

LỜI MỞ ĐẦU

Trong cuộc sống hiện đại, việc mua sắm online đã trở nên quen thuộc với hầu hết mọi người. Từ việc đặt món ăn, mua quần áo cho đến các thiết bị điện tử, người dùng chỉ cần vài thao tác đơn giản trên điện thoại hoặc máy tính là có thể hoàn tất đơn hàng. Điều này cho thấy thương mại điện tử đang ngày càng phát triển và trở thành một phần quan trọng trong hoạt động kinh doanh hiện nay.

Xuất phát từ thực tế đó, em chọn thực hiện đề tài xây dựng một hệ thống web bán hàng đa nền tảng. Hệ thống được xây dựng với mục tiêu không chỉ phục vụ cho việc học tập, mà còn mang tính thực tiễn cao có thể áp dụng cho các cửa hàng vừa và nhỏ trong việc quản lý sản phẩm, đơn hàng, khách hàng cũng như hỗ trợ thanh toán, đánh giá sản phẩm.

Đề tài sử dụng các công nghệ hiện đại như NestJS cho backend, MongoDB để lưu trữ dữ liệu, tích hợp Cloudinary để quản lý hình ảnh và sử dụng JWT để xác thực người dùng. Hệ thống hỗ trợ phân quyền rõ ràng (admin, khách hàng, nhà cung cấp), cho phép đăng nhập, đăng ký, thêm sản phẩm, đặt hàng, đánh giá sản phẩm và nhiều chức năng khác.

Thông qua đề tài này, em mong muốn có cơ hội áp dụng kiến thức lập trình web vào một sản phẩm cụ thể, đồng thời rèn luyện kỹ năng xây dựng hệ thống, làm việc với cơ sở dữ liệu, bảo mật và tổ chức mã nguồn theo hướng chuyên nghiệp hơn.

LỜI CẢM ƠN

Trước hết, em xin bày tỏ lòng biết ơn sâu sắc đến Thầy Nguyễn Khắc Quốc là người đã tận tình hướng dẫn, truyền đạt kiến thức chuyên môn cũng như định hướng thực hiện đề tài trong suốt thời gian qua. Sự tận tâm, trách nhiệm và những góp ý quý báu từ Thầy là yếu tố quan trọng giúp em hoàn thiện khóa luận này.

Em cũng xin chân thành cảm ơn các Thầy Cô trong Khoa Công nghệ thông tin đã giảng dạy và tạo điều kiện thuận lợi để em tích lũy được nền tảng kiến thức vững chắc trong suốt quá trình học tập.

Bên cạnh đó, em xin gửi lời cảm ơn chân thành đến gia đình, bạn bè và những người thân yêu, những người luôn đồng hành, động viên và tiếp thêm tinh thần cho em trong suốt hành trình học tập và thực hiện khóa luận.

Mặc dù đã nỗ lực hoàn thành tốt nhất trong khả năng của mình, nhưng chắc chắn khóa luận vẫn không tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý từ Thầy Cô để em có thể rút kinh nghiệm và tiếp tục hoàn thiện bản thân trong tương lai.

NHẬN XÉT

(Của giảng viên hướng dẫn trong đồ án, khoá luận của sinh viên)

Giảng viên hướng dẫn

(ký và ghi rõ họ tên)

BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP
(Của giảng viên hướng dẫn)

Họ và tên sinh viên: MSSV:

Ngành: Khóa:

Tên đề tài:

Họ và tên Giáo viên hướng dẫn:

Chức danh: Học vị:

NHẬN XÉT

1. Nội dung đề tài:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

2. Ưu điểm:

.....
.....
.....
.....
.....
.....
.....

3. Khuyết điểm:

.....
.....
.....
.....
.....
.....
.....

4. Điểm mới đề tài:

.....
.....
.....
.....
.....
.....

5. Giá trị thực trên đề tài:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

7. Đề nghị sửa chữa bổ sung:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

8. Đánh giá:

.....
.....
.....
.....
.....

Trà Vinh, ngày tháng năm 20...

Giảng viên hướng dẫn
(Ký & ghi rõ họ tên)

NHẬN XÉT

Giảng viên chấm

(ký và ghi rõ họ tên)

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)

MỤC LỤC

MỤC LỤC	IX
DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH	XII
KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT	XV
CHƯƠNG 1: TỔNG QUAN.....	1
1.1 Lý do chọn đề tài.....	1
1.2 Mục tiêu đề tài	1
1.3 Đối tượng nghiên cứu	2
1.4 Phạm vi nghiên cứu	2
1.5 Phương pháp nghiên cứu	3
CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT	4
2.1 Website đa nền tảng	4
2.1.1 Khái niệm	4
2.1.2 Nguyên tắc hoạt động	5
2.1.3 Ưu và nhược điểm.....	5
2.2 API và RESTful API	6
2.2.1 Khái niệm	6
2.2.2 Lợi ích của RESTful API	6
2.2.3 Cách hoạt động	7
2.3 Tổng quan về Node.js	8
2.3.1 Giới thiệu Node.js.....	8
2.3.2 Cách hoạt động của Node.js	8
2.3.3 Ưu điểm của Node.js	9
2.3.4 Nhược điểm của Node.js	9
2.3.5 Ứng dụng của Node.js	10
2.4 Tổng quan về NestJs.....	11
2.4.1 Giới thiệu NestJs	11
2.4.2 Cấu trúc của NestJs	11
2.4.3 Ưu điểm của NestJs	12
2.4.4 Nhược điểm của NestJs.....	13
2.4.5 Cách cài đặt và tạo dự án NestJs	13
2.5 Tổng quan về NextJs	14
2.5.1 Giới thiệu NextJS	14
2.5.2 Các tính năng chính của NextJs.....	14
2.5.3 Ưu điểm của NextJs.....	17
2.5.4 Nhược điểm của NextJs	18
2.5.5 Sử dụng NextJs cơ bản	18
2.6 Tổng quan về Tailwind CSS	20
2.6.1 Giới thiệu Tailwind CSS	20
2.6.2 Ưu điểm của Tailwind CSS	20

2.6.3	Nhược điểm của Tailwind CSS.....	20
2.6.4	Cài đặt và sử dụng Tailwind CSS.....	21
2.7	Tổng quan về MongoDB	22
2.7.1	Sơ lược về NoSQL	22
2.7.2	Giới thiệu MongoDB.....	22
2.7.3	Một số khái niệm trong MongoDB	23
2.7.4	Ưu điểm của MongoDB	23
2.7.5	Nhược điểm của MongoDB	24
CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU.....		25
3.1	Phân tích và đặc tả yêu cầu hệ thống.....	25
3.1.1	Yêu cầu chức năng	25
3.1.2	Yêu cầu phi chức năng	26
3.2	Thiết kế hệ thống	26
3.2.1	Thiết kế cơ sở dữ liệu	26
3.2.2	Sơ đồ Use Case	31
CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU.....		38
4.1	Giao diện website trên desktop	38
4.1.1	Trang chủ	38
4.1.2	Trang xem theo danh mục	39
4.1.3	Trang tìm kiếm sản phẩm	39
4.1.4	Trang chi tiết sản phẩm	40
4.1.5	Trang giỏ hàng.....	40
4.1.6	Form thanh toán	41
4.1.7	Trang đơn hàng.....	41
4.1.8	Trang thông tin cá nhân	42
4.1.9	Trang đổi mật khẩu.....	43
4.1.10	Thanh toán trực tuyến.....	43
4.1.11	Trang đăng nhập	44
4.1.12	Trang đăng ký	44
4.1.13	Trang quên mật khẩu	45
4.2	Giao diện trên điện thoại	46
4.2.1	Trang chủ	46
4.2.2	Trang xem sản phẩm theo danh mục	47
4.2.3	Trang tìm kiếm sản phẩm	48
4.2.4	Trang chi tiết sản phẩm	49
4.2.5	Trang giỏ hàng.....	50
4.2.6	Form thanh toán	51
4.2.7	Thanh toán trực tuyến.....	52
4.2.8	Thông tin tài khoản.....	53
4.2.9	Thông tin đơn hàng.....	54
4.2.10	Trang chi tiết đơn hàng.....	55

4.2.11	Trang đổi mật khẩu.....	56
4.2.12	Trang đăng nhập	57
4.2.13	Trang đăng ký	58
4.2.14	Trang quên mật khẩu	59
4.3	Giao diện quản trị.....	60
4.3.1	Trang tổng quan.....	60
4.3.2	Trang quản lý sản phẩm	61
4.3.3	Trang Quản lý người dùng	62
4.3.4	Trang Quản lý danh mục	63
4.3.5	Trang Quản lý đơn hàng	64
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN		65
5.1	Kết luận	65
5.2	Hướng phát triển	65
DANH MỤC TÀI LIỆU THAM KHẢO		66
PHỤ LỤC		67

DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH

Sơ đồ 3.1 Use Case tổng quát.....	31
Sơ đồ 3.2 Use Case đăng ký	31
Sơ đồ 3.3 Use Case đăng nhập	32
Sơ đồ 3.4 Use Case xem sản phẩm.....	32
Sơ đồ 3.5 Use Case quản lý thông tin cá nhân	33
Sơ đồ 3.6 Use Case thêm sản phẩm vào giỏ hàng.....	33
Sơ đồ 3.7 UseCase đặt hàng	34
Sơ đồ 3.8 Use Case lịch sử mua hàng	34
Sơ đồ 3.9 Use Case quản lý sản phẩm.....	35
Sơ đồ 3.10 Use Case quản lý đơn hàng.....	35
Sơ đồ 3.11 Use Case quản lý danh mục	36
Sơ đồ 3.12 Use Case quản lý người dùng	36
Sơ đồ 3.13 UseCase xem báo cáo thống kê.....	37
 Hình 2.1 Cách hoạt động của REST API	7
Hình 2.2 Cách hoạt động của Node.js	8
Hình 2.2.3. Static Generation trong NextJs	16
Hình 2.2.4. Ví dụ sử dụng CSS Modules trong NextJs.....	16
Hình 2.2.5. Ví dụ sử dụng CSS-in-JS Libraries trong NextJS	17
Hình 2.2.6. Giới thiệu Tailwind CSS	20
Hình 2.2.7. Nội dung tệp postcss.config.js.....	21
Hình 2.2.8. Nội dung tệp styles.css	21
Hình 2.2.9. Thêm tập lệnh vào pagekage.json	21
Hình 2.10 So sánh NoSql với SQL.....	22
Hình 2.11 Ví dụ về Collection trong MongoDB	23
Hình 2.12 Ví dụ minh họa cho Document trong MongoDB	23

Xây dựng website bán hàng đa nền tảng

Hình 4.1 Trang chủ trên desktop	38
Hình 4.2 Trang xem theo danh mục trên desktop	39
Hình 4.3 Trang tìm kiếm sản phẩm trên desktop	39
Hình 4.4 Trang chi tiết sản phẩm trên desktop.....	40
Hình 4.5 Trang giỏ hàng trên desktop	40
Hình 4.6 Trang đơn hàng trên desktop	41
Hình 4.7 Trang chi tiết đơn hàng trên desktop	42
Hình 4.8 Trang Quản lý thông tin cá nhân trên desktop	42
Hình 4.9 Form thanh toán.....	41
Hình 4.10 Trang đổi mật khẩu trên desktop	43
Hình 4.11 Trang thanh toán trực tuyến	43
Hình 4.12 Trang đăng nhập trên desktop	44
Hình 4.13 Trang đăng ký trên desktop	44
Hình 4.14 Trang lấy lại mật khẩu	45
Hình 4.15 Trang chủ trên điện thoại.....	46
Hình 4.16 Trang xem theo danh mục trên điện thoại.....	47
Hình 4.17 Trang tìm kiếm sản phẩm trên điện thoại.....	48
Hình 4.18 Trang chi tiết sản phẩm trên điện thoại	49
Hình 4.19 Trang giỏ hàng trên điện thoại.....	50
Hình 4.20 Form thanh toán trên điện thoại.....	51
Hình 4.21 Thanh toán trực tuyến trên điện thoại.....	52
Hình 4.22 Trang thông tin tài khoản trên điện thoại	53
Hình 4.23 Trang danh sách đơn hàng trên điện thoại.....	54
Hình 4.24 Trang chi tiết đơn hàng trên điện thoại	55
Hình 4.25 Trang thay đổi mật khẩu trên điện thoại.....	56

Xây dựng website bán hàng đa nền tảng

Hình 4.26 Trang đăng nhập trên điện thoại	57
Hình 4.27 Trang đăng ký trên điện thoại.....	58
Hình 4.28 Trang tổng quan.....	60
Hình 4.29 Trang quản lý sản phẩm	61
Hình 4.30 Trang Quản lý người dùng	62
Hình 4.31 Trang Quản lý danh mục	63
Hình 4.32 Trang Quản lý đơn hàng.....	64

KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT

JWT	JSON Web Token
API	Application Programming Interface (Giao diện lập trình ứng dụng)
SMTP	Simple Mail Transfer Protocol (Giao thức truyền thư đơn giản)
CSS	Cascading Style Sheets (Tập tin định kiểu theo tầng)
REST	Representational State Transfer
I/O	Input/Output (Nhập/Xuất)
IoT	Internet of Things (Internet vạn vật)
CLI	Command Line Interface (Giao diện dòng lệnh)
SSR	Server-Side Rendering (Render phía máy chủ)
SSG	Static Site Generation (Tạo trang tĩnh sẵn)
JS	JavaScript
DB	Database (Cơ sở dữ liệu)
RDBMS	Relational Database Management System (Cơ sở dữ liệu quan hệ)

CHƯƠNG 1:TỔNG QUAN

1.1 Lý do chọn đề tài

Trong bối cảnh công nghệ thông tin ngày càng phát triển mạnh mẽ, thương mại điện tử đã và đang trở thành một xu thế tất yếu. Việc xây dựng các nền tảng bán hàng trực tuyến không chỉ giúp doanh nghiệp mở rộng thị trường, tiếp cận khách hàng hiệu quả mà còn tối ưu hóa quy trình quản lý và kinh doanh. Từ thực tiễn đó, em lựa chọn thực hiện đề tài “Xây dựng hệ thống website bán hàng đa nền tảng” với mong muốn tạo ra một giải pháp ứng dụng web hiện đại, dễ sử dụng, có thể áp dụng thực tế cho các cửa hàng vừa và nhỏ, đồng thời giúp em củng cố và vận dụng kiến thức đã học vào một sản phẩm cụ thể.

1.2 Mục tiêu đề tài

Đề tài hướng đến việc xây dựng một hệ thống website bán hàng đa nền tảng hiện đại, có thể hoạt động ổn định trên nhiều thiết bị (máy tính, điện thoại, máy tính bảng), nhằm đáp ứng nhu cầu mua sắm trực tuyến ngày càng phổ biến hiện nay. Việc thiết kế hệ thống không chỉ giúp người dùng dễ dàng tìm kiếm và đặt mua sản phẩm, mà còn hỗ trợ người quản trị quản lý hiệu quả các hoạt động kinh doanh. Cụ thể, đề tài đặt ra các mục tiêu như sau:

- Xây dựng giao diện website thân thiện, dễ sử dụng cho cả người dùng và quản trị viên, đảm bảo hiển thị tốt trên nhiều thiết bị (responsive design).
- Phát triển hệ thống backend sử dụng NestJS, thiết kế kiến trúc mô-đun rõ ràng, dễ mở rộng và bảo trì.
- Thiết kế cơ sở dữ liệu sử dụng MongoDB, đảm bảo lưu trữ hiệu quả thông tin người dùng, sản phẩm, danh mục, đơn hàng...
- Tích hợp chức năng xác thực người dùng bằng JWT, hỗ trợ phân quyền rõ ràng giữa admin, khách hàng và nhà cung cấp.
- Xây dựng các chức năng bán hàng cốt lõi như: xem và tìm kiếm sản phẩm, thêm vào giỏ hàng, đặt hàng, đánh giá sau mua.
- Tích hợp thanh toán trực tuyến thực tế qua PayOS, cho phép người dùng quét mã và thanh toán nhanh chóng.

- Thêm chức năng gửi email tự động khi đặt hàng thành công hoặc khi cập nhật trạng thái đơn hàng.

- Thiết kế giao diện quản trị với chức năng theo dõi đơn hàng, quản lý sản phẩm và thống kê doanh thu thông qua biểu đồ.

1.3 Đối tượng nghiên cứu

Đề tài là quy trình thiết kế và xây dựng hệ thống website bán hàng hiện đại, đa nền tảng. Trong đó, đề tài tập trung nghiên cứu các thành phần chính như:

- Giao diện người dùng với khả năng tương thích đa thiết bị (máy tính, điện thoại)

- Hệ thống xử lý nghiệp vụ bán hàng, xác thực người dùng, phân quyền và quản lý sản phẩm, đơn hàng.

1.4 Phạm vi nghiên cứu

Về chức năng:

- Xây dựng hệ thống website bán hàng với các tính năng cốt lõi: đăng ký, đăng nhập, phân quyền, quản lý sản phẩm và danh mục, giỏ hàng, đơn hàng, đánh giá sản phẩm.

- Hệ thống phân quyền gồm: admin, khách hàng và nhà cung cấp.

- Hỗ trợ thanh toán trực tuyến thông qua PayOS và gửi email thông báo khi đặt hàng hoặc cập nhật đơn hàng.

- Giao diện quản trị có chức năng thống kê doanh thu bằng biểu đồ, quản lý đơn hàng và sản phẩm.

Về công nghệ:

- Frontend: sử dụng Next.js để xây dựng giao diện người dùng, tương thích đa nền tảng.

- Backend: sử dụng NestJS để xây dựng RESTful API với bảo mật JWT, tổ chức theo kiến trúc mô-đun.

- Cơ sở dữ liệu: sử dụng MongoDB, kết hợp thư viện Mongoose.

- Tích hợp Cloudinary để lưu trữ ảnh và Nodemailer (SMTP Gmail) để gửi email.

1.5 Phương pháp nghiên cứu

Phương pháp khảo sát và thu thập yêu cầu:

- Tìm hiểu nhu cầu và hành vi người dùng trên các nền tảng bán hàng hiện tại nhằm xác định những tính năng cốt lõi và yêu cầu của người dùng.

- Xác định đặc điểm của hai nhóm người dùng chính: Admin, Khách hàng từ đó xây dựng các luồng chức năng và phân quyền phù hợp.

Phương pháp phân tích và thiết kế hệ thống:

- Phân tích chức năng: Xây dựng sơ đồ use case để xác định đầy đủ các chức năng của hệ thống như: đăng nhập, quản lý sản phẩm, đặt hàng,...

- Phân tích yêu cầu phi chức năng: Bao gồm khả năng mở rộng, hiệu năng, bảo mật (JWT, phân quyền), khả năng tương thích đa thiết bị (responsive) và tính thân thiện với người dùng.

Phương pháp thiết kế:

- Thiết kế giao diện người dùng (UI/UX): Áp dụng nguyên tắc responsive design, bố cục rõ ràng, màu sắc dễ chịu.

- Thiết kế hệ thống backend: Xây dựng kiến trúc mô-đun sử dụng NestJS, tách biệt rõ các lớp controller, service, module để dễ bảo trì.

- Thiết kế cơ sở dữ liệu: Sử dụng MongoDB kết hợp Mongoose, đảm bảo khả năng mở rộng và linh hoạt trong lưu trữ tài liệu phi cấu trúc.

Phương pháp triển khai: Sử dụng NextJs cùng với Tailwind CSS để xây dựng giao diện website. Xây dựng API Restful với NestJs. Tích hợp các dịch vụ khác: PayOS để thanh toán qua hình thức quét mã QR, Cloudinary để lưu trữ hình ảnh, Nodemailer + SMTP Gmail để gửi email thông báo tự động.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Website đa nền tảng

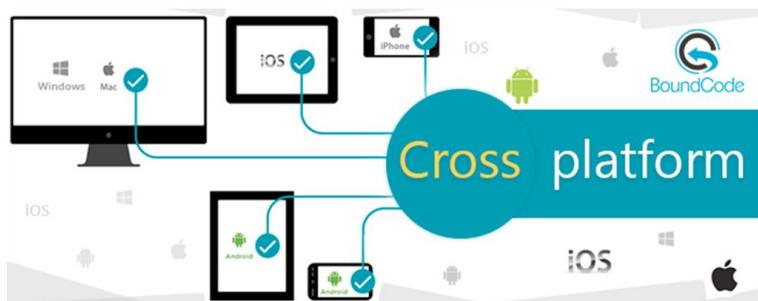
2.1.1 Khái niệm

Website đa nền tảng (Cross-platform Website) là một hệ thống web được thiết kế để hoạt động hiệu quả trên nhiều loại thiết bị và hệ điều hành khác nhau như máy tính để bàn (Windows, macOS), điện thoại thông minh (Android, iOS), máy tính bảng hệ điều hành một trang duy nhất mà không cần cài đặt và các trình duyệt hiện đại.

Không giống như website truyền thống chỉ tối ưu cho một loại màn hình hoặc một hệ điều hành cụ thể, website đa nền tảng đảm bảo:

- Giao diện người dùng hiển thị tốt trên mọi độ phân giải màn hình.
- Chức năng hoạt động ổn định trên nhiều trình duyệt và thiết bị.
- Dữ liệu được xử lý đồng bộ, nhất quán giữa các nền tảng.

Một website đa nền tảng không chỉ cần có giao diện thích ứng (responsive design), mà còn phải có kiến trúc rõ ràng, chia tách giữa giao diện người dùng (frontend) và xử lý dữ liệu (backend), giúp dễ dàng tích hợp với các hệ thống khác như điện thoại app hoặc ứng dụng desktop.



Hình 2.1. Ảnh minh họa Cross về platform

2.1.2 Nguyên tắc hoạt động

Một hệ thống đa nền tảng thường hoạt động theo các bước như sau:

Lớp frontend (giao diện người dùng) được thiết kế sao cho có thể hiển thị tốt trên nhiều thiết bị và hệ điều hành (Windows, macOS, Android, iOS...). Điều này đạt được thông qua: Thiết kế giao diện responsive, sử dụng thư viện framework như React, Next.js.

Lớp backend (xử lý nghiệp vụ) được xây dựng như một hệ thống độc lập, phục vụ qua các API chuẩn như REST hoặc GraphQL. Backend không phụ thuộc vào nền tảng của client, nhờ đó có thể dùng lại cho: Website, ứng dụng điện thoại (React Native, Flutter), ứng dụng desktop.

API trung gian đóng vai trò là cầu nối giữa frontend và backend, truyền tải dữ liệu ở định dạng phổ quát như JSON. Điều này giúp frontend ở bất kỳ nền tảng nào cũng có thể hiểu và xử lý được phản hồi.

Dữ liệu và dịch vụ được chia sẻ thống nhất, nghĩa là mọi thiết bị đều làm việc với một cơ sở dữ liệu và hệ thống xử lý duy nhất. Điều này đảm bảo tính đồng bộ, nhất quán và khả năng mở rộng của toàn bộ hệ thống.

2.1.3 Ưu và nhược điểm

Ưu điểm:

- Phạm vi tiếp cận thị trường rộng lớn
- Cập nhật đơn giản
- Giảm tốc độ phát triển

Nhược điểm

- Gặp phải các vấn đề về hệ xuất
- Tăng độ phức tạp của mã nguồn
- Quyền truy cập hạn chế vào các tính năng dành riêng cho thiết bị

2.2 API và RESTful API

2.2.1 Khái niệm

Giao diện lập trình ứng dụng (API) xác định các quy tắc phải tuân theo để giao tiếp với các hệ thống phần mềm khác. Các nhà phát triển đưa ra hoặc tạo API để các ứng dụng khác có thể giao tiếp với ứng dụng của họ theo cách lập trình. Ví dụ: ứng dụng bảng chấm công đưa ra một API yêu cầu tên đầy đủ của nhân viên và phạm vi ngày. Khi nhận được thông tin này, bảng chấm công của nhân viên sẽ được xử lý nội bộ và trả về số giờ làm việc trong phạm vi ngày đó. Có thể coi API web như một cổng giữa client và tài nguyên trên web [1].

Chuyển trạng thái đại diện (REST) là một kiến trúc phần mềm quy định các điều kiện về cách thức hoạt động của API. REST ban đầu được tạo ra như một hướng dẫn để quản lý giao tiếp trên một mạng phức tạp như Internet. Có thể sử dụng kiến trúc dựa trên REST để hỗ trợ giao tiếp hiệu suất cao và đáng tin cậy trên quy mô lớn. Có thể dễ dàng triển khai và sửa đổi REST, mang lại khả năng hiển thị và tính di động đa nền tảng cho bất kỳ hệ thống API nào.

Các nhà phát triển API có thể thiết kế các API bằng cách sử dụng nhiều kiến trúc khác nhau. Các API tuân theo kiểu kiến trúc REST được gọi là API REST. Các dịch vụ web triển khai kiến trúc REST được gọi là dịch vụ web RESTful. Thuật ngữ RESTful API thường là chỉ các API web RESTful. Tuy nhiên, có thể sử dụng các thuật ngữ API REST và RESTful API thay thế cho nhau.

2.2.2 Lợi ích của RESTful API

RESTful API có những lợi ích sau:

Khả năng thay đổi quy mô: Các hệ thống triển khai API REST có thể thay đổi quy mô một cách hiệu quả vì REST tối ưu hóa các tương tác giữa client và máy chủ. Tình trạng phi trạng thái loại bỏ tải của máy chủ vì máy chủ không phải giữ lại thông tin yêu cầu của client trong quá khứ. Việc lưu bộ nhớ đệm được quản lý tốt sẽ loại bỏ một phần hoặc hoàn toàn một số tương tác giữa client và máy chủ. Tất cả các tính năng này hỗ trợ khả năng thay đổi quy mô mà không gây ra tắc nghẽn giao tiếp làm giảm hiệu suất.

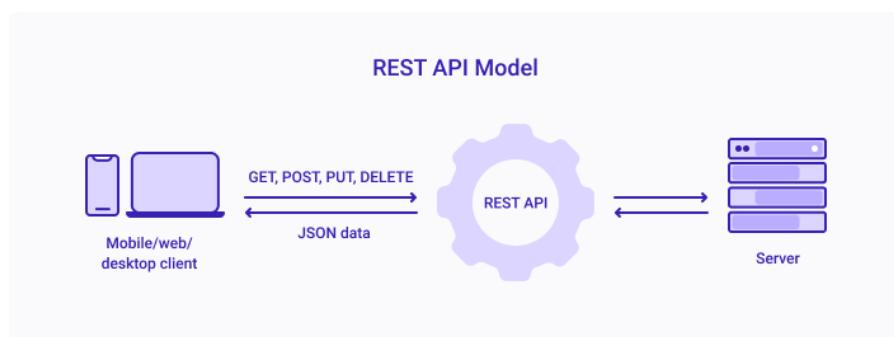
Sự linh hoạt: Các dịch vụ web RESTful hỗ trợ phân tách hoàn toàn giữa client và máy chủ. Các dịch vụ này đơn giản hóa và tách riêng các thành phần máy chủ khác nhau để mỗi phần có thể phát triển độc lập. Các thay đổi ở nền tảng hoặc công nghệ tại ứng dụng máy chủ không ảnh hưởng đến ứng dụng client. Khả năng phân lớp các chức năng ứng dụng làm tăng tính linh hoạt hơn nữa. Ví dụ: các nhà phát triển có thể thực hiện các thay đổi đối với lớp cơ sở dữ liệu mà không cần viết lại logic ứng dụng.

Sự độc lập: Các API REST không phụ thuộc vào công nghệ được sử dụng. Có thể viết cả ứng dụng client và máy chủ bằng nhiều ngôn ngữ lập trình khác nhau mà không ảnh hưởng đến thiết kế API. Đồng thời cũng có thể thay đổi công nghệ cơ sở ở hai phía mà không ảnh hưởng đến giao tiếp.

2.2.3 Cách hoạt động

Chức năng cơ bản của RESTful API cũng giống như việc duyệt Internet. Client liên hệ với máy chủ bằng cách sử dụng API khi yêu cầu tài nguyên. Các nhà phát triển API giải thích cách client nên sử dụng API REST trong tài liệu về API ứng dụng máy chủ. Đây là các bước chung cho bất kỳ lệnh gọi API REST nào:

- Client gửi một yêu cầu đến máy chủ. Client làm theo tài liệu API để định dạng yêu cầu theo cách mà máy chủ hiểu được.
- Máy chủ xác thực và xác nhận máy khách có quyền đưa ra yêu cầu đó.
- Máy chủ nhận yêu cầu và xử lý trong nội bộ.
- Máy chủ trả về một phản hồi đến client. Phản hồi chứa thông tin cho client biết liệu yêu cầu có thành công hay không. Phản hồi cũng bao gồm bất kỳ thông tin nào mà client yêu cầu.



Hình 2.1 Cách hoạt động của REST API

2.3 Tổng quan về Node.js

2.3.1 Giới thiệu Node.js

Node.js là một môi trường chạy (runtime environment) cho JavaScript với mã nguồn mở và đa nền tảng [2].

Node.js là mã nguồn mở: Mã nguồn của Node.js là công khai. Và nó được duy trì bởi những người đóng góp từ khắp nơi trên thế giới.

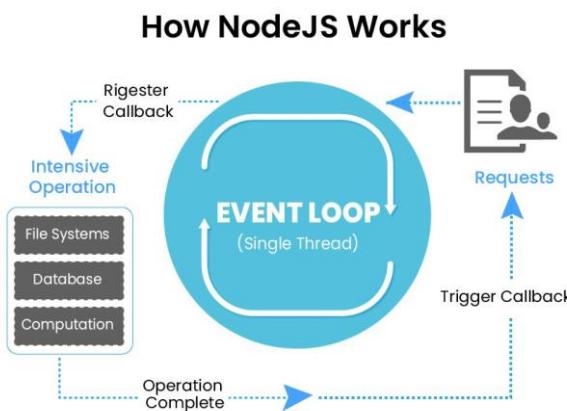
Node.js là môi trường thời gian chạy JavaScript: Nghĩa là khi viết mã JavaScript trong trình soạn thảo văn bản của mình, mã đó sẽ thực hiện tác vụ nào trừ khi thực thi nó. Và để chạy mã cần phải có một môi trường biên dịch.

Các trình duyệt như Chrome và FireFox có môi trường thời gian chạy. Đó là lý do tại sao các trình duyệt có thể chạy mã JavaScript. Trước khi Node.js được tạo ra thì JavaScript chỉ có thể được chạy trên trình duyệt. Và chỉ được sử dụng để xây dựng các ứng dụng front-end.

Node.js cung cấp một môi trường chạy JavaScript ngoài trình duyệt. Nó được xây dựng trên công cụ JavaScript Chrome V8. Điều này cho phép xây dựng các ứng dụng back-end bằng cách sử dụng cùng một ngôn ngữ lập trình JavaScript.

2.3.2 Cách hoạt động của Node.js

Node.js hoạt động dựa trên một số nguyên tắc cơ bản giúp nó hiệu quả trong việc xử lý các ứng dụng có nhiều hoạt động I/O mà không bị chặn, đồng thời giảm đáng kể sự phức tạp trong quản lý các luồng thực thi [3].



Hình 2.2 Cách hoạt động của Node.js

2.3.3 Ưu điểm của Node.js

Hiệu suất cao: Node.js được xây dựng trên động cơ JavaScript V8 của Google Chrome, cho phép biên dịch mã JavaScript thành mã máy nhanh chóng. Nhờ đó, thời gian thực thi của Node.js rất nhanh, làm tăng hiệu suất của các ứng dụng.

Hệ sinh thái phong phú: Với hơn 50,000 gói có sẵn trong Node Package Manager (NPM), các nhà phát triển có thể dễ dàng tìm và sử dụng các thư viện theo nhu cầu của họ mà không cần phải viết lại từ đầu, tiết kiệm đáng kể thời gian và công sức.

Xử lý bất đồng bộ và không chặn (Asynchronous and Non-blocking): Node.js hoạt động một cách bất đồng bộ và không chặn các hoạt động I/O, nghĩa là nó không cần chờ đợi API trả về dữ liệu trước khi tiếp tục xử lý yêu cầu tiếp theo. Điều này làm cho Node.js trở nên lý tưởng cho việc xây dựng các ứng dụng web thời gian thực và xử lý dữ liệu lớn.

Tính nhất quán trong mã nguồn: Node.js cho phép sử dụng cùng một ngôn ngữ lập trình (JavaScript) cho cả phía máy chủ và máy khách. Điều này không chỉ giúp giảm thiểu sự không đồng bộ giữa client và server mà còn làm cho việc bảo trì và quản lý mã nguồn trở nên dễ dàng hơn.

Khả năng mở rộng: Node.js hỗ trợ xây dựng các ứng dụng có khả năng mở rộng cao thông qua mô hình sự kiện và bất đồng bộ của mình. Điều này cho phép xử lý hàng ngàn kết nối đồng thời mà không làm giảm hiệu suất.

Ngôn ngữ quen thuộc: Vì Node.js là một khung làm việc JavaScript, nó trở thành lựa chọn lý tưởng cho những nhà phát triển đã quen thuộc với JavaScript. Điều này làm cho quá trình học tập và phát triển dự án với Node.js trở nên dễ dàng hơn nhiều.

2.3.4 Nhược điểm của Node.js

Đầu tiên phải kể đến chính là nó không có khả năng mở rộng. Do đó mà nó không thể tận dụng được lợi thế mô hình đa lõi ở các phần cứng cấp server trên thị trường hiện nay.

Khó thao tác được với cơ sở dữ liệu quan hệ.

Mỗi callback của nó sẽ đi kèm với nhiều callback lồng nhau khác.

Cần trang bị kiến thức tốt về JavaScript.

Không phù hợp với những tác vụ đòi hỏi nhiều CPU.

2.3.5 Ứng dụng của Node.js

Node.js được sử dụng rộng rãi trong nhiều loại ứng dụng web và server do khả năng xử lý bất đồng bộ, hiệu suất cao và hệ sinh thái phong phú của nó. Dưới đây là một số ứng dụng phổ biến của Node.js:

Ứng dụng Web Thời Gian Thực (Real-time Web Applications): Node.js là lựa chọn lý tưởng cho các ứng dụng web thời gian thực như trò chuyện trực tuyến và trò chơi trực tuyến do khả năng xử lý các sự kiện I/O một cách nhanh chóng và hiệu quả.

APIs Server-side: Node.js thường được sử dụng để xây dựng RESTful APIs do khả năng xử lý đồng thời lớn và tốc độ phản hồi nhanh, làm cơ sở cho các ứng dụng di động và web.

Streaming Data: Node.js hỗ trợ xử lý dữ liệu dạng stream, cho phép ứng dụng xử lý các tệp video, âm thanh hoặc các dữ liệu khác trong khi chúng vẫn đang được truyền, thay vì phải chờ cho đến khi toàn bộ tệp được tải về.

Ứng dụng Một Trang (Single Page Applications): Node.js phù hợp với việc phát triển các ứng dụng một trang như Gmail, Google Maps, hay Facebook, nơi mà nhiều tương tác xảy ra trên một trang duy nhất mà không cần tải lại trang.

Công cụ và Tự Động Hóa: Node.js cũng được sử dụng để phát triển các công cụ dòng lệnh và các script tự động hóa quy trình làm việc, nhờ vào các gói NPM hỗ trợ đa dạng và khả năng tích hợp dễ dàng với các công nghệ khác.

Microservices Architecture: Node.js là một lựa chọn phổ biến cho kiến trúc microservices, nơi các ứng dụng lớn được chia thành các dịch vụ nhỏ, độc lập và dễ quản lý hơn.

Ứng dụng IoT: Node.js thường được sử dụng trong các ứng dụng IoT, nơi cần xử lý một lượng lớn các kết nối đồng thời và các sự kiện từ các thiết bị IoT.

Dashboard và Monitoring: Node.js được sử dụng để xây dựng các dashboard hiển thị dữ liệu thời gian thực và các công cụ giám sát, giúp doanh nghiệp dễ dàng theo dõi hiệu suất và tình trạng của các hệ thống.

2.4 Tổng quan về NestJs

2.4.1 Giới thiệu NestJs

Nest (NestJS) là một framework để xây dựng các ứng dụng phía máy chủ hiệu quả, chạy trên Node.js. Được xây dựng và hỗ trợ TypeScript (nhưng vẫn cho phép các nhà phát triển viết mã bằng JavaScript thuận tiện) và kết hợp các yếu tố của OOP, FP và FRP.

Nest sử dụng các khung HTTP Server mạnh mẽ như Express (mặc định) và tùy chọn cũng có thể được định cấu hình để sử dụng Fastify.

Nest cung cấp một mức độ trùu tượng trên các khung Node.js phổ biến này (Express hoặc Fastify), nhưng cũng hiển thị API của chúng trực tiếp cho nhà phát triển. Điều này cho phép các nhà phát triển tự do sử dụng vô số mô-đun của bên thứ ba có sẵn cho nền tảng cơ bản [4].

2.4.2 Cấu trúc của NestJs

NestJS được xây dựng từ nhiều yếu tố khác nhau, tuy nhiên quan trọng nhất là ba thành phần chính sau:

Modules: được sử dụng để tổ chức code và chia các tính năng thành các đơn vị có thể tái sử dụng hợp lý. Các tệp TypeScript được nhóm bằngDecorator”@Module”.Decorator”@Module”cung cấp siêu dữ liệu cho NestJS giúp nó xác định các thành phần, bộ điều khiển hay những tài nguyên khác để sắp xếp cấu trúc ứng dụng khoa học, hiệu quả.

Providers: Một thành phần không thể thiếu ở NestJS đó là Providers. Nó tương tự như một dịch vụ giúp xử lý những tác vụ mang tính phức tạp, logic mà các trình xử lý Controller không thể làm được. Providers có thể được tạo và đưa vào Controllers hoặc Providers khác.

Controllers: Chức năng chính của Controllers là xử lý các yêu cầu gửi đến và đáp trả lại cho client-side. Sau khi nhận được yêu cầu HTTP nó sẽ soạn thảo câu trả lời

chính xác, phù hợp nhất để gửi đi. Mỗi Controllers sẽ có bộ lô trình riêng để giúp nó thực hiện tốt các tác vụ khác nhau.

2.4.3 Ưu điểm của NestJs

Đối với một lập trình viên Nodejs, việc lựa chọn đúng framework cho project rất quan trọng. Trong khi Expressjs được sử dụng để xây dựng ứng dụng web trong nhiều năm thì NestJS đã xuất hiện và mang lại các lợi ích và tính năng nổi trội hơn:

Ngôn ngữ mạnh về kiểu dữ liệu: NestJS được xây dựng trên nền TypeScript, một phiên bản hỗ trợ kiểu dữ liệu của JavaScript. Sử dụng Typescript khi viết code giúp code sạch hơn và dễ xử lý khi có lỗi.

Kiến trúc modular: NestJS được xây dựng trên kiến trúc modular giúp việc tổ chức và phát triển mã nguồn trên quy mô lớn hơn trở nên dễ dàng. Kiến trúc này bao gồm các khối tách biệt nhau như controller, provider và module, giúp cho việc quản lý dễ dàng hơn. Mình sẽ nói kĩ hơn về các thành phần này ở phần sau của bài viết nhé.

Dependency Injection: NestJS sử dụng DI (Dependency Injection – Tiêm phụ thuộc) để quản lý luồng của các phụ thuộc giữ các module và component. Nó cho phép ứng dụng dễ kiểm thử và bảo trì hơn, do sự độc lập giữa các component với nhau. Cụ thể độc lập như thế nào thì cùng đợi đến phần sau nhé.

Built-In Validation: NestJS được hỗ trợ sẵn tính năng validate dữ liệu đầu vào - thứ đặc biệt quan trọng khi xây dựng các API. Framework này giúp định nghĩa và thực thi các validation rule một cách dễ dàng hơn, giảm đi các sai sót và lỗi không đáng có.

Hỗ trợ các tính năng nâng cao: NestJS hỗ trợ hàng loạt các tính năng được tích hợp sẵn, ví dụ như WebSocket, GraphQL và Microservices. Điều này giúp đơn giản hóa việc xây dựng các ứng dụng yêu cầu tính năng realtime hay kiến trúc microservices.

Cộng đồng hỗ trợ mạnh mẽ: NestJS có một cộng đồng các lập trình viên đóng góp cho framework thường xuyên. Điều này sẽ giúp NestJS luôn được cập nhật và sửa lỗi, trở thành 1 framework đáng tin cậy.

2.4.4 Nhược điểm của NestJS

Việc hỗ trợ nhiều tính năng mạnh mẽ không có nghĩa là NestJS không có bất kỳ điểm yếu nào.

Sự phụ thuộc vòng lặp (circular dependencies): Đây là một vấn đề phổ biến mà hầu hết các dự án NestJS sẽ gặp phải. Vấn đề này rất rắc rối và nó có thể làm chậm quá trình phát triển của phần mềm. May mắn là, một bài báo gần đây của Trilon đã đưa ra một công cụ tên là Madge, giúp xác định sớm circular dependencies.

Logs bị ẩn khi ứng dụng khởi động: Đây cũng là một vấn đề khá phổ biến. Điều này có thể khiến các lập trình viên khó debug khi có lỗi xảy ra. Để giải quyết thì họ thường vô hiệu hóa việc dừng ứng dụng khi gặp lỗi và gửi lại thông báo lỗi.

Khó khăn trong kiểm thử đơn vị (unit test): Unit test được tích hợp sâu vào framework. Việc kiểm thử ở mức đơn vị nhỏ thường gây ra nhiều boilerplate code. Hơn nữa, việc viết test có thể gây khó khăn với một số lập trình viên, vì họ phải hiểu được cơ chế hoạt động của dependency injection tree trong NestJS.

2.4.5 Cách cài đặt và tạo dự án NestJS

Để cài đặt NestJS, cần phải cài đặt Node.js và npm (Node Package Manager) trước tiên. Sau đó, có thể sử dụng npm để cài đặt NestJS CLI và tạo một dự án NestJS mới. Dưới đây là hướng dẫn cài đặt NestJS trên hệ điều hành Windows, macOS và Linux:

Bước 1: Cài đặt Node.js và npm

Trước tiên, hãy truy cập trang chủ của Node.js (<https://nodejs.org/>) để tải về phiên bản mới nhất và cài đặt Node.js và npm theo hướng dẫn trên trang web.

Bước 2: Cài đặt NestJS CLI

Sau khi cài đặt Node.js và npm thành công, mở cửa sổ dòng lệnh hoặc terminal trên máy tính của và chạy lệnh sau để cài đặt NestJS CLI một cách toàn cầu (global): “npm install -g @nestjs/cli”

Bước 3: Tạo một dự án NestJS mới

Sau khi cài đặt NestJS CLI, có thể sử dụng nó để tạo một dự án NestJS mới. Để tạo một dự án mới, hãy chạy lệnh sau trong cửa sổ dòng lệnh hoặc terminal: "nest new project-name"

Trong đó, 'project-name' là tên của dự án muốn tạo. NestJS CLI sẽ tạo ra một cấu trúc dự án NestJS cơ bản với các tập tin và thư mục cần thiết.

Bước 4: Chạy ứng dụng NestJS

Sau khi dự án NestJS đã được tạo thành công, có thể di chuyển vào thư mục dự án bằng cách chạy lệnh: "cd project-name"

Sau đó, có thể chạy ứng dụng NestJS bằng lệnh sau: "npm run start"

Hoặc nếu muốn tự động khởi động lại ứng dụng khi có thay đổi trong mã nguồn, có thể sử dụng lệnh: "npm run start:dev"

Sau khi ứng dụng đã chạy thành công, có thể truy cập vào <http://localhost:3000> (hoặc cổng khác nếu đã thay đổi cài đặt) để xem ứng dụng NestJS.

2.5 Tổng quan về NextJS

2.5.1 Giới thiệu NextJS

NextJS là một framework có mã nguồn mở được xây dựng trên nền tảng của React, cho phép xây dựng các trang web tĩnh có tốc độ siêu nhanh và thân thiện với người dùng, cũng như xây dựng các ứng dụng web React.

NextJS được ra đời vào năm 2016, thuộc sở hữu của Vercel. NextJS bắt đầu trở nên phổ biến vào năm 2018 và tiếp tục tăng trưởng mạnh mẽ trong cộng đồng phát triển web. Sự kết hợp của các tính năng như SSR với SSG đã giúp NextJS trở thành sự lựa chọn hấp dẫn cho nhiều dự án [5].

2.5.2 Các tính năng chính của NextJS

Routing trong NextJS

Automatic Routing: NextJS sẽ tự động tạo các router dựa trên cấu trúc thư mục của dự án. Ví dụ, nếu tạo một file có tên là about.js ở thư mục pages. NextJS sẽ tạo router là /about.

Nested Routing: Có thể tạo các thư mục con để tạo các router lồng nhau. Ví dụ, nếu tạo một folder có tên blog nằm trong folder pages, bên trong folder blog lại có file post.js, đường dẫn sẽ là pages/blog/post.js, thì router mà NextJS tạo ra sẽ là /blog/post.

Dynamic Routes: Có thể tạo các router động bằng cách sử dụng cặp dấu [] trong tên file. Ví dụ nếu đường dẫn là pages/blog/[slug].js thì NextJS sẽ tạo ra các router như /blog/blog-dau-tien hoặc /blog/blog-thu-hai. Với slug là một giá trị bắt kì truyền vào.

Link Component: Để tạo liên kết giữa các trang, có thể sử dụng component Link được cung cấp sẵn bởi NextJS ở thư viện next/link. Sử dụng Link thay cho thẻ <a /> giúp tránh việc tải lại trang và tối ưu hóa hiệu suất.

Query Parameters: Có thể truyền dữ liệu giữa các trang sử dụng query parameters trong router bằng cách sử dụng ký tự dấu chấm hỏi? trong tên file.

Rendering trong NextJS:

Khi user truy cập vào một trang sử dụng SSR của NextJS, browser sẽ gửi một request đến server

Trước khi gửi kết quả ra trình duyệt, NextJS chạy hàm getServerSideProps() để lấy dữ liệu cần thiết cho trang. Hàm này thường sẽ dùng để gọi API hoặc truy vấn vào cơ sở dữ liệu để lấy dữ liệu cần thiết.

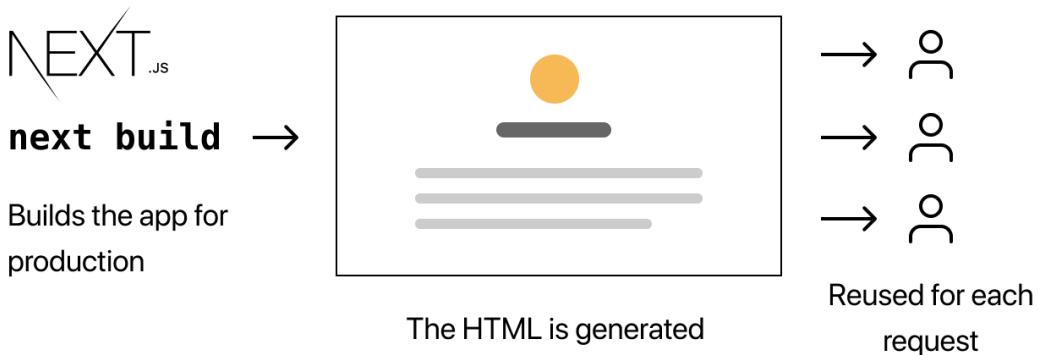
Với dữ liệu cần thiết đã được lấy từ bước trước, NextJS tạo ra một phiên bản đã render đầy đủ cả nội dung và dữ liệu. Sau đó trả phiên bản này ra browser dưới dạng HTML

Browser nhận được code HTML từ server và hiển thị lên giao diện. Nội dung của trang sẽ được hiển thị ngay lập tức, sau đó JS được tải xuống và đổ lên để trang có thể tương tác được, ví dụ như hàm onClick().

Static Site Generation: SSG là một phương pháp mà NextJS cung cấp sẵn, cho phép tạo các trang tĩnh và lưu chúng xuống dưới dạng file html tĩnh. Điều này giúp cải thiện hiệu suất tải trang và cung cấp trải nghiệm người dùng tốt hơn vì nội dung được lấy từ file html và hiển thị ngay lập tức mà không cần đợi việc tải về từ phía server.

Static Generation

The HTML is generated at **build-time** and is reused for each request.



Hình 2.2.3. Static Generation trong NextJs

Styling trong NextJs:

CSS Modules: Để style cho ứng dụng NextJS, cách dễ nhất là tạo các file CSS/SCSS riêng lẻ cho từng component hoặc sử dụng file chung cho toàn dự án.

The screenshot shows a code editor interface with two tabs: "CSS" and "JS".

CSS Tab:

```
1 // styles.module.css
2 .myButton {
3   background-color: blue;
4   color: white;
5 }
```

JS Tab:

```
1 // MyComponent.js
2 import styles from './styles.module.css';
3
4 function MyComponent() {
5   return <button className={styles.myButton}>Click me</button>;
6 }
```

Hình 2.2.4. Ví dụ sử dụng CSS Modules trong NextJs

CSS-in-JS Libraries: Ngoài ra cũng có thể sử dụng các thư viện CSS-in-JS như Styled-Components để viết trực tiếp css vào code.

```
// MyComponent.js
import styled from 'styled-components';

const StyledButton = styled.button`
background-color: blue;
color: white;
`;

function MyComponent() {
  return <StyledButton>Click me</StyledButton>;
}

export default MyComponent;
```

Hình 2.2.5. Ví dụ sử dụng CSS-in-JS Libraries trong NextJS

CSS FrameWorks: Ngoài ra NextJS cũng hỗ trợ sử dụng cùng các CSS framework như TailwindCSS, Bootstrap hoặc MaterialUI.

2.5.3 Ưu điểm của NextJs

NextJS hiện tại đang được sử dụng ở rất nhiều dự án khác nhau, sở dĩ NextJS được tin dùng như vậy là vì một số lý do sau:

Sử dụng SSR và SSG: Giúp cải thiện tốc độ tải trang và khả năng SEO.

Có nhiều tính năng giúp tối ưu hóa hiệu suất như Code Splitting, Lazy Loading, Image Optimization,...

Fast Refresh: Tính năng giúp tự động làm mới giao diện mà không cần load lại toàn bộ trang.

Tự động tạo file CSS dành riêng cho mỗi trang, giúp tránh xung đột trong việc sử dụng và quản lý các file CSS.

Hỗ trợ TypeScript: NextJS cũng hỗ trợ sử dụng Typescript giúp cải thiện tính rõ ràng cho code và thuận tiện cho việc debug về sau.

Cộng đồng lớn: NextJS có một cộng đồng sử dụng đông đảo, điều này được chứng minh ở trên chính trang Github của NextJS khi nó đang đạt khoảng hơn 100k sao. Điều này giúp cho NextJS có thêm nhiều nguồn tài liệu phong phú và các plugin hữu ích.

Hệ sinh thái mạnh mẽ: NextJS kết hợp tốt với các thư viện và công cụ như Redux, React Query, Apollo Client và nhiều thư viện khác nằm trong hệ sinh thái của React.

Tích hợp tốt với React: Nếu đã quen với việc sử dụng React trước đó thì việc làm quen với NextJS sẽ đơn giản hơn rất nhiều.

2.5.4 Nhược điểm của NextJS

Mặc dù có nhiều ưu điểm nêu trên, NextJS vẫn có những khuyết điểm:

Khó học cho người mới: Nếu chưa có hiểu biết cơ bản về Web Fundamentals, JavaScript và React thì việc học NextJS sẽ hơi khó khăn. Nhất là khi gặp các khái niệm như SSR hay SSG.

Khó khăn trong việc tích hợp với một số thư viện bên ngoài: Một số thư viện và plugin có thể cần phải điều chỉnh hoặc tùy chỉnh để hoạt động tốt với Next.js. Ví dụ như để sử dụng Redux trong ứng dụng NextJS, cần cài thêm thư viện next-redux-wrapper để quản lý state trên cả server và client.

Phụ thuộc vào hệ sinh thái của React: NextJs phụ thuộc vào React, vì vậy nếu không quen thuộc với React hoặc không muốn sử dụng React thì NextJS không phải là lựa chọn tốt.

Đòi hỏi chạy trên server NodeJS: Để deploy ứng dụng NextJS, cần có một máy chủ NodeJS, việc này có thể làm tăng chi phí và quá trình triển khai sẽ trở nên phức tạp hơn.

Cấu trúc dự án phức tạp: Với các dự án lớn, việc quản lý cấu trúc dự án không cẩn thận lúc ban đầu sẽ dẫn đến việc khó quản lý sau này.

2.5.5 Sử dụng NextJs cơ bản

Chuẩn bị môi trường: Trước tiên, cần cài đặt NodeJS và npm (hoặc yarn) vào máy của mình. Có thể cài NodeJS ở trang chủ của NodeJS hoặc cài thông qua thư viện nvm.

Tạo một project NextJS mới: Để tạo một project NextJS mới, cần chạy câu lệnh sau: npx create-next-app@latest hoặc yarn create next-app my-nextjs-app (nếu dùng yarn)

Xây dựng website bán hàng đa nền tảng

Chạy project trong NextJS: Để chạy project, cần chạy câu lệnh sau: npm run dev hoặc yarn dev (nếu dùng yarn)

Tạo các trang trong dự án NextJs:

pages là nơi để bạn tạo router của ứng dụng. Bên trong pages có 3 file quan trọng đó là: _app.js, _document.js, index.js

_app.js: Là điểm khởi đầu của toàn ứng dụng. Ở file này có thể đưa các thành phần layout như Header, Footer để chúng bọc tất cả các trang trên app. Hoặc Redux Store cũng có thể bọc ở đây.

_document.js: _document.js không phải là một trang riêng lẻ, nó là một tệp dùng để cấu hình các thẻ html, head và các thẻ khác trong file html. Có thể sửa đổi hoặc thêm các thẻ như <meta>, <link>, hay các đoạn CSS, JS liên quan đến toàn bộ trang web.

index.js: Đây chính là trang homepage của dự án, khi truy cập vào địa chỉ trang chủ http://localhost:3000, NextJS sẽ tự động hiển thị nội dung từ file index.js

Folder api: Được sử dụng để tạo các API Routes trong NextJS, đây là tính năng vô cùng mạnh mẽ của NextJS cho phép tạo các API Endpoint ở ngay trong ứng dụng NextJS.

2.6 Tổng quan về Tailwind CSS

2.6.1 Giới thiệu Tailwind CSS

Tailwind CSS là một framework CSS utility-first giúp tạo kiểu nhanh chóng và hiệu quả cho website mà không cần viết CSS thủ công. Thay vì cung cấp các thành phần (component) hoặc kiểu thiết kế mặc định, Tailwind cung cấp một loạt các lớp tiện ích (utility classes) mà bạn có thể kết hợp linh hoạt để xây dựng giao diện tùy chỉnh.



Hình 2.2.6. Giới thiệu Tailwind CSS

2.6.2 Ưu điểm của Tailwind CSS

Tùy biến cao, hiệu suất cao. Không giống như Bootstrap cung cấp những class giàn như đồng gọi sẵn, chỉ cần gọi ra là dùng. Tailwind giúp bạn định nghĩa những phần phù hợp với dự án của bạn mà không bị gò bó.

Cho phép xây dựng responsive layout phức tạp.

Responsive và phát triển dễ dàng.

Tạo thành phần dễ dàng.

Hỗ trợ cài đặt với nhiều framework front-end khác như react, vuejs,...

2.6.3 Nhược điểm của Tailwind CSS

Thiếu tiêu đề và thành phần điều hướng (navigation).

Cần có thời gian để làm quen, tìm hiểu và nhớ tên các class.

Có kiến thức về CSS thì mới sử dụng tốt được.

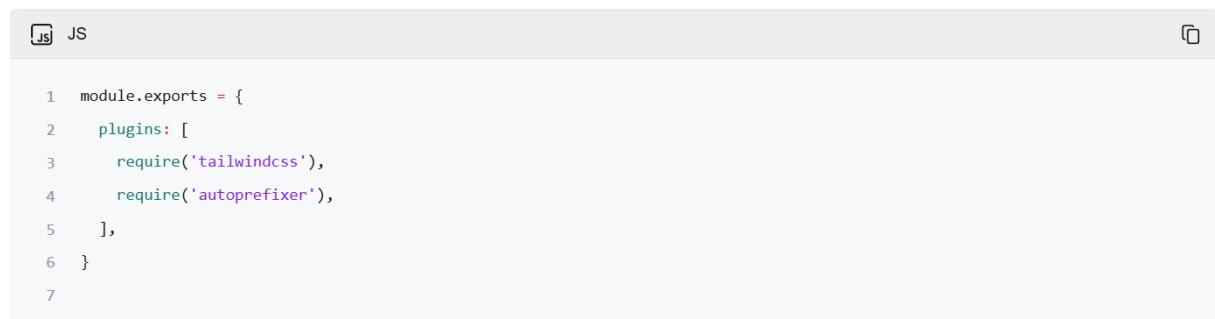
2.6.4 Cài đặt và sử dụng Tailwind CSS

Hướng dẫn cài đặt Tailwind CSS:

Sử dụng lệnh sau để cài đặt: npm install tailwindcss postcss autoprefixer hoặc yarn add tailwindcss postcss autoprefixer (nếu dùng yarn)

Tạo tệp cấu hình: npx tailwindcss init

Tạo một postcss.config.js file trong thư mục gốc của thư mục dự án và thêm nội dung sau:

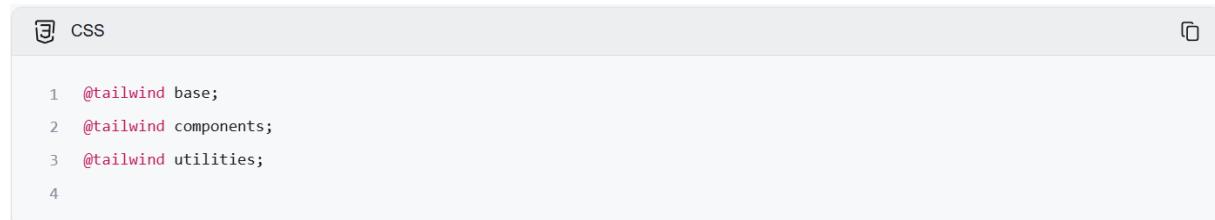


```
JS JS

1 module.exports = {
2   plugins: [
3     require('tailwindcss'),
4     require('autoprefixer'),
5   ],
6 }
7
```

Hình 2.2.7. Nội dung tệp postcss.config.js

Tạo một styles.css tệp trong thư mục dự án của bạn src (hoặc bất kỳ thư mục nào khác mà bạn thích) và thêm nội dung sau:



```
CSS CSS

1 @tailwind base;
2 @tailwind components;
3 @tailwind utilities;
4
```

Hình 2.2.8. Nội dung tệp styles.css

Cuối cùng, thêm tập lệnh vào package.json file của để tạo CSS:



```
JSON JSON

1 "scripts": {
2   "build": "postcss src/styles.css -o dist/styles.css"
3 }
```

Hình 2.2.9. Thêm tập lệnh vào package.json

Chạy lệnh sau để xây dựng CSS: npm run built hoặc yarn built.

2.7 Tổng quan về MongoDB

2.7.1 Sơ lược về NoSQL

Cơ sở dữ liệu NoSQL chuyên dành cho các mô hình dữ liệu cụ thể và lưu trữ dữ liệu trong các sơ đồ linh hoạt dễ dàng điều chỉnh quy mô cho các ứng dụng hiện đại. Cơ sở dữ liệu NoSQL được công nhận rộng rãi vì khả năng dễ phát triển, chức năng cũng như hiệu năng ở quy mô lớn [6].



Hình 2.10 So sánh NoSQL với SQL

2.7.2 Giới thiệu MongoDB

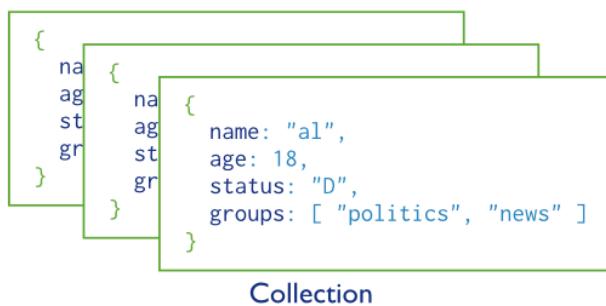
MongoDB là phần mềm cơ sở dữ liệu mã nguồn mở NoSQL hỗ trợ đa nền tảng được thiết kế theo hướng đối tượng. Các bảng (trong MongoDB gọi là collection) có cấu trúc linh hoạt cho phép dữ liệu không cần tuân theo dạng cấu trúc nào.

Vì thế, nó có thể dùng để lưu trữ dữ liệu có cấu trúc phức tạp và đa dạng. Dữ liệu được gọi là Big Data. Đặc biệt, chương trình này lưu trữ dữ liệu vào collection theo hướng tài liệu kiểu JSON thay vì bảng nên có hiệu suất cao và tính khả dụng cao [7].

2.7.3 Một số khái niệm trong MongoDB

Database là một container vật lý cho các collection. Mỗi DB được thiết lập cho riêng nó một danh sách các files hệ thống files. Một máy chủ MongoDB đơn thường có nhiều DB.

Collection là một nhóm các documents của MongoDB. Nó tương đương với một table trong RDBMS. Một Collection tồn tại trong một cơ sở dữ liệu duy nhất. Các collection ko tạo nên một schema. Documents trong collection có thể có các fields khác nhau. Thông thường, tất cả các documents trong collections có mục đích khá giống nhau hoặc liên quan tới nhau.



Hình 2.11 Ví dụ về Collection trong MongoDB

Document là một tập hợp các cặp key-value. Documents có schema động. Schema động có nghĩa là documents trong cùng một collection không cần phải có cùng một nhóm các fields hay cấu trúc giống nhau và các fields phổ biến trong các documents của collection có thể chứa các loại dữ liệu khác nhau.



Hình 2.12 Ví dụ minh họa cho Document trong MongoDB

2.7.4 Ưu điểm của MongoDB

Đầu tiên có thể nhắc đến là tính linh hoạt lưu trữ dữ liệu theo các kích cỡ khác nhau, dữ liệu dưới dạng hướng tài liệu JSON nên có thể chèn vào thoải mái bất cứ thông tin mong muốn.

Khác với RDBMS, dữ liệu trong đây không có sự ràng buộc và không có yêu cầu tuân theo khuôn khổ nhất định, điều này giúp tiết kiệm thời gian cho việc kiểm tra sự thỏa mãn về cấu trúc nếu muốn chèn, xóa, cập nhật hay thay đổi các dữ liệu trong bảng.

MongoDB dễ dàng mở rộng hệ thống bằng cách thêm node vào cluster (cụm các node chứa dữ liệu giao tiếp với nhau).

Ưu điểm thứ tư là tốc độ truy vấn nhanh hơn nhiều so với hệ quản trị cơ sở dữ liệu quan hệ RDBMS do dữ liệu truy vấn được cached lên bộ nhớ RAM để lượt truy vấn sau diễn ra nhanh hơn mà không cần đọc từ ổ cứng.

Cũng là một ưu điểm về hiệu suất truy vấn của MongoDB, trường dữ liệu”_id” luôn được tự động đánh chỉ mục để đạt hiệu suất cao nhất.

2.7.5 Nhược điểm của MongoDB

Dữ liệu trong MongoDB không bị ràng buộc như RDBMS nhưng người sử dụng lưu ý cẩn thận mọi thao tác để không xảy ra các kết quả ngoài ý muốn gây ảnh hưởng đến dữ liệu.

Một nhược điểm mà”dân công nghệ” hay lo ngại là bộ nhớ của thiết bị. Chương trình này thường tồn bộ nhớ do dữ liệu được lưu dưới dạng key-value, trong khi các collection chỉ khác về value nên sẽ lặp lại key dẫn đến thừa dữ liệu.

Thông thường, dữ liệu thay đổi từ RAM xuống ổ cứng phải qua 60 giây thì chương trình mới thực hiện hoàn tất, đây là nguy cơ bị mất dữ liệu nếu bất ngờ xảy ra tình huống mất điện trong vòng 60 giây đó.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1 Phân tích và đặc tả yêu cầu hệ thống

3.1.1 Yêu cầu chức năng

Đối với người dùng (khách hàng):

- Đăng nhập, đăng xuất khỏi hệ thống: Cho phép tạo tài khoản mới, đăng nhập bằng email và mật khẩu và đăng xuất khỏi hệ thống khi không sử dụng.
- Quản lý tài khoản cá nhân: Cập nhật các thông tin cá nhân như tên, email, địa chỉ giao hàng, số điện thoại và đổi mật khẩu tài khoản.
- Xem sản phẩm và tìm kiếm: Duyệt sản phẩm theo danh mục và tìm kiếm sản phẩm theo từ khóa, lọc theo giá, tên, mức đánh giá.
- Chi tiết sản phẩm: Hiển thị thông tin sản phẩm, hình ảnh, giá, mô tả, số lượng còn lại và đánh giá từ người mua khác.
- Giỏ hàng: Thêm sản phẩm vào giỏ, chỉnh sửa số lượng hoặc xóa sản phẩm.
- Đặt hàng: Chọn địa chỉ giao hàng, xác nhận giỏ hàng, tiến hành thanh toán qua PayOS hoặc có thể thanh toán khi nhận hàng. Nhận email xác nhận đơn hàng sau khi đặt thành công.
- Theo dõi đơn hàng: Xem danh sách đơn hàng đã đặt, trạng thái xử lý (đang chờ, đang giao, đã giao...).
- Đánh giá sản phẩm: Gửi nhận xét và chấm sao sau khi đơn hàng hoàn tất.

Đối với người quản trị:

- Quản lý người dùng: Xem danh sách tài khoản người dùng, khóa hoặc mở khóa tài khoản nếu cần.
- Quản lý danh mục sản phẩm: Thêm, sửa, xóa danh mục để tổ chức sản phẩm rõ ràng.
- Quản lý sản phẩm: Thêm mới, chỉnh sửa hoặc xóa sản phẩm từ hệ thống.
- Quản lý đơn hàng: Xem toàn bộ đơn hàng, cập nhật trạng thái đơn hàng (đang xử lý, đã giao...).
- Thống kê doanh thu: Xem báo cáo đơn hàng và doanh thu qua biểu đồ.

3.1.2 Yêu cầu phi chức năng

Khả năng mở rộng: Dễ nâng cấp, thêm tính năng mới trong tương lai.

Bảo mật: Sử dụng JWT để xác thực, phân quyền rõ ràng giữa user và admin.

Hiệu năng: Giao diện tải nhanh, truy vấn dữ liệu tối ưu.

Đáp ứng đa nền tảng (responsive): Hoạt động tốt trên máy tính, điện thoại, máy tính bảng.

Dễ sử dụng: Giao diện thân thiện với người dùng và quản trị viên.

3.2 Thiết kế hệ thống

3.2.1 Thiết kế cơ sở dữ liệu

Cơ sở dữ liệu gồm có các collection chính như sau:

1. Collection "users": (Lưu trữ thông tin người dùng như: tên, email, số điện thoại, địa chỉ,...)

```
“users”: {  
    “_id”：“ObjectId”,  
    “name”：“String”,  
    “email”：“String”,  
    “password”：“String”,  
    “role”：“String (user | admin)”,  
    “address”：“String”,  
    “phone”：“String”,  
    “isVerified”：“Boolean”,  
    “verifyToken”：“String”,  
    “resetToken”：“String”,  
    “resetTokenExpires”：“Date”,  
    “avatarUrl”：“String”,
```

```
    "birthday": "String (yyyy-mm-dd)",  
    "gender": "String (nam | nu)",  
    "createdAt": "Date",  
    "updatedAt": "Date"  
}
```

2. Collection "categories": (Lưu trữ danh mục của sản phẩm ví dụ như: laptop, pc,...)

```
"categories": {  
    "_id": "ObjectId",  
    "name": "String",  
    "slug": "String",  
    "icon": "String",  
    "description": "String",  
    "createdAt": "Date",  
    "updatedAt": "Date"  
}
```

3. Collection "products": (Lưu trữ thông tin sản phẩm)

```
"products": {  
    "_id": "ObjectId",  
    "name": "String",  
    "description": "String",  
    "originalPrice": "Number",  
    "price": "Number",  
    "stock": "Number",  
    "imageUrl": "String",
```

```
“thumbnailUrl”：“String”，  
“gallery”：[“String”]，  
“brand”：“String”，  
“isActive”：“Boolean”，  
“specifications”：{  
    “<key>”：“String”  
},  
“categoryId”：“ObjectId”，  
“ratingAvg”：“Number”，  
“ratingCount”：“Number”，  
“parentId”：“ObjectId | null”，  
“variantAttributes”：{  
    “<key>”：“String”  
},  
“createdAt”：“Date”，  
“updatedAt”：“Date”  
}
```

4. Collection “cart”: (Lưu trữ thông tin giỏ hàng)

```
“carts”：{  
    “_id”：“ObjectId”，  
    “userId”：“ObjectId”，  
    “items”：[  
        {  
            “productId”：“ObjectId”，  
            “quantity”：“Number”
```

}

]

}

5. Collection "orders": (Lưu trữ thông tin đơn hàng)

“orders”: {

“_id”：“ObjectId”,

“userId”：“ObjectId”,

“items”: [

{

“productId”：“ObjectId”,

“name”：“String”,

“price”：“Number”,

“quantity”：“Number”,

“itemTotal”：“Number”,

“imageUrl”：“String”,

“isReviewed”：“Boolean”

}

],

“total”：“Number”,

“orderCode”：“Number”,

“paymentUrl”：“String”,

“checkoutUrl”：“String”,

“status”：“String”,

“paymentMethod”：“String”,

“address”：“String”,

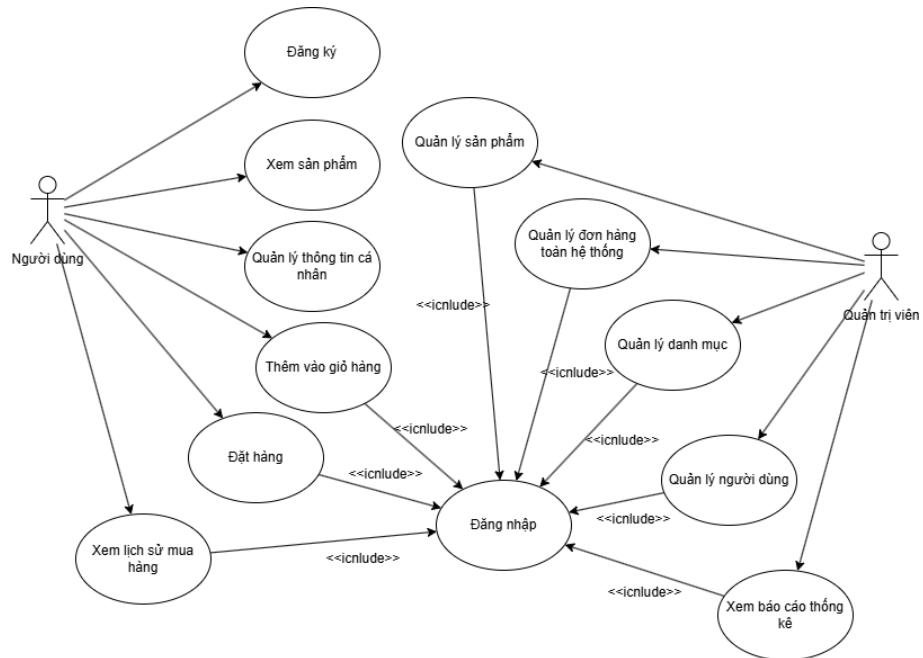
```
    "phone": "String",  
    "createdAt": "Date",  
    "updatedAt": "Date"  
}
```

6. Collection "reviews": (Lưu trữ bình luận, đánh giá về sản phẩm)

```
"reviews": {  
    "_id": "ObjectId",  
    "user": "ObjectId",  
    "product": "ObjectId",  
    "rating": "Number",  
    "comment": "String",  
    "createdAt": "Date",  
    "updatedAt": "Date"  
}
```

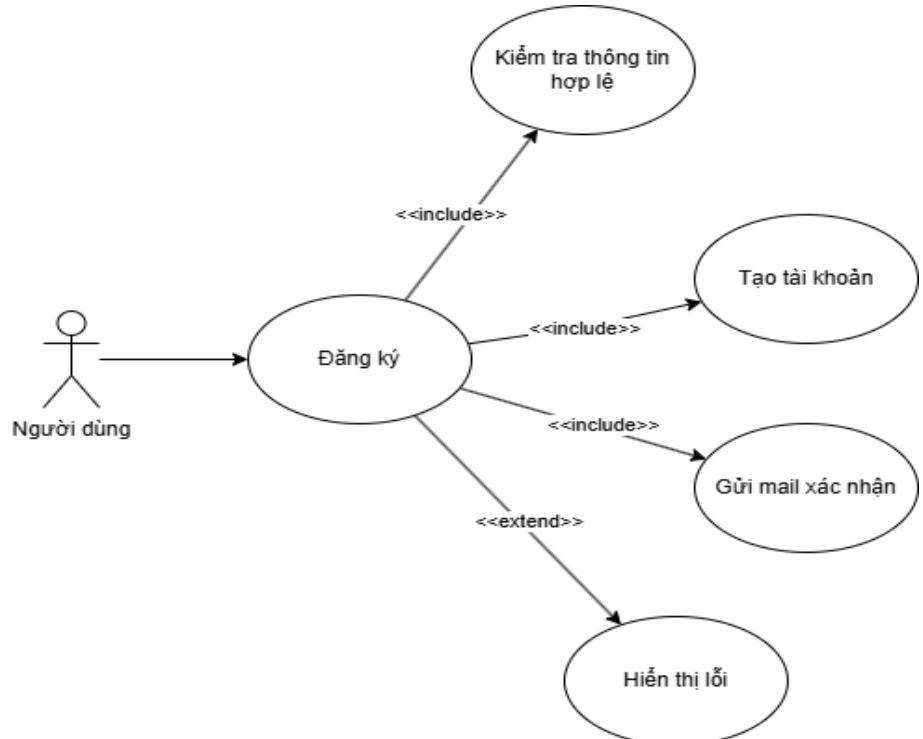
3.2.2 Sơ đồ Use Case

Use Case tổng quát:



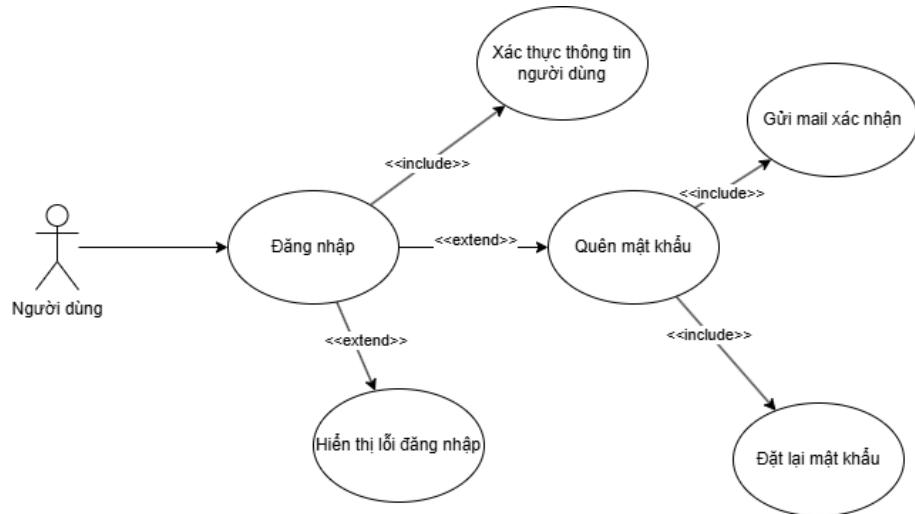
Sơ đồ 3.1 Use Case tổng quát

Use Case Đăng ký:



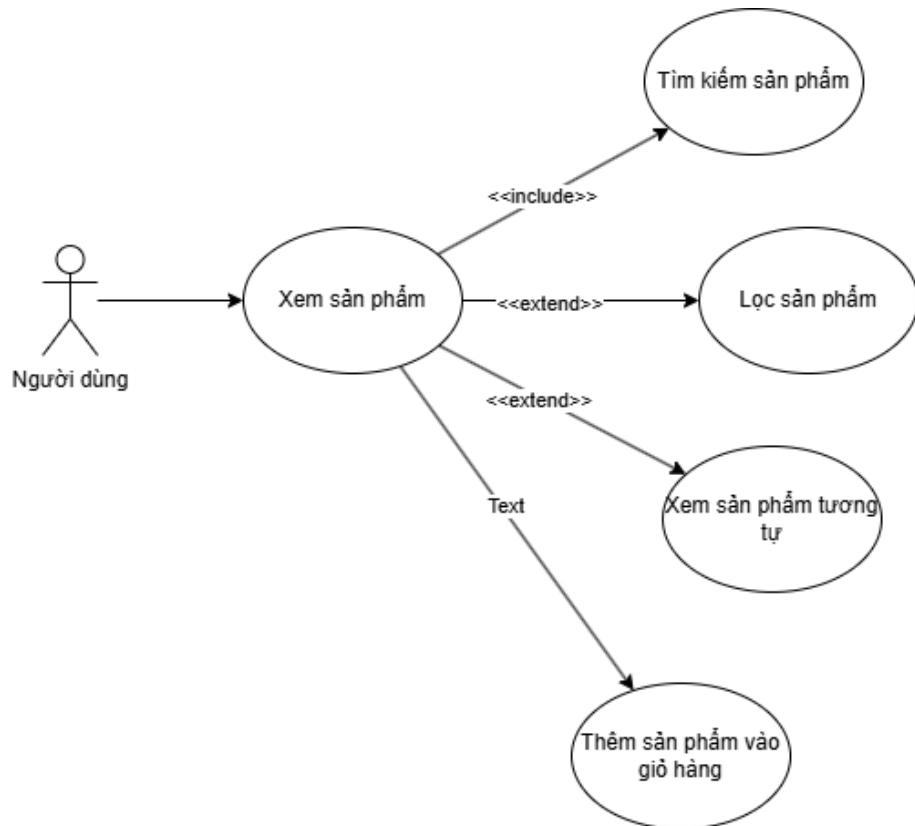
Sơ đồ 3.2 Use Case đăng ký

Use Case đăng nhập:



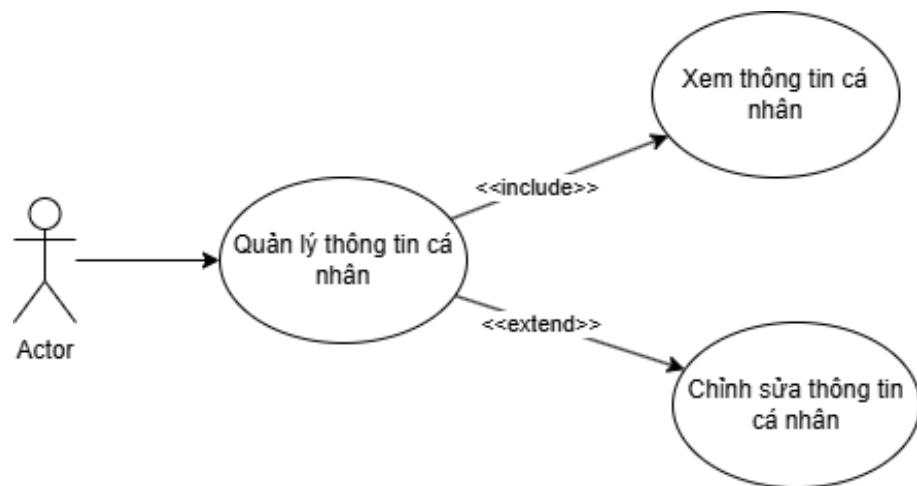
Sơ đồ 3.3 Use Case đăng nhập

Use Case xem sản phẩm:



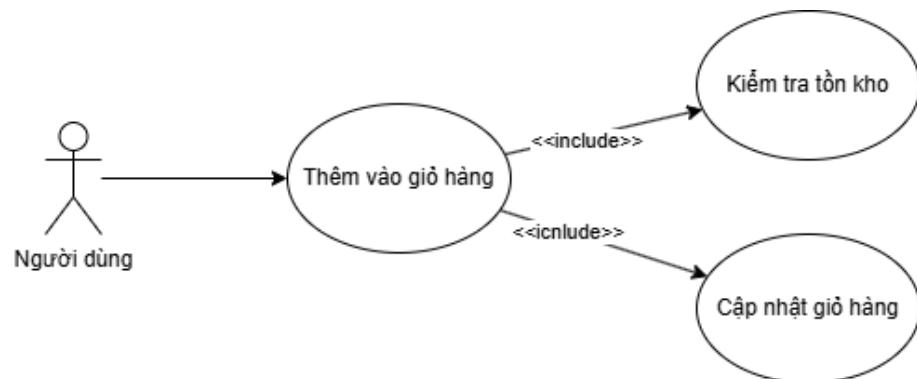
Sơ đồ 3.4 Use Case xem sản phẩm

Use Case quản lý thông tin cá nhân:



Sơ đồ 3.5 Use Case quản lý thông tin cá nhân

Use Case thêm sản phẩm vào giỏ hàng:



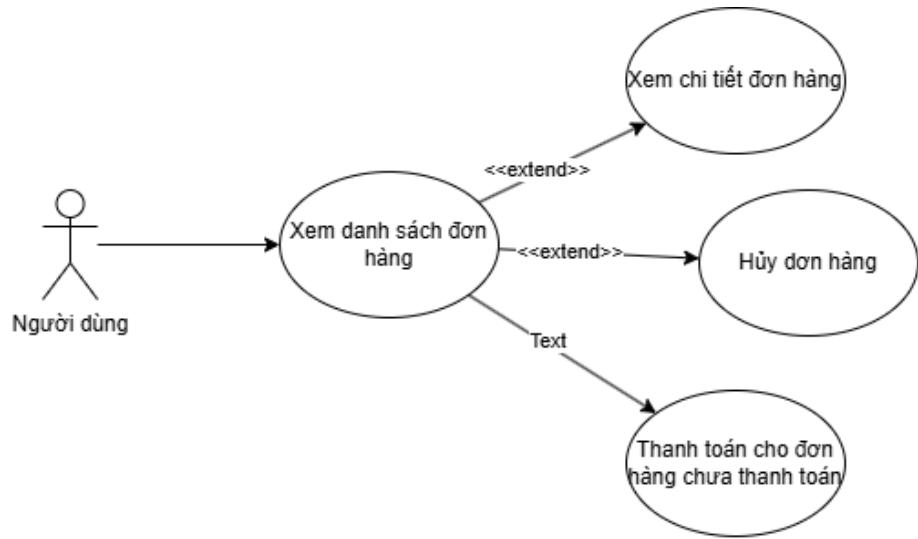
Sơ đồ 3.6 Use Case thêm sản phẩm vào giỏ hàng

Use Case đặt hàng:



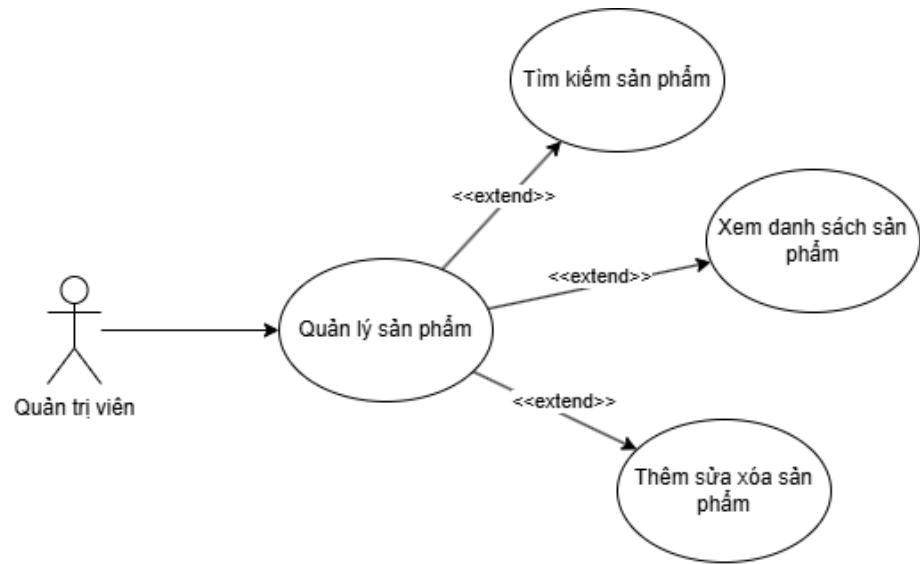
Sơ đồ 3.7 UseCase đặt hàng

Use Case lịch sử mua hàng:



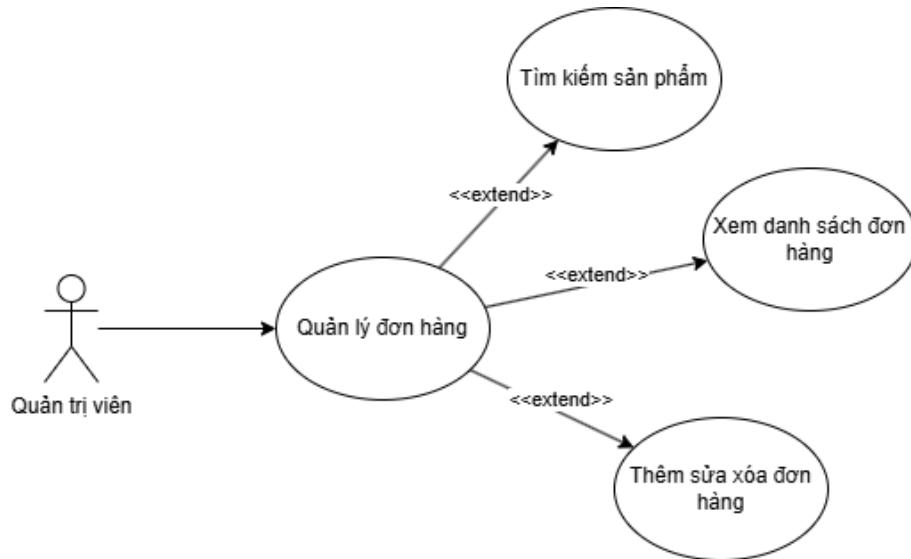
Sơ đồ 3.8 Use Case lịch sử mua hàng

Use Case quản lý sản phẩm:



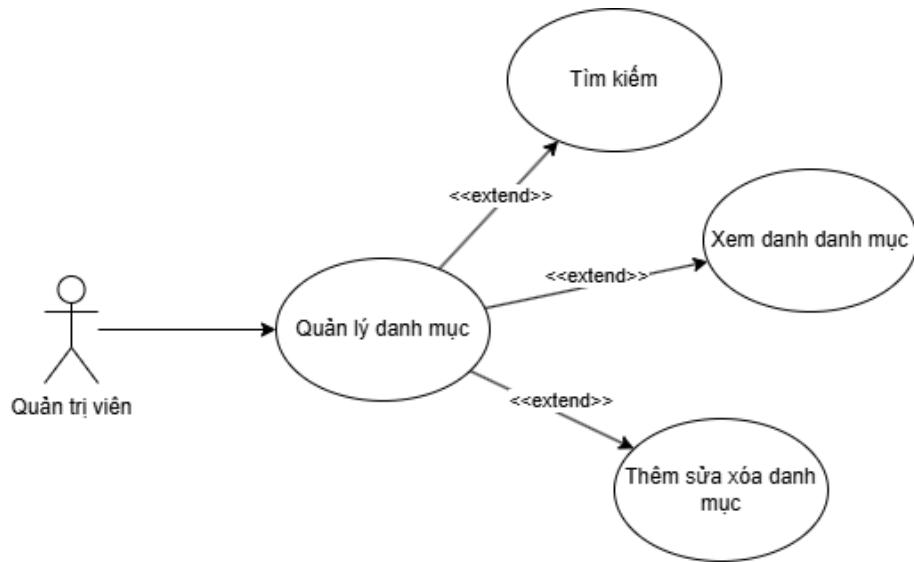
Sơ đồ 3.9 Use Case quản lý sản phẩm

Use Case quản lý đơn hàng:



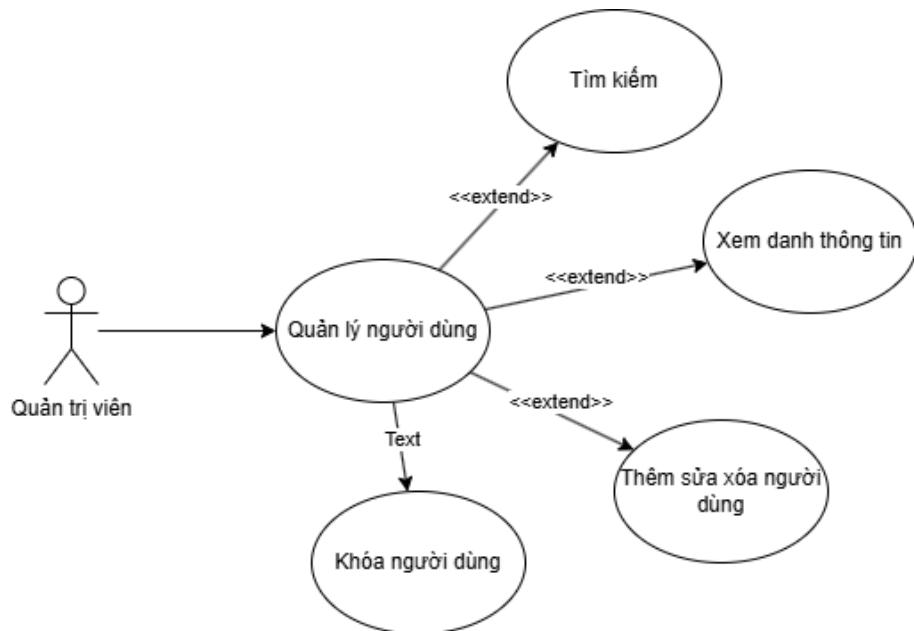
Sơ đồ 3.10 Use Case quản lý đơn hàng

Use Case quản lý danh mục:



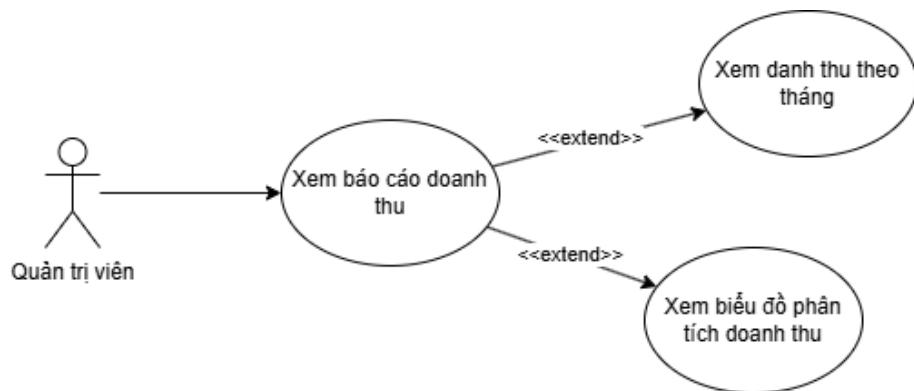
Sơ đồ 3.11 Use Case quản lý danh mục

Use Case quản lý người dùng:



Sơ đồ 3.12 Use Case quản lý người dùng

Use Case xem báo cáo thống kê:



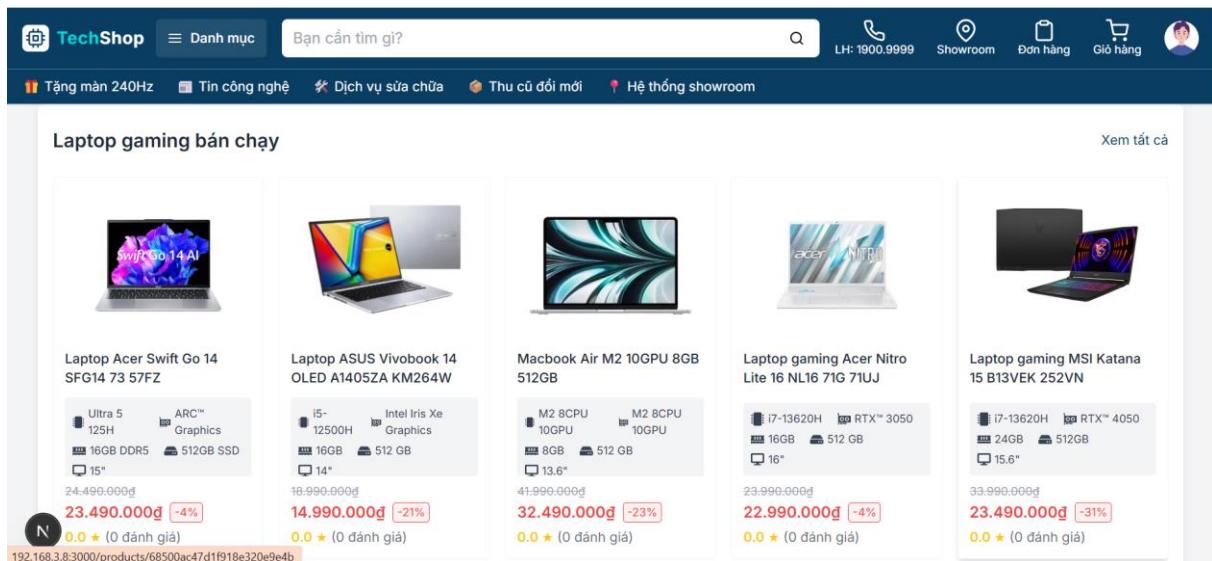
Sơ đồ 3.13 UseCase xem báo cáo thống kê

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1 Giao diện website trên desktop

4.1.1 Trang chủ

Trang chủ đóng vai trò là nơi tập trung các chức năng chính của website. Người dùng có thể dễ dàng tìm kiếm sản phẩm, truy cập các danh mục sản phẩm, quản lý đơn hàng, gioi hàng, tài khoản cá nhân và xem nhanh các sản phẩm bán chạy ngay trên trang chủ.

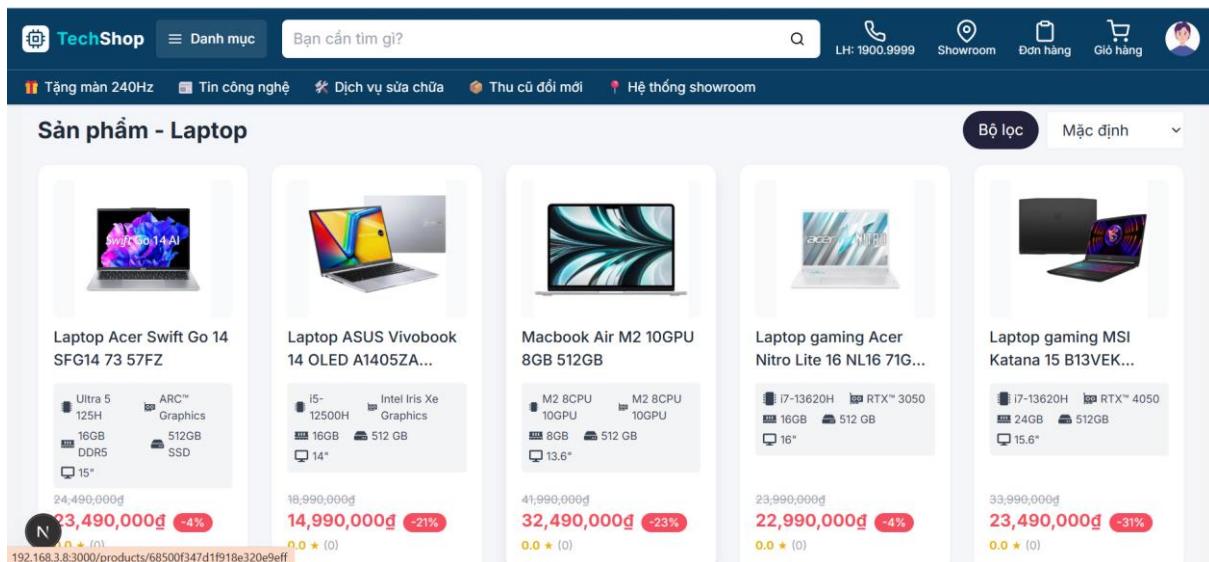


Hình 4.1 Trang chủ trên desktop

Xây dựng website bán hàng đa nền tảng

4.1.2 Trang xem theo danh mục

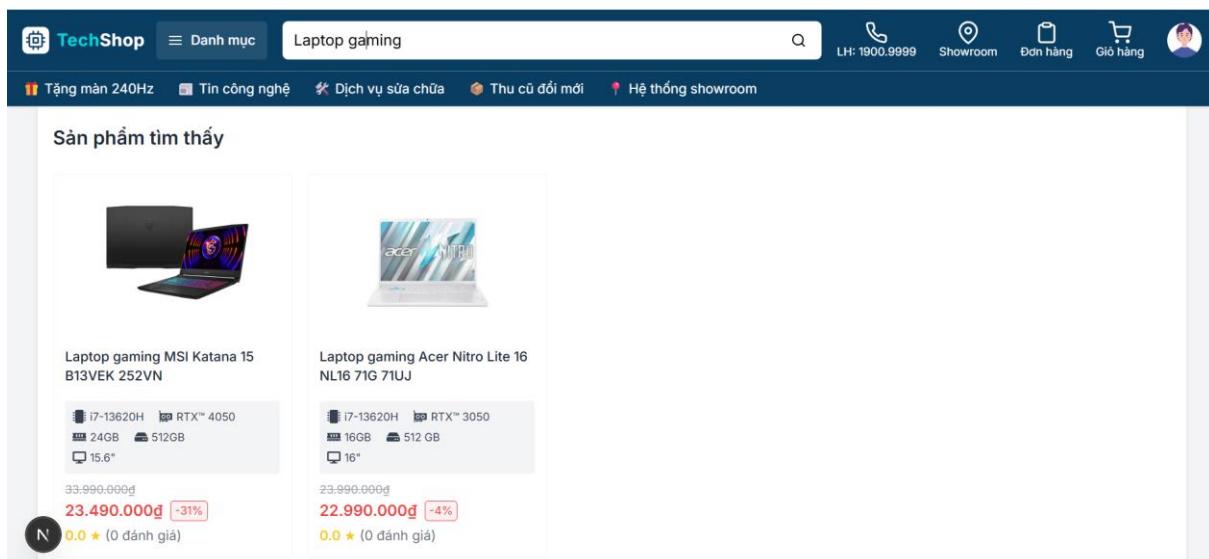
Trang xem theo danh mục sản phẩm là trang hiển thị sản phẩm theo các danh mục của sản phẩm. Ngoài ra trang còn có bộ lọc giúp người dùng lọc sản phẩm theo các tiêu chí như giá, thương hiệu,...



Hình 4.2 Trang xem theo danh mục trên desktop

4.1.3 Trang tìm kiếm sản phẩm

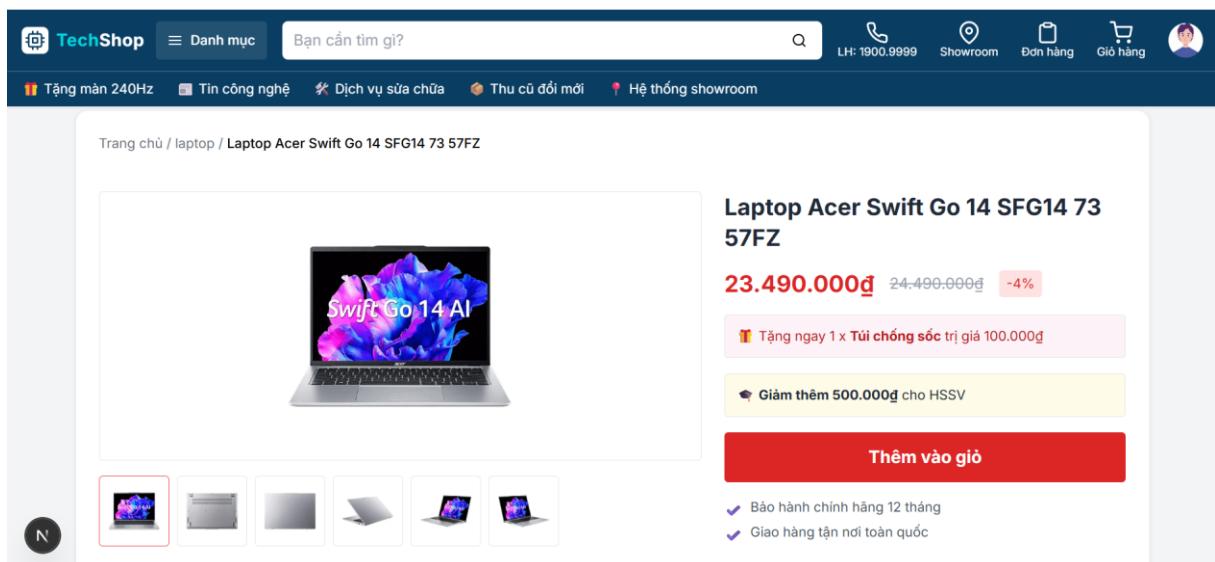
Là trang trả kết quả tìm kiếm về khi dùng chức năng tìm kiếm trên thanh công cụ trên cùng của website.



Hình 4.3 Trang tìm kiếm sản phẩm trên desktop

4.1.4 Trang chi tiết sản phẩm

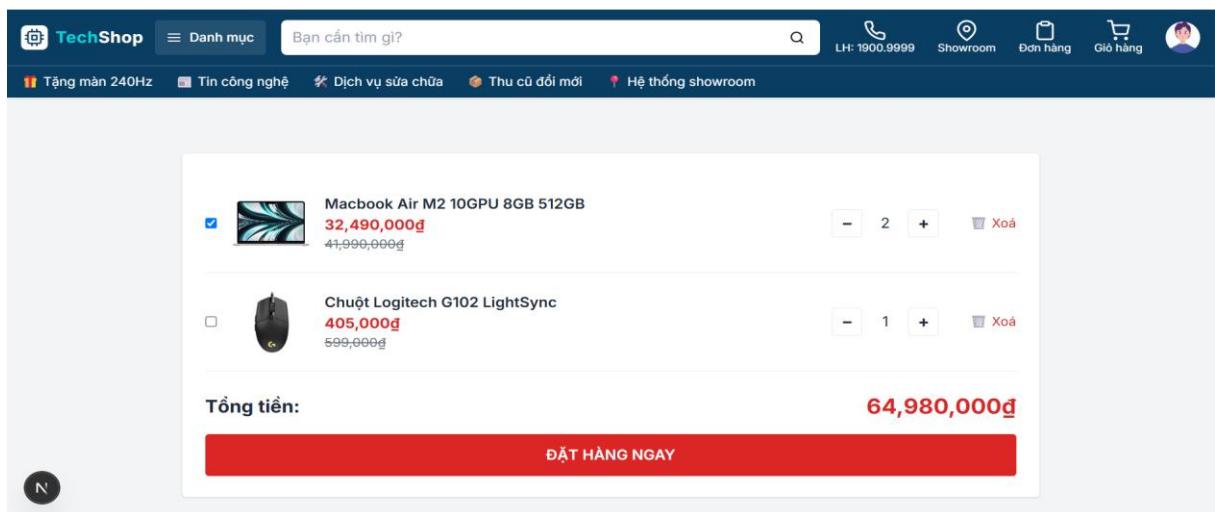
Trang chi tiết sản phẩm hiển thị chi tiết sản phẩm khi người dùng chọn vào một sản phẩm. Ở trang này sẽ hiển thị: tên các hình ảnh sản phẩm, giá tiền, cấu hình, sản phẩm mua kèm, sản phẩm liên quan, đánh giá của những người đã mua sản phẩm này. Nếu muốn mua người dùng có thể thêm sản phẩm vào giỏ hàng.



Hình 4.4 Trang chi tiết sản phẩm trên desktop

4.1.5 Trang giỏ hàng

Là trang hiển thị các sản phẩm người dùng đã thêm vào giỏ hàng.



Hình 4.5 Trang giỏ hàng trên desktop

4.1.6 Form thanh toán

Ở trang giỏ hàng bấm vào thanh toán sẽ hiển thị ra form như sau để người dùng nhập thông tin thanh toán.

The screenshot shows a payment confirmation page titled "Xác nhận đơn hàng". It displays a summary of the purchase: "Laptop ASUS Vivobook 14 OLED A1405ZA KM264W x 1" with a price of "14,990,000đ". The total amount is highlighted in red as "14,990,000đ". Below the summary, there are fields for "Địa chỉ giao hàng" (Delivery address) and "Số điện thoại" (Phone number), both with placeholder text "Nhập địa chỉ..." and "Nhập số điện thoại...". A dropdown menu for "Phương thức thanh toán" (Payment method) is set to "Thanh toán chuyển khoản (PayOS)". At the bottom, there are two buttons: "Hủy" (Cancel) and a red "Xác nhận đặt hàng" (Confirm order) button. The background shows a blurred view of the shopping cart and search bar.

Hình 4.6 Form thanh toán

4.1.7 Trang đơn hàng

Là trang hiển thị lịch sử các đơn hàng của người dùng gồm có các thông tin: Mã đơn, ngày đặt, các sản phẩm, trạng thái số tiền.

The screenshot shows the desktop version of the order history page. On the left, there is a sidebar with a user profile picture and the title "Super Admin". Below it are several navigation links: "Thông tin tài khoản", "Quản lý đơn hàng", "Sản phẩm đã xem", "Đổi mật khẩu", and "Đăng xuất". The main content area is titled "Đơn hàng của tôi". It lists two orders. Order #1750081351908 was placed on June 16, 2025, at 20:42:31. It includes a Macbook Air M2 10GPU 8GB 512GB x 2 and a Logitech G102 LightSync mouse, totaling 64,980,000đ. Order #1749976347249 was placed on June 15, 2025, at 15:32:27. It includes a MacBook Pro M1, totaling 2,000đ. Each order row has a status indicator: a green circle with a checkmark for "Đã giao và thanh toán" (Shipped and paid) and a red circle with an X for "Đã huỷ" (Cancelled). There are also "Xem chi tiết" (View details) and "Tổng cộng" (Total) buttons.

Hình 4.7 Trang đơn hàng trên desktop

Xây dựng website bán hàng đa nền tảng

Trang chi tiết đơn hàng là trang sẽ hiển thị ra khi ấn và nút “Xem chi tiết” có ở từng đơn hàng ở danh sách đơn hàng thì sẽ đến trang chi tiết cho đơn hàng đó. Ở trang này sẽ hiển thị thông tin rõ hơn về đơn hàng và có thể thêm đánh giá nếu đã nhận được sản phẩm và thanh toán rồi.

The screenshot shows the TechShop website's order detail page. At the top, there is a navigation bar with the logo, search bar, and various links like LH: 1900.9999, Showroom, Order, Cart, and Profile. On the left, a sidebar for 'Super Admin' shows options: Thông tin tài khoản (selected), Quản lý đơn hàng, Sản phẩm đã xem, Đổi mật khẩu, and Đăng xuất. The main content area displays the order details for #1750081351908, which was placed on 20:42:31 16/6/2025. The status is 'Đã giao và thanh toán'. It lists two items: 'Macbook Air M2 10GPU 8GB 512GB' (64.980.000đ) and 'Chuột Logitech G102 LightSync' (405.000đ). A total amount of 65.385.000đ is shown at the bottom. There are 'Viết đánh giá' buttons next to each item.

Hình 4.8 Trang chi tiết đơn hàng trên desktop

4.1.8 Trang thông tin cá nhân

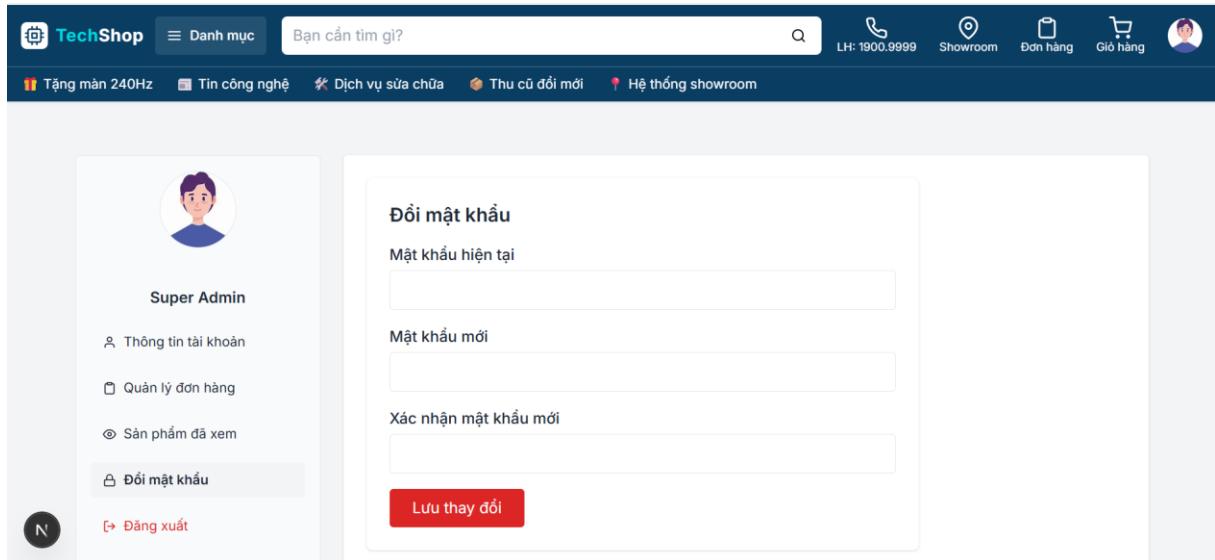
Trang để người dùng Quản lý thông tin cá nhân bản thân người dùng có thể cập nhật ảnh đại diện, thông tin cá nhân khác ở trang này.

The screenshot shows the TechShop website's personal information management page. The sidebar for 'Super Admin' has 'Thông tin tài khoản' selected. The main content area shows a circular profile picture of the user. Below it, there are fields for 'Họ Tên' (Super Admin), 'Giới tính' (radio buttons for Nam and Nữ, with Nam selected), 'Số điện thoại' (0363821946), and 'Email' (empty field). There is also a 'Thay đổi' button.

Hình 4.9 Trang Quản lý thông tin cá nhân trên desktop

4.1.9 Trang đổi mật khẩu

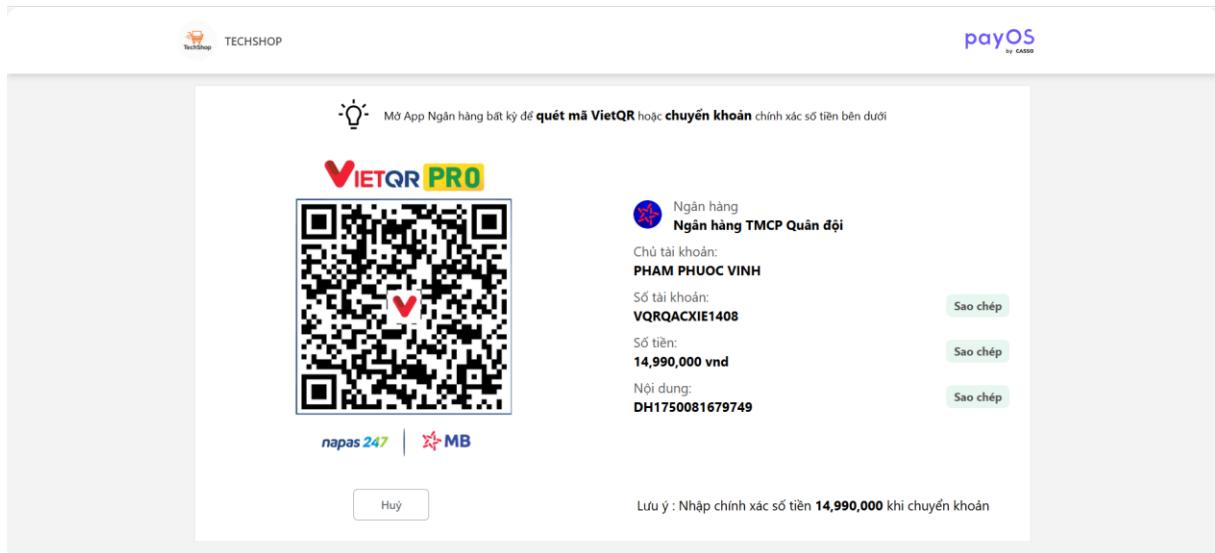
Người dùng có thể đổi lại mật khẩu bản thân ở trang này.



Hình 4.10 Trang đổi mật khẩu trên desktop

4.1.10 Thanh toán trực tuyến

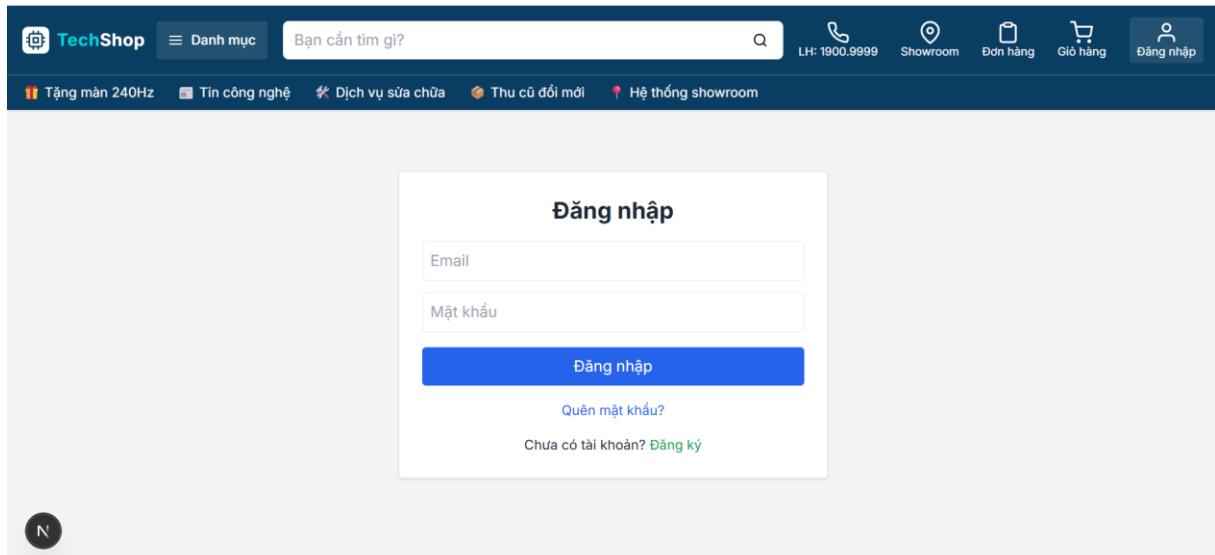
Trang khi thanh toán trực tuyến qua PayOS.



Hình 4.11 Trang thanh toán trực tuyến

4.1.11 Trang đăng nhập

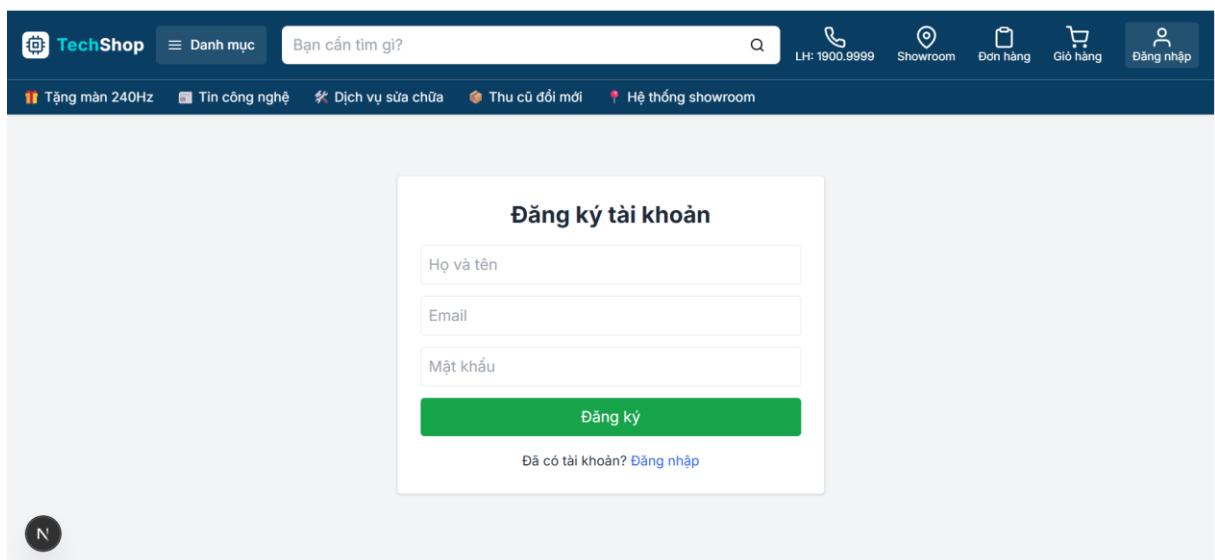
Cho phép người dùng nhập thông tin và đăng nhập vào hệ thống.



Hình 4.12 Trang đăng nhập trên desktop

4.1.12 Trang đăng ký

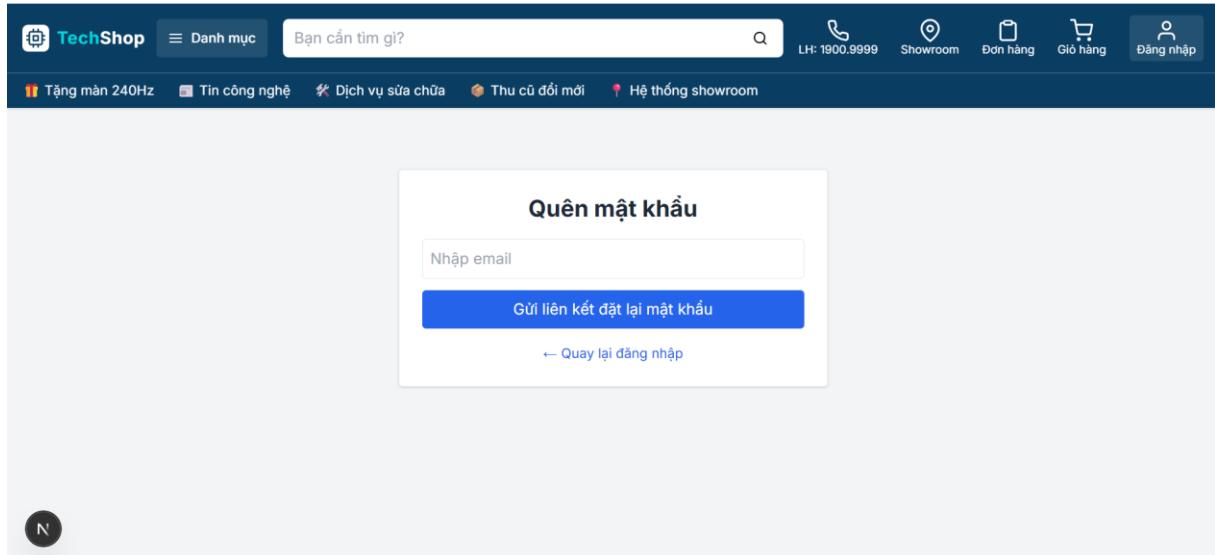
Cho phép người dùng đăng ký tài khoản mới bằng email.



Hình 4.13 Trang đăng ký trên desktop

4.1.13 Trang quên mật khẩu

Trang giúp người dùng lấy lại mật khẩu nếu quên.

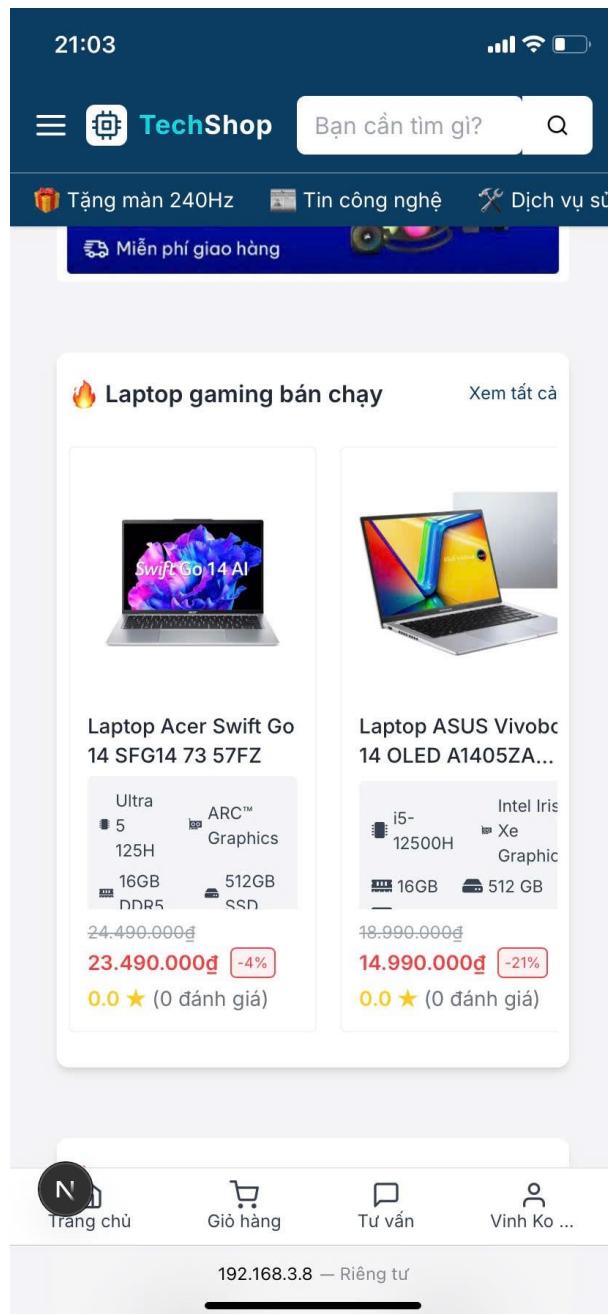


Hình 4.14 Trang lấy lại mật khẩu

4.2 Giao diện trên điện thoại

4.2.1 Trang chủ

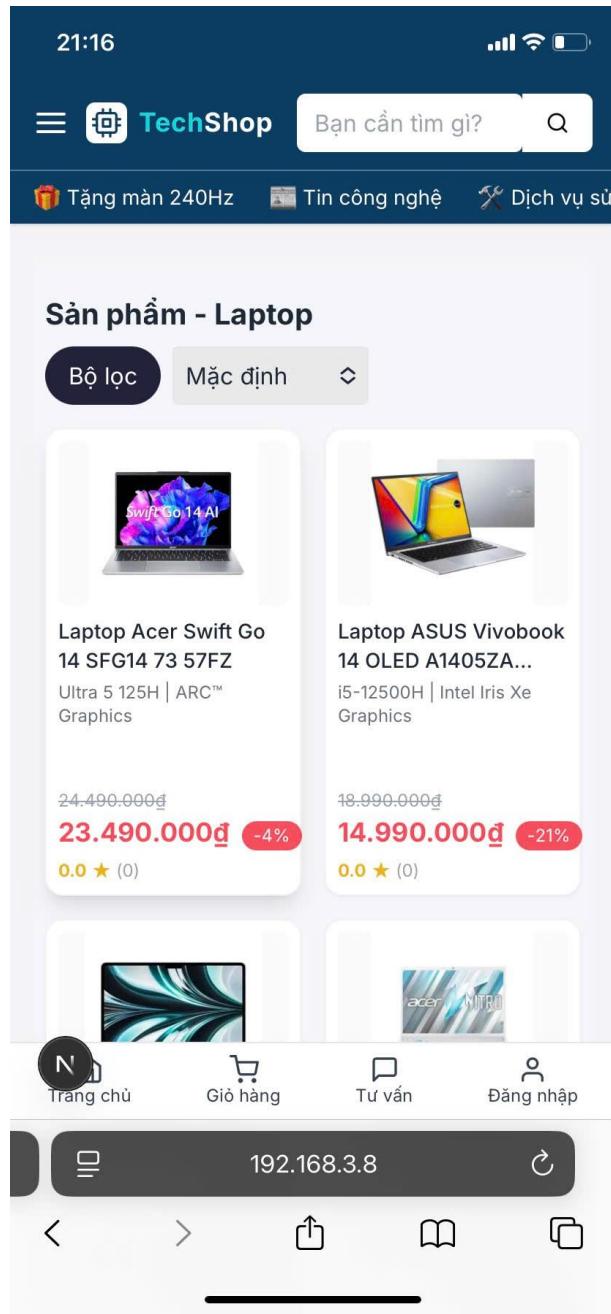
Trang chủ đóng vai trò là nơi tập trung các chức năng chính của website. Người dùng có thể dễ dàng tìm kiếm sản phẩm, truy cập các danh mục sản phẩm, quản lý đơn hàng, giỏ hàng, tài khoản cá nhân và xem nhanh các sản phẩm bán chạy ngay trên trang chủ. Được thiết kế để tương thích với trình duyệt trên điện thoại.



Hình 4.15 Trang chủ trên điện thoại

4.2.2 Trang xem sản phẩm theo danh mục

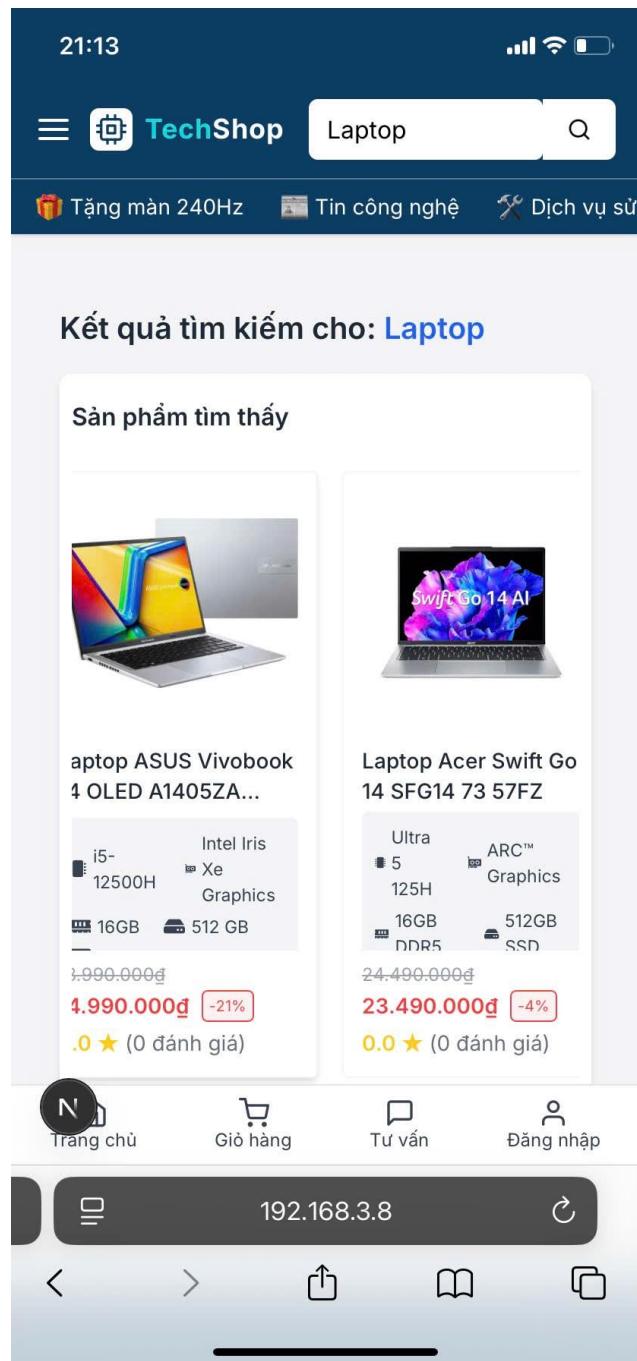
Trang xem theo danh mục sản phẩm là trang hiển thị sản phẩm theo các danh mục của sản phẩm. Ngoài ra trang còn có bộ lọc giúp người dùng lọc sản phẩm theo các tiêu chí như giá, thương hiệu,... Được thiết kế để tương thích với trình duyệt trên điện thoại.



Hình 4.16 Trang xem theo danh mục trên điện thoại

4.2.3 Trang tìm kiếm sản phẩm

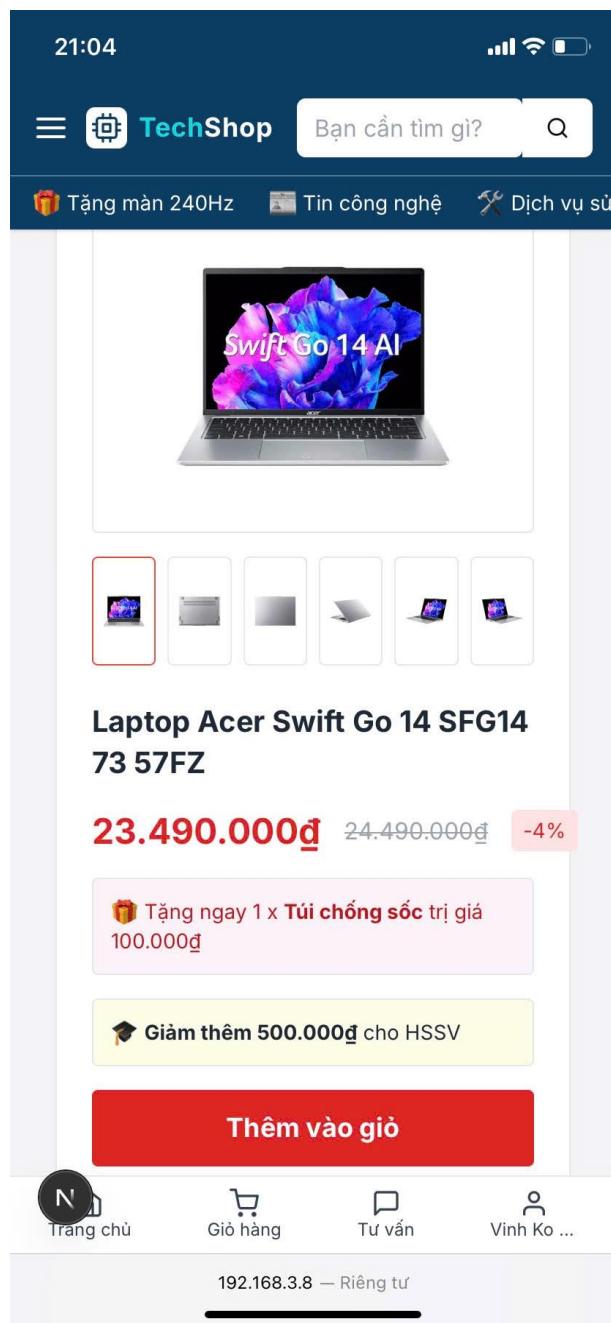
Là trang trả kết quả tìm kiếm về khi dùng chức năng tìm kiếm trên thanh công cụ trên cùng của website.



Hình 4.17 Trang tìm kiếm sản phẩm trên điện thoại

4.2.4 Trang chi tiết sản phẩm

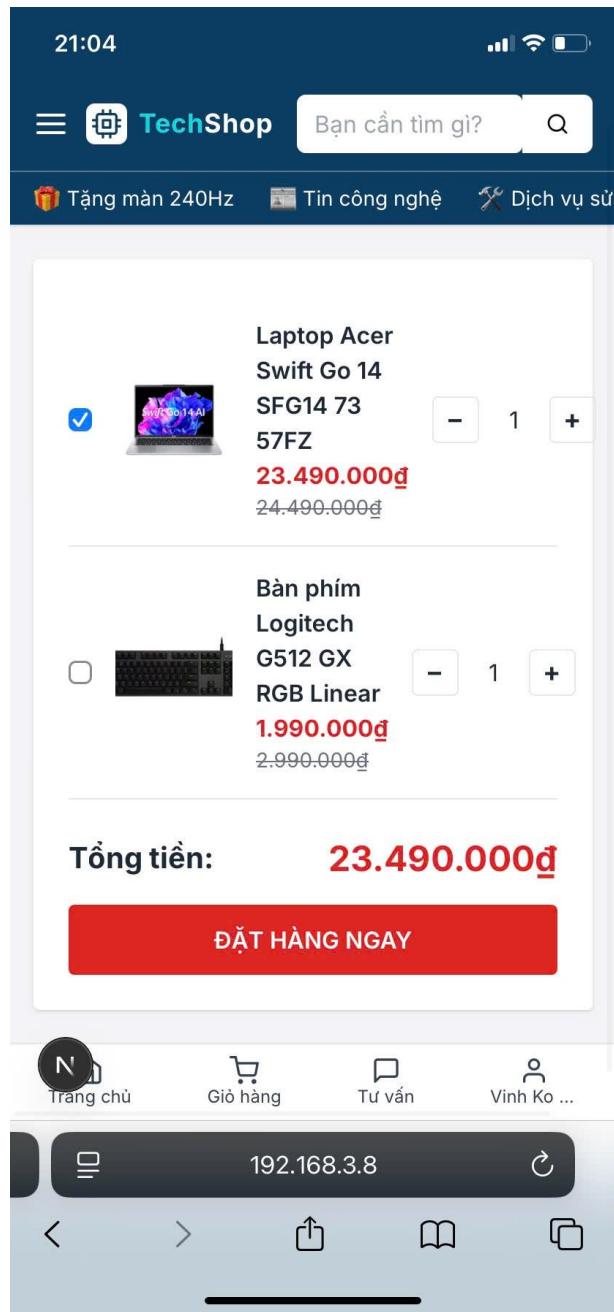
Trang chi tiết sản phẩm hiển thị chi tiết sản phẩm khi người dùng chọn vào một sản phẩm. Ở trang này sẽ hiển thị: tên các hình ảnh sản phẩm, giá tiền, cấu hình, sản phẩm mua kèm, sản phẩm liên quan, đánh giá của những người đã mua sản phẩm này. Nếu muốn mua người dùng có thể thêm sản phẩm vào giỏ hàng. Được thiết kế để tương thích với trình duyệt trên điện thoại.



Hình 4.18 Trang chi tiết sản phẩm trên điện thoại

4.2.5 Trang giỏ hàng

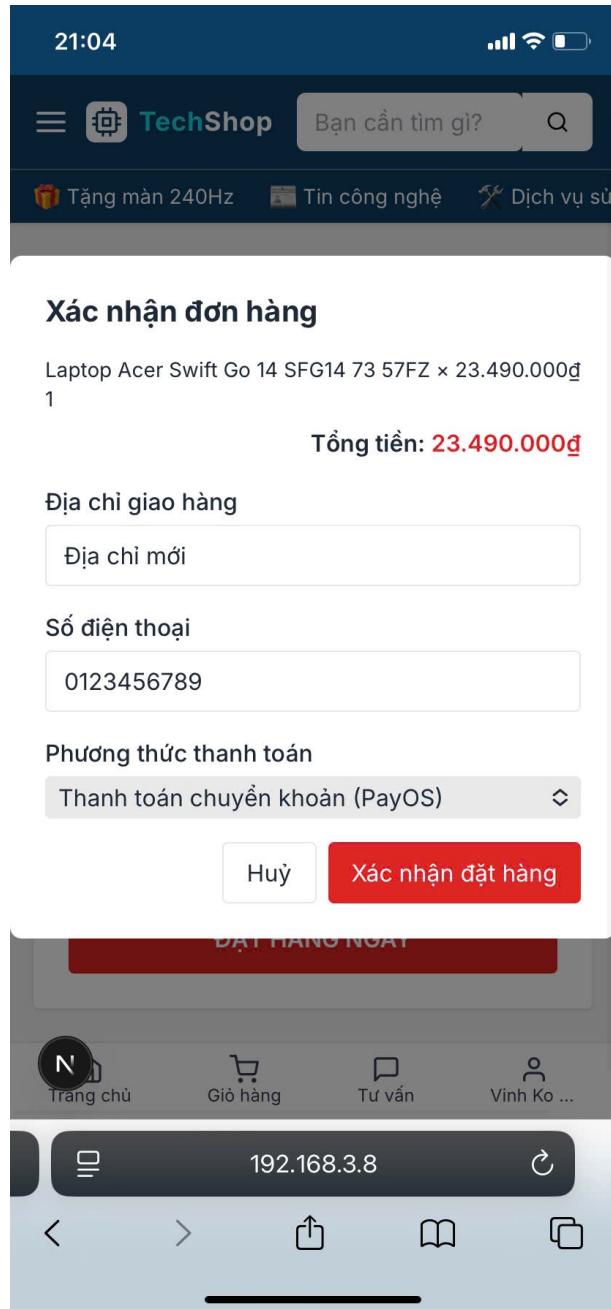
Là trang hiển thị các sản phẩm người dùng đã thêm vào giỏ hàng mà chưa thanh toán ở đây người dùng có thể chọn sản phẩm và sản phẩm mình muốn thanh toán để tiến hành thanh toán. Được thiết kế để phù hợp với trình duyệt trên điện thoại.



Hình 4.19 Trang giỏ hàng trên điện thoại

4.2.6 Form thanh toán

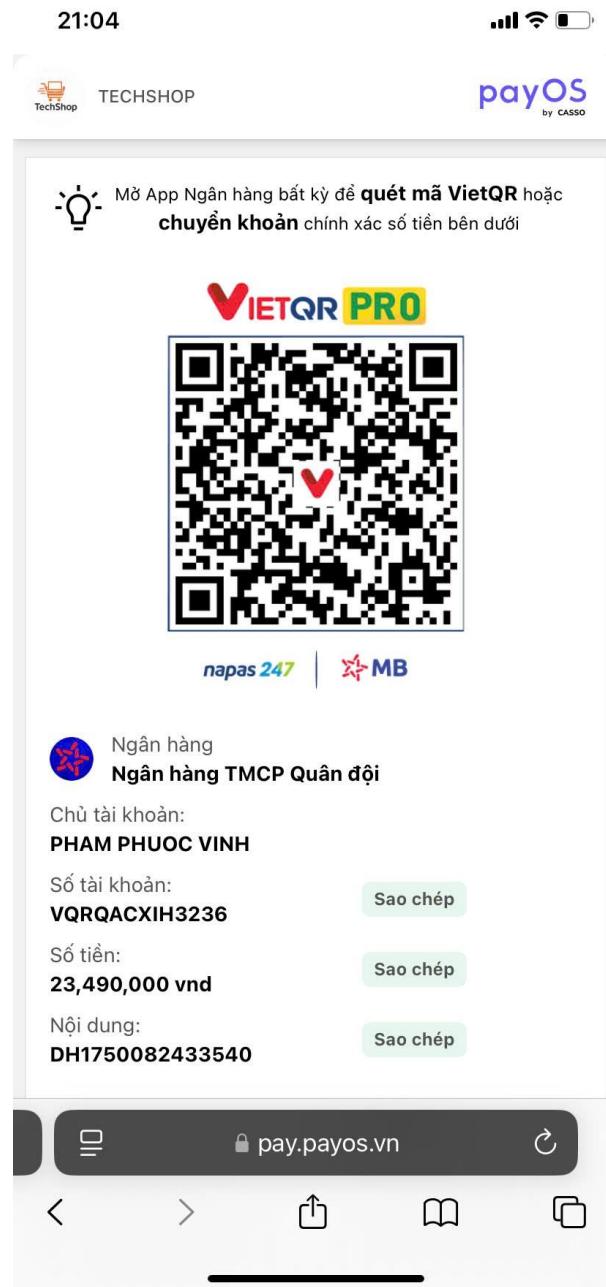
Ở trang giỏ hàng bấm vào thanh toán sẽ hiển thị ra form như sau để người dùng nhập thông tin thanh toán. Được thiết kế để tương thích với trình duyệt trên điện thoại.



Hình 4.20 Form thanh toán trên điện thoại

4.2.7 Thanh toán trực tuyến

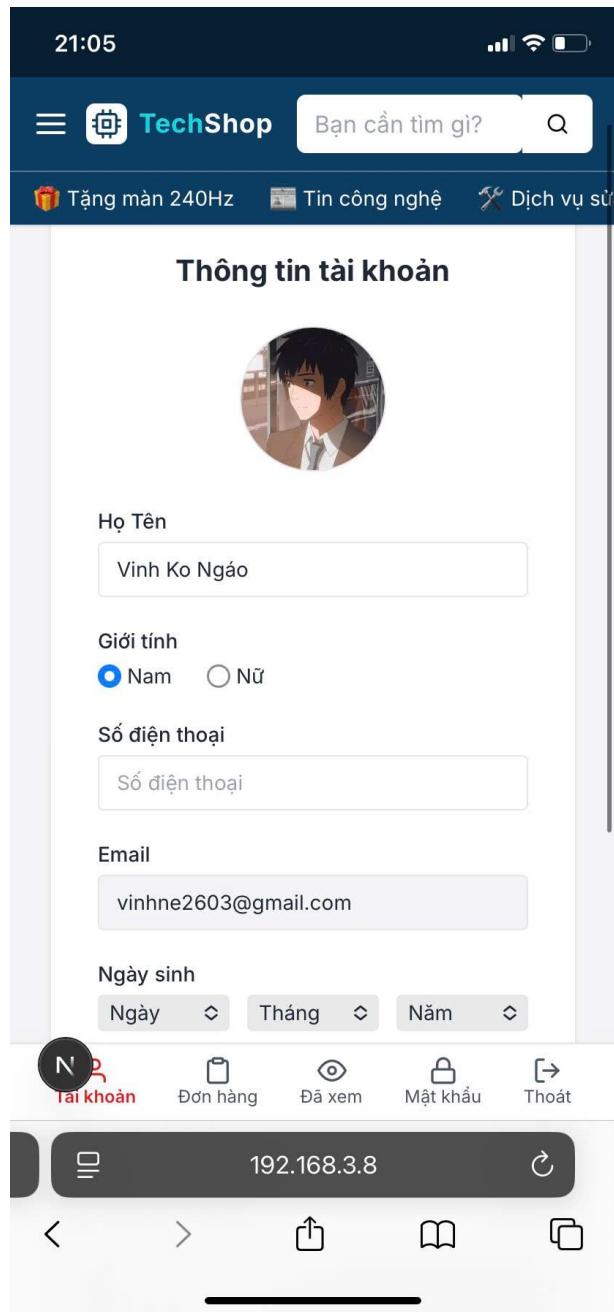
Trang thanh toán trực tuyến qua PayOS.



Hình 4.21 Thanh toán trực tuyến trên điện thoại

4.2.8 Thông tin tài khoản

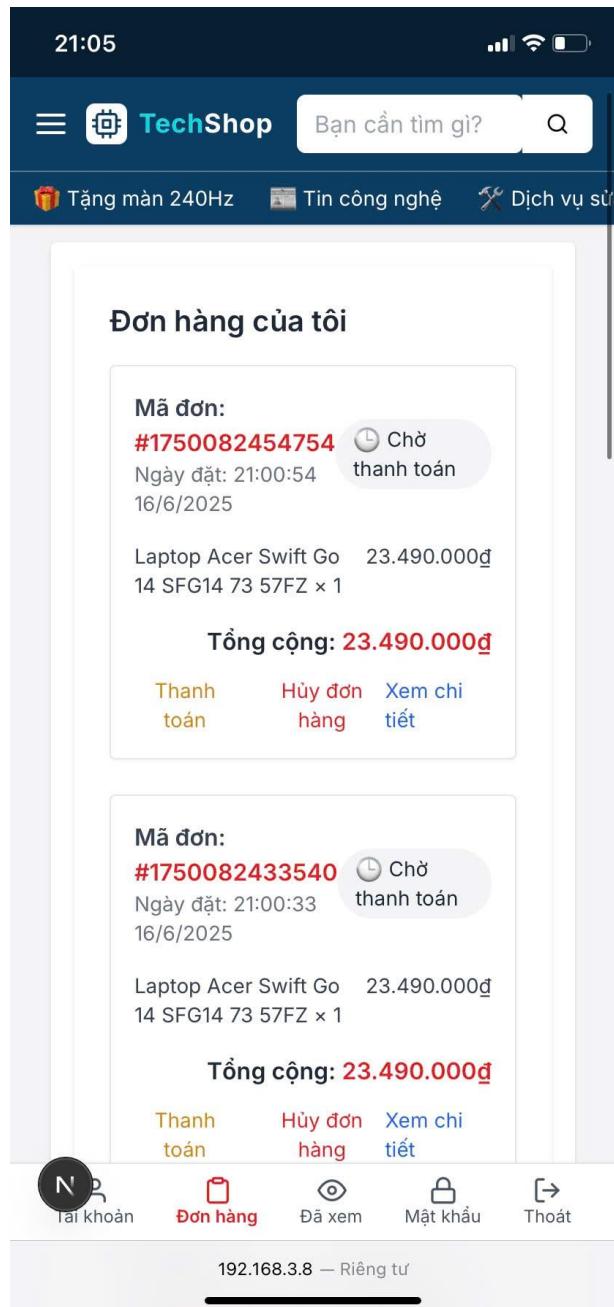
Trang để người dùng Quản lý thông tin cá nhân bản thân người dùng có thể cập nhật ảnh đại diện, thông tin cá nhân khác ở trang này. Được thiết kế để tương thích với trình duyệt trên điện thoại.



Hình 4.22 Trang thông tin tài khoản trên điện thoại

4.2.9 Thông tin đơn hàng

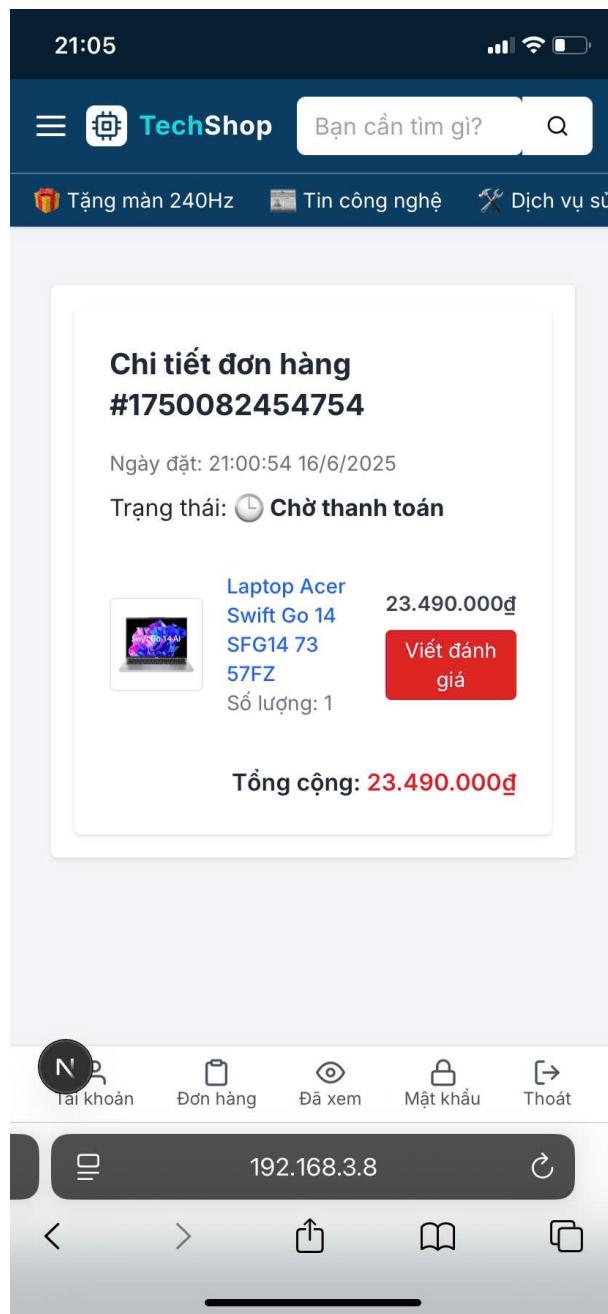
Là trang hiển thị lịch sử các đơn hàng của người dùng gồm có các thông tin: Mã đơn, ngày đặt, các sản phẩm, trạng thái số tiền. Được tối ưu hóa giao diện trên điện thoại.



Hình 4.23 Trang danh sách đơn hàng trên điện thoại

4.2.10 Trang chi tiết đơn hàng

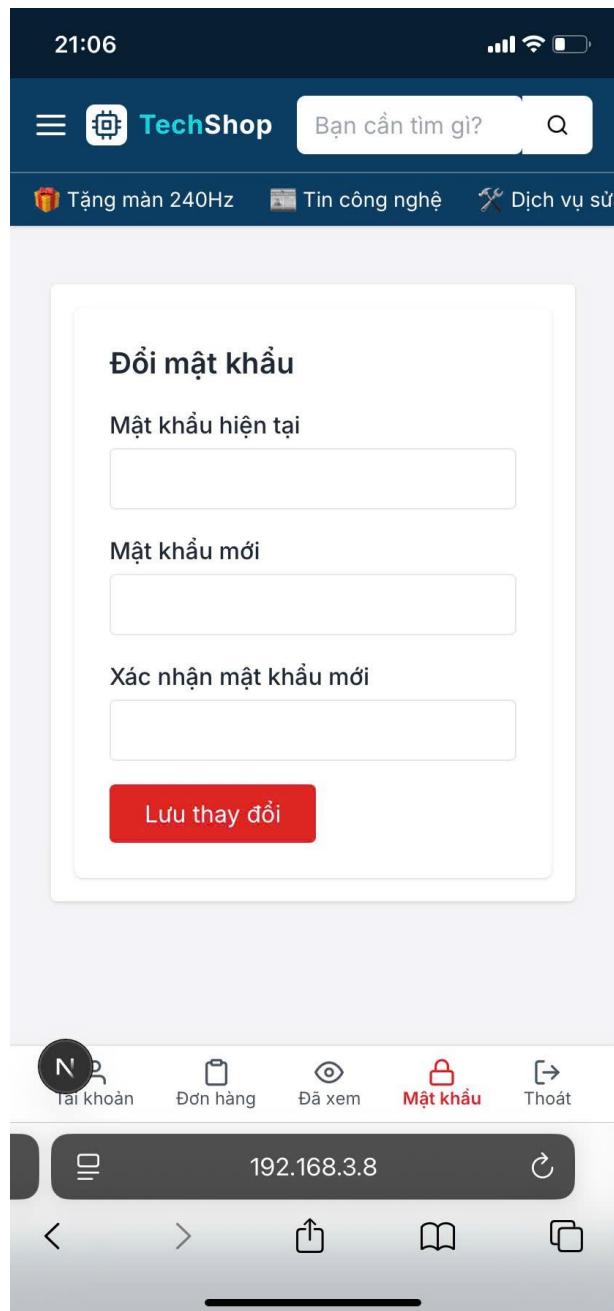
Trang chi tiết đơn hàng là trang sẽ hiển thị ra khi ấn và nút “Xem chi tiết” có ở từng đơn hàng ở danh sách đơn hàng thì sẽ đến trang chi tiết cho đơn hàng đó. Ở trang này sẽ hiển thị thông tin rõ hơn về đơn hàng và có thể thêm đánh giá nếu đã nhận được sản phẩm và thanh toán rồi. Được tối ưu hóa cho giao diện điện thoại.



Hình 4.24 Trang chi tiết đơn hàng trên điện thoại

4.2.11 Trang đổi mật khẩu

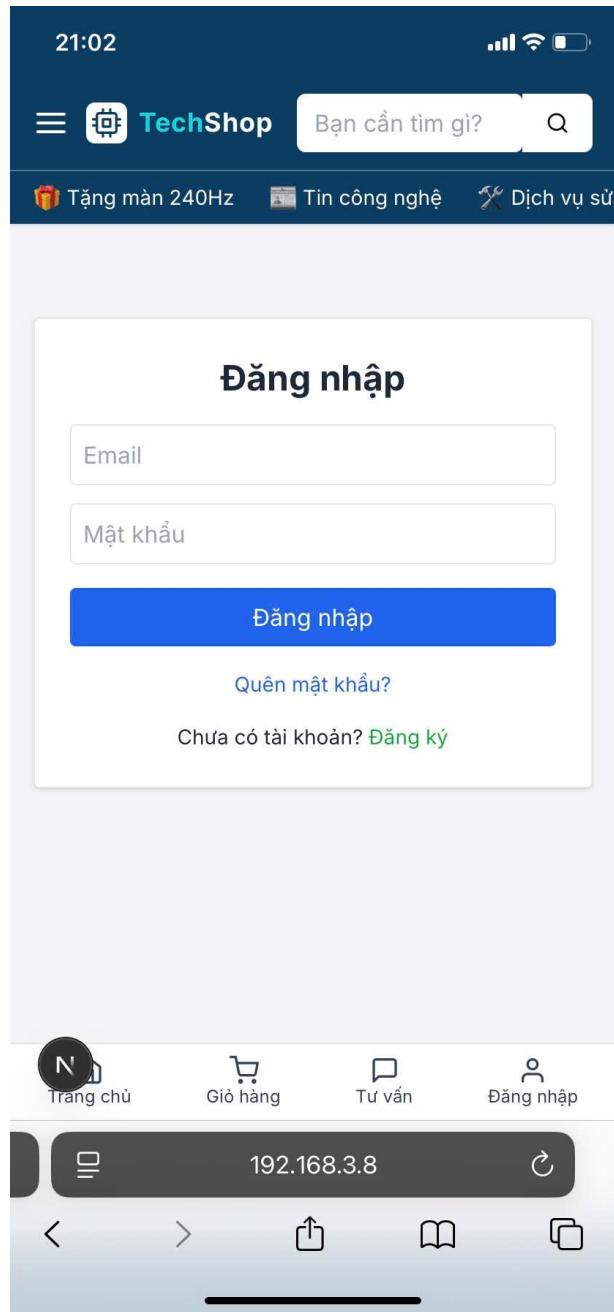
Giúp người dùng có thể đổi mật khẩu.



Hình 4.25 Trang thay đổi mật khẩu trên điện thoại

4.2.12 Trang đăng nhập

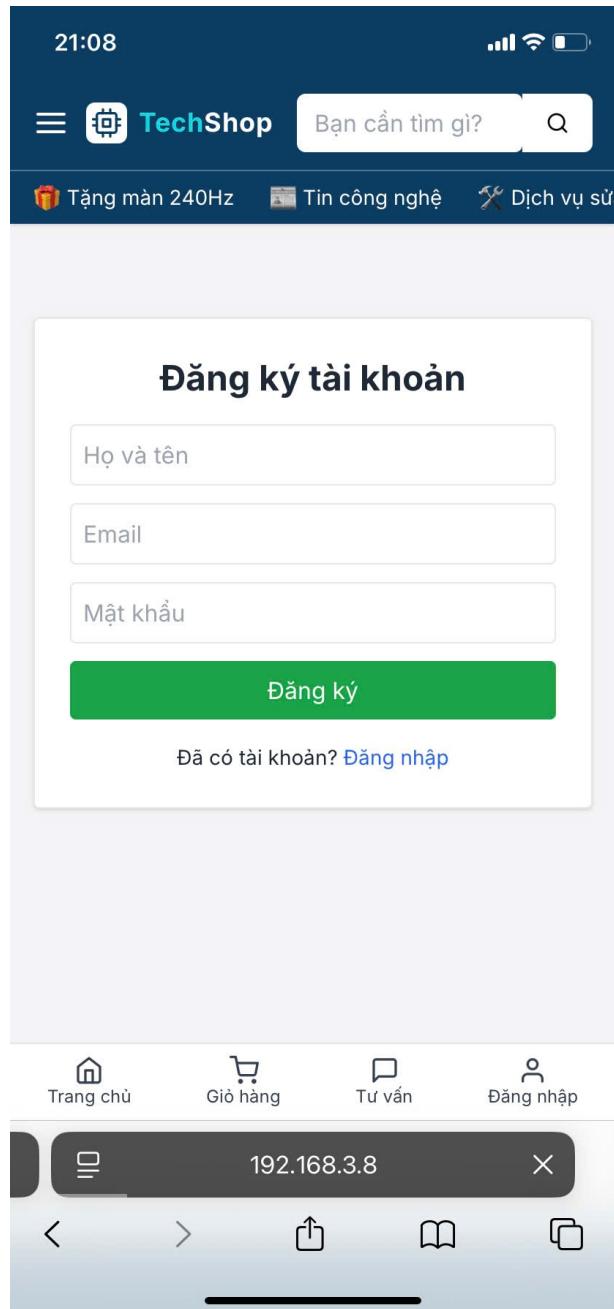
Trang giúp người dùng có thể đăng nhập vào hệ thống.



Hình 4.26 Trang đăng nhập trên điện thoại

4.2.13 Trang đăng ký

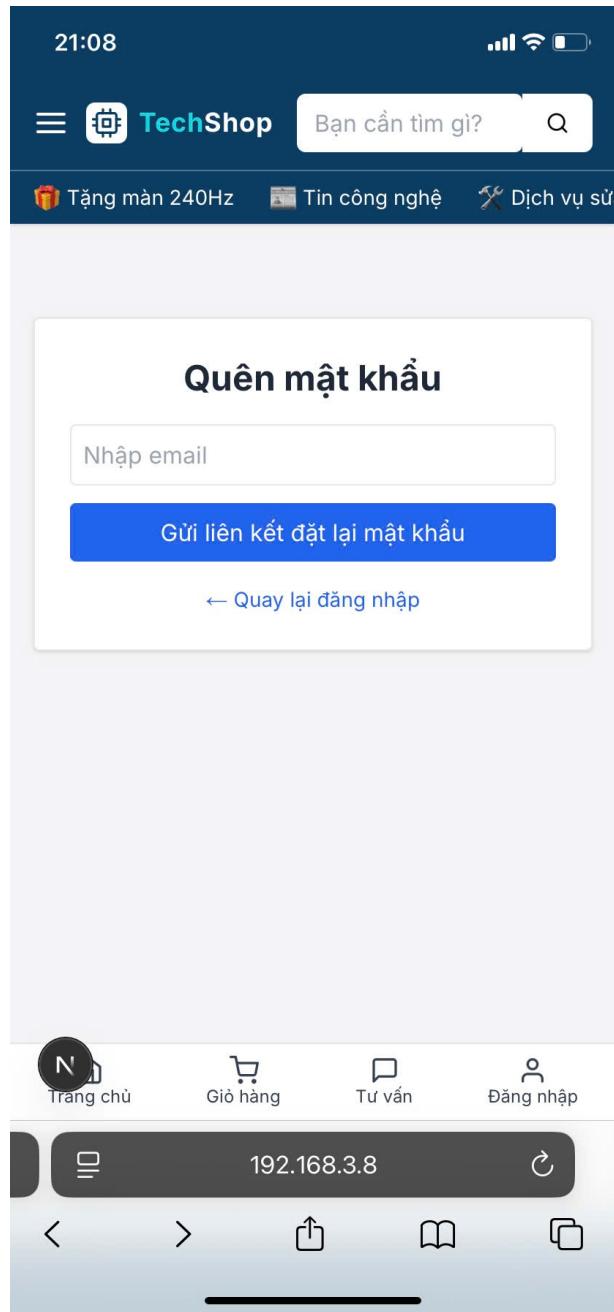
Trang giúp người dùng đăng ký tài khoản mới.



Hình 4.27 Trang đăng ký trên điện thoại

4.2.14 Trang quên mật khẩu

Trang giúp người dùng lấy lại mật khẩu nếu quên.

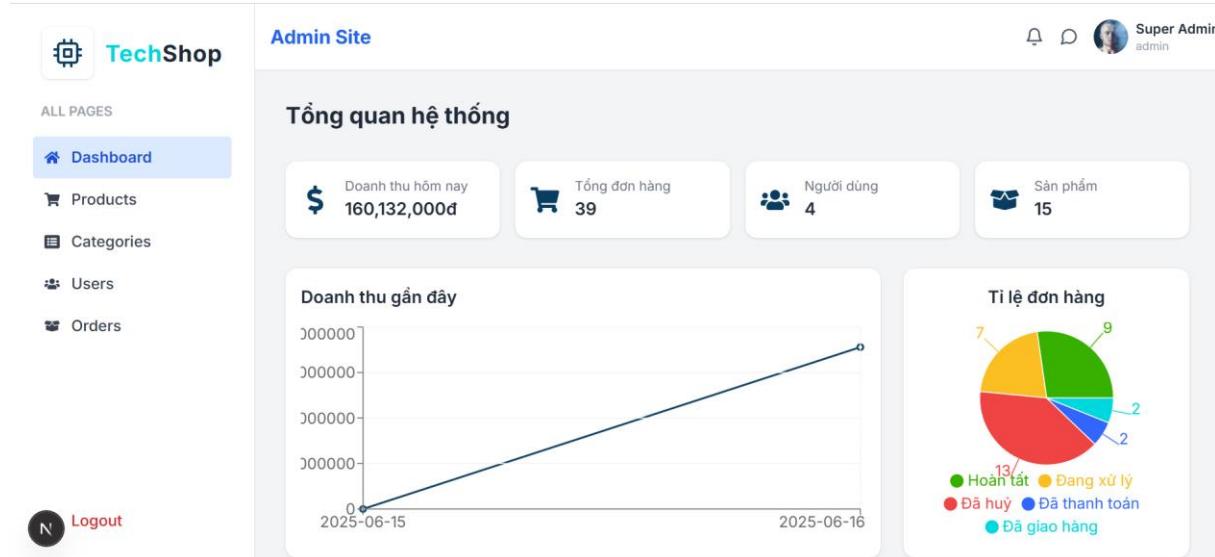


Hình 4.28 Trang quên mật khẩu trên điện thoại

4.3 Giao diện quản trị

4.3.1 Trang tổng quan

Trang tổng quan trong giao diện quản trị cung cấp cái nhìn tổng thể về hoạt động của hệ thống. Tại đây, quản trị viên có thể nhanh chóng theo dõi các thông tin quan trọng như tổng doanh thu, số lượng đơn hàng, người dùng, và sản phẩm hiện có. Ngoài ra, trang còn hiển thị biểu đồ doanh thu gần đây giúp đánh giá xu hướng kinh doanh. Nhờ đó, quản trị viên dễ dàng giám sát tình hình hoạt động và đưa ra các quyết định quản lý kịp thời, hiệu quả.



Hình 4.29 Trang tổng quan

4.3.2 Trang quản lý sản phẩm

Trang quản lý sản phẩm cho phép quản trị viên theo dõi và kiểm soát toàn bộ danh mục sản phẩm trên hệ thống. Giao diện hiển thị danh sách sản phẩm với các thông tin quan trọng như hình ảnh, tên sản phẩm, giá bán, tồn kho, danh mục, trạng thái hoạt động và các thao tác quản lý như: chỉnh sửa, ẩn/hiện, xóa. Nhờ đó, quản trị viên có thể dễ dàng cập nhật thông tin, điều chỉnh số lượng tồn kho hoặc thay đổi trạng thái hiển thị của sản phẩm một cách thuận tiện và nhanh chóng, đảm bảo việc quản lý sản phẩm luôn hiệu quả và chính xác.

The screenshot shows the 'Admin Site' interface for 'TechShop'. On the left is a sidebar with 'ALL PAGES' and links to 'Dashboard', 'Products' (which is highlighted), 'Categories', 'Users', and 'Orders'. On the right is a main content area titled 'Admin Site' with a header for 'Super Admin admin'. It displays a table of products:

Ảnh	Tên sản phẩm	Giá bán	Tồn kho	Danh mục	Trạng thái	Thao tác
	Laptop Acer Swift Go 14 SFG14 73 57FZ	23,490,000đ	199	laptop	Hiện	⋮
	PC GVN Intel i5-12400F/ VGA RTX 4060	18,690,000đ	50	laptop	Hiện	⋮
	PC GVN AMD R5-5600X/ VGA RX 6600	16,490,000đ	100	laptop	Hiện	⋮
	PC GVN Intel i5-12400F/ VGA RX 7600	16,990,000đ	200	laptop	Hiện	⋮
	PC GVN AMD R5-5500/ VGA RTX 3050	14,990,000đ	150	laptop	Hiện	⋮
	Laptop ASUS Vivobook 14 OLED A1405ZA	14,990,000đ	115	laptop	Hiện	⋮

Hình 4.30 Trang quản lý sản phẩm

4.3.3 Trang Quản lý người dùng

Trang quản lý người dùng cho phép quản trị viên theo dõi và kiểm soát toàn bộ tài khoản trên hệ thống. Giao diện hiển thị danh sách người dùng với các thông tin như avatar, tên, email, số điện thoại, vai trò (user/admin), trạng thái hoạt động và các thao tác quản lý. Quản trị viên có thể dễ dàng tìm kiếm, thêm mới, chỉnh sửa hoặc thay đổi trạng thái của người dùng, đảm bảo hệ thống luôn an toàn và cập nhật đầy đủ thông tin thành viên.

The screenshot shows the Admin Site interface for TechShop. On the left, there's a sidebar with 'ALL PAGES' and links to 'Dashboard', 'Products', 'Categories', 'Users' (which is highlighted in blue), and 'Orders'. At the bottom of the sidebar is a 'Logout' button. The main area is titled 'Admin Site' and 'Quản lý người dùng'. It features a search bar with placeholder 'Tim kiếm tên hoặc email...'. Below the search bar is a table with columns: Avatar, Tên, Email, Số điện thoại, Vai trò, Trạng thái, and Thao tác. The table contains four rows of user data:

Avatar	Tên	Email	Số điện thoại	Vai trò	Trạng thái	Thao tác
	Phạm Phước Vinh	pvinh@example.com	-	user	Hoạt động	...
	Super Admin	admin@example.com	-	admin	Hoạt động	...
	Vinh Ko Ngáo	vinhne2603@gmail.com	0123456789	user	Hoạt động	...
	Vinh abc	vinh260104@gmail.com	0363821946	user	Hoạt động	...

Hình 4.31 Trang Quản lý người dùng

4.3.4 Trang Quản lý danh mục

Trang quản lý danh mục giúp quản trị viên tổ chức và kiểm soát các nhóm sản phẩm trên hệ thống. Tại đây, quản trị viên có thể xem danh sách các danh mục hiện có, tìm kiếm, thêm mới, chỉnh sửa hoặc xóa danh mục sản phẩm. Việc quản lý danh mục hiệu quả giúp việc phân loại sản phẩm rõ ràng, hỗ trợ người dùng dễ dàng tìm kiếm và lựa chọn sản phẩm khi mua sắm trên website.

The screenshot shows the Admin Site interface for managing categories. The left sidebar includes links for Dashboard, Products, Categories (selected), Users, and Orders. The main area is titled "Quản lý danh mục" and features a search bar. A button "+ Thêm danh mục" is visible. The table lists categories with columns for Name, Number of products, Status, and Actions (Edit, Delete).

Tên danh mục	Số sản phẩm	Trạng thái	Thao tác
laptop	7	Hiện	[Edit] [Delete]
màn hình	1	Hiện	[Edit] [Delete]
bàn phím	1	Hiện	[Edit] [Delete]
chuột	1	Hiện	[Edit] [Delete]
pc	4	Hiện	[Edit] [Delete]

Hình 4.32 Trang Quản lý danh mục

4.3.5 Trang Quản lý đơn hàng

Trang quản lý đơn hàng cho phép quản trị viên theo dõi và xử lý các đơn đặt hàng của khách hàng trên hệ thống. Giao diện hiển thị danh sách đơn hàng với các thông tin như tên khách hàng, ngày đặt, tổng tiền, số điện thoại, địa chỉ, trạng thái đơn hàng và các thao tác quản lý (xem chi tiết, cập nhật, xóa đơn hàng). Quản trị viên có thể tìm kiếm, kiểm tra trạng thái và xử lý đơn hàng một cách dễ dàng, góp phần nâng cao hiệu quả quản lý và phục vụ khách hàng.

The screenshot shows the Admin Site interface for TechShop. On the left, there's a sidebar with 'ALL PAGES' and links to Dashboard, Products, Categories, Users, and Orders (which is highlighted). At the bottom of the sidebar is a 'Logout' button. The main area is titled 'Admin Site' and 'Quản lý đơn hàng'. It features a search bar with placeholder text 'Tim tên khách, trạng thái, sđt, địa c' and a table with columns: Tên khách, Ngày đặt, Tổng tiền, Số điện thoại, Địa chỉ, Trạng thái, and Thao tác. Two orders are listed:

Tên khách	Ngày đặt	Tổng tiền	Số điện thoại	Địa chỉ	Trạng thái	Thao tác
Vinh Ko Ngáo	6/16/2025, 9:00:54 PM	23,490,000đ	0123456789	Địa chỉ mới	Chờ thanh toán	View Delete
Vinh Ko Ngáo	6/16/2025, 9:00:33 PM	23,490,000đ	0123456789	Địa chỉ mới	Chờ thanh toán	View Delete

Hình 4.33 Trang Quản lý đơn hàng

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Về thực hiện, em đã xây dựng được một hệ thống bán hàng đa nền tảng gồm website người dùng, trang quản trị (admin) và giao diện ứng dụng di động. Hệ thống đáp ứng được các chức năng cốt lõi như hiển thị sản phẩm, giỏ hàng, đặt hàng và quản lý đơn hàng, góp phần hỗ trợ hoạt động kinh doanh trực tuyến một cách thuận tiện. Ngoài ra, phần giao diện điện thoại được thiết kế để mô phỏng trải nghiệm người dùng, phục vụ cho mục đích trình bày và mở rộng trong tương lai.

Về kiến thức, quá trình triển khai giúp em nắm bắt và vận dụng thành thạo các công nghệ hiện đại như NestJS (backend), ReactJS (frontend) và React Native (giao diện điện thoại). Đồng thời, em cũng tích lũy được nhiều kỹ năng quan trọng trong quá trình phân tích yêu cầu, thiết kế hệ thống, tổ chức mã nguồn và kiểm thử tính năng. Những trải nghiệm này là nền tảng quý giá để em tự tin tham gia và phát triển các dự án phần mềm chuyên nghiệp sau này.

5.2 Hướng phát triển

Trong thời gian tới, hệ thống bán hàng có thể được mở rộng và nâng cấp với các tính năng hiện đại, đáp ứng tốt hơn nhu cầu thực tế của người dùng và doanh nghiệp. Một số hướng phát triển tiềm năng bao gồm:

- Phát triển hoàn thiện ứng dụng di động: Kết nối giao diện điện thoại với backend API để cho phép người dùng thực hiện thao tác mua hàng thực tế.
- Tối ưu hiệu năng và trải nghiệm người dùng: Điều chỉnh cấu trúc mã nguồn, tối ưu tốc độ tải trang, cải thiện giao diện.
- Bổ sung tính năng nâng cao: Cho phép đánh giá sản phẩm, gợi ý sản phẩm liên quan, mã giảm giá và hệ thống khuyến mãi tự động.
- Hệ thống báo cáo thông minh: Phát triển chức năng phân tích và thống kê chi tiết cho quản trị viên theo thời gian, doanh số, sản phẩm bán chạy, ...

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Amazon Web Services,"What is a RESTful API?"*AWS*. [Trực tuyến]. Có sẵn tại: <https://aws.amazon.com/what-is/RESTful-api/>. [Truy cập: 8/4/2025].
- [2] M. Hann,"What is Node.js?,"*freeCodeCamp*. [Trực tuyến]. Có sẵn tại: <https://www.freecodecamp.org/news/what-is-node-js/>. [Truy cập: 8/4/2025].
- [3] TopDev,"Node.js là gì? Tại sao Node.js lại được ưa chuộng đến vậy?,"*TopDev Blog*. [Trực tuyến]. Có sẵn tại: <https://topdev.vn/blog/node-js-la-gi/>. [Truy cập: 9/4/2025].
- [4] NestJS,"Documentation,"*NestJS Docs*. [Trực tuyến]. Có sẵn tại: <https://docs.nestjs.com/>. [Truy cập: 9/4/2025].
- [5] 200Lab,"NextJS là gì? Tìm hiểu NextJS từ A đến Z,"*200Lab Blog*. [Trực tuyến]. Có sẵn tại: <https://200lab.io/blog/nextjs-la-gi>. [Truy cập: 10/4/2025].
- [6] [12] Amazon Web Services,"NoSQL là gì?,"*AWS Vietnam*. [Trực tuyến]. Có sẵn tại: <https://aws.amazon.com/vi/nosql/>. [Truy cập: 11/4/2025].
- [7] Mắt Bão,"MongoDB là gì? Tính năng nổi bật từ MongoDB bạn cần biết,"*Mắt Bão Wiki*. [Trực tuyến]. Có sẵn tại: <https://wiki.matbao.net/mongodb-la-gi-tinh-nang-noi-bat-tu-mongodb-ban-can-biet/>. [Truy cập: 12/4/2025].

PHỤ LỤC