**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**



**ISO 9001:2015**

**NGUYỄN TÍN THÀNH**

**XÂY DỰNG WEBSITE BÁN LAPTOP**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

VĨNH LONG**, NĂM 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

**XÂY DỰNG WEBSITE BÁN LAPTOP**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Sinh viên thực hiện: **NGUYỄN TÍN THÀNH**

Lớp: **DA21TTA**

Mã số sinh viên: **110121104**

Giảng viên hướng dẫn: **Ths. VÕ THÀNH C**

VĨNH LONG**, NĂM 2025**

LỜI CAM ĐOAN

Tôi xin cam đoan rằng toàn bộ nội dung của báo cáo này là kết quả của quá  
trình học tập, nghiên cứu và thực hiện đồ án một cách nghiêm túc dưới sự hướng dẫn của giảng viên Võ Thành C. Các tài liệu, dữ liệu và hình ảnh sử dụng trong báo cáo đều được trích dẫn rõ ràng nguồn gốc.

Tôi hoàn toàn chịu trách nhiệm về tính trung thực và nguyên bản của báo cáo.  
Nếu phát hiện có sự sao chép, gian lận hay vi phạm bản quyền, tôi xin hoàn toàn chịu  
trách nhiệm trước hội đồng và quy định của Nhà trường.

Vĩnh Long, ngày ... tháng ... năm 2025

Sinh viên thực hiện

(Ký và ghi rõ họ tên)

**LỜI CẢM ƠN**

Tôi xin gửi lời tri ân sâu sắc đến Trường Đại học Trà Vinh, nơi đã tạo điều kiện thuận lợi, cung cấp môi trường học tập và nghiên cứu lý tưởng giúp tôi có cơ hội phát triển bản thân và hoàn thành dự án này. Trường đã mang lại cho tôi không chỉ những kiến thức chuyên môn quý báu mà còn là động lực để tôi không ngừng học hỏi và phấn đấu.

Tôi đặc biệt biết ơn Thầy Võ Thành C, người đã dành thời gian quý báu để hướng dẫn, chia sẻ kinh nghiệm và đồng hành cùng tôi trong suốt hành trình thực hiện dự án. Những chỉ dẫn tận tâm, sự động viên kịp thời và những lời khuyên quý giá từ cô đã giúp tôi định hình rõ ràng hướng đi cho dự án và vượt qua nhiều khó khăn, thử thách.

Ngoài ra, tôi cũng xin cảm ơn tất cả những người đã trực tiếp hoặc gián tiếp hỗ trợ tôi trong quá trình thực hiện dự án. Những lời gợi ý, góp ý chân thành từ các anh chị, bạn bè, và thầy cô khác đã giúp tôi hoàn thiện dự án này tốt hơn. Dù là sự hỗ trợ nhỏ nhất, đó đều là nguồn động viên to lớn giúp tôi vượt qua những áp lực và khó khăn trong quá trình thực hiện.

Thực hiện một dự án cá nhân là một hành trình không dễ dàng, nhưng nhờ có sự giúp đỡ từ mọi người, tôi đã có thể học hỏi thêm nhiều điều và hoàn thiện kỹ năng của bản thân. Tôi rất tự hào về thành quả mình đạt được, đồng thời trân trọng những mối quan hệ và sự hỗ trợ mà tôi đã nhận được trong suốt thời gian qua.

Một lần nữa, tôi xin chân thành cảm ơn Trường Đại học Trà Vinh, Thầy Võ Thành C và tất cả những người đã đồng hành cùng tôi trong suốt hành trình này. Tôi hy vọng rằng trong tương lai, tôi sẽ tiếp tục nhận được sự hỗ trợ, động viên và góp ý của mọi người để có thể tiến xa hơn trên con đường học tập và sự nghiệp.

Trân trọng.

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN 1](#_Toc206405879)

[1.1. Đặt vấn đề 1](#_Toc206405880)

[1.2. Mục đích 2](#_Toc206405881)

[Đối với người dùng cuối 2](#_Toc206405882)

[Đối với người quản trị 2](#_Toc206405883)

[Mục tiêu kỹ thuật 3](#_Toc206405884)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 4](#_Toc206405885)

[2.1. Kiến trúc Microservices 4](#_Toc206405886)

[2.1.1. Tổng quan 4](#_Toc206405887)

[2.1.2. Ưu điểm 4](#_Toc206405888)

[2.1.3. Nhược điểm 5](#_Toc206405889)

[2.2. Restful API 6](#_Toc206405890)

[2.2.1. Tổng quan 6](#_Toc206405891)

[2.2.2. Đặc điểm 6](#_Toc206405892)

[2.2.3. Ưu điểm 7](#_Toc206405893)

[2.2.4. Nhược điểm 7](#_Toc206405894)

[2.3. Ngôn ngữ backend 8](#_Toc206405895)

[2.3.1. Nodejs 8](#_Toc206405896)

[2.3.2. Express.js 8](#_Toc206405897)

[2.4. Ngôn ngữ frontend 9](#_Toc206405898)

[2.4.1. Reactjs 9](#_Toc206405899)

[2.4.2. Material UI library 10](#_Toc206405900)

[2.5. Cơ sở dữ liệu MySQL 12](#_Toc206405901)

[2.5.1. Khái niệm 12](#_Toc206405902)

[2.5.2. Cấu trúc 12](#_Toc206405903)

[2.6. Docker 13](#_Toc206405904)

[2.6.1. Khái niệm 13](#_Toc206405905)

[2.6.2. Lợi ích 14](#_Toc206405906)

[2.7. Prisma 15](#_Toc206405907)

[2.7.1. Khái niệm 15](#_Toc206405908)

[2.7.2. Các thành phần chính 15](#_Toc206405909)

[2.7.3. Lợi ích 15](#_Toc206405910)

[2.8. Swagger 15](#_Toc206405911)

[2.8.1 Khái niệm 15](#_Toc206405912)

[2.8.2 Các thành phần chính 16](#_Toc206405913)

[2.8.3 Lợi ích 16](#_Toc206405914)

[CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU 17](#_Toc206405915)

[3.1. Mô tả bài toán 17](#_Toc206405916)

[Bối cảnh và lý do chọn đề tài 17](#_Toc206405917)

[Mục tiêu bài toán 17](#_Toc206405918)

[3.2. Yêu cầu chức năng của hệ thống 18](#_Toc206405919)

[3.2.1. Đối với người quản trị 18](#_Toc206405920)

[3.2.2. Đối với người dùng 19](#_Toc206405921)

[3.3. Yêu cầu phi chức năng của hệ thống 20](#_Toc206405922)

[3.4. Phân tích thiết kế hệ thống 21](#_Toc206405923)

[3.4.1. Sơ đồ phân cấp chức năng 21](#_Toc206405924)

[3.4.2. Sơ đồ use case chung cho cả hệ thống 24](#_Toc206405925)

[3.4.3. Sơ đồ thực thể kết hợp ERD 26](#_Toc206405926)

[3.4.4. Sơ đồ vật lý 27](#_Toc206405927)

[3.4.5. Thông tin các bảng của cơ sở dữ liệu 28](#_Toc206405928)

[3.5. Các bước cài đặt 34](#_Toc206405929)

[3.5.1 Cài đặt trên local 34](#_Toc206405930)

[3.5.2 Cài đặt trên Docker 35](#_Toc206405931)

[CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU 37](#_Toc206405932)

[4.1 Giao diện cho người quản lý 37](#_Toc206405933)

[4.2 Giao diện cho người dùng 42](#_Toc206405934)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 50](#_Toc206405935)

[5.1. Kết quả và đóng góp của đồ án 50](#_Toc206405936)

[5.2. Hạn chế 51](#_Toc206405937)

[5.3. Hướng phát triển 51](#_Toc206405938)

**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)

UBND TỈNH TRÀ VINH **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: Nguyễn Tín Thành MSSV: 110121104

Ngành: Công nghệ Thông tin Khóa: 2021

Tên đề tài: Xây dựng website bán laptop

Họ và tên Giáo viên hướng dẫn: Võ Thành C

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:

Trà Vinh*, ngày tháng năm 20…*

Giảng viên hướng dẫn

*(Ký & ghi rõ họ tên)*

**NHẬN XÉT**

**(Của giảng viên chấm trong đồ án, khoá luận của sinh viên)**

**Giảng viên chấm**

(ký và ghi rõ họ tên)

UBND TỈNH TRÀ VINH **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 HÌNH ẢNH**

[Hình 3. 1 Sơ đồ phân cấp chức năng 21](#_Toc205843765)

[Hình 3. 2 Sơ đồ useCase 22](#_Toc205843766)

[Hình 3. 3 Sơ đồ thực thể kết hợp ERD 23](#_Toc205843767)

[Hình 3. 4 Sơ đồ vật lý 24](#_Toc205843768)

[Hình 4. 1 Giao diện đăng nhập admin 32](#_Toc205843798)

[Hình 4. 2 Giao diện thống kê trang chủ admin 32](#_Toc205843799)

[Hình 4. 3 Trang quản lý người dùng 33](#_Toc205843800)

[Hình 4. 4 Trang thêm thông tin người dùng mới 33](#_Toc205843801)

[Hình 4. 5 Sau khi thêm sản phẩm 34](#_Toc205843802)

[Hình 4. 6 Quản lý danh mục 34](#_Toc205843803)

[Hình 4. 7 Quản lý danh mục 34](#_Toc205843804)

[Hình 4. 8 Quản lý đơn hàng 35](#_Toc205843805)

[Hình 4. 9 Thông tin chi tiết đơn hàng 35](#_Toc205843806)

[Hình 4. 10 Quản lý mã khuyến mãi 36](#_Toc205843807)

[Hình 4. 11 Modal nhập mã khuyến mãi 36](#_Toc205843808)

[Hình 4. 12 Trang chủ 37](#_Toc205843809)

[Hình 4. 13 Giao diện đăng nhập 37](#_Toc205843810)

[Hình 4. 14 Giao diện đăng ký 38](#_Toc205843811)

[Hình 4. 15 Giao diện giỏ hàng 39](#_Toc205843812)

[Hình 4. 16 Giao diện thanh toán 40](#_Toc205843813)

[Hình 4. 17 Giao diện quản lý đơn hàng cá nhân 41](#_Toc205843814)

[Hình 4. 18 Giao diện quản lý thông tin cá nhân 42](#_Toc205843815)

[Hình 4. 19 Giao diện quản lý sản phẩm yêu thích 42](#_Toc205843816)

[Hình 4. 20 Quản lý điểm tích lũy 43](#_Toc205843817)

[Hình 4. 21 Thông tin chi tiết sản phẩm 44](#_Toc205843818)

[Hình 4. 22 giao diện footer 44](#_Toc205843819)

**DANH MỤC BẢNG BIỂU**

[Bảng 3. 1 Thông tin bảng user 25](#_Toc205585412)

[Bảng 3. 2 Thông tin bảng role 25](#_Toc205585413)

[Bảng 3. 3 Thông tin bảng product 26](#_Toc205585414)

[Bảng 3. 4 Thông tin bảng laptop 27](#_Toc205585415)

[Bảng 3. 5 Thông tin bảng accessory 27](#_Toc205585416)

[Bảng 3. 6 Thông tin bảng Cart 28](#_Toc205585417)

[Bảng 3. 7 Thông tin bảng wishlist 28](#_Toc205585418)

[Bảng 3. 8 Thông tin bảng order 29](#_Toc205585419)

[Bảng 3. 9 Thông tin bảng user 30](#_Toc205585420)

[Bảng 3. 10 Thông tin bảng comment 30](#_Toc205585421)

[Bảng 3. 11 Thông tin bảng feedback 31](#_Toc205585422)

**DANH MỤC TỪ VIẾT TẮT**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Từ viết tắt** | **Ý nghĩa đầy đủ (Full Form)** | **Giải thích / Ghi chú** |
| 1 | API | Application Programming Interface | Giao diện lập trình ứng dụng – thường dùng để giao tiếp giữa front-end và back-end |
| 2 | CRUD | Create, Read, Update, Delete | Các thao tác cơ bản với dữ liệu |
| 3 | DB | Database | Cơ sở dữ liệu |
| 4 | UI | User Interface | Giao diện người dùng |
| 5 | UX | User Experience | Trải nghiệm người dùng |
| 6 | JWT | JSON Web Token | Chuỗi mã hóa dùng cho xác thực và phân quyền |
| 7 | MVC | Model – View – Controller | Mô hình kiến trúc phần mềm |
| 8 | RESTful API | Representational State Transfer | Giao thức truyền thông thường dùng trong Web API |
| 9 | JSON | JavaScript Object Notation | Định dạng trao đổi dữ liệu |
| 10 | HTML | HyperText Markup Language | Ngôn ngữ đánh dấu siêu văn bản |
| 11 | CSS | Cascading Style Sheets | Ngôn ngữ định kiểu cho HTML |
| 12 | JS | JavaScript | Ngôn ngữ lập trình front-end (cũng dùng cả ở back-end với Node.js) |
| 15 | SQL | Structured Query Language | Ngôn ngữ truy vấn dữ liệu trong DB |
| 16 | XAMPP | Cross-platform, Apache, MySQL, PHP, Perl | Bộ phần mềm giả lập máy chủ web cục bộ |
| 17 | ORM | Object Relational Mapping | Công cụ ánh xạ giữa CSDL và đối tượng trong mã (VD: Sequelize, Prisma) |
| 20 | CSR | Client-Side Rendering | Kết xuất giao diện phía trình duyệt |
| 21 | SSR | Server-Side Rendering | Kết xuất giao diện phía máy chủ |
| 23 | NPM | Node Package Manager | Trình quản lý gói của Node.js |
| 24 | IDE | Integrated Development Environment | Môi trường phát triển tích hợp (VD: VS Code) |
| 25 | UX/UI | User Experience / User Interface | Giao diện và trải nghiệm người dùng |

**TÓM TẮT**

Ngày nay, công nghệ thông tin đang phát triển mạnh mẽ, nhu cầu thương mại điện tử của các cá nhân, doanh nghiệp, cũng như các tổ chức ngày càng gia tăng. Việc ứng dụng công nghệ thông tin vào hoạt động kinh doanh không chỉ giúp cho doanh nghiệp dễ dàng quản lý thông tin sản phẩm, khách hàng và đơn hàng, mà còn góp phần mở rộng thị trường, nâng cao hiệu quả bán hàng, tiết kiệm chi phí và thời gian vận hành.

Sự phát triển vượt bậc của internet và các nền tảng trực tuyến đã tạo điều kiện cho việc mua bán, trao đổi hàng hóa diễn ra nhanh chóng, tiện lợi. Trong đó, mô hình kinh doanh thông qua website thương mại điện tử đang trở thành xu hướng phổ biến, giúp người bán và người mua có thể giao dịch mọi lúc, mọi nơi. Mỗi website được xây dựng sẽ có các tính năng chuyên biệt tùy theo mục tiêu kinh doanh và nhu cầu của doanh nghiệp.

Đối với lĩnh vực kinh doanh laptop, việc xây dựng một website chuyên dụng không chỉ giúp giới thiệu sản phẩm đến khách hàng một cách trực quan, chi tiết, mà còn hỗ trợ các nghiệp vụ như quản lý kho hàng, xử lý đơn đặt hàng, tính toán giá bán, quản lý khuyến mãi và chăm sóc khách hàng một cách hiệu quả.

Thay vì quản lý sản phẩm, đơn hàng và khách hàng bằng phương pháp thủ công hoặc những công cụ cơ bản như Word, Excel, một hệ thống website được thiết kế và lập trình chuyên nghiệp sẽ tự động hóa nhiều quy trình, giúp tiết kiệm thời gian, công sức và giảm thiểu sai sót.

Từ nhu cầu thực tế đó, đề tài “Xây dựng website bán laptop” được lựa chọn với mục tiêu tạo ra một nền tảng thương mại điện tử hiện đại, đáp ứng nhu cầu quản lý sản phẩm, đơn hàng, khách hàng và thanh toán trực tuyến. Hệ thống sẽ áp dụng công nghệ web tiên tiến nhằm nâng cao trải nghiệm người dùng, tối ưu hoạt động kinh doanh và đảm bảo tính bảo mật trong giao dịch.

# TỔNG QUAN

## Đặt vấn đề

Trong bối cảnh xã hội hiện đại không ngừng phát triển và hội nhập, sự bùng nổ của công nghệ thông tin đã kéo theo những thay đổi sâu sắc trong mọi lĩnh vực, đặc biệt là thương mại. Thương mại điện tử (TMĐT) đã trở thành một xu thế tất yếu, định hình lại cách thức giao dịch mua bán, trao đổi hàng hóa và dịch vụ. Sự tiện lợi, khả năng tiếp cận không giới hạn về không gian và thời gian, chỉ với một thiết bị thông minh có kết nối internet, đã biến TMĐT thành kênh phân phối hàng đầu, mang lại lợi thế cạnh tranh đáng kể cho doanh nghiệp và người tiêu dùng.

Trong thị trường TMĐT sôi động hiện nay, ngành hàng công nghệ - đặc biệt là laptop - luôn giữ một vai trò quan trọng nhờ nhu cầu sử dụng ngày càng cao từ học sinh, sinh viên đến nhân viên văn phòng, lập trình viên và doanh nhân. Nhu cầu này tạo ra một cơ hội lớn cho các doanh nghiệp triển khai hệ thống kinh doanh trực tuyến chuyên biệt, hiệu quả để tiếp cận và phục vụ khách hàng một cách tối ưu.

Tuy nhiên, để một website bán hàng thực sự hiệu quả và có khả năng cạnh tranh trong môi trường TMĐT ngày càng khốc liệt, việc chỉ cung cấp các tính năng cơ bản như giỏ hàng, thanh toán hay quản lý đơn hàng là chưa đủ. Người dùng hiện đại kỳ vọng một trải nghiệm mua sắm cá nhân hóa, thông minh và thuận tiện hơn. Trong bối cảnh đó, việc tích hợp Trí tuệ nhân tạo (AI) trở thành một giải pháp đột phá, giúp tối ưu hóa trải nghiệm người dùng thông qua các tính năng như hệ thống gợi ý sản phẩm, chatbot hỗ trợ khách hàng, phân tích hành vi mua sắm, và quản lý tồn kho thông minh.

Về mặt kiến trúc hệ thống, dự án sử dụng kiến trúc phân lớp kết hợp với các công nghệ hiện đại nhằm đảm bảo hiệu năng, bảo mật và khả năng mở rộng trong tương lai. Cụ thể:

* Backend được xây dựng bằng Node.js với framework Express, kết nối với MySQL để lưu trữ và truy xuất dữ liệu.
* Frontend sử dụng React.js, một thư viện JavaScript phổ biến để phát triển các ứng dụng web hiện đại, tối ưu hiệu suất và nâng cao trải nghiệm người dùng.
* Hệ thống còn tích hợp AI ở một số tính năng như gợi ý sản phẩm dựa trên lịch sử mua hàng, giúp tăng mức độ hài lòng và giữ chân khách hàng.
* RESTful API được thiết kế để dễ dàng mở rộng và tích hợp với các dịch vụ khác.

Từ những phân tích trên, đề tài “xây dựng Website bán laptop” được đề xuất với mục tiêu phát triển một hệ thống thương mại điện tử hiện đại, thông minh, đáp ứng kỳ vọng của người tiêu dùng trong thời đại số, đồng thời mang lại hiệu quả vận hành và lợi thế cạnh tranh cho doanh nghiệp.

## Mục đích

Đề tài “xây dựng Website bán laptop” được thực hiện với mục tiêu nghiên cứu và phát triển một nền tảng thương mại điện tử hiện đại, thông minh và có khả năng mở rộng. Trang web đảm bảo có khả năng đáp ứng các chức năng cơ bản của một hệ thống bán hàng trực tuyến để nâng cao trải nghiệm người dùng, tối ưu hóa hoạt động vận hành, đồng thời mang lại lợi thế cạnh tranh cho doanh nghiệp trong kỷ nguyên số.

### Đối với người dùng cuối

Người dùng là trung tâm của hệ thống, là nền tảng của một cửa hàng nếu muốn hoạt động ổn định, do đó đề tài hướng tới việc cung cấp một nền tảng mua sắm trực tuyến đáp ứng nhu cầu thực tế và ngày càng khắt khe của khách hàng hiện đại:

* **Tìm kiếm và lọc nâng cao**: Tối ưu giao diện và chức năng tìm kiếm, giúp người dùng dễ dàng tiếp cận sản phẩm mong muốn.
* **Giao diện trực quan, thân thiện**: Ứng dụng React.js để xây dựng giao diện hiện đại, phản hồi nhanh, dễ sử dụng trên cả desktop và thiết bị di động.
* **Tương tác cộng đồng và đánh giá sản phẩm**: Khuyến khích người dùng chia sẻ trải nghiệm, đóng góp nội dung và hỗ trợ các khách hàng khác thông qua chức năng bình luận, đánh giá.
* **Tăng cường gắn kết thông qua ưu đãi và mini game**: Thêm tính năng trò chơi quay thưởng, nhận mã giảm giá nhằm tạo sự hấp dẫn và giữ chân khách hàng lâu dài.

### Đối với người quản trị

Hệ thống cung cấp một bảng điều khiển chuyên nghiệp và đầy đủ tính năng, hỗ trợ quản lý vận hành hiệu quả:

* **Quản lý sản phẩm và danh mục linh hoạt**: Dễ dàng thêm, sửa, xóa, tìm kiếm và kiểm kê sản phẩm/danh mục.
* **Quản lý người dùng và nội dung cộng đồng**: Duyệt bình luận, xử lý báo cáo vi phạm, quản lý thông tin người dùng và quyền truy cập.
* **Theo dõi và xử lý đơn hàng**: Kiểm soát toàn bộ quá trình mua hàng từ đặt hàng, thanh toán, giao nhận.
* **Quản lý mã giảm giá và chương trình khuyến mãi**: Tạo, chỉnh sửa, theo dõi hiệu quả sử dụng mã khuyến mãi, tự động hủy mã hết hạn.

### Mục tiêu kỹ thuật

* **Kiến trúc linh hoạt, dễ mở rộng**: Ứng dụng kiến trúc phân lớp với RESTful API để hỗ trợ các dịch vụ trong tương lai như microservices.
* **Bảo mật người dùng:** Áp dụng các kỹ thuật bảo mật cơ bản như mã hóa mật khẩu bằng bcrypt, xác thực người dùng bằng JWT (JSON Web Token), và phân quyền truy cập theo vai trò (user, admin).
* **Tối ưu hiệu năng giao diện người dùng:** Sử dụng React.js để xây dựng giao diện nhanh, mượt mà, kết hợp lazy loading, phân trang, debounce tìm kiếm nhằm tăng tốc độ phản hồi và giảm tải server.
* **Tích hợp cơ sở dữ liệu quan hệ:** Sử dụng MySQL để lưu trữ dữ liệu có cấu trúc như sản phẩm, đơn hàng, người dùng,... kết hợp prisma ORM hoặc truy vấn SQL thuần tùy theo yêu cầu của hệ thống.

# CƠ SỞ LÝ THUYẾT

## Kiến trúc Microservices

### Tổng quan

Microservices Architecture (kiến trúc vi dịch vụ) là một mô hình kiến trúc phần mềm trong đó hệ thống được chia thành nhiều dịch vụ nhỏ, độc lập, mỗi dịch vụ đảm nhiệm một chức năng riêng biệt. Mỗi microservice có thể được phát triển, triển khai và mở rộng một cách riêng rẽ, cho phép tối ưu hóa hiệu năng, khả năng mở rộng và tính bảo trì của toàn hệ thống.

Việc lựa chọn xây dựng hệ thống theo hướng microservices, sử dụng Node.js làm nền tảng phát triển phía backend và MySQL làm hệ quản trị cơ sở dữ liệu cho từng dịch vụ sẽ giúp dễ dàng triển khai các dịch vụ mà không bị xung đột đối với các dịch vụ khác

#### Đặc điểm của kiến trúc Microservices

* **Tách biệt chức năng**: Mỗi microservice đảm nhiệm một chức năng cụ thể như quản lý người dùng, sản phẩm, đơn hàng, giỏ hàng,...
* **Triển khai độc lập**: Các dịch vụ có thể được build và triển khai riêng biệt mà không ảnh hưởng đến các dịch vụ khác.
* **Cơ sở dữ liệu riêng biệt**: Mỗi service có cơ sở dữ liệu riêng (ở đây là MySQL), giúp đảm bảo tính toàn vẹn dữ liệu và tránh phụ thuộc giữa các service.
* **Giao tiếp qua giao thức HTTP hoặc Message Queue:** Các service giao tiếp với nhau thông qua RESTful API hoặc các hàng đợi tin nhắn như RabbitMQ, Kafka trong các tình huống cần xử lý bất đồng bộ.

### Ưu điểm

* **Phân chia rõ ràng, dễ quản lý:** Kiến trúc Microservices giúp chia hệ thống lớn thành các dịch vụ nhỏ, độc lập, mỗi dịch vụ phụ trách một chức năng cụ thể. Việc này giúp quản lý code dễ dàng hơn, giảm độ phức tạp cho từng thành phần, và tăng khả năng làm việc theo nhóm.
* **Dễ dàng mở rộng (scalability):** Do mỗi dịch vụ được tách biệt, hệ thống có thể mở rộng linh hoạt từng service tùy theo nhu cầu về tài nguyên hoặc lưu lượng người dùng. Ví dụ, nếu service xử lý đơn hàng có nhu cầu cao, chỉ cần mở rộng riêng service này mà không ảnh hưởng đến toàn bộ hệ thống.
* **Triển khai độc lập:** Mỗi dịch vụ có thể được triển khai riêng biệt, không cần chờ các phần còn lại của hệ thống. Điều này hỗ trợ mạnh mẽ cho các quy trình phát triển theo DevOps và triển khai liên tục (CI/CD).
* **Tăng tính ổn định:** Một lỗi xảy ra trong một service sẽ không làm sập toàn bộ hệ thống, giúp hệ thống duy trì tính sẵn sàng cao hơn. Khả năng cô lập lỗi tốt hơn cũng giúp quá trình bảo trì và khắc phục sự cố dễ dàng hơn.
* **Dễ áp dụng công nghệ đa dạng:** Các service có thể sử dụng ngôn ngữ lập trình hoặc công nghệ riêng phù hợp với nghiệp vụ của chúng. Trong đề tài này, nhóm sử dụng Node.js và MySQL nhưng vẫn có thể tích hợp các công nghệ khác nếu cần thiết mà không ảnh hưởng đến tổng thể kiến trúc.

### Nhược điểm

* **Độ phức tạp cao**: Việc phân tách hệ thống thành nhiều service làm gia tăng độ phức tạp trong quản lý, triển khai và vận hành. Hệ thống cần các công cụ hỗ trợ theo dõi logs, giám sát, phân phối cấu hình, đồng bộ dữ liệu và đảm bảo giao tiếp ổn định giữa các service.
* **Khó khăn trong quản lý giao tiếp giữa các service:** Việc gọi API giữa các service (service-to-service communication) có thể dẫn đến vấn đề về độ trễ (latency), lỗi mạng, hoặc lỗi timeout. Nếu không xử lý tốt, các service sẽ dễ bị phụ thuộc vào nhau, đi ngược lại nguyên tắc tách biệt.
* **Yêu cầu cao về DevOps**: Microservices đòi hỏi môi trường DevOps mạnh mẽ để tự động hóa việc build, test, deploy và monitor từng service. Với những nhóm chưa có kinh nghiệm hoặc thiếu công cụ phù hợp, việc triển khai có thể gặp khó khăn.
* **Khó đảm bảo tính nhất quán dữ liệu (data consistency):** Vì mỗi service có cơ sở dữ liệu riêng, việc đảm bảo tính toàn vẹn dữ liệu khi thực hiện các thao tác liên quan đến nhiều service trở nên khó khăn. Việc triển khai các mô hình như Saga hoặc Eventual Consistency là cần thiết nhưng không đơn giản.
* **Chi phí tài nguyên cao hơn:** Việc vận hành nhiều service chạy song song có thể làm tăng tài nguyên hệ thống, yêu cầu nhiều container hoặc máy chủ hơn so với kiến trúc nguyên khối (monolithic), dẫn đến chi phí cao hơn nếu không tối ưu tốt**.**

## Restful API

### Tổng quan

**RESTful API** (Representational State Transfer) là một phong cách kiến trúc phần mềm được sử dụng phổ biến trong việc phát triển các hệ thống dịch vụ web, đặc biệt phù hợp với các hệ thống phân tán như **Microservices**. REST định nghĩa một bộ quy tắc để xây dựng API sử dụng các phương thức HTTP tiêu chuẩn như **GET**, **POST**, **PUT**, **DELETE** nhằm thao tác với tài nguyên (resources) được đại diện dưới dạng dữ liệu (thường là JSON).

Trong kiến trúc RESTful API, mỗi tài nguyên (ví dụ: sản phẩm, người dùng, đơn hàng...) được biểu diễn bởi một URL duy nhất. Các thao tác tương ứng sẽ được thực hiện thông qua HTTP method. Điều này giúp API trở nên đơn giản, dễ hiểu và thân thiện với lập trình viên front-end.

RESTful API đặc biệt phù hợp với **kiến trúc Microservices**, nơi mỗi dịch vụ (service) phụ trách một phần chức năng riêng biệt, và giao tiếp với nhau thông qua HTTP.

### Đặc điểm

* **Dựa trên giao thức HTTP:** gồm các phương thức là GET, POST, PUT, DELETE và PATCH
* **Mỗi tài nguyên có một URL duy nhất:** ví dụ như api/product là một URL duy nhất
* **Sử dụng JSON để trao đổi dữ liệu:** JSON là định dạng phổ biến để truyền dữ liệu giữa client và server trong RESTful API vì nó nhẹ, dễ đọc và dễ xử lý ở cả hai phía.
* **Stateless (phi trạng thái):** Mỗi request từ client tới server đều độc lập và không lưu trữ trạng thái giữa các request. Điều này giúp hệ thống dễ mở rộng và đơn giản trong việc xử lý.
* **Khả năng phân tách rõ ràng giữa client và server:** Client (giao diện người dùng) và server (logic xử lý) hoạt động độc lập, giao tiếp thông qua API, giúp tăng khả năng mở rộng và bảo trì hệ thống.
* **Dễ dàng tích hợp với hệ thống khác:** RESTful API là chuẩn phổ biến và được hỗ trợ rộng rãi nên dễ tích hợp với các hệ thống, dịch vụ bên ngoài hoặc app mobile.

### Ưu điểm

* **Đơn giản và dễ hiểu**: RESTful API sử dụng các phương thức HTTP cơ bản như GET, POST, PUT, DELETE nên rất trực quan và dễ tiếp cận đối với các lập trình viên, kể cả người mới bắt đầu.
* **Tính linh hoạt cao:** REST không bị ràng buộc bởi ngôn ngữ lập trình hay công nghệ cụ thể, giúp API có thể triển khai bằng bất kỳ backend nào (Node.js, Java, Python...) và dễ dàng tích hợp với nhiều client khác nhau như web, mobile, IoT,...
* **Dễ mở rộng:** RESTful API giúp chia nhỏ hệ thống thành các tài nguyên độc lập, phù hợp với kiến trúc microservices. Việc thêm mới, cập nhật hoặc xoá dịch vụ không ảnh hưởng đến toàn hệ thống.
* **Hiệu suất cao và nhẹ:** RESTful thường sử dụng JSON làm định dạng dữ liệu nên nhẹ hơn so với XML, giúp tiết kiệm băng thông và cải thiện hiệu suất truyền tải dữ liệu.
* **Dễ kiểm thử và debug:** Do RESTful API sử dụng chuẩn HTTP và tài nguyên có URL rõ ràng, nên dễ dàng kiểm thử qua công cụ như Postman, cURL, hoặc tích hợp CI/CD để tự động hoá việc kiểm tra.

### Nhược điểm

* **Thiếu tính chuẩn hóa nghiêm ngặt:** REST chỉ là một phong cách kiến trúc nên không có quy chuẩn chặt chẽ, điều này dẫn đến các dự án RESTful API có thể triển khai không nhất quán nếu không có định hướng hoặc quy ước rõ ràng từ đầu.
* **Không hỗ trợ tốt các thao tác phức tạp**: RESTful API có thể gặp khó khăn khi xử lý các yêu cầu có logic nghiệp vụ phức tạp, chẳng hạn như thao tác đồng thời trên nhiều tài nguyên (transactional logic), vì REST thường chỉ thao tác từng tài nguyên một cách độc lập.
* **Không có tính trạng thái (Stateless):** Mặc dù stateless giúp đơn giản hóa và mở rộng dễ dàng, nhưng đồng thời cũng gây khó khăn khi cần lưu trữ phiên người dùng hoặc các thao tác liên quan đến trạng thái dài hạn.
* **Quản lý version phức tạp:** Khi hệ thống mở rộng và thay đổi, việc quản lý version API (v1, v2,...) cần có chiến lược rõ ràng nếu không sẽ gây rối loạn hoặc xung đột cho các client đang sử dụng.

## Ngôn ngữ backend

### Nodejs

Node.js là một môi trường thực thi JavaScript phía máy chủ (server-side), được xây dựng dựa trên công cụ V8 JavaScript Engine của Google. Được phát triển lần đầu vào năm 2009 bởi Ryan Dahl, Node.js cho phép JavaScript – vốn là một ngôn ngữ chỉ chạy trên trình duyệt – có thể được sử dụng để xây dựng các ứng dụng backend mạnh mẽ, hiệu quả và dễ mở rộng.

Node.js áp dụng mô hình **non-blocking I/O** (nhập xuất không đồng bộ) và **event-driven** (dựa trên sự kiện), giúp tối ưu hoá hiệu suất xử lý trong các ứng dụng có nhiều kết nối đồng thời, như API backend, hệ thống chat, hoặc ứng dụng real-time.

Nhờ sử dụng cùng một ngôn ngữ JavaScript cho cả frontend và backend, Node.js giúp giảm sự phân mảnh về công nghệ trong một dự án, đồng thời tăng tốc độ phát triển và bảo trì hệ thống.

Một số ưu điểm nổi bật của Node.js:

* Hiệu năng cao: Nhờ vào engine V8 và cơ chế I/O không đồng bộ.
* Dễ mở rộng: Thích hợp cho các kiến trúc như microservices hoặc serverless.
* Hệ sinh thái phong phú: Có sẵn hàng nghìn package thông qua npm (Node Package Manager).
* Tính nhất quán về ngôn ngữ: Giúp các đội ngũ frontend có thể dễ dàng làm việc với backend.

Node.js hiện là một trong những công nghệ backend phổ biến nhất, được sử dụng bởi nhiều công ty lớn như Netflix, LinkedIn, Uber và PayPal.

### Express.js

Express.js là một framework tối giản và linh hoạt dành cho Node.js, giúp xây dựng các ứng dụng web và API một cách nhanh chóng. Express hỗ trợ đầy đủ các tính năng để xây dựng một hệ thống RESTful API chuyên nghiệp và là một lựa chọn phổ biến trong các dự án Node.js.

Ưu điểm của Express:

* **Cú pháp đơn giản, dễ dùng**: Giúp khởi tạo nhanh các endpoint HTTP (GET, POST, PUT, DELETE).
* **Middleware mạnh mẽ:** Cho phép xử lý logic giữa các bước như xác thực, logging, kiểm tra lỗi,...
* **Dễ dàng mở rộng**: Có thể tích hợp các thư viện khác như JWT, multer, morgan,... để mở rộng chức năng.
* **Tối ưu cho Microservices**: Dễ chia nhỏ thành các service độc lập với khả năng mở rộng linh hoạt.

## Ngôn ngữ frontend

### Reactjs

ReactJS là một thư viện JavaScript mã nguồn mở được phát triển và duy trì bởi Facebook từ năm 2013. React được thiết kế nhằm hỗ trợ việc xây dựng giao diện người dùng (UI) một cách hiệu quả, đặc biệt trong các ứng dụng web có tính tương tác cao. Với triết lý component-based (dựa trên thành phần), React giúp chia nhỏ giao diện thành các thành phần độc lập, có thể tái sử dụng và dễ bảo trì.

Khác với các framework toàn diện như Angular, React chỉ tập trung vào tầng View trong mô hình MVC (Model – View – Controller). Tuy nhiên, nhờ hệ sinh thái phong phú (React Router, Redux, Zustand, React Query, Material UI,...), React có thể dễ dàng mở rộng để đáp ứng đầy đủ nhu cầu phát triển các ứng dụng web hiện đại.

#### Ưu điểm của ReactJS:

* Component hóa mạnh mẽ: React cho phép chia nhỏ giao diện thành nhiều component nhỏ, độc lập, có thể tái sử dụng, từ đó tăng khả năng quản lý và bảo trì mã nguồn.
* Virtual DOM: React sử dụng cơ chế Virtual DOM để cập nhật giao diện một cách tối ưu, giúp tăng hiệu suất và giảm chi phí tái render không cần thiết.
* Khả năng tái sử dụng và mở rộng cao: Các component React có thể dễ dàng được tái sử dụng trong nhiều phần khác nhau của ứng dụng hoặc thậm chí trong các dự án khác.
* JSX – cú pháp mở rộng: JSX kết hợp giữa JavaScript và HTML giúp việc viết UI trở nên trực quan, đồng thời tăng tính mô-đun và dễ kiểm soát logic giao diện.
* Cộng đồng lớn mạnh và tài liệu phong phú: React có một cộng đồng người dùng rộng lớn cùng với tài liệu chính thức chi tiết, hỗ trợ quá trình học tập và phát triển ứng dụng dễ dàng hơn.
* Thích hợp với SPA (Single Page Application): React rất phù hợp để xây dựng các ứng dụng một trang, nơi người dùng có thể tương tác mà không cần tải lại trang.
* Tích hợp tốt với RESTful API hoặc GraphQL: React có thể kết nối dữ liệu dễ dàng thông qua các thư viện như Axios, Fetch, hay Apollo Client, phù hợp với các kiến trúc backend hiện đại.

#### Nhược điểm của ReactJS:

* Chỉ tập trung vào View: React không cung cấp sẵn các giải pháp về routing, state management, hay form validation, buộc lập trình viên phải chọn thư viện bên ngoài, dẫn đến sự phân mảnh và cần thời gian để lựa chọn công nghệ phù hợp.
* Cập nhật thường xuyên: React và hệ sinh thái đi kèm cập nhật liên tục. Mặc dù giúp cải tiến tính năng và hiệu năng, nhưng cũng đòi hỏi nhà phát triển phải luôn cập nhật kiến thức, nhất là khi có thay đổi lớn (ví dụ như hooks, concurrent mode).
* Khó khăn cho người mới: Việc kết hợp giữa JavaScript, JSX, hooks, context và các thư viện ngoài như Redux có thể khiến người mới bắt đầu cảm thấy khó khăn khi tiếp cận và xây dựng ứng dụng React hoàn chỉnh.
* Quản lý state phức tạp: Trong các ứng dụng lớn, việc quản lý trạng thái (state) toàn cục có thể trở nên phức tạp nếu không có kiến trúc rõ ràng.

### Material UI library

**Material UI** (thường được viết tắt là MUI) là một thư viện giao diện người dùng (UI) nổi bật dành cho React, được xây dựng dựa trên triết lý thiết kế Material Design của Google. Thư viện này cung cấp một tập hợp các component UI phong phú, dễ sử dụng, có tính nhất quán cao và khả năng tuỳ biến linh hoạt, giúp tăng tốc quá trình xây dựng giao diện người dùng hiện đại cho các ứng dụng web.

Không giống như việc tự thiết kế từng thành phần từ đầu, Material UI mang đến một bộ công cụ thiết kế hoàn chỉnh với các thành phần như nút bấm, thanh điều hướng, hộp thoại, biểu mẫu, bảng dữ liệu, biểu tượng, và nhiều hơn nữa – tất cả đều tuân thủ chặt chẽ chuẩn thiết kế Material Design. Với việc tích hợp sâu sắc vào React và khả năng mở rộng mạnh mẽ, MUI trở thành lựa chọn phổ biến trong cộng đồng phát triển front-end.

#### Ưu điểm:

Một trong những điểm mạnh lớn nhất của Material UI là khả năng **tăng tốc độ phát triển giao diện**. Các component đã được tối ưu sẵn về mặt giao diện lẫn hành vi, giúp lập trình viên không phải tốn thời gian "mò thiết kế" từ đầu, đồng thời vẫn đảm bảo tính thẩm mỹ và sự thống nhất trên toàn hệ thống.

Material UI cũng **hỗ trợ tuỳ biến rất tốt**. Nhờ hệ thống theme linh hoạt, nhà phát triển có thể dễ dàng thay đổi màu sắc, kiểu chữ, kích thước, hình dạng component, thậm chí override style chi tiết bằng hệ thống CSS-in-JS (Emotion hoặc Styled Components). Điều này giúp MUI phù hợp với cả các dự án cá nhân lẫn các hệ thống thương mại có yêu cầu cao về thương hiệu (branding).

Bên cạnh đó, Material UI đi kèm với **bộ icon phong phú** (Material Icons), hỗ trợ hiệu ứng chuyển động mượt mà (animation), khả năng responsive cao, và tích hợp tốt với các thư viện quản lý trạng thái như Redux hoặc Context API.

Một điểm cộng nữa là **cộng đồng người dùng rộng lớn**, tài liệu rõ ràng, ví dụ dễ hiểu và có sự hỗ trợ từ đội ngũ phát triển chuyên nghiệp. Việc cập nhật phiên bản và các tính năng mới cũng diễn ra thường xuyên, giúp đảm bảo tính ổn định lâu dài.

#### Nhược điểm:

Tuy sở hữu nhiều ưu điểm, Material UI cũng tồn tại một số điểm hạn chế nhất định. Thứ nhất, **kích thước bundle tương đối lớn** nếu chỉ sử dụng một vài component, điều này có thể ảnh hưởng đến hiệu suất nếu không cấu hình tree-shaking hoặc lazy load hợp lý.

Thứ hai, việc **tuỳ biến nâng cao** đôi khi yêu cầu người dùng phải hiểu rõ cấu trúc bên trong của component MUI, đặc biệt khi cần override sâu CSS hoặc áp dụng các kỹ thuật như slot customization. Điều này có thể gây khó khăn cho người mới bắt đầu hoặc những ai chưa quen với hệ thống CSS-in-JS.

Cuối cùng, **Material Design** tuy rất phổ biến, nhưng đôi khi không phù hợp với một số sản phẩm yêu cầu thiết kế mang tính cá nhân hoá cao hoặc đi theo hướng "phá cách", bởi phong cách Material khá đặc trưng và có phần ràng buộc về mặt hình thức.

Tóm lại, Material UI là một lựa chọn mạnh mẽ cho việc xây dựng giao diện React hiện đại, giúp giảm thiểu thời gian phát triển, tăng tính nhất quán trong UI, đồng thời vẫn giữ được sự linh hoạt trong tuỳ biến và mở rộng.

## Cơ sở dữ liệu MySQL

### Khái niệm

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) mã nguồn mở được phát triển ban đầu bởi công ty MySQL AB, sau đó được Sun Microsystems mua lại và hiện nay thuộc quyền sở hữu của Oracle Corporation. MySQL nổi bật nhờ hiệu suất cao, dễ sử dụng và khả năng mở rộng tốt, là lựa chọn phổ biến cho các hệ thống web, đặc biệt là trong kiến trúc LAMP (Linux – Apache – MySQL – PHP/Python/Perl).

Không giống như cái tên dễ gây nhầm lẫn là "MySQL server" ám chỉ phần cứng, MySQL thực chất là một phần mềm hệ quản trị cơ sở dữ liệu. Nó hoạt động như một “máy chủ dữ liệu” có nhiệm vụ tiếp nhận các yêu cầu từ ứng dụng (client), xử lý truy vấn và trả về kết quả tương ứng. Việc này cho phép các ứng dụng như hệ thống quản lý, trang web thương mại điện tử, blog, hoặc nền tảng phân tích dữ liệu vận hành trên một cơ sở dữ liệu tập trung, nhanh chóng và an toàn.

MySQL sử dụng ngôn ngữ SQL (Structured Query Language) để thao tác và truy vấn dữ liệu. Đây là một ngôn ngữ chuẩn, phổ biến trên hầu hết các hệ quản trị cơ sở dữ liệu quan hệ. Dù không có hệ mở rộng như T-SQL của SQL Server, MySQL vẫn hỗ trợ đầy đủ các tính năng cần thiết như stored procedures, triggers, views và transaction để xử lý các nghiệp vụ phức tạp trong thực tế.

### Cấu trúc

MySQL có kiến trúc mô-đun rõ ràng, giúp dễ dàng mở rộng và tối ưu hóa, bao gồm các thành phần chính như sau:

* **Connection Manager & Protocol Layer**: Là tầng đầu tiên tiếp nhận kết nối từ client. MySQL hỗ trợ nhiều giao thức như TCP/IP, socket Unix, hoặc Named Pipes (trên Windows). Tầng này đảm bảo giao tiếp hiệu quả giữa client và server.
* **SQL Layer**: Đây là nơi xử lý tất cả các truy vấn SQL. Các chức năng chính bao gồm phân tích cú pháp, lập kế hoạch thực thi (query optimization), kiểm tra quyền truy cập, và xử lý các biểu thức SQL.
* **Storage Engine Layer**: Đây là tầng cốt lõi phụ trách việc lưu trữ, truy xuất và ghi dữ liệu vào ổ cứng. MySQL hỗ trợ nhiều loại storage engine như:
  + InnoDB: Engine mặc định, hỗ trợ ACID, khóa ngoại (foreign key), transaction, và row-level locking.
  + MyISAM: Engine truyền thống, nhẹ hơn nhưng không hỗ trợ transaction hay khóa ngoại.
  + Ngoài ra còn có MEMORY, ARCHIVE, CSV, v.v., tùy thuộc vào nhu cầu sử dụng.
* **File System Layer**: Là tầng giao tiếp cuối cùng giữa MySQL và hệ điều hành, đảm nhận nhiệm vụ đọc/ghi dữ liệu vào tệp vật lý, quản lý bảng, chỉ mục, log file, và các dữ liệu hệ thống khác.

Một đặc điểm quan trọng của MySQL là thiết kế kiến trúc plug-in storage engine, cho phép người dùng lựa chọn hoặc thậm chí phát triển engine phù hợp với mục đích sử dụng cụ thể.

## Docker

### Khái niệm

Trong bối cảnh phát triển phần mềm hiện đại, nơi các ứng dụng cần được triển khai linh hoạt và nhanh chóng trên nhiều môi trường khác nhau, Docker nổi lên như một nền tảng mã nguồn mở mạnh mẽ, cho phép tự động hóa việc đóng gói, phân phối và triển khai ứng dụng dưới dạng các container. Với triết lý “viết một lần, chạy ở mọi nơi”, Docker đã trở thành một công cụ không thể thiếu trong hệ sinh thái DevOps và triển khai phần mềm.

Về bản chất, Docker ứng dụng công nghệ container hóa – một phương pháp đóng gói toàn bộ mã nguồn, thư viện, môi trường runtime và các phụ thuộc khác của ứng dụng vào một đơn vị độc lập gọi là container. Chính khả năng đóng gói này đảm bảo rằng ứng dụng sẽ hoạt động nhất quán bất kể môi trường triển khai là máy tính cá nhân, máy chủ cục bộ, hay đám mây.

Một trong những thành phần cốt lõi của Docker là Docker Engine – phần mềm nền tảng giúp tạo, quản lý và chạy các container. Docker Engine bao gồm ba thành phần chính: đầu tiên là Docker Daemon, một tiến trình chạy nền có nhiệm vụ xử lý các thao tác khởi tạo, dừng, quản lý và xóa container. Tiếp theo là Docker CLI (Command Line Interface), giao diện dòng lệnh cho phép lập trình viên và quản trị viên hệ thống tương tác với Docker một cách hiệu quả thông qua các lệnh đơn giản. Cuối cùng là Docker API, cung cấp khả năng truy cập và điều khiển Docker thông qua các giao thức HTTP, giúp tích hợp Docker vào các hệ thống hoặc công cụ khác.

Bên cạnh Docker Engine, khái niệm Docker Container giữ vai trò trung tâm trong toàn bộ kiến trúc. Mỗi container là một môi trường tách biệt, nơi một ứng dụng có thể chạy độc lập cùng với tất cả các thành phần cần thiết đi kèm. Mặc dù các container này chia sẻ chung hạt nhân hệ điều hành (kernel), chúng vẫn được cách ly về mặt quy trình, hệ thống tệp và tài nguyên, từ đó đảm bảo tính an toàn và ổn định trong vận hành.

Để tạo ra các container, Docker sử dụng Docker Image – một bản mẫu không thay đổi, chứa toàn bộ hệ điều hành, cấu hình, mã ứng dụng và các phần mềm hỗ trợ cần thiết. Các image này được xây dựng từ một tập tin có tên Dockerfile, nơi định nghĩa các bước cần thiết để thiết lập môi trường chạy cho ứng dụng. Dockerfile có thể được ví như một “công thức” để xây dựng image, cho phép tái tạo container một cách nhất quán và có thể kiểm soát được.

Ngoài ra, Docker còn cung cấp một nền tảng chia sẻ và phân phối các image thông qua Docker Hub – một kho lưu trữ trực tuyến cho phép người dùng tải về các image đã được chuẩn bị sẵn, hoặc chia sẻ những image do họ tự xây dựng. Docker Hub đóng vai trò như một chợ ứng dụng, giúp tiết kiệm thời gian và tối ưu quy trình phát triển.

### Lợi ích

Một trong những lợi thế nổi bật của Docker chính là khả năng khởi tạo và dừng container một cách nhanh chóng chỉ trong vài giây, giúp tăng tốc quá trình phát triển và triển khai ứng dụng. Không những vậy, Docker có thể chạy nhất quán trên nhiều hệ điều hành và nền tảng khác nhau, từ máy cá nhân đến hệ thống máy chủ hay nền tảng đám mây, nhờ kiến trúc container hóa độc lập với môi trường. Đặc biệt, Docker cho phép thiết lập môi trường làm việc một cách dễ dàng và ổn định, khi toàn bộ cấu hình và các phụ thuộc chỉ cần cài đặt một lần duy nhất trong file cấu hình (Dockerfile hoặc docker-compose), từ đó loại bỏ hoàn toàn nhu cầu cài lại phần mềm hay thư viện mỗi khi chuyển sang máy khác hoặc làm việc nhóm.

## Prisma

### Khái niệm

Prisma là một ORM (Object Relational Mapping) thế hệ mới dành cho Node.js và TypeScript. Nó giúp lập trình viên dễ dàng làm việc với cơ sở dữ liệu bằng cách cung cấp API trực quan, mạnh mẽ và an toàn về kiểu dữ liệu (type-safe). Prisma hỗ trợ nhiều hệ quản trị cơ sở dữ liệu phổ biến như MySQL, PostgreSQL, SQL Server, SQLite và MongoDB.

Nhờ Prisma, thay vì phải viết các truy vấn SQL phức tạp, lập trình viên có thể thao tác với dữ liệu thông qua các phương thức được sinh ra tự động từ mô hình dữ liệu (schema).

### Các thành phần chính

* Prisma Client: Là thư viện được sinh ra tự động từ file schema.prisma, cung cấp API type-safe để thực hiện các thao tác CRUD và query dữ liệu.
* Prisma Migrate: Công cụ quản lý migration, giúp định nghĩa, tạo và áp dụng các thay đổi cấu trúc cơ sở dữ liệu dựa trên schema.
* Prisma Studio: Giao diện người dùng trực quan (UI) cho phép xem và chỉnh sửa dữ liệu trong cơ sở dữ liệu một cách trực tiếp, tương tự như phpMyAdmin nhưng dành riêng cho Prisma.
* Prisma Schema: Tệp cấu hình trung tâm (thường là schema.prisma) định nghĩa mô hình dữ liệu, nguồn dữ liệu (datasource) và generator để sinh Prisma Client.

### Lợi ích

* Dễ sử dụng, cú pháp trực quan.
* Hỗ trợ TypeScript, đảm bảo an toàn kiểu dữ liệu.
* Tích hợp sẵn công cụ migration tiện lợi.
* Hỗ trợ nhiều hệ quản trị cơ sở dữ liệu.
* Hiệu suất tốt nhờ tối ưu hóa query.

## Swagger

1. **Khái niệm**

Swagger là một bộ công cụ mã nguồn mở giúp **thiết kế, xây dựng, mô tả và trực quan hóa API**. Với Swagger, lập trình viên có thể định nghĩa API bằng một chuẩn thống nhất (OpenAPI Specification - OAS), từ đó tự động sinh ra tài liệu API, mock server, và cả client SDK.

Swagger giúp tăng khả năng giao tiếp giữa backend, frontend và tester nhờ tài liệu API rõ ràng, dễ hiểu, có thể chạy thử trực tiếp trên trình duyệt.

1. **Các thành phần chính**

* OpenAPI Specification (OAS): Chuẩn định nghĩa API (thường ở dạng YAML/JSON) mô tả endpoints, request, response, models…
* Swagger UI: Giao diện web trực quan cho phép xem, tìm hiểu và thử nghiệm các API ngay trên trình duyệt.
* Swagger Editor: Công cụ trực tuyến hoặc cài đặt cục bộ, dùng để viết và chỉnh sửa tài liệu OpenAPI Specification.
* Swagger Codegen: Công cụ tự động sinh mã nguồn client SDK hoặc server stub từ file OpenAPI Specification.
* SwaggerHub: Nền tảng cộng tác trực tuyến của SmartBear, cho phép quản lý, chia sẻ và phát triển API tập trung.

1. **Lợi ích**

* Chuẩn hóa và thống nhất tài liệu API.
* Hỗ trợ tạo tài liệu tự động, dễ bảo trì.
* Tích hợp UI trực quan, có thể test API trực tiếp.
* Giảm thời gian phát triển nhờ sinh mã tự động.
* Tăng tính minh bạch và khả năng cộng tác giữa các nhóm.

# HIỆN THỰC HÓA NGHIÊN CỨU

## Mô tả bài toán

### ****Bối cảnh và lý do chọn đề tài****

Trong thời đại công nghệ số hiện nay, nhu cầu mua sắm trực tuyến ngày càng trở nên phổ biến và thiết yếu, đặc biệt là đối với các sản phẩm công nghệ như laptop. Người tiêu dùng có xu hướng tìm kiếm thông tin, so sánh cấu hình, giá cả, đánh giá từ người dùng khác trước khi đưa ra quyết định mua hàng. Do đó, việc xây dựng một website thương mại điện tử chuyên biệt trong lĩnh vực kinh doanh laptop là hoàn toàn cần thiết nhằm đáp ứng nhu cầu thị trường, đồng thời nâng cao hiệu quả kinh doanh cho doanh nghiệp.

Tuy nhiên, hiện nay nhiều website bán laptop vẫn còn tồn tại những hạn chế như giao diện khó sử dụng, không tối ưu cho thiết bị di động, thiếu tính năng tìm kiếm nâng cao, và không hỗ trợ cá nhân hóa trải nghiệm người dùng. Xuất phát từ thực tiễn đó, nhóm tác giả lựa chọn đề tài **“Xây dựng website bán laptop”** với mong muốn thiết kế và phát triển một hệ thống website chuyên nghiệp, thân thiện với người dùng, đáp ứng các yêu cầu chức năng cơ bản và mở rộng, đồng thời có thể tích hợp với các công nghệ hiện đại như quản lý kho, xử lý đơn hàng, và hỗ trợ khách hàng.

### ****Mục tiêu bài toán****

Mục tiêu của đề tài là thiết kế và xây dựng một hệ thống website bán laptop đáp ứng đầy đủ các chức năng của một trang thương mại điện tử cơ bản, đồng thời đảm bảo hiệu suất, bảo mật và tính dễ sử dụng. Cụ thể:

* Cung cấp nền tảng trực tuyến để khách hàng có thể dễ dàng tra cứu, tìm kiếm và mua sản phẩm laptop theo nhu cầu.
* Hỗ trợ người quản trị trong việc quản lý sản phẩm, đơn hàng, khách hàng, kho hàng và các hoạt động liên quan.
* Tích hợp hệ thống thanh toán trực tuyến và quản lý đơn hàng tự động.
* Đảm bảo khả năng mở rộng, bảo mật thông tin và khả năng sử dụng ổn định trên nhiều thiết bị (PC, tablet, mobile).

## Yêu cầu chức năng của hệ thống

Hệ thống website bán laptop được thiết kế nhằm phục vụ hai nhóm người dùng chính: **khách hàng (người mua)** và **quản trị viên (admin).** Các chức năng được phân chia và mô tả như sau:

### Đối với người quản trị

#### Đăng nhập hệ thống quản trị

* Quản trị viên đăng nhập vào hệ thống qua giao diện riêng.
* Có cơ chế bảo vệ truy cập trái phép bằng xác thực tài khoản.

#### Quản lý sản phẩm

* Thêm mới sản phẩm: tên, giá, hình ảnh, mô tả, cấu hình chi tiết (RAM, CPU, ổ cứng,...).
* Cập nhật thông tin sản phẩm (giá, kho hàng, mô tả).
* Xóa sản phẩm khi không còn kinh doanh.
* Quản lý số lượng tồn kho của từng sản phẩm.

#### Quản lý danh mục và thương hiệu

* Tạo, sửa, xóa danh mục sản phẩm (theo dòng máy, mục đích sử dụng).
* Quản lý danh sách các thương hiệu laptop (Dell, Asus, HP, Lenovo,...).

#### Quản lý đơn hàng

* Xem danh sách đơn hàng của khách hàng.
* Cập nhật trạng thái đơn hàng (đang xử lý, đang giao, đã giao, hủy).
* Xem chi tiết từng đơn hàng, khách hàng, sản phẩm liên quan.

#### Quản lý người dùng

* Xem danh sách người dùng đã đăng ký.
* Cập nhật, khóa hoặc xóa tài khoản vi phạm.
* Quản lý phản hồi, đánh giá của khách hàng.

#### Quản lý đánh giá

* Kiểm duyệt bình luận, đánh giá sản phẩm của người dùng.
* Xóa những bình luận vi phạm hoặc có nội dung không phù hợp.

### Đối với người dùng

#### Đăng ký tài khoản

* Người dùng có thể đăng ký tài khoản bằng cách cung cấp các thông tin: họ tên, email, mật khẩu, số điện thoại, địa chỉ.
* Hệ thống kiểm tra email đã được đăng ký trước đó hay chưa.
* Mật khẩu được mã hóa trước khi lưu trữ.

#### Đăng nhập và đăng xuất

* Người dùng đăng nhập bằng email và mật khẩu.
* Kiểm tra thông tin đăng nhập hợp lệ, thông báo lỗi nếu sai.
* Cho phép đăng xuất khỏi hệ thống một cách an toàn.

#### Quản lý thông tin cá nhân

* Cho phép người dùng xem và cập nhật thông tin cá nhân: họ tên, địa chỉ, số điện thoại.
* Đổi mật khẩu sau khi xác thực mật khẩu cũ.

#### Duyệt và tìm kiếm sản phẩm

* Hiển thị danh sách laptop theo nhiều danh mục (theo hãng, theo giá, theo cấu hình).
* Tìm kiếm sản phẩm theo tên, từ khóa.
* Lọc sản phẩm theo các tiêu chí: mức giá, RAM, CPU, thương hiệu, loại ổ cứng, kích thước màn hình,…

#### Xem chi tiết sản phẩm

* Hiển thị đầy đủ thông tin sản phẩm: tên, giá, cấu hình chi tiết, mô tả, hình ảnh.
* Cho phép xem đánh giá, nhận xét từ người dùng khác.

#### Giỏ hàng

* Thêm sản phẩm vào giỏ hàng.
* Cập nhật số lượng hoặc xóa sản phẩm khỏi giỏ hàng.
* Tính tổng giá trị đơn hàng trong giỏ.

#### Đặt hàng

* Chọn sản phẩm từ giỏ hàng để tiến hành đặt hàng.
* Cung cấp thông tin giao hàng và chọn hình thức thanh toán (thanh toán khi nhận hàng hoặc ví điện tử).
* Hệ thống tạo đơn hàng, gửi thông báo xác nhận và chuyển đơn hàng vào hệ thống quản lý.

#### Quản lý đơn hàng

* Xem danh sách đơn hàng đã đặt: mã đơn hàng, ngày đặt, trạng thái (đang xử lý, đã giao, đã hủy,...).
* Xem chi tiết đơn hàng, sản phẩm đã mua, tổng tiền.
* Hủy đơn hàng nếu đơn đang trong trạng thái "chờ xử lý".

#### Đánh giá sản phẩm

* Sau khi đơn hàng hoàn tất, người dùng có thể để lại đánh giá (sao, bình luận) cho sản phẩm đã mua.
* Hệ thống kiểm duyệt đánh giá trước khi hiển thị công khai.

## Yêu cầu phi chức năng của hệ thống

Hệ thống website bán laptop không chỉ cần đảm bảo các chức năng chính mà còn phải đáp ứng các yêu cầu phi chức năng nhằm nâng cao trải nghiệm người dùng, đảm bảo hiệu suất và an toàn. Cụ thể:

**Hiệu năng:**

* + Hệ thống phải có khả năng xử lý ít nhất 100 lượt truy cập đồng thời mà không bị gián đoạn.
  + Thời gian phản hồi của mỗi yêu cầu không được vượt quá 3 giây trong điều kiện tải bình thường.
  + Các thao tác như thêm sản phẩm vào giỏ hàng, thanh toán, và tìm kiếm phải được thực hiện trong thời gian tối đa 2 giây.

**Tính khả dụng:**

* + Website cần hoạt động liên tục 24/7, thời gian ngừng bảo trì không quá 2 giờ mỗi tháng.
  + Hệ thống phải hiển thị thông báo bảo trì nếu có, và lưu lại dữ liệu người dùng đang thao tác nếu có thể.

**Tính bảo mật:**

* + Dữ liệu người dùng phải được mã hóa (ví dụ: mật khẩu sử dụng thuật toán băm như bcrypt).
  + Hệ thống yêu cầu xác thực và phân quyền người dùng (quản trị viên, khách hàng, nhà cung cấp nếu có).
  + Phải chống lại các lỗ hổng phổ biến như SQL Injection, Cross-Site Scripting (XSS), CSRF,...
  + Hỗ trợ xác thực 2 lớp (nếu có) hoặc xác minh qua email trong quá trình đăng ký.

**Tính dễ sử dụng:**

* + Giao diện người dùng thân thiện, dễ thao tác trên cả máy tính và thiết bị di động.
  + Hỗ trợ tiếng Việt (và có thể mở rộng thêm tiếng Anh nếu cần).
  + Có chức năng hỗ trợ khách hàng trực tuyến hoặc chatbot hỗ trợ tư vấn.

**Khả năng mở rộng:**

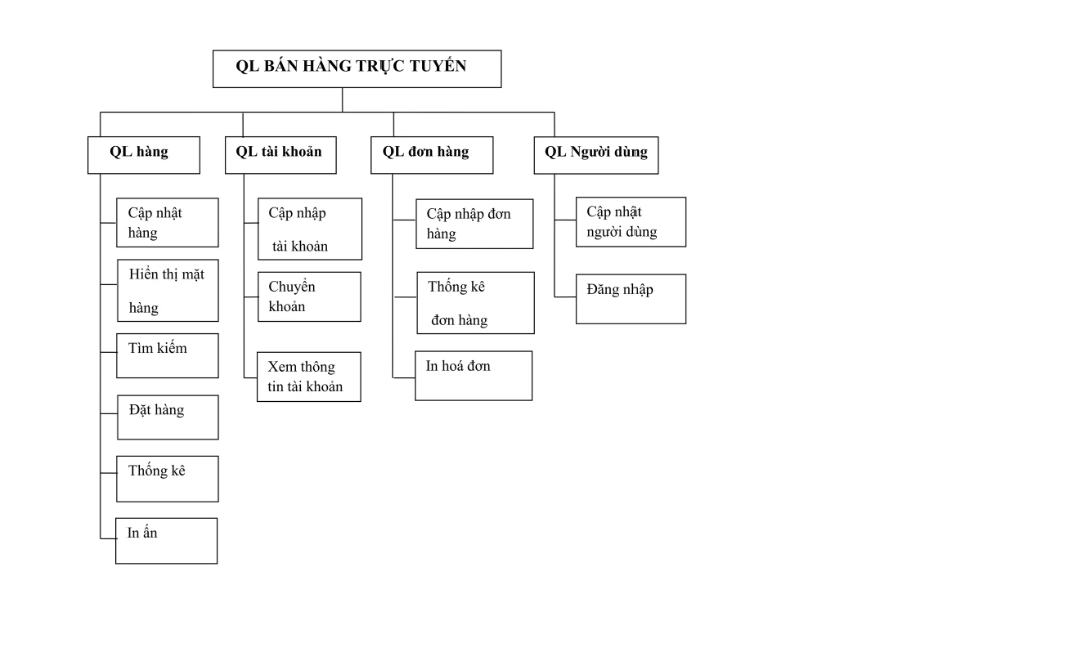
* + Hệ thống cần được thiết kế theo hướng dễ dàng mở rộng như thêm sản phẩm, phân loại mới, module khuyến mãi hoặc tích hợp các cổng thanh toán khác.
  + Cấu trúc cơ sở dữ liệu và mã nguồn đảm bảo khả năng mở rộng mà không ảnh hưởng đến hiệu suất.

**Tính tương thích:**

* + Website phải hoạt động tốt trên các trình duyệt phổ biến như Chrome, Firefox, Edge, Safari.
  + Đảm bảo tương thích trên cả thiết bị máy tính, máy tính bảng và điện thoại di động.

## Phân tích thiết kế hệ thống

### Sơ đồ phân cấp chức năng



Hình 3. 1 Sơ đồ phân cấp chức năng

Sơ đồ phân cấp chức năng mô tả các chức năng cơ bản của một website bán hàng sẽ có, cơ bản sẽ có bốn chức năng cơ bản là quản lý hàng, quản lý tài khoản, quản lý đơn hàng đặt vào, quản lý người dùng cuối

- Quản lý hàng hay còn gọi là quản lý sản phẩm bao gồm các chức năng cơ bản:

+ Cập nhật hàng

+ Hiển thị hàng

+ Tìm kiếm hàng

+ Đặt hàng

+ Thống kê hàng hóa

+ In ấn

- Quản lý tài khoản sẽ quản lý tài khoản của người dùng bao gồm các chức năng cơ bản:

+ Cập nhật tài khoản

+ Chuyển khoản hay gọi là thanh toán online

+ Xem thông tin tài khoản cá nhân

- Quản lý đơn hàng bao gồm quản lý đơn hàng đã đặt và có các chức năng cơ bản:

+ Cập nhật đơn hàng

+ Thống kê đơn hàng

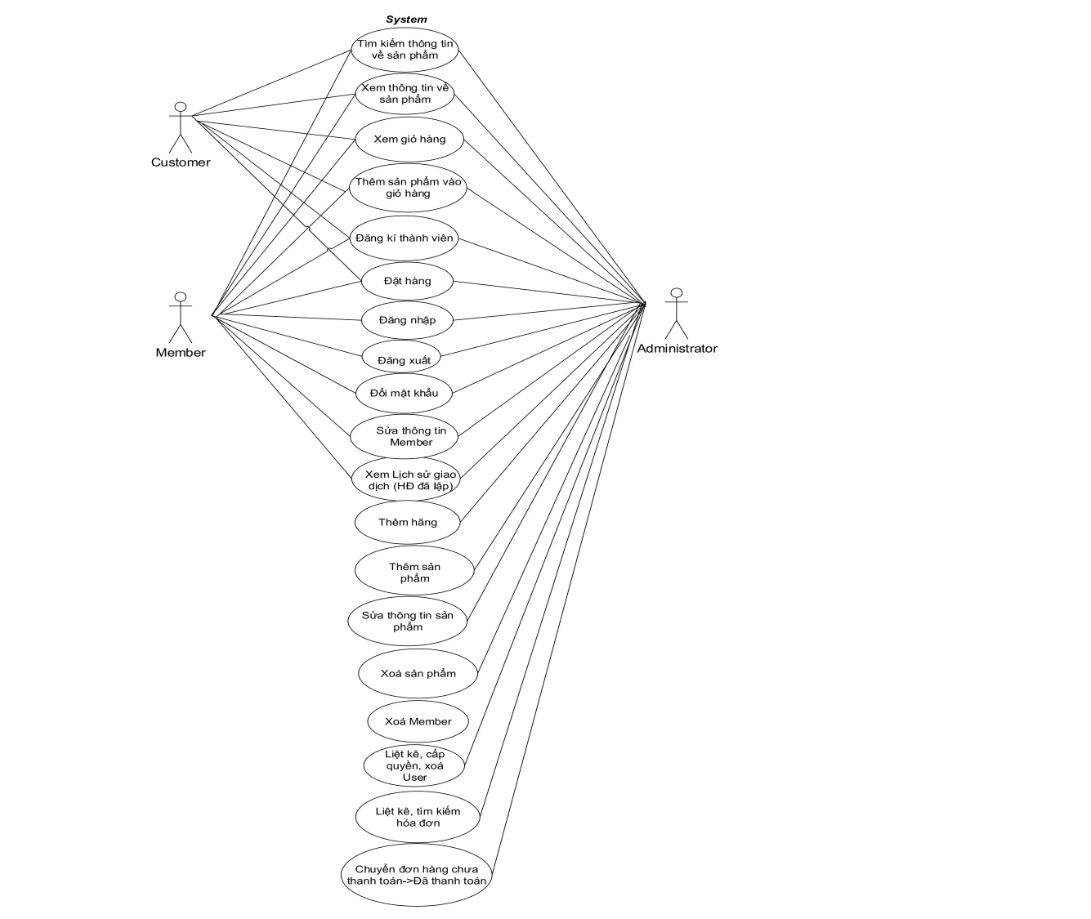
+ In hóa đơn

- Quản lý người dùng:

+ Cập nhật người dùng

+ Đăng nhập

### Sơ đồ use case chung cho cả hệ thống



Hình 3. 2 Sơ đồ useCase

Sơ đồ useCase gồm ba tác nhân chính:

- **Tác nhân Customer (khách vãng lai):** tác nhân này là khách mua hàng không đăng nhập vào hệ thống và có ít chức năng nhất, tác nhân này có các chức năng chính:

+ Tìm kiếm thông tin sản phẩm: cho phép tìm thông tin của một sản phẩm

+ Xem thông tin chi tiết của một sản phẩm có trong hệ thống

+ Xem số lượng sản phẩm bên trong giỏ hàng của mình

+ Thêm sản phẩm vào giỏ hàng

+ Và có thể tự nâng cấp thành tác nhân Member (thành viên) nếu tự đăng ký thành viên

- **Tác nhân Member (Thành viên):** tác nhân này là khách hàng chính của hệ thống . Và sở hữu các quyền của tác nhân Customer và nhiều hơn Customer một số quyền như:

+ Đăng nhập, đăng xuất tài khoản cá nhân

+ Quản lý tài khoản cá nhân của mình, có thể đổi thông tin tài khoản,…

+ Xem lại lịch sử giao dịch cá nhân của mình để quản lý thông tin đơn hàng cá nhân

**- Tác nhân Administrator:** hay còn gọi là quản trị viên sẽ có tất cả chức năng của các tác nhân trên và thêm các chức năng khác như là:

+ Thêm sửa xóa thông tin sản phẩm

+ Nhập sản phẩm

+ Thêm sửa xóa và phân quyền cho một người dùng khác

+ Quản lý các chức năng khác như voucher, danh mục, thống kê doanh thu, in hóa đơn,….

+ Quản lý đơn hàng và cập nhật đơn hàng cho người dùng

### Sơ đồ thực thể kết hợp ERD

Hình 3. 3 Sơ đồ thực thể kết hợp ERD

### Sơ đồ vật lý



Hình 3. 4 Sơ đồ vật lý

### Thông tin các bảng của cơ sở dữ liệu

**Mô tả chi tiết các bảng**

Bảng 3. Thông tin bảng user

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | emailAddress | Email |  | Text |  |  |
| 3 | avatar | Hình dại diện |  | Text |  |  |
| 4 | firstName | Tên |  | Text |  |  |
| 5 | lastName | Họ |  | Text |  |  |
| 6 | fullName | Họ và tên |  | Text |  |  |
| 7 | phoneNumber | Số điện thoại |  | Text |  |  |
| 8 | Address | Địa chỉ |  | Text |  |  |
| 9 | CodeAddress | Mã địa chỉ |  | Text |  |  |
| 10 | points | Điểm |  | Integer |  |  |
| 11 | passwordHash | Mật khẩu |  | Text |  |  |
| 12 | status | Trạng thái |  | Text |  |  |

Bảng 3. Thông tin bảng role

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** |  | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1. | id | id | Bắt buộc | Int |  | Khóa chính |  |
| 2. | roleName | Tên phân quyền |  | Text |  |  |  |
| 3 | listPermission | Danh sách quyền |  | Text |  |  |  |

Bảng 3. Thông tin bảng product

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | Name | Tên |  | Text |  |  |
| 3 | Price | Giá |  | Text |  |  |
| 4 | image | Hình ảnh |  | Text |  |  |
| 5 | Description | Mô tả |  | Text |  |  |
| 6 | importPrice | Giá nhập |  | Text |  |  |
| 7 | manufactureYear | Năm sản xuất |  | Text |  |  |
| 8 | importQuantity | Số lượng nhập |  | Text |  |  |
| 9 | Type | Loại |  | Text |  |  |
| 10 | categoryId | Mã danh mục |  | Integer |  |  |

Bảng 3. Thông tin bảng laptop

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | productId | Mã sản phẩm | Bắt buộc | Int | Khóa ngoại |  |
| 3 | Cpu | Cpu |  | Text |  |  |
| 4 | Ram | Ram |  | Text |  |  |
| 5 | Storage | Bộ nhớ |  | Text |  |  |
| 6 | Screen | Màn hình |  | Text |  |  |
| 7 | graphics | Đồ họa |  | Text |  |  |
| 8 | Os | Hệ điều hành |  | Text |  |  |
| 9 | Ports | Cổng |  | Text |  |  |
| 10 | battery | Pin |  | Integer |  |  |
| 11 | Weight | Cân nặng |  | Integer |  |  |
| 12 | Warranty | Bảo hành |  | Text |  |  |
| 13 | brand | Thương hiệu |  | Text |  |  |

Bảng 3. Thông tin bảng accessory

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | productId | Mã sản phẩm |  | Text | Khóa ngoại |  |
| 3 | accessoryType | Loại thiết bị |  | Text |  |  |
| 4 | Connection | Kết nối |  | Text |  |  |
| 5 | compatibleWith | Tương thích |  | Text |  |  |
| 6 | brand | Thương hiệu |  | Text |  |  |
| 7 | warranty | Bảo hành |  | Text |  |  |

Bảng 3. Thông tin bảng Cart

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | userId | Id người dùng |  | Text | Khóa ngoại |  |
| 3 | quantity | Số lượng |  | Text |  |  |
| 4 | productId | Id sản phẩm |  | Text |  |  |

Bảng 3. Thông tin bảng wishlist

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | userId | Id người dùng |  | Text | Khóa ngoại |  |
| 3 | productId | Id sản phẩm |  | Text |  |  |

Bảng 3. Thông tin bảng order

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | userId | Id người dùng |  | Text |  |  |
| 3 | Quantity | Số lượng |  | Integer |  |  |
| 4 | orderStatus | Trạng thái đơn hàng |  | Text |  |  |
| 5 | paymentStatus | Trạng thái thanh toán |  | Text |  |  |
| 6 | paymentMethod | Phương thức thanh toán |  | Text |  |  |
| 7 | Promotion | Khuyến mãi |  | Text |  |  |
| 8 | listProducts | Danh sách sản phẩm |  | Text |  |  |
| 9 | userInfo | Thông tin user |  | Text |  |  |
| 10 | orderId | Mã đơn hàng |  | Integer |  |  |
| 11 | plusPoints | Điểm |  | Integer |  |  |
| 12 | note | Ghi chú |  | Text |  |  |

Bảng 3. Thông tin bảng user

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | Name | Tên voucher |  | Text |  |  |
| 3 | Code | Mã code |  | Text |  |  |
| 4 | Type | Loại |  | Text |  |  |
| 5 | startDate | Ngày bắt đầu |  | date |  |  |
| 6 | expiryDate | Ngày kết thúc |  | date |  |  |
| 7 | usageCount | Số lượng dùng |  | Integer |  |  |
| 8 | minOrderValue | Giá trị tối thiểu |  | Integer |  |  |
| 9 | discountAmount | Lượng giảm |  | Integer |  |  |
| 10 | discountPercent | Phần trăm giảm |  | Integer |  |  |
| 11 | maxDiscountValue | Giảm tối đa |  | Integer |  |  |

Bảng 3. Thông tin bảng comment

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | content | Nội dung |  | Text |  |  |
| 3 | Status | Trạng thái |  | Text |  |  |
| 4 | Rating | Điểm |  | Text |  |  |
| 5 | productId | Mã sản phẩm |  | date |  |  |
| 6 | userId | Id người dùng |  | date |  |  |

Bảng 3. Thông tin bảng feedback

| **TT** | **Thuộc tính** | **Diễn giải** | **Loại giá trị** | **Kiểu dữ liệu** | **Miền giá trị** | **Chiều dài** |
| --- | --- | --- | --- | --- | --- | --- |
| 1. | Id | id | Bắt buộc | Int | Khóa chính |  |
| 2. | content | Nội dung |  | Text |  |  |
| 3 | Status | Trạng thái |  | Text |  |  |
| 4 | Rating | Điểm |  | Text |  |  |
| 5 | productId | Mã sản phẩm |  | date |  |  |
| 6 | userId | Id người dùng |  | date |  |  |

## Các bước cài đặt

1. Cài đặt trên local

Để có thể cài đặt và chạy được dự án trên máy trước hết phải cài đặt node.js qua  
đường dẫn: <https://nodejs.org/en>.

Bước 1: Clone dự án từ github qua câu lệnh từ repo:  
 https://github.com/tinthanhtv2014/KHOALUANTOTNGHIEP  
 Bước 2: Mở terminal tại thư mục microservice và chạy dòng lệnh cho từng terminal tương ứng với từng service. Bao gồm năm service chính là order, user, product, content và address, trỏ vào từng package.json và chạy lệnh bên dưới để cài đặt

$ npm install

Bước 3: Mở terminal tại thư mục client và chạy dòng lệnh.

$ npm install

$ npm start

Bước 4: Mở terminal tại thư mục admin cho từng service và chạy dòng lệnh.

$ npm install

$ npm run dev

Bước 5: Với frontend Reactjs thì sẽ được chạy ở cổng 3001

- Với các service khác:

+ User service: chạy ở cổng 5001

+ Product service: chạy ở cổng 5002

+ Order service: chạy ở cổng 5003

+ Content service: chạy ở cổng 5004

+ Address service: chạy ở cổng 5005

1. Cài đặt trên Docker

Bước 1: Clone dự án từ github qua câu lệnh từ repo:  
 <https://github.com/tinthanhtv2014/KHOALUANTOTNGHIEP>

Bước 2: Mở terminal tại thư mục microservice và chạy dòng lệnh bên dưới để tiến hành chạy build src dự án lên docker

$ docker-compose up --build

Lệnh này sẽ giúp vừa build và vừa chạy docker cùng lúc

Bước 3: Mở terminal tại thư mục client và chạy dòng lệnh.

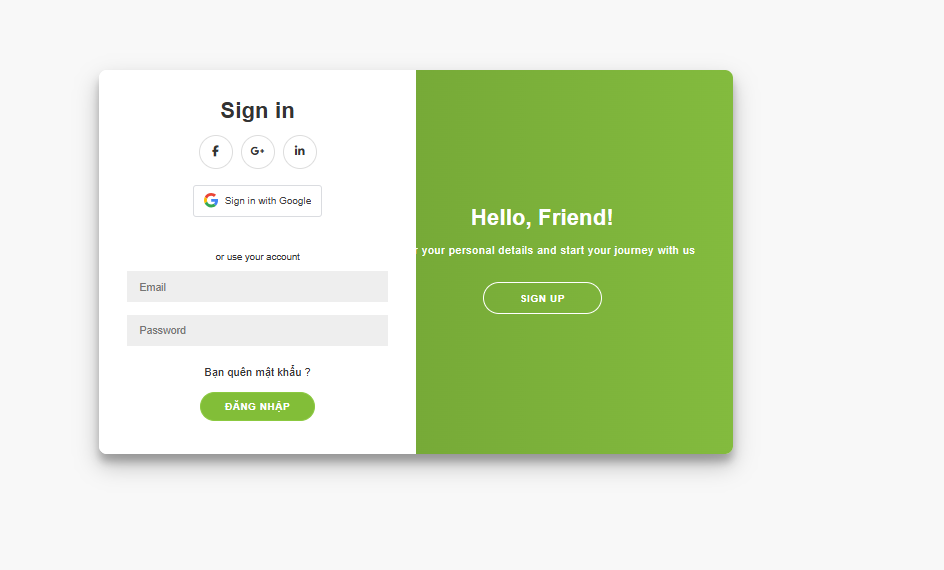
$ docker build -t react-client .

Lệnh này sẽ giúp build docker dành cho client

Bước 4: Vào docker desktop (nếu có) hoặc có thể vào terminal của client để có thể chạy vào đường link bên dưới terminal để có thể bật lại dự án trên docker

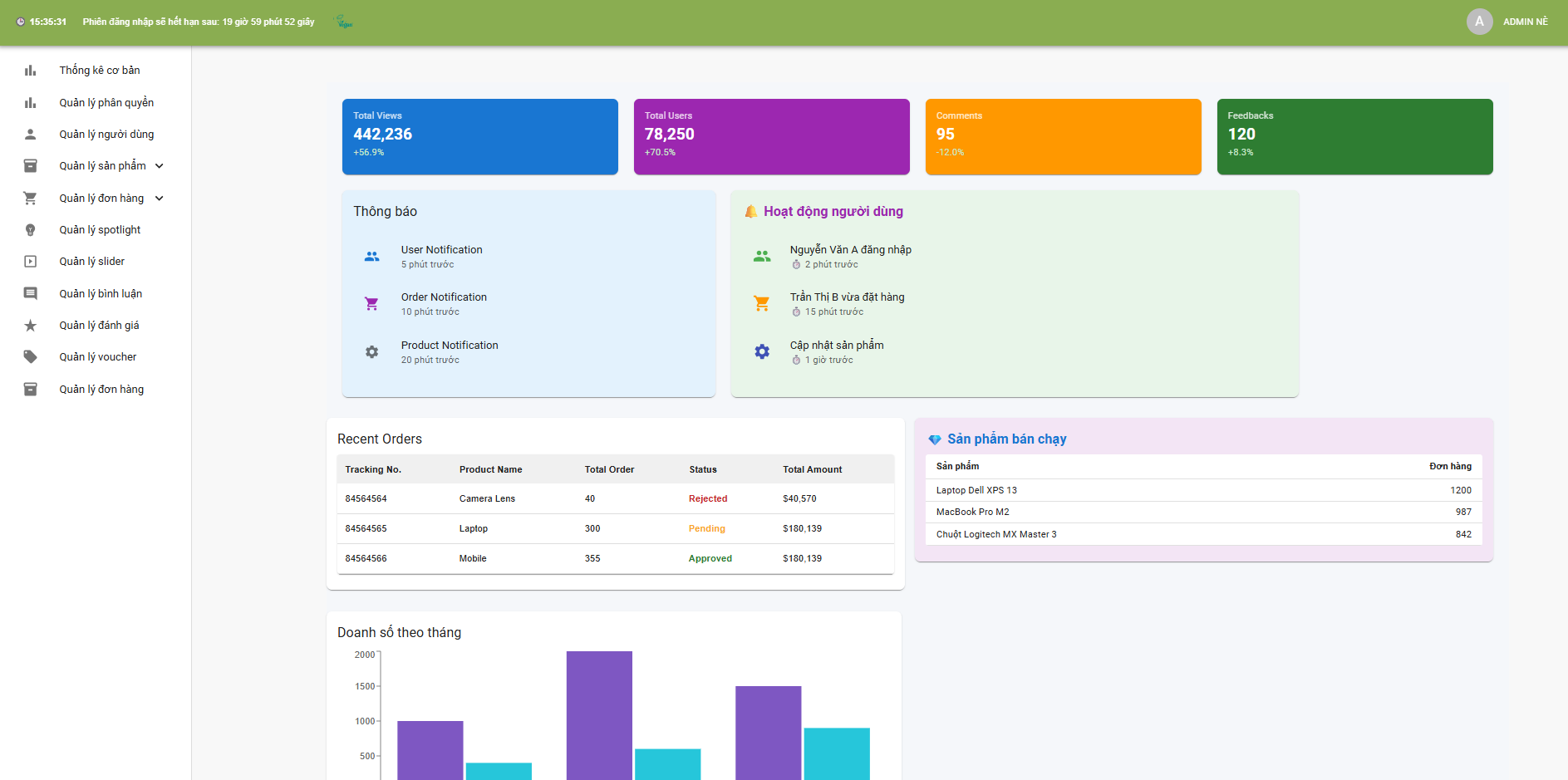
# KẾT QUẢ NGHIÊN CỨU

1. **Giao diện cho người quản lý**



Hình 4. 1 Giao diện đăng nhập admin

Đây là giao diện đăng nhập dành cho người quản lý hệ thống, tài khoản và mật khẩu chỉ có thể được cấp thông qua quản trị viên

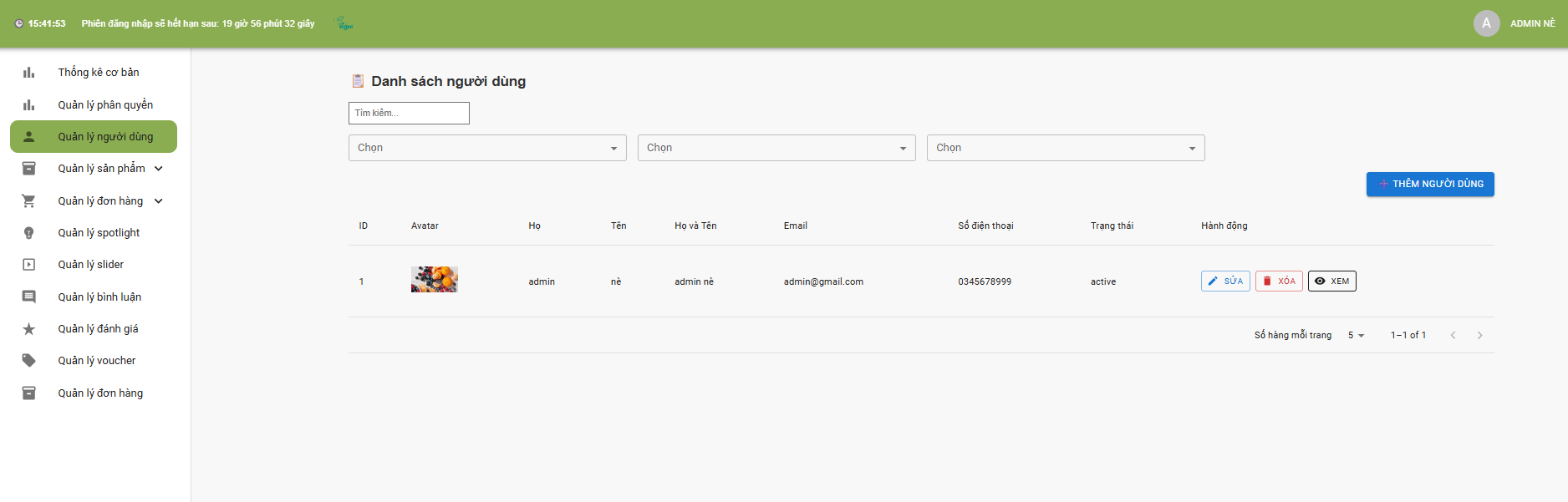


Hình 4. 2 Giao diện thống kê trang chủ admin

Đây là giao diện trang dashboard sau khi đăng nhập vào hệ thống, nó sẽ bao gồm thống kê trang chủ, doanh thu của cửa hàng và các trang quản lý khác bao gồm: Quản lý phân quyền. quản lý người dùng, quản lý sản phẩm sẽ bao gồm quản lý danh mục và sản phẩm, quản lý đơn hàng, quản lý bình luận, quản lý mã khuyến mãi, và quản lý đánh giá.

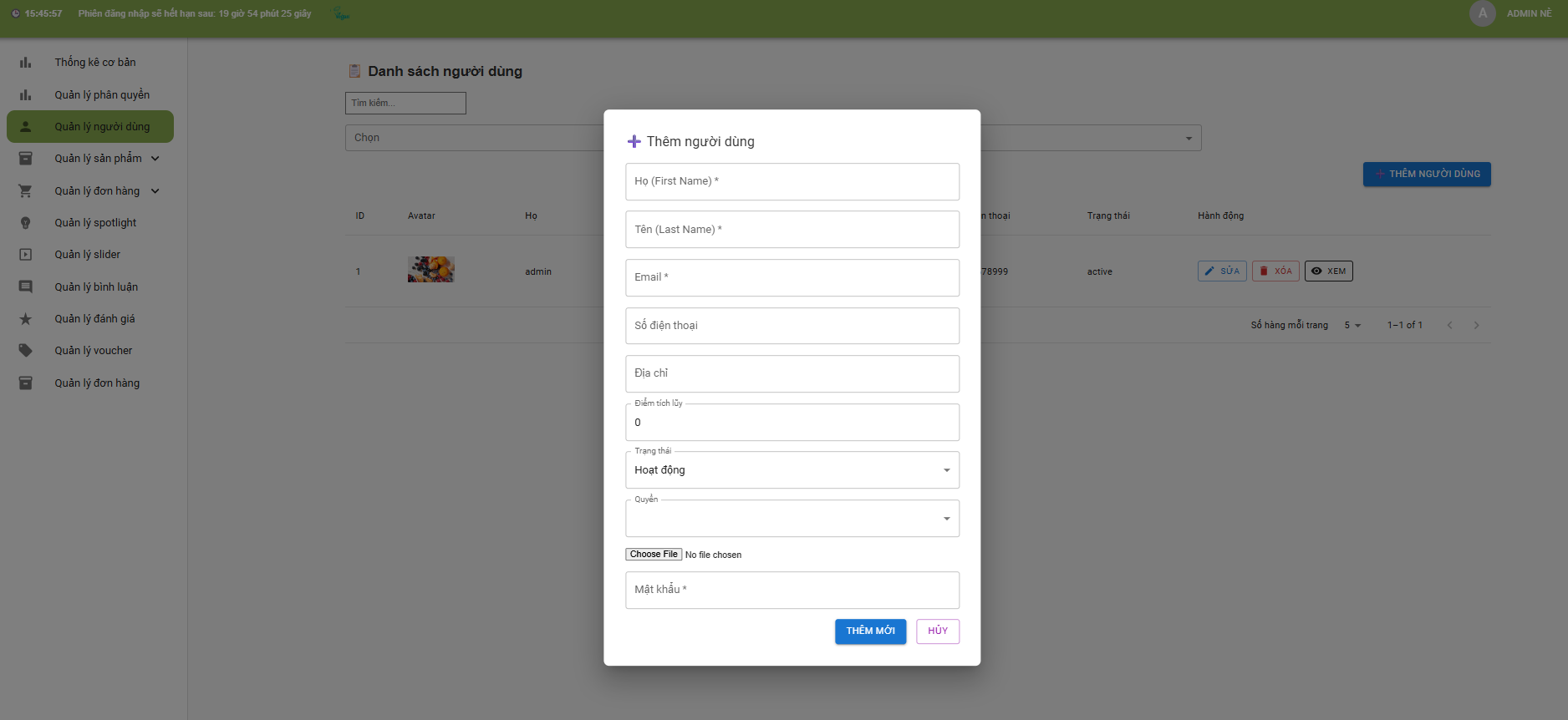
**Quản lý người dùng:**

Bao gồm danh sách người dùng và các thông tin tìm kiếm, lọc và tìm kiếm người dùng, bên dưới là danh sách người dùng và thông tin cơ bản của người dùng đó



Hình 4. 3 Trang quản lý người dùng

Nút thêm và sửa có chức năng thêm và sửa thông tin một người dùng mới, thông tin của người dùng cần thêm cần có các thông tin như: Họ, tên, email, số điện thoại, địa chỉ, điểm tích lũy, trạng thái, phân quyền, hình ảnh và mật khẩu.

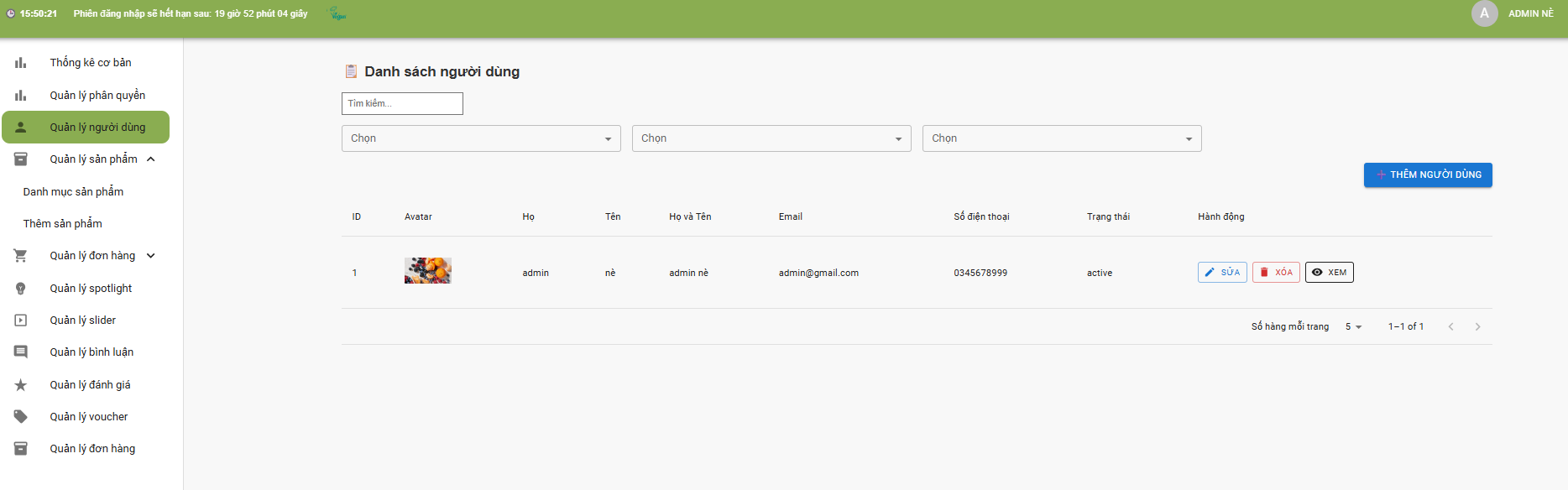


Hình 4. 4 Trang thêm thông tin người dùng mới

Nút xóa sẽ xóa thông tin của người dùng đó khỏi hệ thống.

**Quản lý sản phẩm:**

Bao gồm 2 phần quản lý chính là quản lý danh mục sản phẩm và quản lý sản phẩm

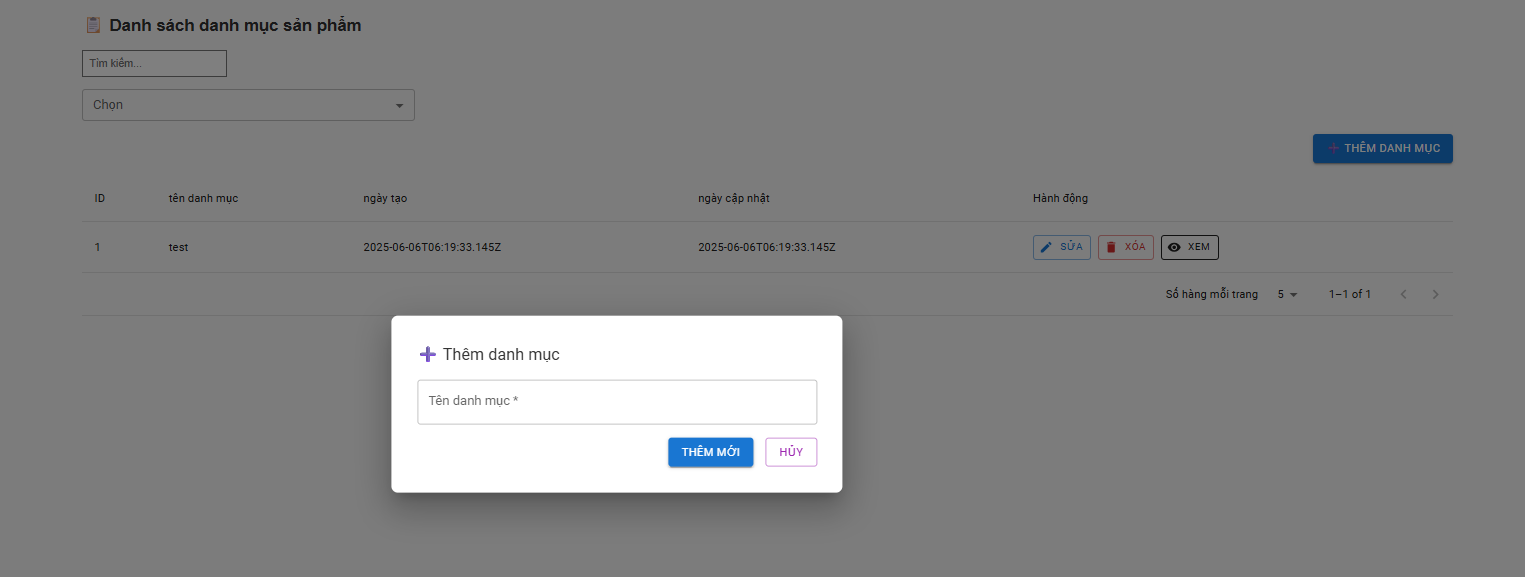


Hình 4. 5 Sau khi thêm sản phẩm

- Quản lý danh mục sẽ quản lý các danh mục của sản phẩm

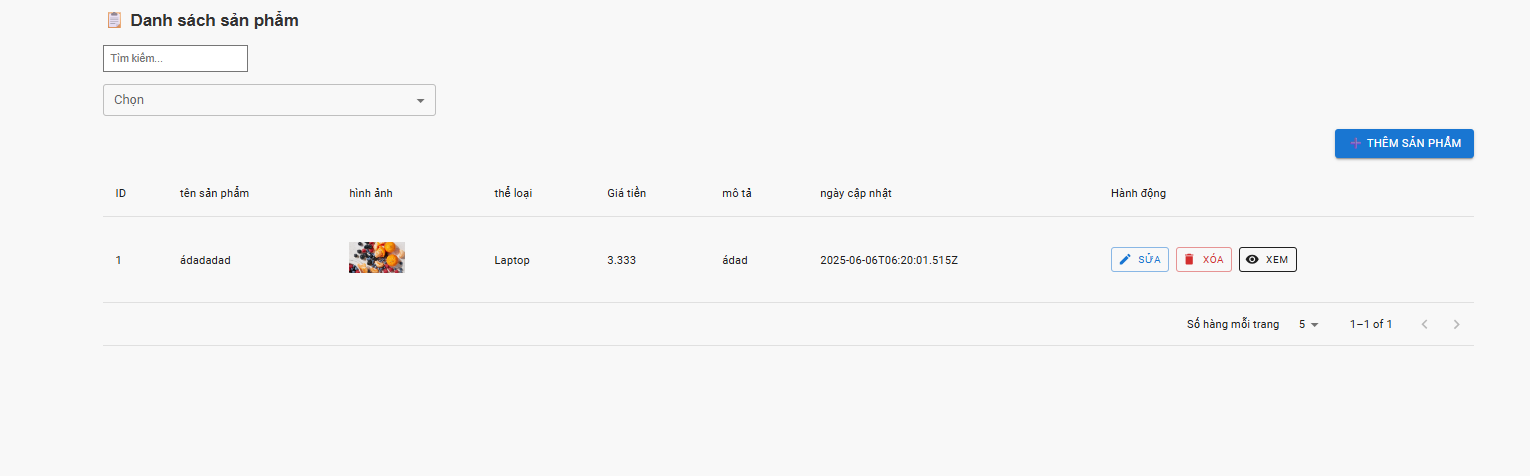
+ Nút thêm và sửa sẽ giúp thêm và sửa thông tin của một danh mục mới

+ Nút xóa sẽ xóa danh mục của sản phẩm đó



Hình 4. 6 Quản lý danh mục

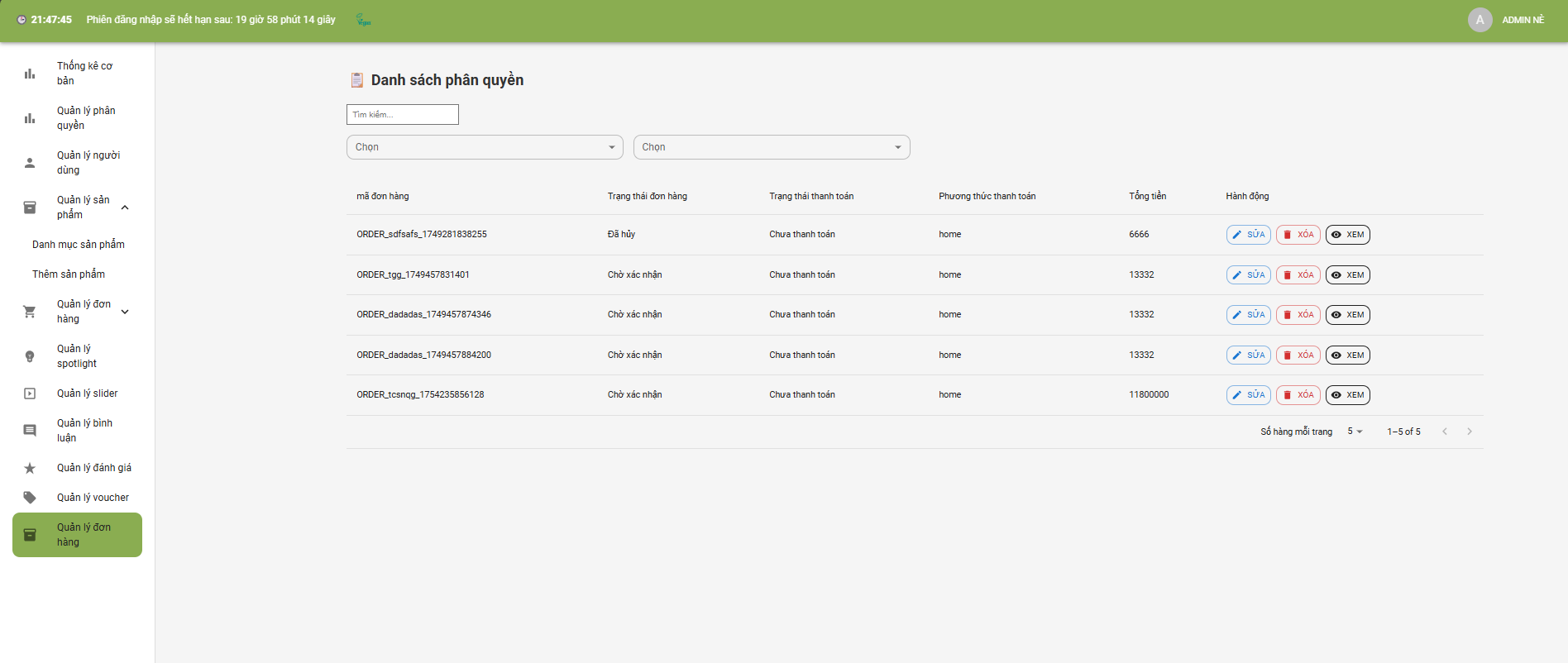
Quản lý danh sách sản phẩm sẽ quản lý sản phẩm có bên trong hệ thống



Hình 4. 7 Quản lý danh mục

**Quản lý đơn hàng:**

Quản lý đơn hàng sẽ thống kê các đơn hàng do người dùng tạo, bên trong sẽ có thông tin của đơn hàng

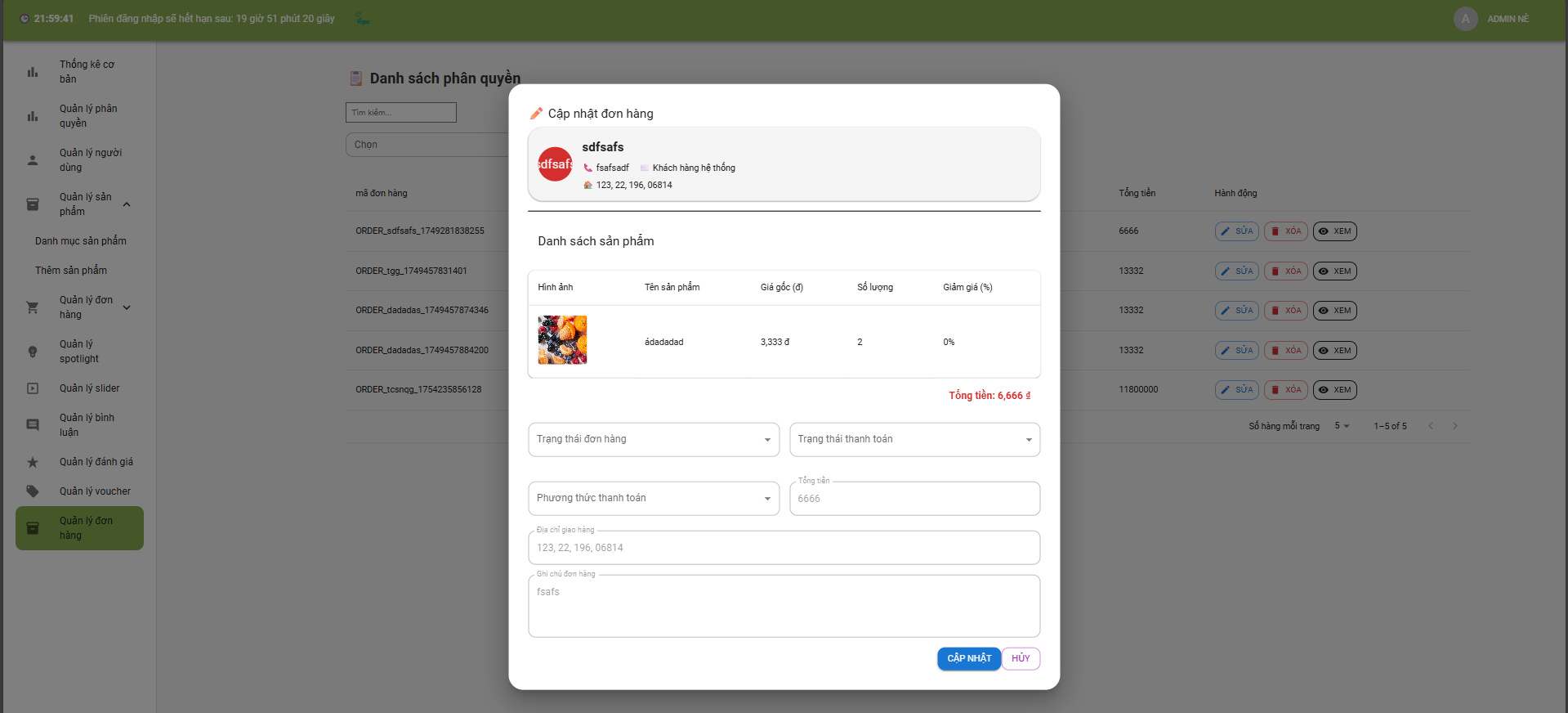


Hình 4. 8 Quản lý đơn hàng

Khi bấm vào nút cập nhật thì quản trị viên có thể cập nhật các trường như phương thức thanh toán, trạng thái thanh toán, trạng thái đơn hàng, các trường còn lại chỉ có thể xem để xác nhận với người dùng.

Khi bấm nút xem thì sẽ xem lại thông tin hiện tại, không thể cập nhật.

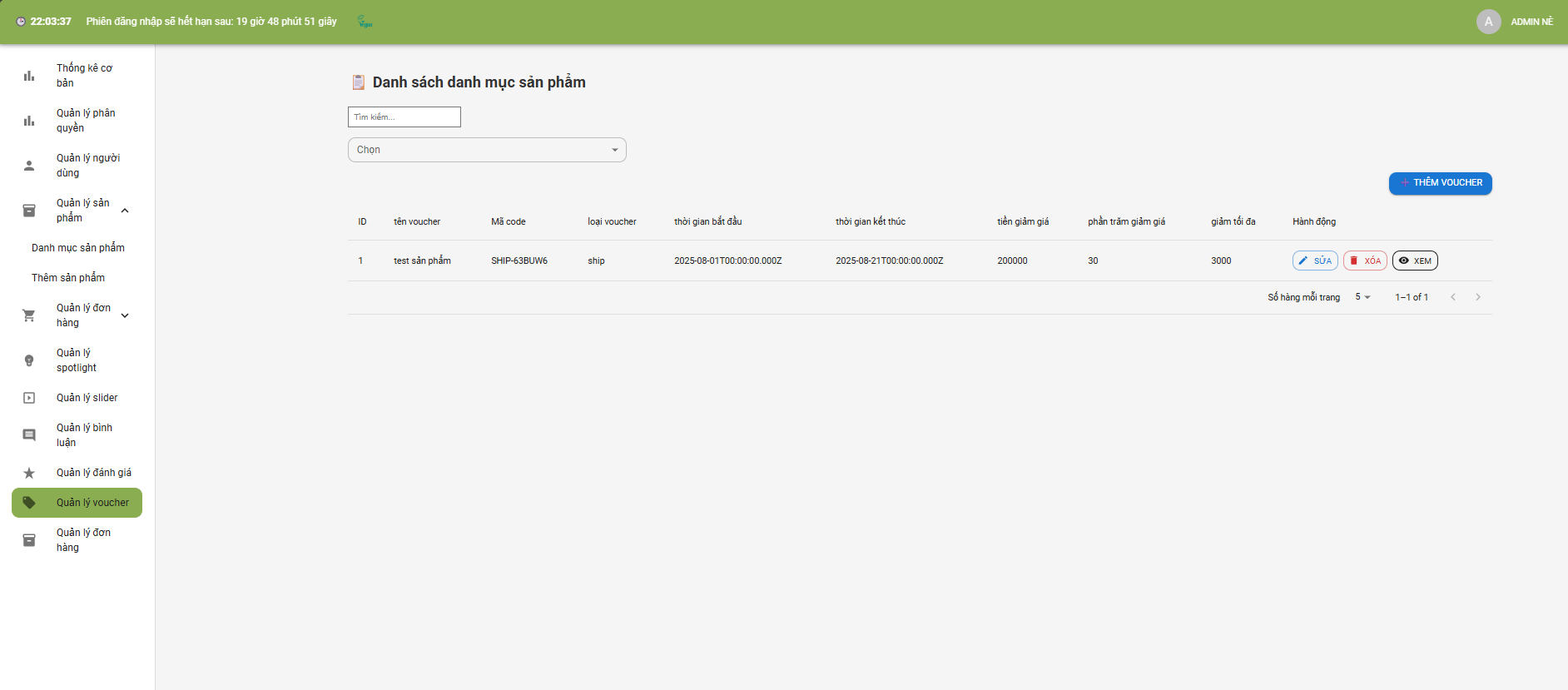
Khi bám nút xóa thì sẽ xóa đơn hàng khỏi hệ thống



Hình 4. 9 Thông tin chi tiết đơn hàng

**Quản lý mã khuyến mãi**

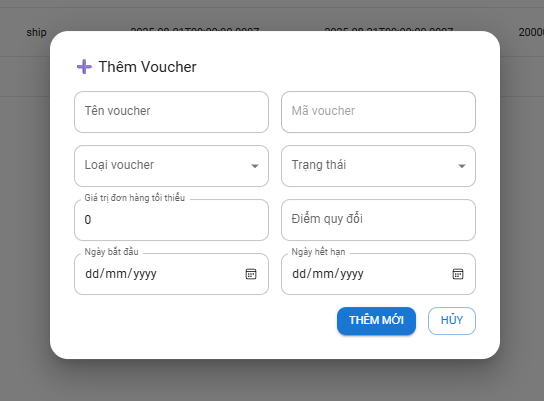
Đây là giao diện thêm mã khuyến mãi mới cho hệ thống



Hình 4. 10 Quản lý mã khuyến mãi

Bên trong là thông tin của danh sách mã khuyến mãi của hệ thống, sẽ có thông tin của từng mã khuyến mãi,

Khi bấm nút tạo thì sẽ tạo thông tin của một mã khuyến mãi mới



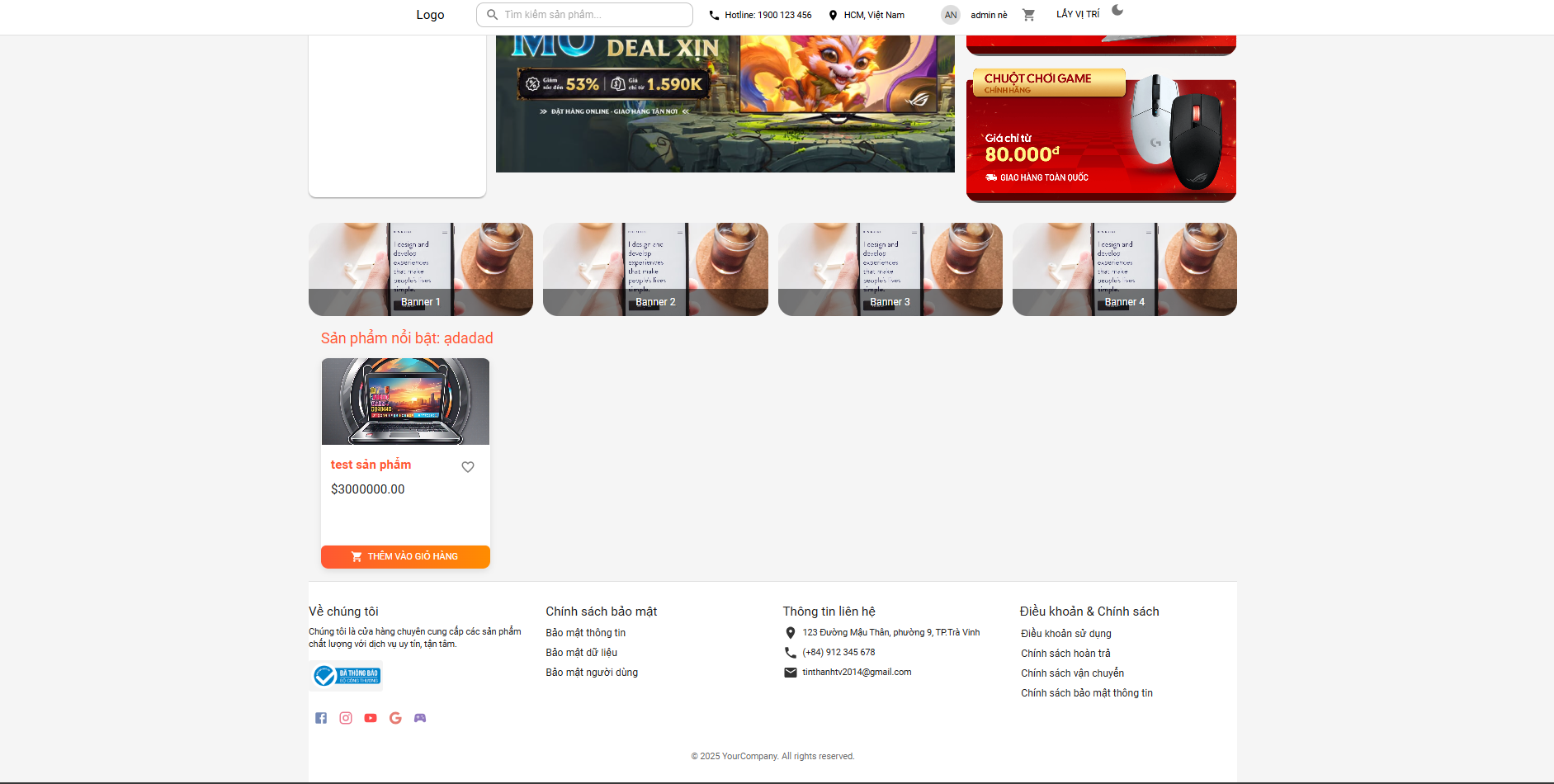
Hình 4. 11 Modal nhập mã khuyến mãi

Khi tạo sẽ nhập thông tin cần thiết để tạo voucher như là tên, mã, loại voucher, trạng thái, điểm, ngày tháng bắt đầu và kết thúc,…Khi nhập xong sẽ có thể tạo được voucher

Khi bấm vào cập nhật sẽ có thể cập nhật lại thông tin của một voucher

Khi xóa sẽ xóa voucher khỏi hệ thống

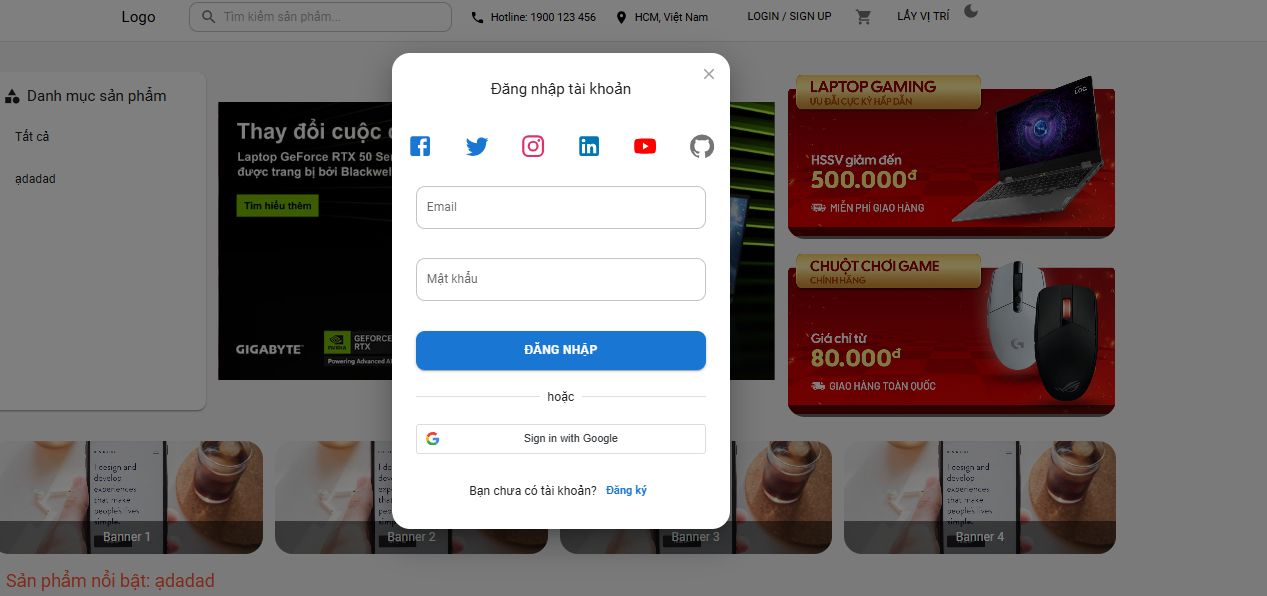
1. **Giao diện cho người dùng**

****

Hình 4. 12 Trang chủ

Đây là giao diện trang chủ của hệ thống dành cho người dùng

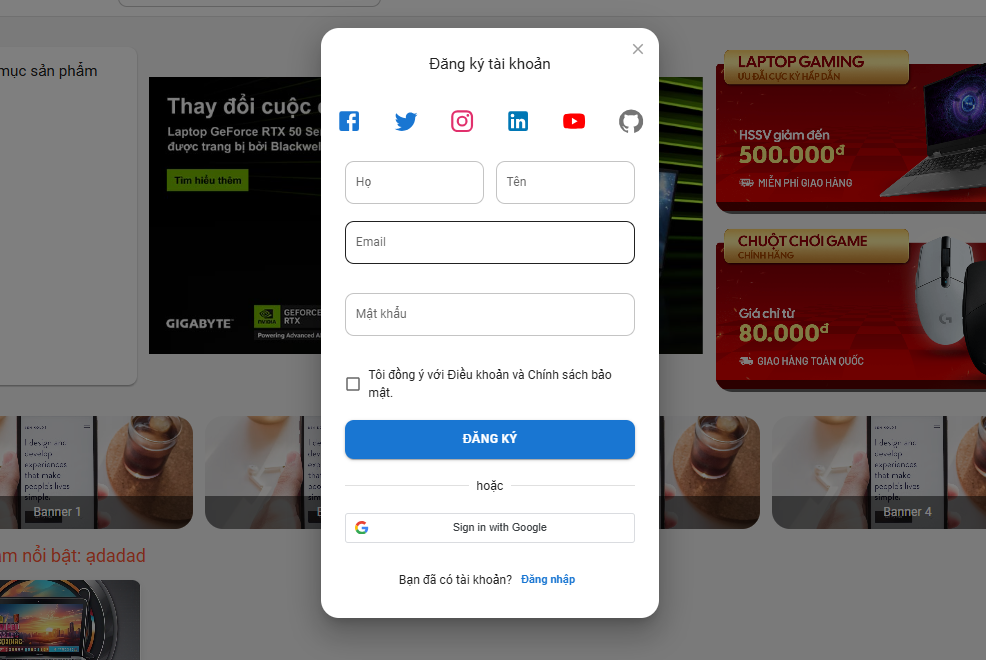
Ở phần trên cùng là thanh header của hệ thống, ở trên sẽ gồm phần logo của cửa hàng, thanh tìm kiếm sản phẩm, thông tin số điện thoại, địa chỉ và nút đăng nhập, dăng ký. Trường hợp nếu người dùng đã đăng nhập thì sẽ hiện tên và cái icon tên của người dùng, kế bên là nút giỏ hàng và nút lấy vị trí hiện tại của người dùng, kế bên là nút chuyển nền trắng đen.



Hình 4. 13 Giao diện đăng nhập

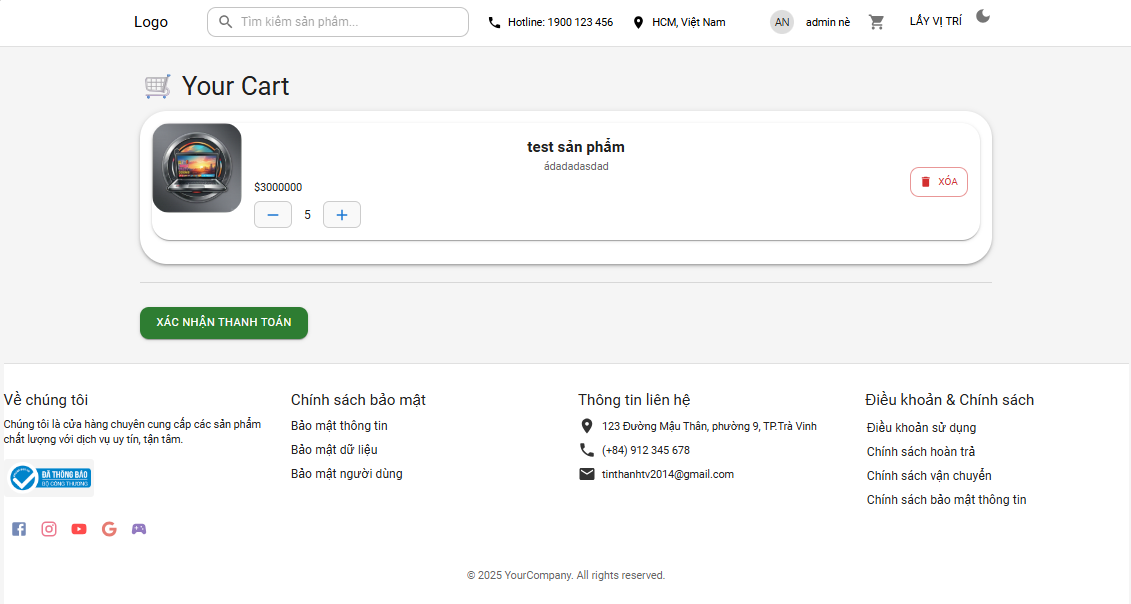
Khi bấm vào nút login sẽ hiện lên modal đăng nhập, người dùng cần nhập email và mật khẩu để đăng nhập vào hệ thống, hoặc có thể chọn đăng nhập bằng google nếu có tài khoản.

Trường hợp chưa có tài khoản có thể bấm vào nút đăng ký để chuyển qua modal đăng ký tài khoản mới



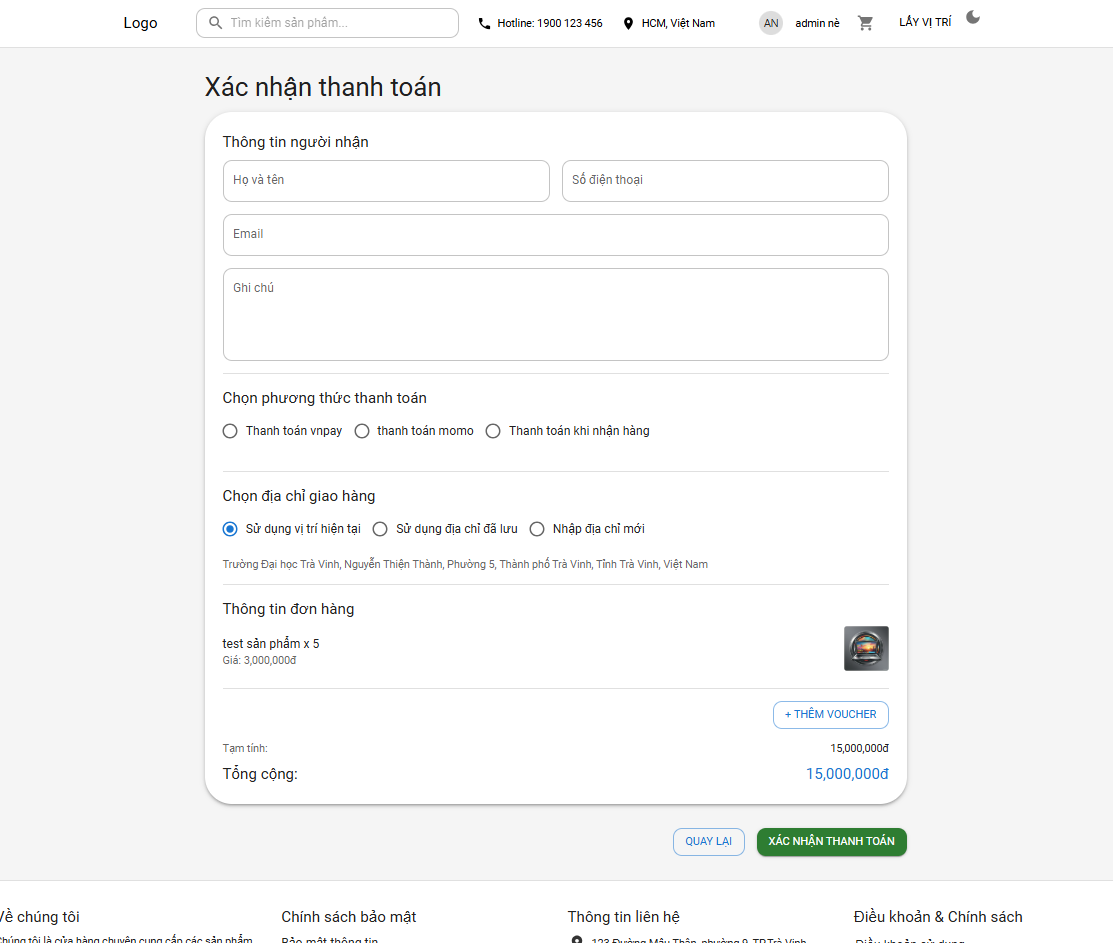
Hình 4. 14 Giao diện đăng ký

Ở bên dưới danh sách sản phẩm có thể bấm thêm sản phẩm vào giỏ hàng để thêm sản phẩm vào giỏ hàng



Hình 4. 15 Giao diện giỏ hàng

Khi thêm vào giỏ hàng thành công thì bấm vào icon giỏ hàng trên header để vào trong giỏ hàng xem thông tin sẽ thấy danh sách sản phẩm thêm vào giỏ hàng, có thể tăng số lượng hoặc xóa sản phẩm đó đi, nếu đã xác nhận đúng có thể chuyển sang thanh toán.



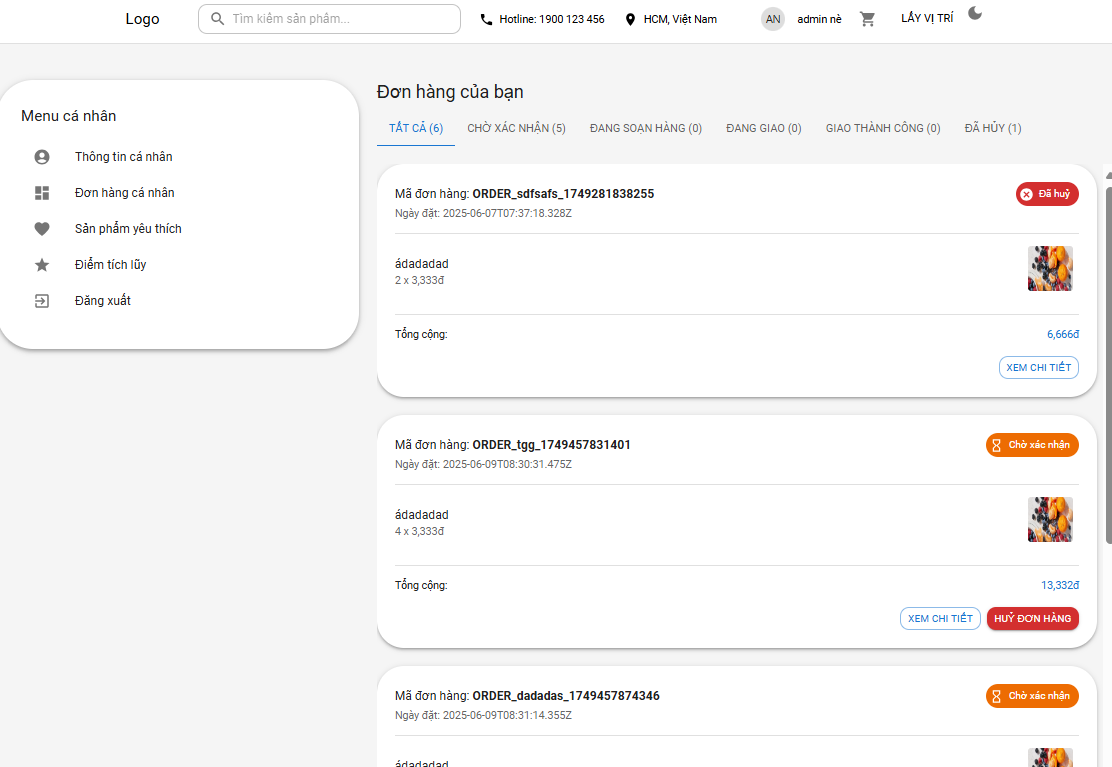
Hình 4. 16 Giao diện thanh toán

Khi chuyển sang trang thanh toán, người dùng cần điền thông tin cá nhân của mình vào để tiến hành thanh toán

Người dùng có thể chọn các phương thức thanh toán như momo, vnpay, hoặc giao tại nhà để tiến hành thanh toán

Về phần địa chỉ người dùng có thể lấy địa chỉ hiện tại, hoặc địa chỉ trong thông tin cá nhân, hoặc địa chỉ tự nhập để giao hàng

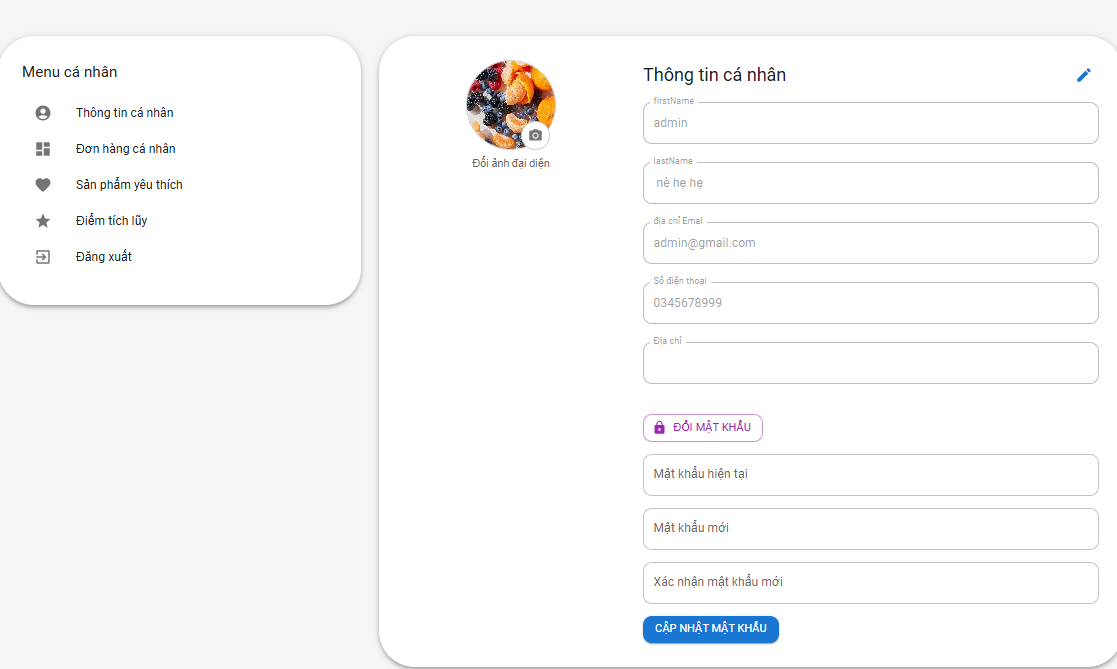
Sau khi xác nhận đúng thông tin thì người dùng có thể thanh toán để xác nhận đơn hàng



Hình 4. 17 Giao diện quản lý đơn hàng cá nhân

Khi đặt hàng xong người dùng đăng nhập có thể vào thông tin cá nhân bằng cách bấm vào icon trên header để vào thông tin cá nhân (chỉ dành cho người dùng đăng nhập)

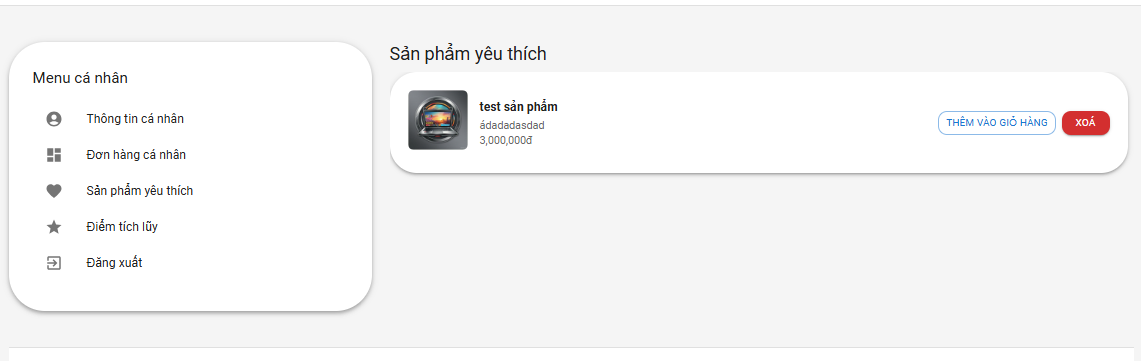
Bên trong sẽ có thanh menu bên trái ở mục đơn hàng cá nhân người dùng có thể xem đơn hàng mình vừa đặt và trạng thái của nó, người dùng có thể tiến hành hủy đơn nếu đơn hàng ở trạng thái chờ duyệt, khi đã duyệt rồi thì không hủy được nữa.



Hình 4. 18 Giao diện quản lý thông tin cá nhân

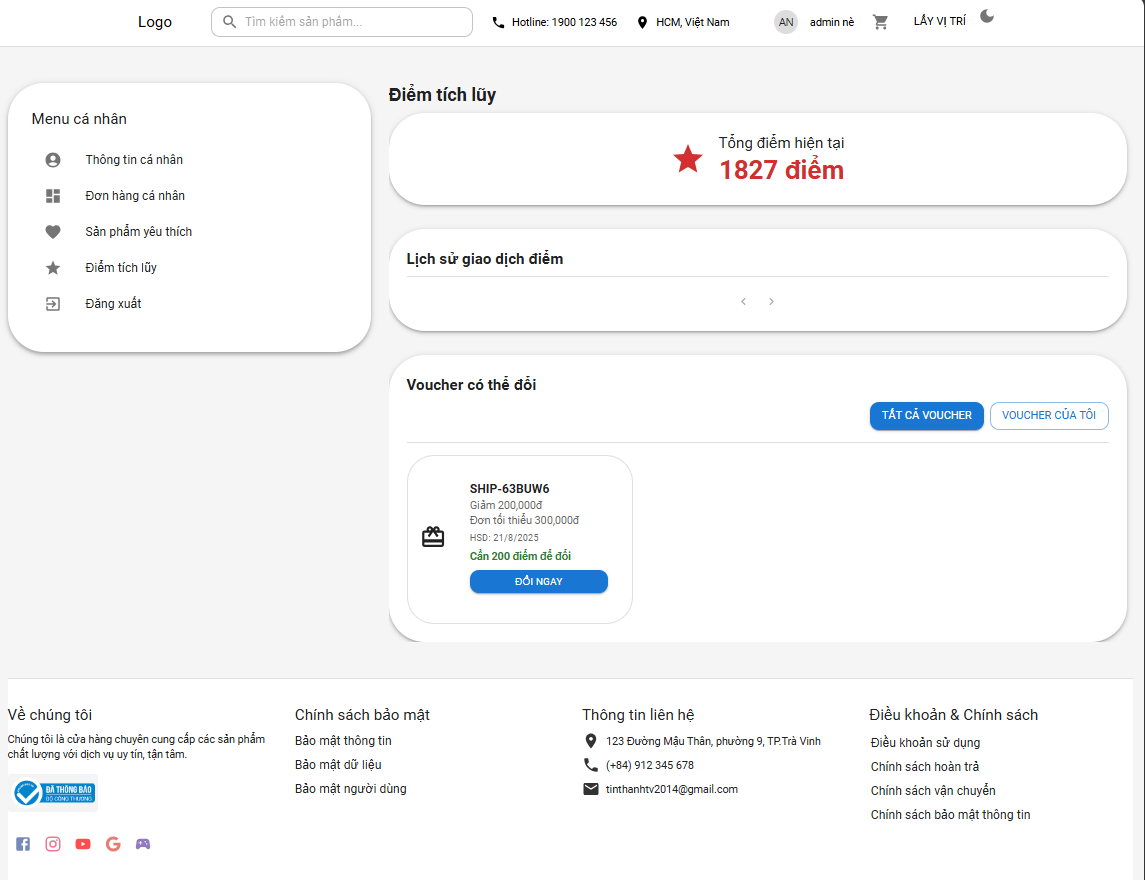
Về phần menu thông tin cá nhân khi bấm vào người dùng có thể xem thông tin hiện tại của mình

Người dùng có thể đổi thông tin bằng cách bấm vào cây bút màu xanh để cập nhật thông tin, cũng có thể đổi mật khẩu bằng cách bấm vào nút đổi mật khẩu



Hình 4. 19 Giao diện quản lý sản phẩm yêu thích

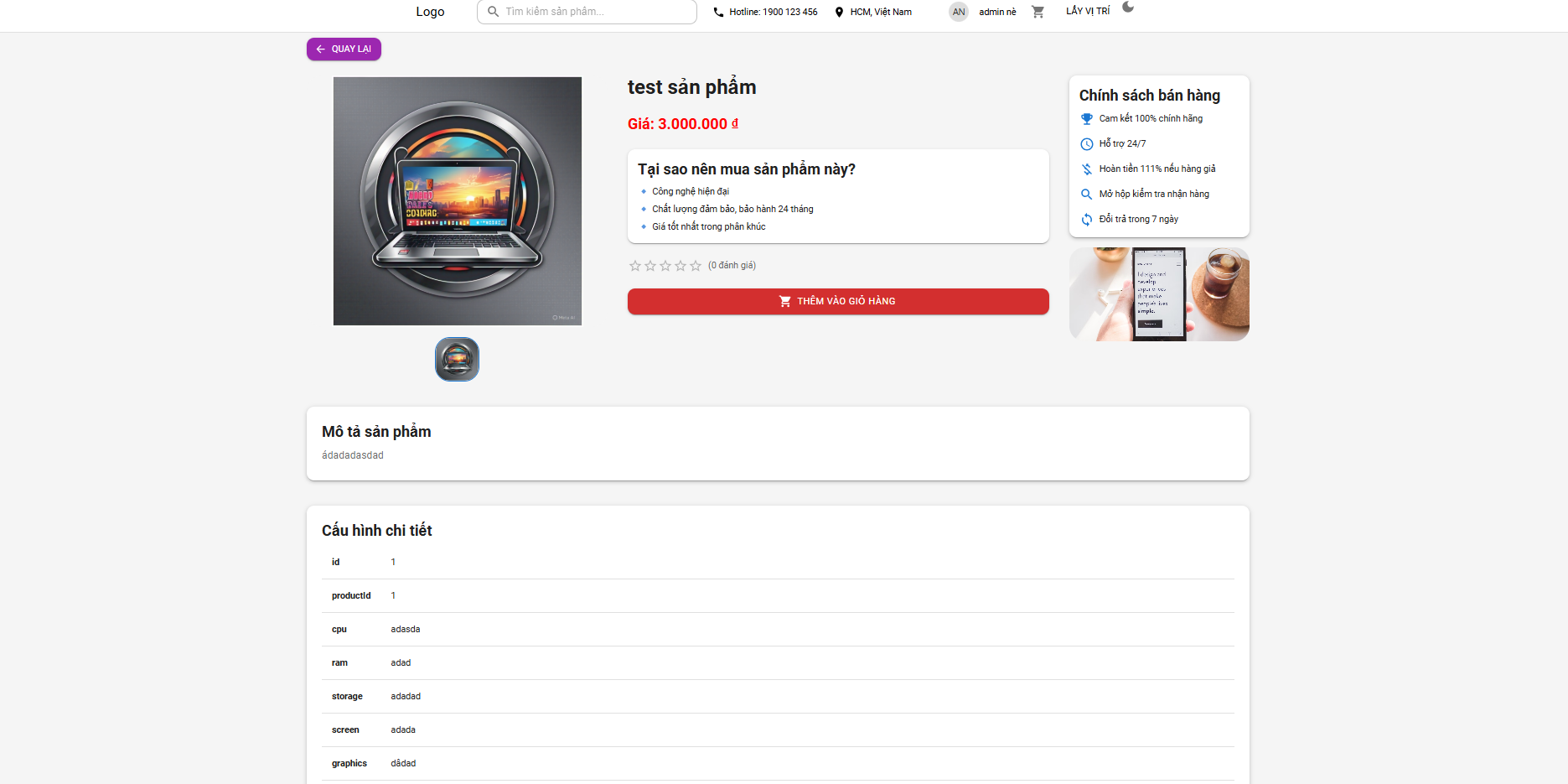
Khi bấm vào menu sản phẩm yêu thích người dùng có thể xem sản phẩm yêu thích cá nhân, và có thể thêm sản phẩm yêu thích vào giỏ hàng của mình hoặc xóa nó đi



Hình 4. 20 Quản lý điểm tích lũy

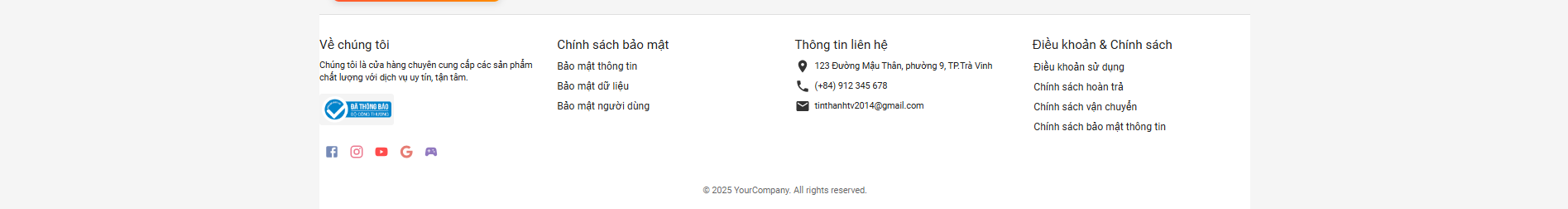
Khi bấm vào điểm tích lũy, người dùng có thể kiểm tra các đơn hàng hoàn thành của mình để xem điểm, điểm có thể được dùng để đổi các mã giảm giá cho các đơn hàng khác.

Người dùng có thể kiểm tra các trạng thái điểm bằng các đơn hàng có trạng thái là thành công.



Hình 4. 21 Thông tin chi tiết sản phẩm

Và trước khi mua hàng người dùng có thể vào chi tiết sản phẩm để xem thông tin chi tiết của sản phẩm đó, các thông số của sản phẩm, nếu xác nhận mua thì có thể bấm vào nút thêm vào giỏ hàng



Hình 4. 22 giao diện footer

Về cuối cùng là footer sẽ ghi các thông tin cơ bản của cửa hàng, các chính sách bảo mật và thông tin cơ bản của cửa hàng.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết quả và đóng góp của đồ án

Sau quá trình nghiên cứu và phát triển, đồ án “Xây dựng hệ thống website bán laptop” đã đạt được các kết quả và đóng góp cụ thể như sau:

**Kết quả đạt được:**

* + Xây dựng thành công một website thương mại điện tử chuyên bán laptop, bao gồm các chức năng cơ bản như:
  + Đăng ký, đăng nhập và quản lý tài khoản người dùng.
  + Xem danh sách sản phẩm, chi tiết sản phẩm, phân loại theo hãng, giá, cấu hình.
  + Tìm kiếm và lọc sản phẩm theo nhu cầu người dùng.
  + Thêm vào giỏ hàng, đặt hàng, theo dõi đơn hàng.
  + Quản trị hệ thống: quản lý sản phẩm, đơn hàng, người dùng và khuyến mãi.
  + Hệ thống hỗ trợ giao diện thân thiện, dễ sử dụng, tương thích với nhiều thiết bị khác nhau.
  + Đảm bảo một số yêu cầu phi chức năng quan trọng như hiệu năng ổn định, bảo mật thông tin người dùng, và khả năng mở rộng về sau.

**Đóng góp của đồ án:**

* + Góp phần cung cấp một nền tảng bán hàng trực tuyến hiệu quả dành cho các cửa hàng kinh doanh laptop, đặc biệt trong bối cảnh thương mại điện tử ngày càng phát triển.
  + Là cơ sở thực tiễn cho việc triển khai các giải pháp thương mại điện tử trong lĩnh vực công nghệ, giúp tiết kiệm chi phí vận hành và mở rộng thị trường.
  + Đồ án là tài liệu tham khảo hữu ích cho sinh viên, lập trình viên hoặc cá nhân có nhu cầu xây dựng website thương mại điện tử từ nền tảng ban đầu.
  + Qua quá trình thực hiện, nhóm thực hiện đồ án đã nâng cao kiến thức và kỹ năng về lập trình web, phân tích hệ thống, thiết kế giao diện, bảo mật cũng như triển khai cơ sở dữ liệu thực tế.

## Hạn chế

Mặc dù đồ án đã hoàn thành đúng mục tiêu đề ra, tuy nhiên trong quá trình thực hiện vẫn còn tồn tại một số hạn chế nhất định như sau:

* + Giao diện người dùng chưa thực sự tối ưu về trải nghiệm (UX/UI): Mặc dù hệ thống hoạt động đầy đủ chức năng, nhưng bố cục, màu sắc và hiệu ứng tương tác còn đơn giản, chưa tạo được ấn tượng mạnh mẽ và chuyên nghiệp như các website thương mại điện tử lớn.
  + Chưa có chức năng hỗ trợ khách hàng trực tuyến: Hệ thống hiện chưa tích hợp chatbot hoặc hệ thống hỗ trợ tư vấn trực tiếp cho khách truy cập.
  + Thiếu tính năng phân tích dữ liệu và báo cáo: Hệ thống chưa có module thống kê doanh thu, lượt truy cập, sản phẩm bán chạy,… hỗ trợ quản trị viên ra quyết định kinh doanh.
  + Thiếu kiểm thử trên quy mô lớn: Hệ thống mới được kiểm thử với số lượng người dùng và dữ liệu hạn chế, chưa đánh giá được hiệu suất trong môi trường thật với lưu lượng truy cập lớn.

## Hướng phát triển

Để nâng cao chất lượng hệ thống và đáp ứng tốt hơn nhu cầu thực tế, trong tương lai website bán laptop có thể được phát triển thêm theo các hướng sau:

* + Phát triển chatbot hỗ trợ khách hàng: Triển khai hệ thống chatbot tự động trả lời câu hỏi, tư vấn sản phẩm hoặc kết nối với nhân viên tư vấn giúp nâng cao trải nghiệm người dùng.
  + Nâng cấp giao diện người dùng (UI/UX): Cải thiện thiết kế giao diện hiện đại, tối ưu cho thiết bị di động (responsive), đồng thời đảm bảo tiêu chuẩn về khả năng truy cập (accessibility).
  + Tích hợp hệ thống đánh giá và phản hồi sản phẩm: Cho phép người dùng để lại nhận xét, đánh giá sao sản phẩm sau khi mua để tạo sự tin cậy và tăng tương tác.
  + Thêm chức năng thống kê và báo cáo: Xây dựng các biểu đồ và bảng thống kê giúp quản trị viên theo dõi doanh số bán hàng, lượt truy cập, sản phẩm phổ biến, hành vi người dùng,...
  + Tối ưu SEO và quảng cáo: Nâng cấp hệ thống để thân thiện với các công cụ tìm kiếm (SEO) và dễ dàng tích hợp với các nền tảng quảng cáo như Google Ads, Facebook Ads.
  + Hỗ trợ nhiều ngôn ngữ: Mở rộng khả năng hiển thị đa ngôn ngữ (ví dụ: tiếng Việt, tiếng Anh) để tiếp cận thị trường quốc tế hoặc khách hàng nước ngoài.
  + Triển khai ứng dụng di động: Xây dựng phiên bản mobile app (Android/iOS) để người dùng có thể mua sắm tiện lợi hơn và nhận thông báo khuyến mãi nhanh chóng.

**DANH MỤC TÀI LIỆU THAM KHẢO**

|  |  |
| --- | --- |
| [1] | https://topdev.vn/blog/restful-api-la-gi/, RESTful API là gì? Cách thiết kế RESTful API, Viet Nam: Lập Trình: RESTful API là gì? Cách thiết kế RESTful API, 2024. |
| [2] | https://topdev.vn/blog/react-la-gi-lo-trinh/, React là gì? Lộ trình trở thành lập trình viên ReactJS, Viet Nam: Phạm Minh Khoa, 2024. |
| [3] | https://topdev.vn/blog/node-js-la-gi/, Node.js là gì? Tổng hợp kiến thức NodeJS từ A-Z, Viet Nam: topdev, 2024. |
| [4] | https://www.mcivietnam.com/blog-detail/mysql-la-gi/, MySQL là gì? Cẩm nang toàn tập về MySQL cho người mới bắt đầu, Viet Nam: MCI tổng hợp, 2024. |
| [5] | P. T. P. Nam, tài liệu giảng dạy môn “Hệ quản trị Cơ sở dữ liệu”, trường Đại học Trà Vinh, 2015. |
| [6] | P. M. Đương, tài liệu giảng dạy môn “Phân tích và thiết kế hệ thống, trường Đại học Trà Vinh, 2014. |
| [7] | N. K. Quốc, tài liệu giảng dạy môn “Công nghệ phần mềm”, 2015, trường Đại học Trà Vinh. |