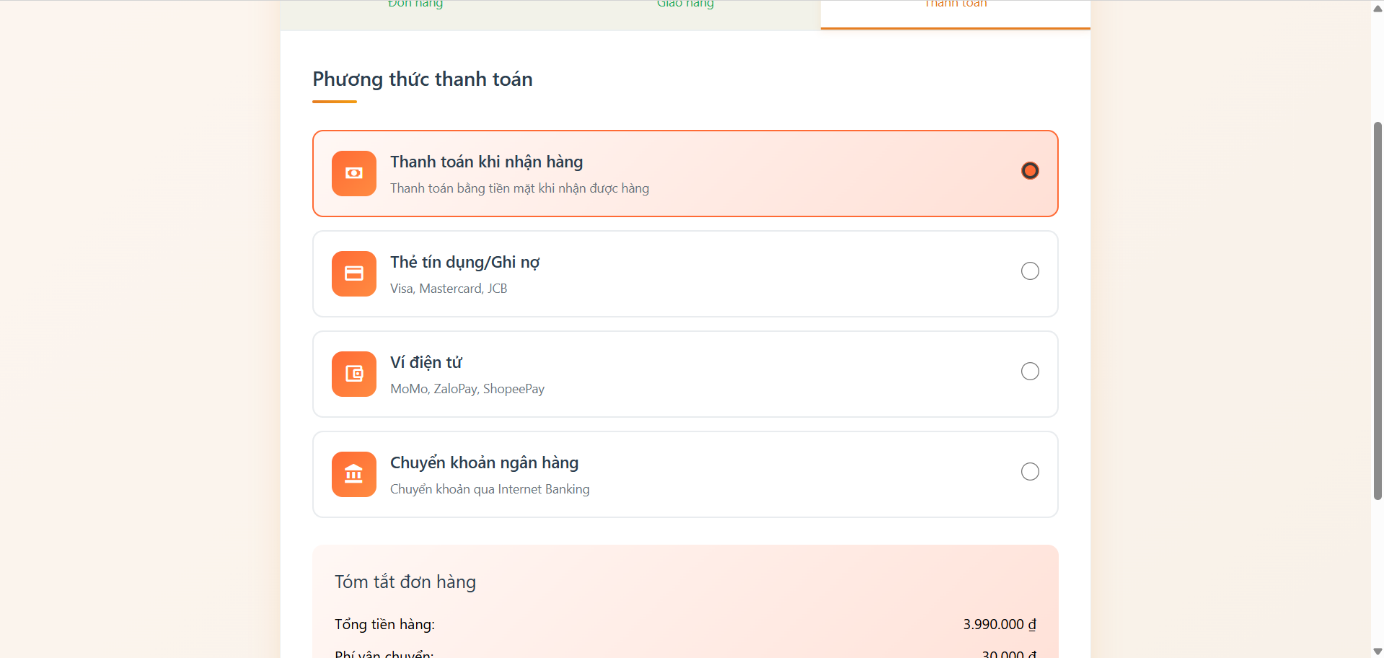
Trang mua hàng được hiển thị trong quá trình mua sản phẩm, gồm ba bước chính: kiểm tra giỏ hàng, nhập thông tin giao hàng và chọn phương thức thanh toán. Ở bước đầu, người dùng xem lại các sản phẩm đã chọn, điều chỉnh số lượng và kiểm tra tổng tiền cùng phí vận chuyển. Tiếp theo, người dùng xem thông tin giao hàng gồm họ tên, số điện thoại, địa chỉ nhận hàng, chọn đơn vị vận chuyển, có thể thay đổi địa chỉ nhận hàng. Cuối cùng, trang cung cấp nhiều phương thức thanh toán như trả tiền khi nhận hàng hoặc chuyển khoản ngân hàng, kèm theo phần tóm tắt đơn hàng trước khi xác nhận đặt mua. Giao diện được thiết kế theo từng bước rõ ràng, giúp quá trình đặt hàng dễ hiểu và thuận tiện cho người dùng.



Hình 4.12: Giao diện mua sản phẩm - đặt hàng



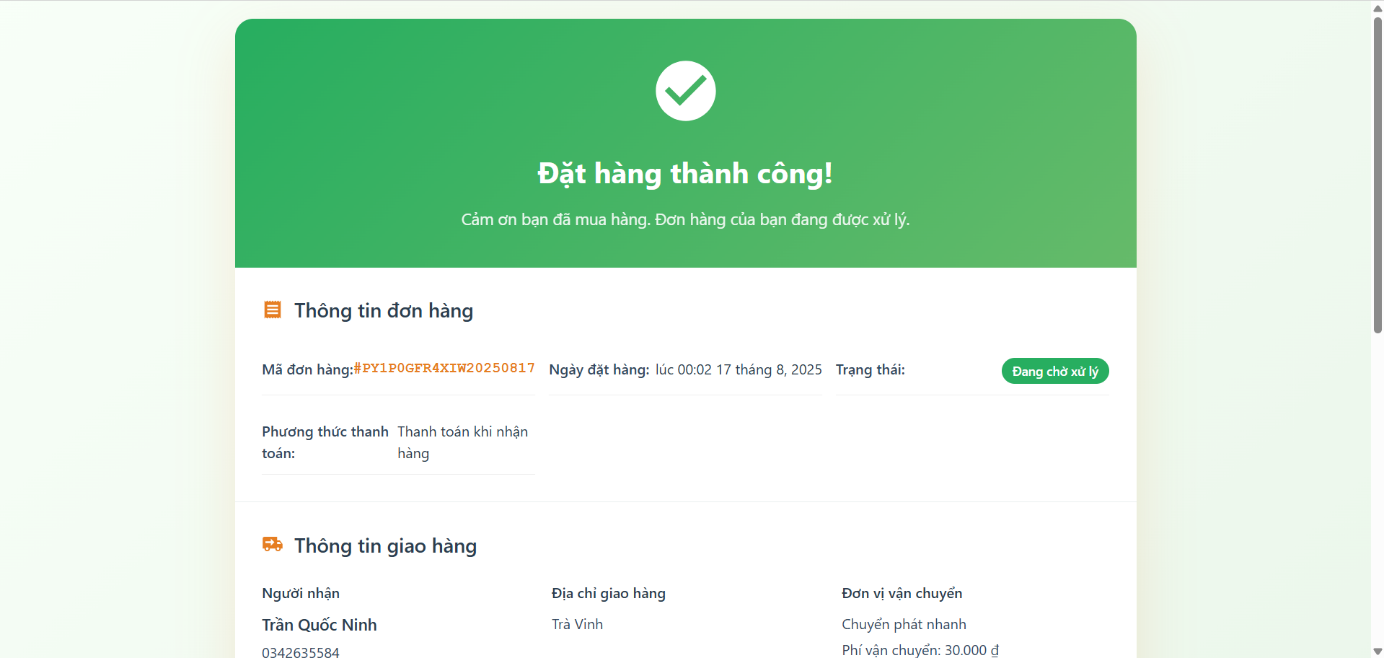
Hình 4.13: Giao diện mua sản phẩm - thông tin giao hàng



Hình 4.14: Giao diện mua sản phẩm - phương thức thanh toán

### Giao diện khi đặt hàng thành công

Trang đặt hàng thành công xác nhận người dùng đã hoàn tất quá trình mua hàng với thông báo nổi bật và biểu tượng thành công ở phía trên. Trang hiển thị đầy đủ thông tin đơn hàng vừa đặt, bao gồm mã đơn hàng, ngày đặt, trạng thái đơn hàng, phương thức thanh toán, thông tin giao hàng và đơn vị vận chuyển. Giao diện giúp người dùng dễ dàng kiểm tra lại chi tiết đơn hàng và yên tâm chờ hệ thống xử lý, đồng thời mang lại cảm giác tin cậy và chuyên nghiệp sau mỗi lần giao dịch.

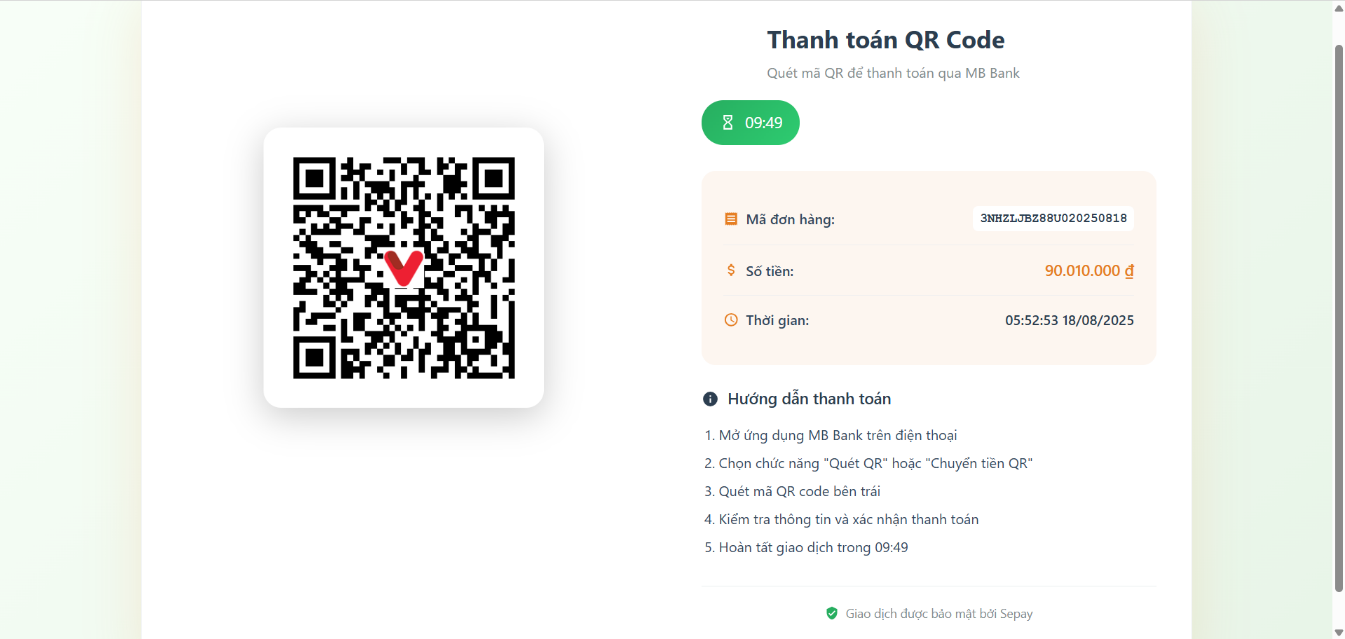


Hình 4.15: Giao diện đặt hàng thành công

### Giao diện thanh toán online

Trang **giao diện thanh toán QR Code** dành cho khách hàng thực hiện giao dịch qua ngân hàng. Ở trung tâm giao diện là một mã QR lớn, người dùng chỉ cần mở ứng dụng ngân hàng trên điện thoại, chọn chức năng “Quét QR” hoặc “Chuyển tiền QR”, sau đó quét mã QR hiển thị để tiến hành thanh toán.

Bên phải màn hình, hệ thống hiển thị rõ các thông tin quan trọng gồm: Mã đơn hàng, số tiền phải thanh toán và thời gian giao dịch. Ngoài ra, còn có đồng hồ đếm ngược giúp khách hàng biết được thời gian còn lại để hoàn tất thanh toán. Phía dưới là hướng dẫn chi tiết từng bước, đảm bảo người dùng dễ dàng thao tác, kiểm tra lại thông tin trước khi xác nhận thanh toán. Giao diện này giúp tối ưu trải nghiệm người dùng, vừa tiện lợi vừa đảm bảo an toàn cho các giao dịch online.

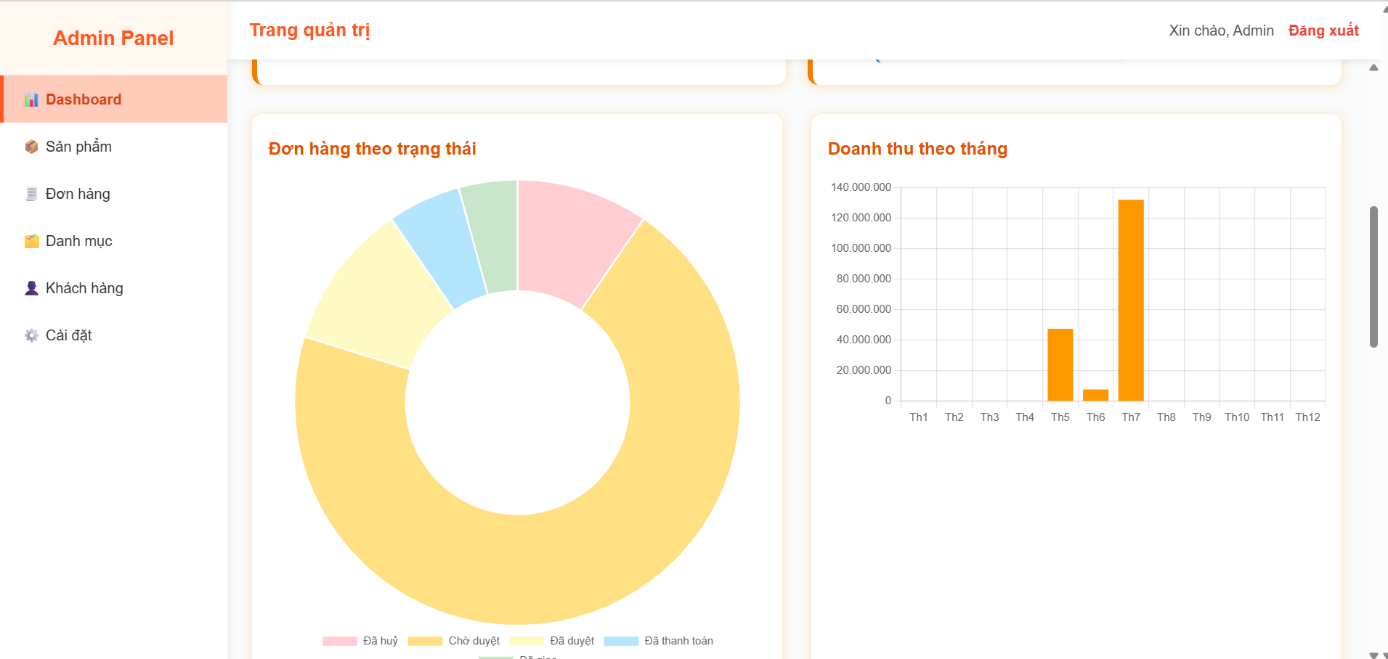


Hình 4.16: Giao diện thanh toán QR Code

## Giao diện quản trị

### Giao diện trang dashboard

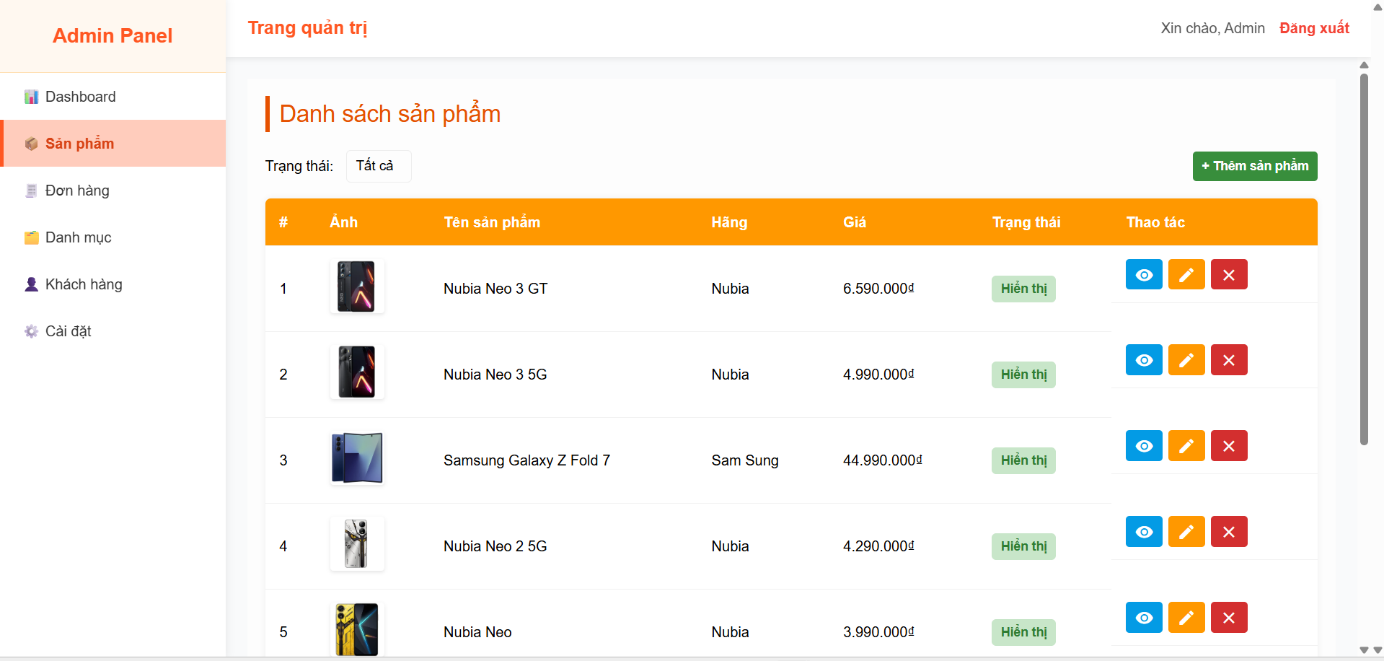
Trang dashboard của giao diện người quản trị cung cấp cái nhìn tổng quan về hoạt động của hệ thống. Bên trái là menu điều hướng với các mục quản lý chính như sản phẩm, đơn hàng, danh mục, khách hàng và cài đặt. Trung tâm dashboard hiển thị các thông tin về doanh thua, đơn hàng, người dùng và các biểu đồ trực quan: Biểu đồ tròn thể hiện số lượng đơn hàng theo trạng thái đơn hàng và biểu đồ cột thống kê doanh thu theo từng tháng. Nhờ đó, người quản trị có thể dễ dàng theo dõi tình hình kinh doanh và quản lý hiệu quả hoạt động của cửa hàng ngay từ trang tổng quan.



Hình 4.17: Giao diện trang dashboard

### Giao diện trang quản lý sản phẩm

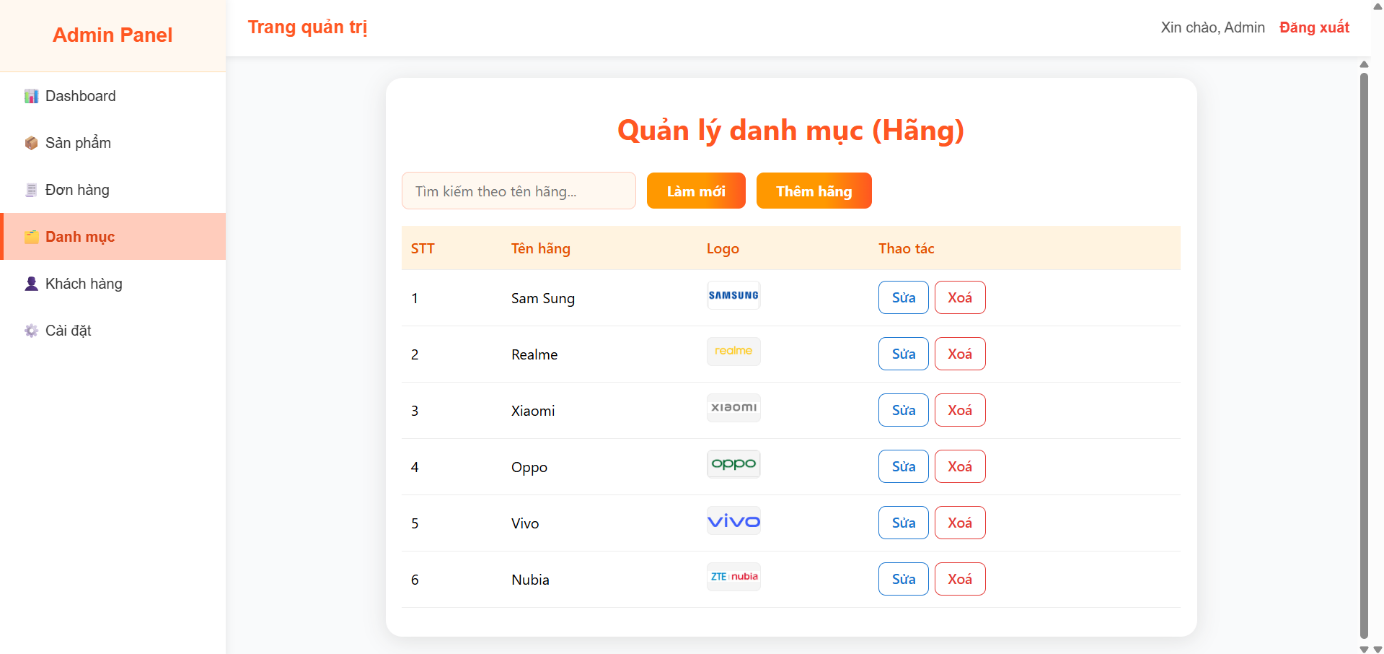
Trang quản lý sản phẩm của người quản trị hiển thị danh sách tất cả sản phẩm trên hệ thống với các thông tin như hình ảnh, tên sản phẩm, hãng, giá bán và trạng thái hiển thị. Người quản trị có thể dễ dàng thực hiện các thao tác như xem chi tiết, chỉnh sửa, xóa sản phẩm hoặc thêm sản phẩm mới thông qua các nút thao tác tương ứng. Giao diện được tổ chức dạng bảng, rõ ràng và thuận tiện cho việc quản lý, cập nhật dữ liệu sản phẩm một cách hiệu quả.



Hình 4.18: Giao diện trang quản lý sản phẩm

### Giao diện trang quản lý danh mục

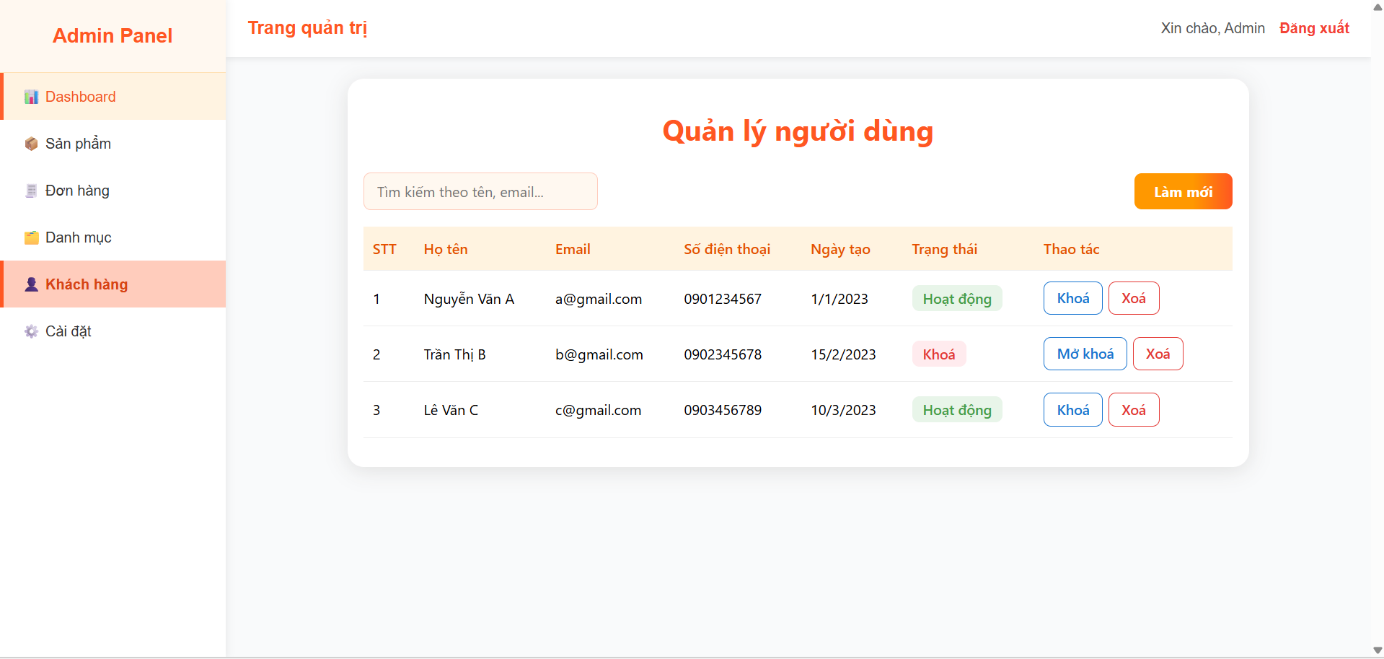
Trang quản lý danh mụccho phép người quản trị xem, thêm mới, sửa hoặc xóa các thương hiệu điện thoại trong hệ thống. Danh sách hiển thị tên hãng, logo đại diện và các nút thao tác tương ứng, hỗ trợ tìm kiếm nhanh theo tên hãng. Giao diện đơn giản, dễ sử dụng, giúp quản lý danh mục sản phẩm hiệu quả và thuận tiện.



Hình 4.19: Giao diện trang quản lý danh mục

### Giao diện trang quản lý người dùng

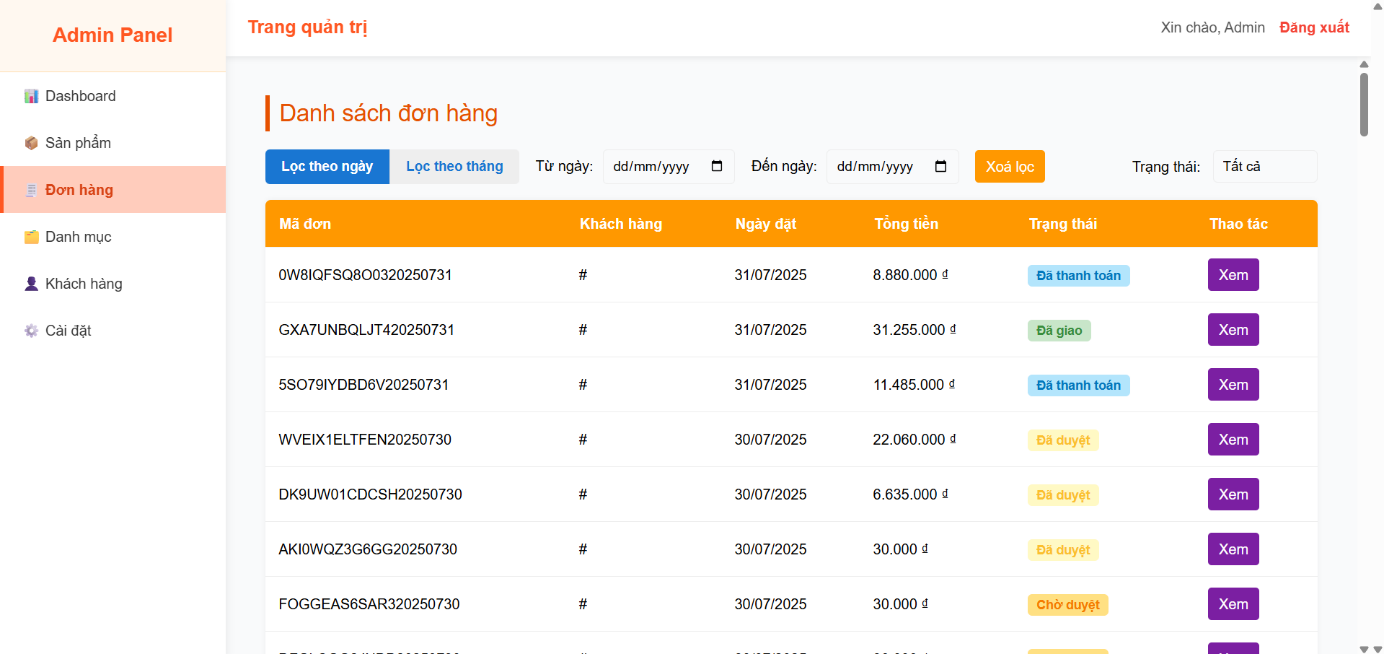
Trang quản lý người dùng cho phép người quản trị theo dõi và quản lý danh sách khách hàng trên hệ thống. Danh sách hiển thị các thông tin cơ bản như họ tên, email, số điện thoại, ngày tạo tài khoản, trạng thái hoạt động và cung cấp các nút thao tác như khóa, mở khóa hoặc xóa tài khoản. Ngoài ra, trang còn hỗ trợ tìm kiếm nhanh theo tên hoặc email, giúp việc quản lý người dùng hiệu quả, thuận tiện và an toàn.



Hình 4.20: Giao diện trang quản lý người dùng

### Giao diện trang quản lý đơn hàng

Trang quản lý đơn hàng của người quản trị cho phép theo dõi, tìm kiếm và kiểm soát toàn bộ đơn hàng trên hệ thống. Danh sách đơn hàng hiển thị các thông tin chính như mã đơn, khách hàng, ngày đặt, tổng tiền, trạng thái đơn hàng, cùng các nút thao tác để xem chi tiết. Ngoài ra, giao diện hỗ trợ lọc đơn hàng theo ngày, theo tháng và theo trạng thái, giúp người quản trị dễ dàng tra cứu và quản lý tiến độ xử lý đơn hàng một cách hiệu quả và trực quan.



Hình 4.21: Giao diện trang quản lý đơn hàng

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

**Kết quả đạt được**

Thiết kế CSDL hoàn chỉnh, CSDL được thiết kế đáp ứng hiệu quả yêu cầu của ứng dụng, đảm bảo tính chính xác và độ tin cậy của dữ liệu.

Xây dựng website bán hàng đầy đủ chức năng.

Giao diện website được thiết kế trực quan, linh hoạt, dễ dàng sử dụng cho cả người quản trị và khách hàng.

**Ưu điểm**

Website được trang bị đầy đủ các tính năng cần thiết cho một hệ thống bán hàng trực tuyến, đáp ứng nhu cầu quản lý và kinh doanh của người dùng.

Sử dụng nhiều phương thức thanh toán trực tuyến, tăng trải nghiệm mua sắm trực tuyến.

Website hoạt động ổn định, xử lý nhanh chóng các tác vụ và đảm bảo tính bảo mật thông tin cho cả người dùng và doanh nghiệp.

Giao diện thân thiện, dễ sử dụng, giúp khách hàng dễ dàng tìm kiếm và mua sản phẩm, đồng thời hỗ trợ người quản trị trong việc quản lý hoạt động kinh doanh.

**Khuyết điểm**

Website chưa có các tính năng nâng cao như chương trình khách hàng thân thiết, tích điểm, khuyến mãi. Những tính năng này có thể giúp tăng trải nghiệm mua sắm.

Website chưa có khả năng cá nhân hóa trải nghiệm mua sắm cho từng khách hàng dựa trên lịch sử mua hàng, sở thích. Điều này có thể làm giảm sự hài lòng và khả năng quay lại mua hàng của khách hàng.

## Hướng phát triển

Thời gian sắp tới, tôi sẽ dành nhiều thời gian để tiếp tục nghiên cứu sâu hơn về .NET Framework và VueJS, đồng thời cố gắng tìm hiểu và nắm vững các kiến thức nâng cao liên quan đến hai công nghệ này cũng như những thành phần mở rộng hỗ trợ. Sau đó sẽ vận dụng những kiến thức đã nghiên cứu để xây dựng và hoàn thiện một website đầy đủ chức năng.

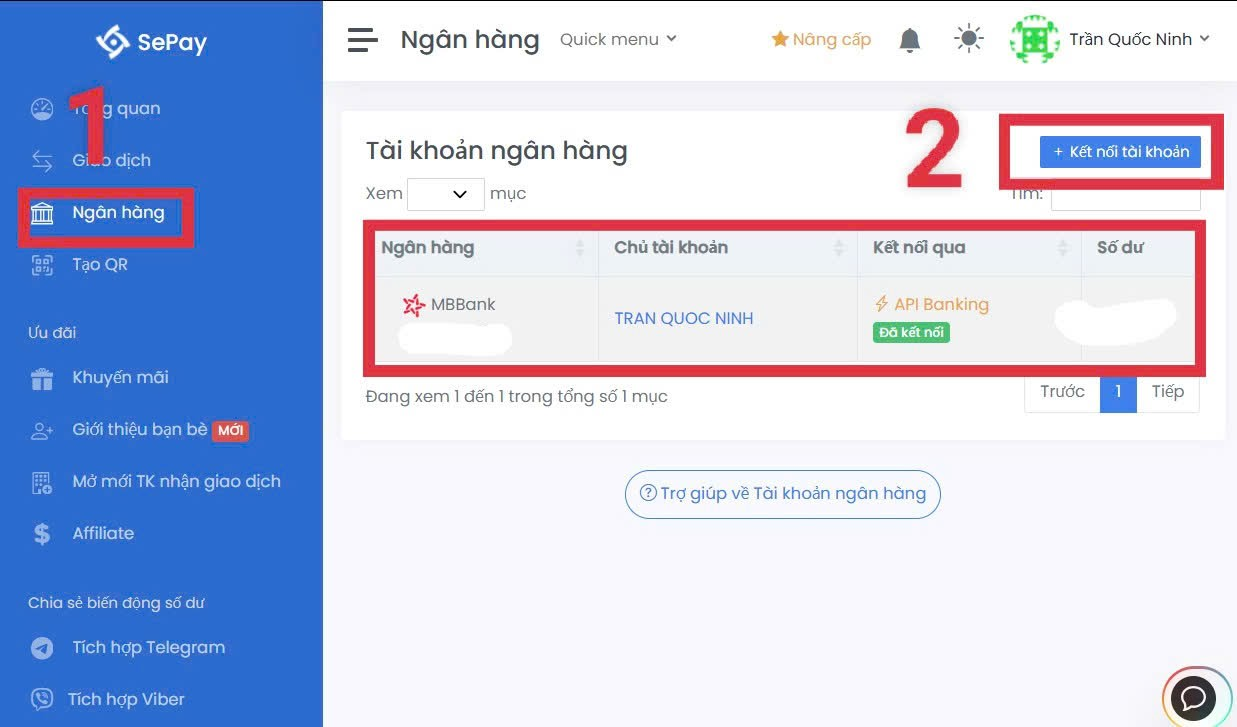
Nếu điều kiện cho phép, tôi sẽ tiếp tục nghiên cứu cũng như tìm hiểu chi tiết hơn về nghiệp vụ bán hàng cũng như cách hình thành trang web bán hàng, tìm hiểu những kiến thức mới góp phần cải tiến website bán hàng hoàn chỉnh. Mục tiêu là không ngừng cải tiến trang web bán hàng của mình, đưa vào hoạt động những chức năng mới và tối ưu hóa hiệu suất để mang lại trải nghiệm tốt nhất cho người sử dụng.

Tôi tin rằng với sự nỗ lực và kiên trì, cùng với việc không ngừng học hỏi và áp dụng những kiến thức mới, tôi sẽ đạt được mục tiêu xây dựng một website bán hàng chuyên nghiệp, hiệu quả và mang lại giá trị thực sự cho khách hàng.

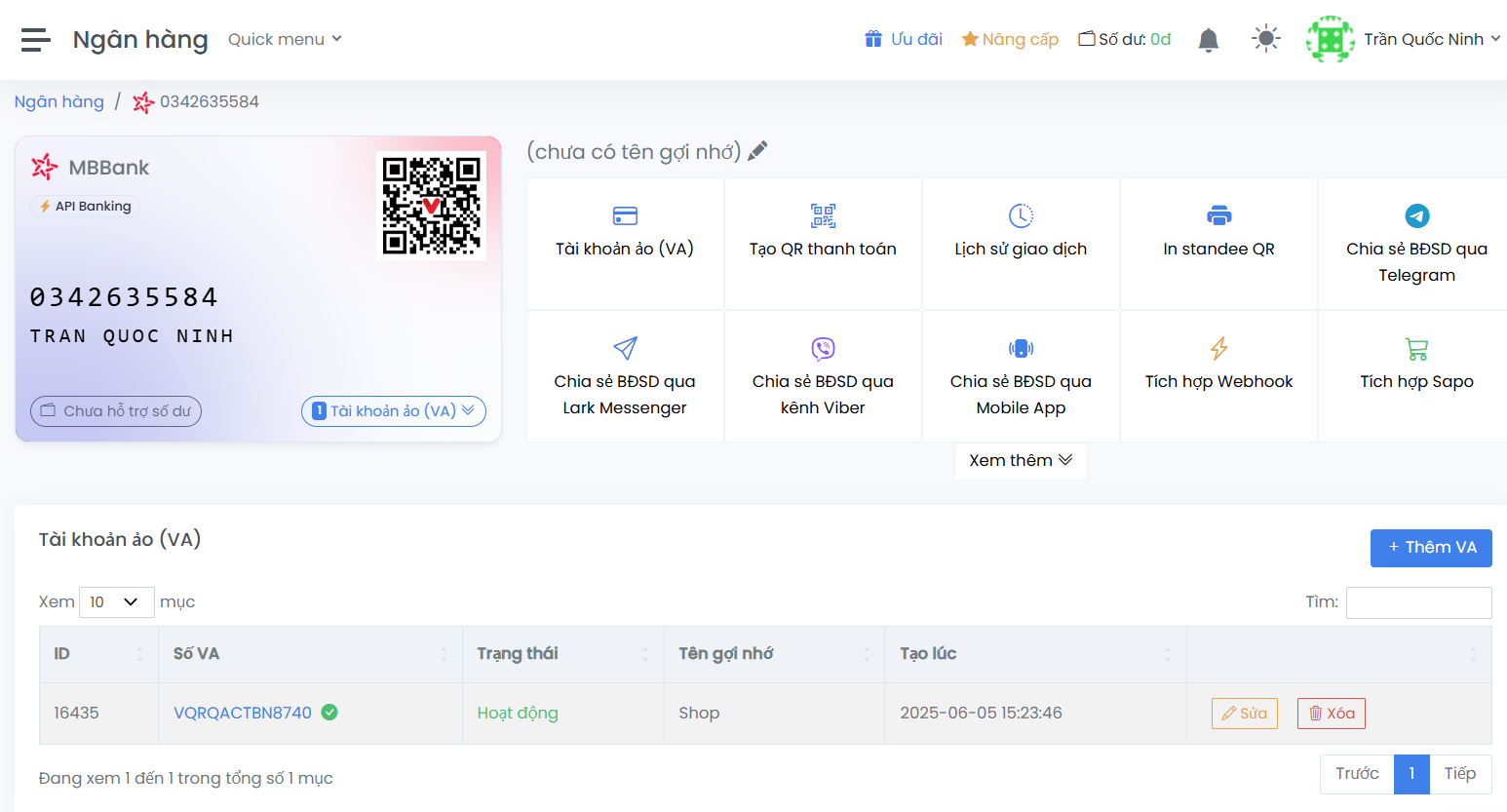
PHỤ LỤC

**1. Cấu hình Sepay**

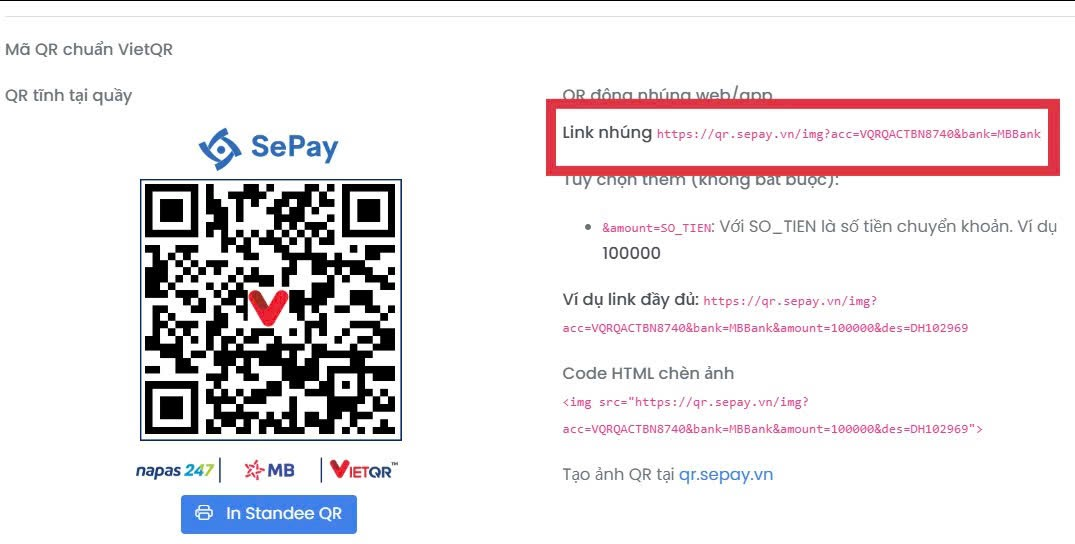
Để tích hợp thanh toán qua Sepay cho website, đầu tiên cần kết nối tài khoản ngân hàng với Sepay. Sepay cung cấp khả năng kết nối với nhiều ngân hàng phổ biến hiện nay như MBBank, BIDV, TPBank, VietinBank, ACB…



Trong trường hợp không muốn sử dụng tài khoản thật để tiếp nhận các giao dịch hay lo ngại nguy hiểm khi để lộ số tài khoản, sepay cho phép bạn sử dụng tài khoản ảo (VA) để thực hiện giao dịch.



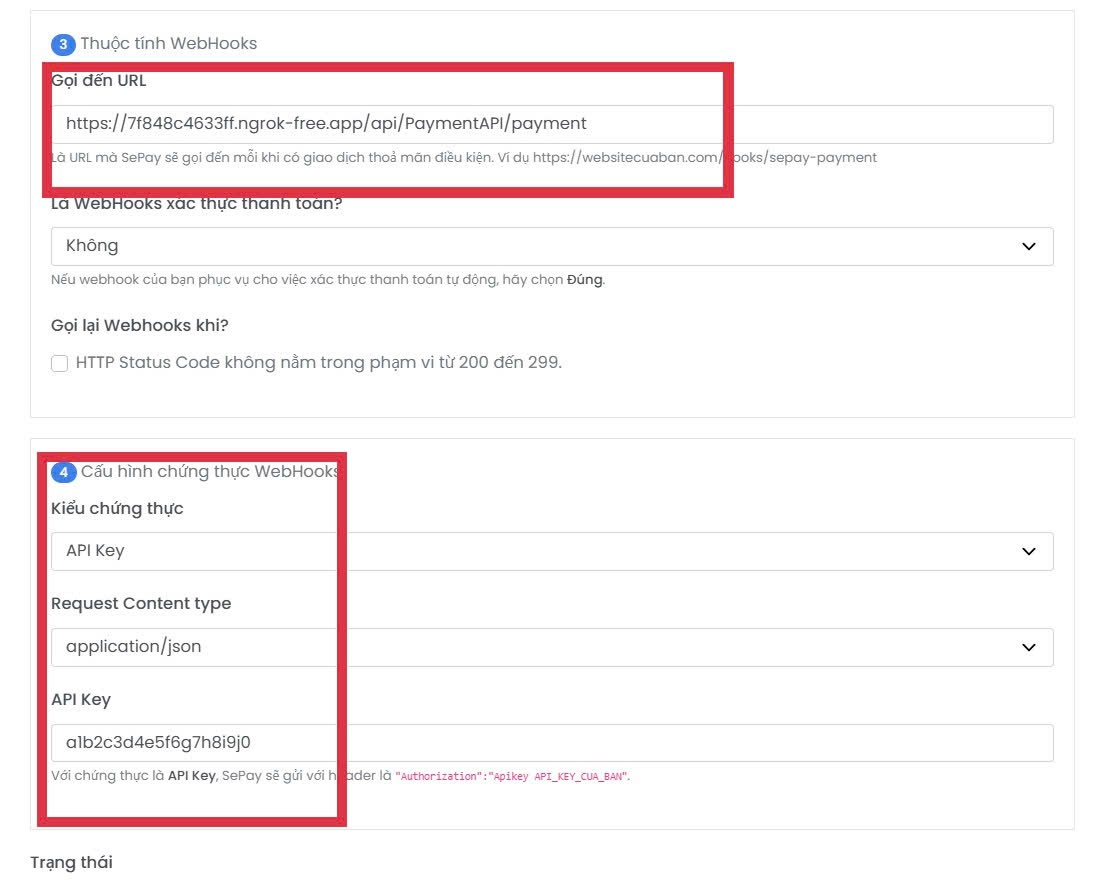
Chọn vào một tài khoản ảo, xuất hiện một số thông tin bao gồm hướng dẫn sử dụng cùng với một mã QR động, sử dụng QR này để đưa vào giao diện giao dịch của website.



Tiếp theo là tích hợp WebHooks, đây là phần rất quan trọng nên đòi hỏi phải thực hiện chính xác.



Tại giao diện thêm WebHooks, hãy điền và chọn các thông tin như đặt tên, điều kiện để bắn WebHooks, tài khoản ngân hàng… Đặc biệt, tại phần thuộc tính WebHooks, điền chính xác URL mà Sepay sẽ gọi đến mỗi khi có giao dịch thỏa mãn điều kiện (đường dẫn đến API xử lý thanh toán của website). Trong phần chứng thực, chọn kiểu chứng thực API Key, Oauth 2.0 hoặc không cần chứng thực.



DANH MỤC TÀI LIỆU THAM KHẢO

[1] Bùi Văn Quang và Lê Tuấn Khải (2020), *Tìm hiểu framwork VueJS, Java Spring Boot và xây dựng website xem phim online*, TP. Hồ Chí Minh,

https://www.slideshare.net/man2017/tm-hiu-framework-vuejs-java-spring-v-xy-dng-website-xem-phim-onlinepdf

[2] Đoàn Phước Miền và Phạm Thị Trúc Mai (2014), *Tài liệu giảng dạy môn Thiết kế và Lập trình web*, trường Đại học Trà Vinh

[3] Giao hàng nhanh (2023), *Bán hàng online là gì? Xu hướng mới trong kinh doanh online*, https://ghn.vn/blogs/tip-ban-hang/ban-hang-online-la-gi-xu-huong-moi-trong-kinh-doanh-online

[4] Microsoft (2024), *Create web APIs with ASP.NET Core*,

https://learn.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-9.0&WT.mc\_id=dotnet-35129-website

[5] Microsoft (2024), *Machine learning tasks in ML.NET*,

https://learn.microsoft.com/en-us/dotnet/machine-learning/resources/tasks

[6] Nguyễn Hồng Mai Thảo (2016), *Xây dựng website bán hàng với công nghệ ASP.NET MVC và Web API*, TP. Hồ Chí Minh, https://tailieuthamkhao.org/khoa-luan-xay-dung-website-ban-hang-su-dung-asp-net-mvc-va-web-api-phan-1-13367/?utm\_source=

[7] Nguyễn Hữu Tiến, Nguyễn Lương Toàn, Đặng Ngọc Liêm và Trương Quốc Bảo (2020), *Giới thiệu framework VueJS*, TP. Hồ Chí Minh,

https://www.studocu.com/vn/document/truong-dai-hoc-cong-nghe-thong-tin-dai-hoc-quoc-gia-thanh-pho-ho-chi-minh/cong-nghe-web-va-ung-dung/bao-cao-vue-js-bao-cao-tim-hieu-vue-js/70164451

[8] Nguyễn Xuân Phúc, Hồ Văn Quang và Nguyễn Trọng Quý (2017), *Lập trình web bán hàng công nghệ*, TP. Hồ Chí Minh, https://www.slideshare.net/LunVnTt/bo-co-n-chuyn-ngnh-lp-trnh-web-net-mvc-5-bn-hng-cng-nghdocx?utm\_source=

[9] Okta (n.d), *Introduction to JSON Web Tokens*, https://jwt.io/introduction

[10] Phan Thanh Nguyên (2021), *Xây dựng ứng dụng chat với VueJS và Firebase*, Đà Nẵng, https://123docz.net/document/12143823-do-an-tot-nghiep-xay-dung-ung-dung-chat-voi-vuejs-va-firebase.htm

[11] Phương Nam ViNa (n.d), *Tìm hiểu về các chức năng cần thiết của website bán hàng*,  
https://websitechuyennghiep.vn/4-chuc-nang-can-thiet-cua-website-ban-hang.html

[12] Thanh Hung (2022), *Giới thiệu về hệ thống gợi ý (Recommender systems hoặc Recommendation systems)*,

https://viblo.asia/p/gioi-thieu-ve-he-thong-goi-y-recommender-systems-hoac-recommendation-systems-1Je5EALG5nL

[13] Thư Viện Pháp Luật (n.d), *Website thương mại điện tử bán hàng là gì? Người bán hàng trên website thương mại điện tử phải cung cấp thông tin nào?*,  
https://thuvienphapluat.vn/phap-luat/website-thuong-mai-dien-tu-ban-hang-la-gi-nguoi-ban-hang-tren-website-thuong-mai-dien-tu-phai-cung--455672-182030.html

[14] Tiep Vu (2021), *Hệ thống gợi ý*,

https://machinelearningcoban.com/tabml\_book/ch\_recommendation\_system/introduce.html

[15] TopDev (n.d), *.NET Core là gì? Lịch sử hình thành và phát triển của .NET Core*,  
https://topdev.vn/blog/dotnet-core-la-gi/

[16] TopDev (n.d), *Giới thiệu nền tảng máy học ML.NET của Microsoft*,  
https://topdev.vn/blog/gioi-thieu-nen-tang-may-hoc-ml-net-cua-microsoft-bai-1/

[17] TopDev (n.d), *JSON Web Token (JWT) là gì ?*,

https://topdev.vn/blog/jwt-la-gi/

[18] TopDev (n.d), *RESTful API là gì? Cách thiết kế RESTful API*,

https://topdev.vn/blog/restful-api-la-gi/

[19] Tran Vuong Minh (2016), *Tìm hiểu về json web token(JWT)*,

https://viblo.asia/p/tim-hieu-ve-json-web-token-jwt-7rVRqp73v4bP

[20] Trituenhantao.github.io (2020), *Hệ thống gợi ý sản phẩm*,

https://trituenhantao.github.io/2020/09/05/he-thong-goi-y-san-pham/

[21] Vue.js (n.d), *Giới thiệu*, https://vi.vuejs.org/v2/guide/

[22] Vue.js (n.d), *Introduction*, https://vuejs.org/guide/introduction.html