Học viện Công nghệ Bưu chính Viễn Thông

------□□□------

BÁO CÁO PHÁT TRIỂN HỆ THỐNG THƯƠNG MẠI ĐIỆN TỬ

ĐỀ TÀI

PHÁT TRIỂN WEBSITE B2C BÁN ĐỒ ĐIỆN TỬ

Giảng viên hướng dẫn : Kim Ngọc Bách

Nhóm : 02

Nhóm BTL : 05

Danh sách thành viên:

B21DCCN304 - Nguyễn Minh Giang

B21DCCN184 - Phạm Minh Công

B21DCCN484 - Nguyễn Khánh Linh

Hà Nội, 2025

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy – người đã đồng hành, hướng dẫn và truyền đạt cho chúng em những kiến thức quý báu trong suốt quá trình học tập và thực hiện đề tài này.

Thông qua xuyên suốt quá trình học, thầy không chỉ giúp chúng em hiểu rõ kiến thức lý thuyết mà còn định hướng cách ứng dụng thực tế, rèn luyện kỹ năng làm việc nhóm, tư duy giải quyết vấn đề và trách nhiệm trong học tập. Đây chính là hành trang quý giá để chúng em tiếp tục vững bước trên những chặng đường tiếp theo.

Một lần nữa, chúng em xin được bày tỏ lòng biết ơn sâu sắc đến thầy vì sự tận tâm, nhiệt huyết và sự hỗ trợ tận tình trong suốt thời gian qua.

Xin trân trọng cảm ơn!

MỤC LỤC

Lời Mở Đầu

Thương mại điện tử (TMĐT) đang bùng nổ mạnh mẽ, định hình lại cách chúng ta kinh doanh và mua sắm. Với sự phát triển của công nghệ, việc có một nền tảng TMĐT hiệu quả là yếu tố then chốt để doanh nghiệp cạnh tranh và phát triển.

Đề tài này tập trung vào việc xây dựng một nền tảng TMĐT đa năng (website hoặc ứng dụng di động) trong lĩnh vực thời trang, hỗ trợ các mô hình B2B (doanh nghiệp với doanh nghiệp), B2C (doanh nghiệp với khách hàng), C2C (khách hàng với khách hàng) hoặc kết hợp. Mục tiêu là tạo ra một không gian mua sắm trực tuyến tiện lợi, an toàn, với các tính năng cốt lõi như khuyến nghị sản phẩm, quản lý giỏ hàng, đơn hàng, thanh toán và đổi trả. Nền tảng này không chỉ mở rộng kênh tiếp cận khách hàng mà còn nâng cao trải nghiệm mua sắm, tạo lợi thế cạnh tranh đáng kể trên thị trường thời trang đầy sôi động.

I. TỔNG QUAN VÀ CƠ SỞ LÝ THUYẾT

1.1. Tổng Quan về Thương Mại Điện Tử

1.1.1. Định nghĩa và Đặc điểm của TMĐT

Thương mại điện tử (TMĐT) là việc mua bán hàng hóa và dịch vụ, thực hiện giao dịch thông tin qua mạng internet và các phương tiện điện tử. Nó bao gồm mọi hoạt động trao đổi trực tuyến, từ tìm kiếm thông tin, lựa chọn sản phẩm đến thanh toán và giao nhận.

Các đặc điểm nổi bật:

- Không giới hạn địa lý & thời gian: Giao dịch 24/7, không biên giới.
- Tiếp cận khách hàng rộng lớn: Mở rộng thị trường vượt xa giới hạn cửa hàng vật lý.
- Tiết kiệm chi phí: Giảm thiểu chi phí vận hành, mặt bằng, nhân viên.
- Cá nhân hóa: Khả năng phân tích dữ liệu để đưa ra đề xuất phù hợp từng khách hàng.
- Tương tác cao: Tương tác trực tiếp qua bình luận, đánh giá, hỗ trợ trực tuyến.
- Thanh toán điện tử: Giao dịch nhanh chóng, an toàn qua các cổng trực tuyến.

1.1.2. Các Mô hình TMĐT Phổ biến

TMĐT được phân loại dựa trên đối tượng tham gia giao dịch:

- B2C (Business-to-Consumer): Doanh nghiệp bán hàng trực tiếp cho người tiêu dùng cá nhân.
    - *Ví dụ:* Các sàn TMĐT (Shopee, Lazada), website của các thương hiệu thời trang.
- B2B (Business-to-Business): Doanh nghiệp giao dịch với doanh nghiệp khác (mua sỉ, cung cấp nguyên liệu).
    - *Ví dụ:* https://www.google.com/search?q=Alibaba.com, các website phân phối sỉ.
- C2C (Consumer-to-Consumer): Người tiêu dùng cá nhân mua bán/trao đổi với nhau qua nền tảng trung gian.
    - *Ví dụ:* Chợ Tốt, Facebook Marketplace.
- C2B (Consumer-to-Business): Cá nhân cung cấp dịch vụ/sản phẩm cho doanh nghiệp (freelancer, bán ảnh stock).
- G2C (Government-to-Citizen): Chính phủ cung cấp dịch vụ công trực tuyến cho công dân (nộp thuế online).

- B2G (Business-to-Government): Doanh nghiệp cung cấp hàng hóa/dịch vụ cho cơ quan chính phủ (đấu thầu điện tử).

1.1.3. Xu hướng phát triển của TMĐT tại Việt Nam và Thế giới

TMĐT đang tăng trưởng mạnh mẽ, đặc biệt tại Việt Nam:

- Tại Việt Nam:
    - Tốc độ tăng trưởng vượt trội: Là một trong những thị trường TMĐT phát triển nhanh nhất Đông Nam Á.
    - Mobile Commerce (M-commerce): Mua sắm qua di động chiếm ưu thế.
    - Cạnh tranh khốc liệt: Các sàn lớn liên tục cải tiến và khuyến mãi.
    - Logistics & Thanh toán: Hệ thống giao nhận và thanh toán không tiền mặt ngày càng hoàn thiện.
    - Live Commerce & Shoppertainment: Xu hướng bán hàng kết hợp giải trí (livestream) bùng nổ.
    - SMEs chuyển dịch số: Nhiều doanh nghiệp nhỏ và vừa tham gia TMĐT.
- Trên Thế giới:
    - Cá nhân hóa & AI: Ứng dụng trí tuệ nhân tạo để cá nhân hóa trải nghiệm mua sắm.
    - Social Commerce: Mua sắm trực tiếp trên mạng xã hội.
    - O2O (Online-to-Offline): Kết hợp trải nghiệm trực tuyến và tại cửa hàng vật lý.
    - Công nghệ mới: VR/AR tăng cường trải nghiệm thử đồ ảo, xem sản phẩm 3D.
    - Bền vững & Đạo đức: Người tiêu dùng quan tâm hơn đến sản phẩm xanh, có trách nhiệm xã hội.
    - Mô hình D2C (Direct-to-Consumer): Thương hiệu bán hàng trực tiếp đến khách hàng.

1.2. Các Công Nghệ và Nền Tảng Phát Triển Website TMĐT

1.2.1. Phân tích các công nghệ web cơ bản

- Shopify (SaaS – Software as a Service):
    - Shopify cung cấp một môi trường hoàn chỉnh, đám mây hóa, nơi mọi hạ tầng kỹ thuật (máy chủ, cơ sở dữ liệu, bảo mật) được Shopify quản lý.
    - Người dùng không cần cài đặt XAMPP hay bất kỳ máy chủ cục bộ nào.
    - Frontend được xây dựng bằng Liquid (ngôn ngữ template của Shopify), HTML, CSS, JavaScript, cho phép tùy chỉnh giao diện thông qua trình chỉnh sửa theme hoặc mã nguồn.

- Node.js (cho các tích hợp mở rộng, nếu cần):
    - Mặc dù Shopify xử lý phần lớn backend, Node.js (môi trường chạy JavaScript phía máy chủ) có thể được sử dụng để xây dựng các ứng dụng (apps) hoặc API tùy chỉnh kết nối với Shopify (thông qua Shopify API).
    - Điều này cho phép phát triển các tính năng độc đáo không có sẵn trên Shopify App Store hoặc tích hợp với các hệ thống bên ngoài.
    - Với cấu trúc thư mục được cung cấp (controllers, models, routes), điều này cho thấy có thể bạn đang phát triển một ứng dụng/dịch vụ bên ngoài Shopify sử dụng Node.js Express để xử lý một số logic nghiệp vụ phức tạp hoặc quản lý dữ liệu riêng biệt, sau đó tích hợp với Shopify.

1.2.2. Ngôn ngữ lập trình Backend

- Shopify (Nền tảng tích hợp): Shopify tự động xử lý phần backend, không yêu cầu bạn viết mã backend trực tiếp cho các chức năng cốt lõi.
- Node.js với Express.js (cho các ứng dụng/API mở rộng):
    - Nếu cần xây dựng các tính năng nâng cao hoặc xử lý dữ liệu phức tạp ngoài khả năng mặc định của Shopify, Node.js kết hợp với Express.js (một framework web linh hoạt cho Node.js) là lựa chọn lý tưởng.
    - Ưu điểm: Sử dụng JavaScript cho cả frontend và backend (giúp thống nhất ngôn ngữ), hiệu suất cao với mô hình I/O không chặn, hệ sinh thái thư viện lớn (npm).

1.2.3. Hệ quản trị cơ sở dữ liệu

- Shopify (Tích hợp sẵn): Shopify tự quản lý toàn bộ cơ sở dữ liệu của bạn trên hệ thống của họ. Bạn không cần phải chọn, cài đặt hay quản lý RDBMS (như MySQL) trực tiếp.
- MySQL (nếu sử dụng Node.js cho dữ liệu riêng biệt):
    - Nếu các ứng dụng Node.js mở rộng của bạn yêu cầu lưu trữ dữ liệu riêng biệt (ví dụ: dữ liệu không thuộc phạm vi quản lý của Shopify, hoặc dữ liệu nhạy cảm cần được xử lý riêng), MySQL là một lựa chọn phổ biến.
    - MySQL là RDBMS mã nguồn mở, ổn định và hiệu suất cao, thường được quản lý thông qua XAMPP trong môi trường phát triển cục bộ.

1.2.4. Giới thiệu các Framework/CMS phổ biến cho TMĐT

- Shopify (SaaS/CMS chuyên biệt cho TMĐT):

    - Là một nền tảng Software as a Service (SaaS) dựa trên đám mây, cung cấp toàn bộ công cụ để xây dựng, vận hành và quản lý cửa hàng trực tuyến mà không cần kiến thức lập trình sâu.
    - Ưu điểm: Triển khai siêu nhanh, dễ sử dụng, hosting và bảo mật tích hợp, App Store phong phú (hàng ngàn ứng dụng mở rộng), hỗ trợ khách hàng tốt, khả năng mở rộng tốt.
    - Nhược điểm: Phí thuê bao định kỳ, hạn chế tùy biến sâu ở cấp độ mã nguồn lõi, phụ thuộc vào nền tảng.
- Các CMS TMĐT khác (để so sánh):
    - WooCommerce (Plugin WordPress): Mã nguồn mở, linh hoạt, nhưng yêu cầu tự quản lý hosting và bảo trì.
    - Magento (Adobe Commerce): Mạnh mẽ, linh hoạt, phù hợp doanh nghiệp lớn, nhưng phức tạp và đòi hỏi chi phí phát triển cao.
    - OpenCart, PrestaShop: Các CMS mã nguồn mở miễn phí khác.
- Các Framework (cho phát triển tùy chỉnh):
    - Laravel (PHP), Django (Python), Ruby on Rails (Ruby): Cung cấp cấu trúc và công cụ để xây dựng nền tảng TMĐT từ đầu, mang lại sự linh hoạt tối đa nhưng đòi hỏi thời gian và chi phí phát triển lớn.

1.2.5. Lý do lựa chọn công nghệ/nền tảng cụ thể cho dự án của bạn (Shopify + Node.js/MySQL cho mở rộng)

Việc lựa chọn Shopify làm nền tảng chính, kết hợp với khả năng phát triển các ứng dụng/API tùy chỉnh bằng Node.js và MySQL khi cần, là một chiến lược tối ưu cho dự án TMĐT thời trang này:

- Tốc độ triển khai và Dễ quản lý (Nhờ Shopify):
    - Shopify cho phép khởi tạo cửa hàng trực tuyến nhanh chóng, giúp dự án sớm đi vào hoạt động để nắm bắt thị trường.
    - Giao diện quản trị trực quan giúp quản lý sản phẩm, đơn hàng, khách hàng dễ dàng, tiết kiệm thời gian vận hành.
    - Loại bỏ gánh nặng về hạ tầng, bảo mật, và bảo trì máy chủ, cho phép tập trung vào kinh doanh và marketing.
- Đầy đủ tính năng cốt lõi (Nhờ Shopify):
    - Shopify cung cấp sẵn các module thiết yếu cho TMĐT: quản lý sản phẩm đa dạng, giỏ hàng, đơn hàng, thanh toán tích hợp (có thể mở rộng qua các cổng thanh toán Việt Nam), và hỗ trợ quy trình đổi trả cơ bản.
- Linh hoạt và Khả năng mở rộng (Kết hợp Shopify Apps và Node.js/MySQL):
    - Shopify App Store cung cấp hàng ngàn ứng dụng để mở rộng nhanh chóng các tính năng bổ sung (marketing, SEO, quản lý kho, dịch vụ khách hàng) mà không cần lập trình.

    - Đối với các yêu cầu đặc thù hoặc logic nghiệp vụ phức tạp không có sẵn trong ứng dụng Shopify, việc sử dụng Node.js và Express.js để phát triển các API riêng biệt (như đã thấy trong cấu trúc thư mục của bạn) là giải pháp lý tưởng. Điều này cho phép dự án có sự linh hoạt cao trong việc tạo ra các chức năng độc đáo, như các hệ thống quản lý tài khoản/thông báo phức tạp hoặc tích hợp sâu với các dịch vụ bên thứ ba.
    - Nếu các ứng dụng Node.js cần lưu trữ dữ liệu riêng biệt, việc sử dụng MySQL (có thể thông qua XAMPP để quản lý trong môi trường dev) cung cấp một giải pháp cơ sở dữ liệu mạnh mẽ và đáng tin cậy.
- Tập trung vào kinh doanh: Sự kết hợp này cho phép tận dụng sức mạnh của Shopify để vận hành chính, đồng thời có khả năng tùy chỉnh sâu khi cần thiết thông qua các công nghệ hiện đại như Node.js, đảm bảo dự án có thể đáp ứng mọi yêu cầu kinh doanh và phát triển trong tương lai.

1.3. Nghiên cứu các Website TMĐT Tương tự

Nghiên cứu các website thương mại điện tử hiện có là một bước quan trọng để học hỏi từ những người đi trước, hiểu rõ hơn về các tiêu chuẩn ngành, và xác định được những điểm mạnh cần phát huy cũng như những điểm yếu cần tránh. Trong lĩnh vực điện tử gia dụng và thiết bị điện tử, các sàn TMĐT lớn và các nhà bán lẻ chuyên biệt đã xây dựng được những trải nghiệm đáng học hỏi.

1.3.1. Phân tích các website TMĐT có mô hình tương đồng

Chúng ta sẽ phân tích một số website TMĐT lớn và phổ biến tại Việt Nam, những nơi cũng kinh doanh đa dạng các mặt hàng điện tử, từ đó rút ra các bài học chung:

- Tiki: Một trong những sàn TMĐT hàng đầu tại Việt Nam, ban đầu nổi tiếng với sách, nay đã mở rộng ra nhiều ngành hàng, trong đó có điện tử - điện lạnh.
- Điện Máy Xanh: Một chuỗi siêu thị điện máy và điện lạnh lớn, có website TMĐT mạnh mẽ, chuyên biệt về các sản phẩm điện tử, điện gia dụng.
- FPT Shop / Thế Giới Di Động: Các chuỗi bán lẻ thiết bị di động, laptop, và điện tử tiêu dùng khác với hệ thống website TMĐT phát triển.

1.3.2. Đánh giá điểm mạnh, điểm yếu về giao diện, tính năng, trải nghiệm người dùng

Tiêu chí	Shopee	Điện Máy Xanh	FPT Shop/Thế Giới Di Động

Giao diện	- Điểm mạnh: Sạch sẽ, hiện đại, bố cục rõ ràng, màu sắc thương hiệu nổi bật. - Điểm yếu: Đôi khi quá nhiều banner quảng cáo, dễ bị phân tâm.	- Điểm mạnh: Chuyên nghiệp, trực quan, hình ảnh sản phẩm lớn, thông tin sản phẩm chi tiết dễ tìm. - Điểm yếu: Có thể hơi "nặng" hình ảnh, một số trang chi tiết sản phẩm dài.	- Điểm mạnh: Tối giản, tập trung vào sản phẩm chính (điện thoại, laptop), dễ nhìn. - Điểm yếu: Đôi khi quá tập trung vào ưu đãi, có thể làm lu mờ thông tin sản phẩm.
Tính năng	- Điểm mạnh: + Tìm kiếm & Khuyến nghị: Bộ lọc chi tiết (giá, thương hiệu, loại sản phẩm), gợi ý tìm kiếm thông minh, khuyến nghị sản phẩm liên quan/đã xem. + Giỏ hàng: Quản lý đơn giản, dễ dàng thêm/bớt, hiển thị tổng tiền rõ ràng. + Đơn hàng: Theo dõi trạng thái chi tiết, lịch sử mua hàng. + Thanh toán: Đa dạng phương thức (COD, thẻ, ví điện tử), quy trình rõ ràng. + Đổi trả/Hoàn tiền: Chính sách rõ ràng, quy trình hướng dẫn chi tiết. + Tính năng khác: Đánh giá sản phẩm, hỏi đáp cộng đồng, TikiNOW (giao	- Điểm mạnh: + Tìm kiếm & Khuyến nghị: Lọc chuyên sâu theo thông số kỹ thuật (dung tích tủ lạnh, công suất máy giặt), so sánh sản phẩm trực quan. + Giỏ hàng/Đơn hàng: Tương tự Shopee + Thanh toán: Đa dạng, tích hợp trả góp. + Đổi trả/Hoàn tiền: Chính sách bảo hành/đổi trả rõ ràng, hỗ trợ tại cửa hàng. + Tính năng khác: Tư vấn mua hàng, đặt lịch hẹn, xem sản phẩm tại siêu thị.	- Điểm mạnh: + Tìm kiếm & Khuyến nghị: Lọc theo các thông số kỹ thuật đặc trưng của điện thoại/laptop (RAM, bộ nhớ, chip), so sánh cấu hình. + Giỏ hàng/Đơn hàng/Thanh toán: Tối ưu cho mua sắm thiết bị điện tử. + Đổi trả/Hoàn tiền: Chính sách bảo hành, đổi trả được nhấn mạnh. + Tính năng khác: Cửa hàng gần nhất, tin tức công nghệ, so sánh giá linh hoạt.

	nhanh).		
Trải nghiệm Người dùng (UX)	- Điểm mạnh: Dễ điều hướng, quy trình mua hàng mạch lạc, thông báo rõ ràng. Phù hợp cho người mua tìm kiếm sản phẩm đa dạng. - Điểm yếu: Có thể gây quá tải thông tin với người dùng mới.	- Điểm mạnh: Thông tin sản phẩm chuyên sâu, hình ảnh 360 độ, video sản phẩm, giúp người dùng hiểu rõ hơn về mặt hàng điện tử. Phù hợp cho người muốn nghiên cứu kỹ trước khi mua. - Điểm yếu: Đôi khi thông tin quá nhiều có thể làm người dùng cảm thấy choáng ngợp.	- Điểm mạnh: Nhanh, gọn, tập trung vào việc giới thiệu sản phẩm mới và ưu đãi. Phù hợp cho người mua có mục tiêu rõ ràng (ví dụ: mua điện thoại đời mới). - Điểm yếu: Cảm giác "săn sale" có thể lấn át trải nghiệm tìm hiểu sản phẩm.

1.3.3. Định hướng cho website

Dựa trên phân tích các website TMĐT lớn trong lĩnh vực điện tử, chúng ta có thể rút ra những bài học và định hướng quan trọng cho website của mình, đặc biệt với mục tiêu "giao diện đơn giản để người dùng có thể sử dụng ngay":

1. Giao diện "sạch" và trực quan:
    - Bài học: Người dùng điện tử thường cần thông tin rõ ràng. Tránh quá nhiều quảng cáo hoặc banner gây nhiễu.
    - Định hướng: Thiết kế giao diện tối giản, tập trung vào hình ảnh sản phẩm chất lượng cao và thông tin sản phẩm nổi bật. Sử dụng khoảng trắng hợp lý để tạo cảm giác thoáng đãng, dễ nhìn. Đảm bảo nút bấm và các thành phần tương tác rõ ràng, dễ nhận biết.
2. Hệ thống tìm kiếm và bộ lọc mạnh mẽ, thông minh:
    - Bài học: Với các sản phẩm điện tử, người dùng thường tìm kiếm theo các thông số kỹ thuật cụ thể (dung tích, công suất, RAM, CPU, loại màn hình...).
    - Định hướng: Xây dựng tính năng tìm kiếm với gợi ý thông minh (autocomplete). Đặc biệt quan trọng là các bộ lọc chi tiết theo thuộc tính sản phẩm (ví dụ: "thương hiệu", "giá", "loại tủ lạnh", "dung tích", "chế

độ giặt", "màu sắc", "kích thước màn hình"). Tính năng so sánh sản phẩm cũng rất hữu ích cho đồ điện tử.

3. Thông tin sản phẩm chi tiết nhưng dễ hiểu:
    - Bài học: Người mua điện tử thường đọc kỹ thông số.
    - Định hướng: Mỗi trang sản phẩm cần có:
        - Hình ảnh/video chất lượng cao, đa góc nhìn.
        - Thông số kỹ thuật rõ ràng, dễ đọc (có thể dùng bảng so sánh).
        - Mô tả sản phẩm ngắn gọn, làm nổi bật lợi ích chính.
        - Giá cả, thông tin khuyến mãi/ưu đãi nổi bật.
        - Chính sách bảo hành, đổi trả được hiển thị rõ ràng.
        - Phần đánh giá, bình luận của khách hàng.
4. Quy trình mua hàng (Giỏ hàng, Đặt hàng, Thanh toán) đơn giản, ít bước:
    - Bài học: Giảm thiểu các rào cản trong quá trình thanh toán giúp tăng tỷ lệ chuyển đổi.
    - Định hướng:
        - Giỏ hàng: Đơn giản, dễ dàng thêm/bớt/cập nhật số lượng, hiển thị tổng tiền rõ ràng.
        - Đặt hàng: Quy trình Checkout (thanh toán) chỉ qua vài bước, yêu cầu thông tin tối thiểu cần thiết.
        - Thanh toán: Tích hợp đa dạng các phương thức thanh toán phổ biến tại Việt Nam (COD, chuyển khoản ngân hàng, ví điện tử như Momo, ZaloPay, thẻ tín dụng/ghi nợ), đảm bảo an toàn.
        - Theo dõi đơn hàng: Cung cấp tính năng theo dõi trạng thái đơn hàng rõ ràng, dễ dàng truy cập.
5. Chính sách Đổi trả/Hoàn tiền minh bạch:
    - Bài học: Đối với mặt hàng điện tử có giá trị cao, chính sách bảo hành, đổi trả rõ ràng sẽ tạo dựng niềm tin cho khách hàng.
    - Định hướng: Xây dựng trang thông tin chi tiết về chính sách đổi trả, bảo hành. Quy trình đổi trả/hoàn tiền cần đơn giản, dễ thực hiện, có hướng dẫn cụ thể.
6. Tối ưu hóa cho thiết bị di động (Mobile-first):
    - Bài học: Phần lớn người dùng TMĐT truy cập từ điện thoại di động.
    - Định hướng: Đảm bảo website hiển thị tốt và hoạt động mượt mà trên mọi kích thước màn hình, đặc biệt là điện thoại thông minh. Giao diện đơn giản trên di động sẽ giúp trải nghiệm tốt hơn.

1.4. Mô hình doanh thu của website

1.4.1. Doanh thu từ việc bán sản phẩm trực tiếp

Bán lẻ sản phẩm đồ điện tử, gia dụng: là nguồn doanh thu chính của website, từ việc bán các sản phẩm đồ điện tử như như TV, điện thoại, tủ lạnh,.... Sản phẩm sẽ được bán theo từng món hoặc theo bộ, với mức giá từ thấp đến cao, đáp ứng nhu cầu đa dạng của khách hàng. Doanh thu từ bán sản phẩm sẽ chiếm tỷ lệ lớn trong tổng doanh thu của website.

1.4.2. Doanh thu từ các chương trình khuyến mãi, giảm giá

Chương trình giảm giá theo dịp: website có thể tổ chức các chương trình giảm giá hoặc khuyến mãi vào các dịp đặc biệt như lễ hội, Tế, Black Friday, v.v. Mặc dù mức giá giảm, website vẫn có thể thu được doanh thu lớn từ lượng khách hàng mua sắm trong các đợt khuyến mãi này. Giảm giá theo số lượng mua: các chương trình giảm giá theo số lượng như "Mua 2 tặng 1" hoặc giảm giá khi mua theo bộ cũng có thể là một nguồn doanh thu hiệu quả cho website.

1.4.3. Doanh thu từ quảng cáo và tiếp thị liên kết

Quảng cáo của đối tác: nếu website có lượng truy cập lớn, có thể phát triển mô hình quảng cáo bằng cách hợp tác với các thương hiệu khác để hiển thị quảng cáo sản phẩm của họ trên website. Mô hình này có thể bao gồm quảng cáo banner, quảng cáo pop-up hoặc liên kết tiếp thị.

Tiếp thị liên kết: website có thể kiếm hoa hồng từ các sản phẩm liên kết hoặc các dịch vụ do các đối tác cung cấp, ví dụ như các dịch vụ thanh toán hoặc dịch vụ bảo hiểm.

II. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1. Phân tích yêu cầu

2.1.1. Yêu cầu nghiệp vụ (Business Requirements)

Phân tích yêu cầu nghiệp vụ là bước đầu tiên và quan trọng để xác định rõ mục tiêu, đối tượng, và cách thức hoạt động của nền tảng TMĐT, đảm bảo sản phẩm cuối cùng đáp ứng đúng nhu cầu kinh doanh và người dùng.

a. Mục đích kinh doanh của website

Mục đích chính của dự án là cung cấp một nền tảng thương mại điện tử uy tín, chuyên biệt về các sản phẩm điện thoại và đồ điện tử gia đình. Cụ thể, website sẽ tập trung vào phân khúc sản phẩm tầm trung, đáp ứng nhu cầu của một lượng lớn khách hàng mong muốn sản phẩm chất lượng với mức giá phải chăng.

Các mục tiêu kinh doanh cụ thể bao gồm:

- Xây dựng uy tín và lòng tin: Trở thành địa chỉ đáng tin cậy cho người dùng khi mua sắm các thiết bị điện tử.
- Mở rộng kênh phân phối: Tạo ra một kênh bán hàng trực tuyến hiệu quả để tiếp cận khách hàng rộng hơn.
- Tối ưu hóa trải nghiệm mua sắm: Cung cấp một giao diện đơn giản, dễ sử dụng để khách hàng có thể tìm kiếm, đặt hàng và thanh toán nhanh chóng.
- Đảm bảo chất lượng sản phẩm và dịch vụ hậu mãi: Cung cấp sản phẩm chính hãng, đi kèm với chính sách bảo hành, đổi trả rõ ràng.

b. Đối tượng khách hàng mục tiêu

Website tập trung vào mô hình B2C (Business-to-Consumer) với đối tượng khách hàng mục tiêu chính là người đã đi làm, trong độ tuổi từ 24 đến 45 tuổi. Nhóm đối tượng này thường có:

- Khả năng chi trả: Có thu nhập ổn định, đủ khả năng mua sắm các sản phẩm điện tử tầm trung.
- Nhu cầu rõ ràng: Tìm kiếm các thiết bị phục vụ công việc, giải trí cá nhân (điện thoại) và các thiết bị thiết yếu cho gia đình (tủ lạnh, máy giặt, nồi cơm điện).
- Kinh nghiệm sử dụng internet: Thành thạo trong việc tìm kiếm thông tin và mua sắm trực tuyến.
- Quan tâm đến giá trị và chất lượng: Có xu hướng tìm hiểu kỹ sản phẩm, cân nhắc giữa giá cả và hiệu năng, cũng như dịch vụ hậu mãi.

c. Các quy trình kinh doanh chính

Dưới đây là các quy trình nghiệp vụ cốt lõi mà nền tảng TMĐT này cần hỗ trợ:

1. Quy trình Đặt hàng:
    - Yêu cầu tài khoản: Người dùng cần phải đăng ký tài khoản và đăng nhập để có thể tiến hành đặt hàng, đảm bảo việc theo dõi đơn hàng và cá nhân hóa trải nghiệm về sau.
    - Thêm sản phẩm vào giỏ hàng: Người dùng duyệt qua các danh mục sản phẩm, tìm kiếm sản phẩm mong muốn và thêm vào giỏ hàng. Số lượng có thể điều chỉnh trong giỏ hàng.

- Xem giỏ hàng và tiến hành thanh toán: Người dùng truy cập giỏ hàng để kiểm tra các sản phẩm đã chọn, số lượng và tổng tiền.
- Chọn phương thức thanh toán: Sau khi xác nhận giỏ hàng, người dùng chọn "Thanh toán". Hệ thống sẽ chuyển hướng đến trang chọn phương thức thanh toán.
- Xác nhận đặt hàng: Sau khi chọn phương thức thanh toán, người dùng tiến hành đặt hàng, hoàn tất quy trình.

2. Quy trình Thanh toán:

    - Thanh toán trực tuyến qua MoMo: Website sẽ hỗ trợ thanh toán online thông qua ví điện tử MoMo.
    - Tích hợp và xác nhận tự động: Hệ thống đã có tích hợp với MoMo để xử lý giao dịch và sẽ nhận được xác nhận thanh toán tự động từ phía MoMo, cập nhật trạng thái đơn hàng.
3. Quy trình Quản lý Đơn hàng (từ phía Người bán/Quản trị viên):

    - Tiếp nhận đơn hàng: Khi có đơn hàng mới được đặt và thanh toán thành công, hệ thống sẽ tự động ghi nhận và thông báo (ví dụ: qua giao diện quản trị) cho quản trị viên.
    - Cập nhật trạng thái: Quản trị viên có thể xem danh sách các đơn hàng, cập nhật trạng thái của đơn hàng (ví dụ: Đang xử lý, Đã xác nhận, Đang giao hàng, Đã giao hàng, Đã hủy).
    - *Lưu ý:* Vì đây là bài tập lớn tập trung vào trải nghiệm người dùng, quy trình quản lý đơn hàng cho admin sẽ được giữ ở mức cơ bản, đủ để theo dõi và xử lý các trạng thái chính.
4. Quy trình Quản lý Kho (từ phía Người bán/Quản trị viên):

    - Đơn giản hóa: Với mục tiêu tập trung vào trải nghiệm người dùng, quy trình quản lý kho cho admin sẽ được thiết kế đơn giản.
    - Cập nhật số lượng tồn kho: Admin có thể nhập hoặc cập nhật số lượng tồn kho của từng sản phẩm một cách thủ công thông qua giao diện quản trị.
    - Hiển thị trạng thái tồn kho: Hệ thống sẽ tự động hiển thị trạng thái "Hết hàng" cho người dùng khi sản phẩm không còn tồn kho.
    - *Lưu ý:* Không yêu cầu các tính năng quản lý kho phức tạp như nhập/xuất kho tự động theo lô, cảnh báo tồn kho thấp chi tiết, hay tích hợp với hệ thống ERP bên ngoài.
5. Quy trình Đổi trả/Hoàn tiền:

- Yêu cầu trực tuyến: Khách hàng có thể gửi yêu cầu đổi trả hoặc hoàn tiền thông qua một form trực tuyến trên website. Form này sẽ yêu cầu các thông tin cần thiết như mã đơn hàng, lý do đổi trả, hình ảnh sản phẩm (nếu có).
- Duyệt yêu cầu từ phía Admin: Yêu cầu đổi trả/hoàn tiền sẽ được gửi về phía quản trị viên để xem xét và duyệt. Quản trị viên sẽ đánh giá yêu cầu dựa trên chính sách của website và đưa ra quyết định (duyệt/từ chối).
- Xử lý thủ công: Sau khi yêu cầu được duyệt, quá trình đổi trả/hoàn tiền thực tế sẽ được thực hiện thủ công (ví dụ: liên hệ khách hàng để nhận lại sản phẩm, sau đó tiến hành hoàn tiền qua MoMo hoặc chuyển khoản ngân hàng).

2.1.2. Yêu cầu chức năng (Functional Requirements)

Các yêu cầu chức năng mô tả cụ thể các hành vi và khả năng mà hệ thống phải thực hiện để đáp ứng các yêu cầu nghiệp vụ đã đề ra. Dưới đây là danh sách các chức năng chính cho từng loại người dùng trong hệ thống TMĐT điện tử gia dụng này:

a. Đối với Người dùng phổ thông (Khách hàng)

Nhóm chức năng	Mã chức năng	Tên chức năng	Mô tả chi tiết
Quản lý tài khoản	F-USR-001	Đăng ký tài khoản	Cho phép người dùng mới tạo tài khoản bằng email/số điện thoại và mật khẩu.
	F-USR-002	Đăng nhập	Cho phép người dùng đăng nhập vào hệ thống bằng tài khoản đã đăng ký.
	F-USR-003	Quên mật khẩu	Cho phép người dùng đặt lại mật khẩu qua email/số điện thoại.
	F-USR-004	Quản lý thông tin cá nhân	Cho phép người dùng xem và chỉnh sửa thông tin cá nhân (tên, số điện

			thoại, địa chỉ giao hàng).
	F-USR-005	Đổi mật khẩu	Cho phép người dùng thay đổi mật khẩu hiện tại.
Tìm kiếm & Xem sản phẩm	F-USR-010	Tìm kiếm sản phẩm	Cho phép người dùng tìm kiếm sản phẩm theo từ khóa (tên, thương hiệu, loại sản phẩm).
	F-USR-011	Xem danh sách sản phẩm	Hiển thị danh sách sản phẩm theo danh mục, kết quả tìm kiếm hoặc bộ lọc.
	F-USR-012	Xem chi tiết sản phẩm	Hiển thị trang chi tiết sản phẩm bao gồm hình ảnh, mô tả, thông số kỹ thuật, giá, thông tin bảo hành, đánh giá.
	F-USR-013	Xem sản phẩm liên quan/đã xem	Gợi ý các sản phẩm tương tự hoặc sản phẩm người dùng đã xem gần đây.
	F-USR-014	Đánh giá/Bình luận sản phẩm	Cho phép người dùng đăng tải đánh giá, bình luận và xếp hạng sao cho sản phẩm đã mua.
Giỏ hàng & Đặt hàng	F-USR-020	Thêm sản phẩm vào giỏ hàng	Cho phép thêm sản phẩm từ trang chi tiết hoặc danh sách vào giỏ hàng.
	F-USR-021	Cập nhật số lượng sản phẩm trong giỏ hàng	Cho phép thay đổi số lượng sản phẩm hoặc xóa sản phẩm

			khỏi giỏ hàng.
	F-USR-022	Xem giỏ hàng	Hiển thị danh sách các sản phẩm trong giỏ hàng, tổng tiền tạm tính.
	F-USR-023	Đặt hàng	Bắt đầu quy trình đặt hàng từ giỏ hàng, chuyển đến trang thanh toán.
Thanh toán	F-USR-030	Chọn phương thức thanh toán	Cho phép người dùng lựa chọn phương thức thanh toán (MoMo).
	F-USR-031	Thanh toán qua MoMo	Tích hợp cổng thanh toán MoMo để xử lý giao dịch.
	F-USR-032	Xác nhận thanh toán	Hiển thị thông báo xác nhận thanh toán thành công hoặc thất bại.
Quản lý đơn hàng	F-USR-040	Xem lịch sử đơn hàng	Hiển thị danh sách các đơn hàng đã đặt, bao gồm trạng thái và chi tiết từng đơn hàng.
	F-USR-041	Xem chi tiết đơn hàng	Cho phép người dùng xem chi tiết một đơn hàng cụ thể (sản phẩm, giá, địa chỉ, trạng thái).
	F-USR-042	Yêu cầu đổi trả/hoàn tiền	Cho phép người dùng gửi yêu cầu đổi trả/hoàn tiền cho một đơn hàng đã giao qua form trực tuyến.
Hỗ trợ & Thông tin	F-USR-050	Liên hệ hỗ trợ	Cung cấp các kênh liên hệ (form liên hệ, số điện thoại,

			email) để được hỗ trợ.

b. Đối với Người quản trị (Admin)

Nhóm chức năng	Mã chức năng	Tên chức năng	Mô tả chi tiết
Quản lý hệ thống	F-ADM-001	Đăng nhập Admin	Cho phép người quản trị đăng nhập vào hệ thống quản trị.
Quản lý sản phẩm	F-ADM-010	Quản lý danh mục sản phẩm	Cho phép thêm, sửa, xóa các danh mục sản phẩm (ví dụ: Điện thoại, Tủ lạnh, Máy giặt, …).
	F-ADM-011	Thêm sản phẩm mới	Cho phép nhập thông tin sản phẩm mới (tên, mô tả, giá, hình ảnh, thông số kỹ thuật, danh mục).
	F-ADM-012	Xóa sản phẩm	Cho phép xóa sản phẩm khỏi danh mục (có thể là ẩn sản phẩm thay vì xóa vĩnh viễn).
	F-ADM-013	Quản lý thuộc tính sản phẩm	Cho phép thêm/sửa/xóa các thuộc tính sản phẩm (ví dụ: RAM, ROM, Dung tích, Màu sắc).
Quản lý người dùng	F-ADM-030	Xem danh sách người dùng	Hiển thị danh sách tất cả các tài khoản khách hàng đã đăng ký.
Báo cáo & Thống kê	F-ADM-060	Xem báo cáo doanh thu	Cung cấp báo cáo về doanh thu theo thời gian (ngày,

			tháng, năm).

2.1.3. Yêu cầu phi chức năng (Non-Functional Requirements)

Yêu cầu phi chức năng mô tả các tiêu chí chất lượng của hệ thống, xác định mức độ hiệu quả, đáng tin cậy, an toàn, dễ sử dụng và khả năng mở rộng của nền tảng TMĐT. Đây là những yếu tố then chốt góp phần tạo nên trải nghiệm người dùng tốt và đảm bảo sự ổn định của hệ thống.

a. Hiệu năng (Performance)

Hiệu năng của website ảnh hưởng trực tiếp đến trải nghiệm người dùng và khả năng giữ chân khách hàng.

- Tốc độ tải trang:
    - Mục tiêu: Thời gian tải trang (Page Load Time) cho các trang quan trọng (trang chủ, trang danh mục sản phẩm, trang chi tiết sản phẩm, giỏ hàng) phải dưới 3 giây trên kết nối internet trung bình (ví dụ: 4G hoặc cáp quang dân dụng).
    - Thực hiện: Tối ưu hóa hình ảnh, sử dụng CDN (Content Delivery Network - do Shopify cung cấp và/hoặc tùy chỉnh nếu dùng Node.js cho tài nguyên riêng), tối ưu mã nguồn frontend (HTML, CSS, JavaScript) và tối ưu truy vấn cơ sở dữ liệu (đối với phần Node.js/MySQL).
- Số lượng người dùng đồng thời:
    - Mục tiêu: Hệ thống phải có khả năng xử lý ít nhất 100-200 người dùng đồng thời tại bất kỳ thời điểm nào mà không làm giảm đáng kể hiệu năng hoặc gây lỗi.
    - Thực hiện: Sử dụng kiến trúc mạnh mẽ (đối với Shopify là nền tảng đã tối ưu sẵn), tối ưu hóa code backend (Node.js) để xử lý bất đồng bộ, sử dụng caching hiệu quả.

b. Bảo mật (Security)

Bảo mật là yếu tố tối quan trọng đối với một nền tảng TMĐT, đặc biệt khi xử lý dữ liệu cá nhân và thông tin thanh toán của người dùng.

- Bảo mật dữ liệu:
    - Mục tiêu: Toàn bộ dữ liệu người dùng (thông tin cá nhân, địa chỉ, lịch sử đơn hàng) và dữ liệu thanh toán phải được mã hóa khi truyền tải (sử

dụng HTTPS/SSL). Dữ liệu nhạy cảm (như mật khẩu) phải được lưu trữ dưới dạng mã hóa một chiều (hashing).
    - Thực hiện: Shopify tự động cung cấp chứng chỉ SSL và quản lý mã hóa dữ liệu. Đối với phần Node.js/MySQL tùy chỉnh, cần đảm bảo sử dụng HTTPS, mã hóa mật khẩu bằng các thuật toán mạnh (ví dụ: bcrypt) và tuân thủ các nguyên tắc bảo mật cơ sở dữ liệu.
- Chống tấn công XSS (Cross-Site Scripting) và SQL Injection:
    - Mục tiêu: Hệ thống phải có khả năng ngăn chặn các lỗ hổng bảo mật phổ biến như XSS (chèn mã độc vào trình duyệt người dùng) và SQL Injection (chèn mã SQL độc hại để truy cập/thao túng cơ sở dữ liệu).
    - Thực hiện:
        - XSS: Thực hiện sanitize và validate tất cả dữ liệu đầu vào từ người dùng trước khi hiển thị hoặc lưu trữ (đặc biệt là các trường bình luận, đánh giá, form liên hệ).
        - SQL Injection: Sử dụng Prepared Statements hoặc ORM (Object-Relational Mapping) khi truy vấn cơ sở dữ liệu (với Node.js/MySQL) để ngăn chặn việc chèn mã SQL độc hại. Shopify đã xử lý vấn đề này ở cấp độ nền tảng.

c. Khả năng mở rộng (Scalability)

Hệ thống phải có khả năng dễ dàng mở rộng để đáp ứng nhu cầu tăng trưởng trong tương lai (ví dụ: tăng số lượng sản phẩm, người dùng, giao dịch).

- Dễ dàng thêm tính năng mới:
    - Mục tiêu: Cấu trúc mã nguồn và kiến trúc hệ thống phải được thiết kế theo module, cho phép thêm các tính năng mới hoặc mở rộng các chức năng hiện có mà không ảnh hưởng lớn đến các phần khác của hệ thống.
    - Thực hiện:
        - Với Shopify: Khai thác Shopify App Store và Shopify API để tích hợp các tính năng mở rộng.
        - Với Node.js/MySQL: Áp dụng kiến trúc module (ví dụ: MVC/MVVM), phân tách rõ ràng các concerns (Controller, Model, Route, Middleware), sử dụng các thư viện và framework có khả năng mở rộng.
- Xử lý lượng truy cập tăng lên:
    - Mục tiêu: Hệ thống cần có khả năng xử lý lượng người dùng và giao dịch tăng đột biến (ví dụ: trong các đợt khuyến mãi lớn) mà không bị sập hoặc suy giảm hiệu năng nghiêm trọng.

    - Thực hiện: Shopify cung cấp khả năng mở rộng tự động. Đối với các thành phần Node.js tùy chỉnh, cần thiết kế hệ thống theo hướng phân tán (microservices nếu phức tạp), sử dụng Load Balancer và có khả năng scale horizontally (thêm nhiều instance của ứng dụng).

d. Khả năng tương thích (Compatibility)

Website cần hoạt động ổn định và hiển thị chính xác trên nhiều môi trường khác nhau.

- Tương thích trình duyệt:
    - Mục tiêu: Website phải tương thích và hiển thị đúng trên các trình duyệt web phổ biến hiện nay (Chrome, Firefox, Edge, Safari phiên bản mới nhất và một số phiên bản cũ hơn).
    - Thực hiện: Sử dụng các tiêu chuẩn web, kiểm thử trên nhiều trình duyệt, đảm bảo CSS và JavaScript hoạt động nhất quán.
- Tương thích thiết bị di động (Responsive Design):
    - Mục tiêu: Giao diện website phải thân thiện và hiển thị tốt trên các loại thiết bị di động khác nhau (điện thoại thông minh, máy tính bảng) với nhiều kích thước màn hình.
    - Thực hiện: Áp dụng thiết kế đáp ứng (Responsive Web Design), sử dụng các framework UI/CSS thân thiện với di động. Shopify themes thường đã được tối ưu Responsive.

e. Tính dễ sử dụng (Usability - UI/UX)

Tính dễ sử dụng là yếu tố cốt lõi để thu hút và giữ chân người dùng, đặc biệt với mục tiêu "giao diện đơn giản để người dùng có thể sử dụng ngay".

Giao diện người dùng (UI - User Interface):

1. Mục tiêu: Giao diện phải sạch sẽ, trực quan, dễ nhìn, và thể hiện sự chuyên nghiệp, uy tín của một nền tảng bán đồ điện tử.
2. Thực hiện: Sử dụng thiết kế tối giản, màu sắc hài hòa, font chữ dễ đọc, bố cục hợp lý. Các nút chức năng phải rõ ràng, dễ bấm. Hình ảnh sản phẩm chất lượng cao.

Trải nghiệm người dùng (UX - User Experience):

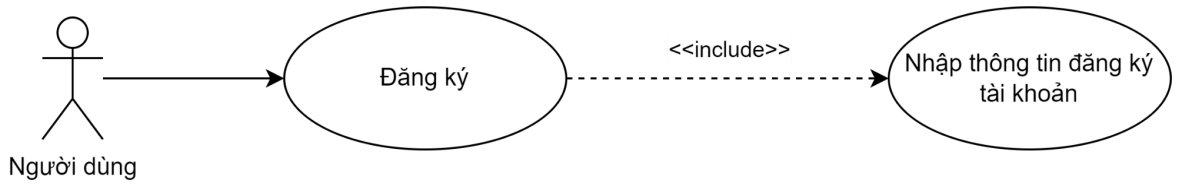
1. Mục tiêu: Quy trình từ tìm kiếm sản phẩm đến thanh toán phải liền mạch, đơn giản và ít bước nhất có thể. Người dùng phải cảm thấy thoải mái và tự tin khi sử dụng website.

2. Thực hiện:
- Điều hướng rõ ràng, menu dễ hiểu.
- Quy trình đăng ký/đăng nhập, giỏ hàng, thanh toán được tối ưu hóa.
- Thông báo lỗi hoặc thông tin phản hồi phải rõ ràng, thân thiện.
- Tính năng tìm kiếm và lọc sản phẩm hiệu quả.
- Thời gian phản hồi của hệ thống nhanh chóng.
- Tạo cảm giác an toàn và đáng tin cậy cho người dùng trong suốt quá trình mua sắm.

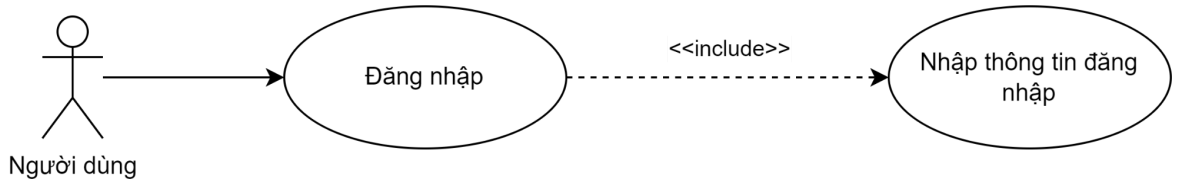
2.2. Biểu đồ use case của các chức năng

a, Biểu đồ usecase cho các chức năng chung của người dùng

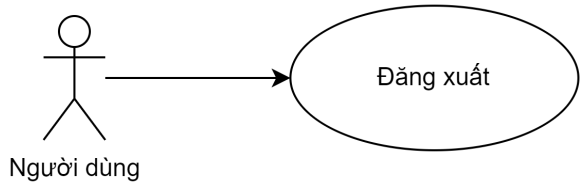
- Đăng ký tài khoản



- Đăng nhập



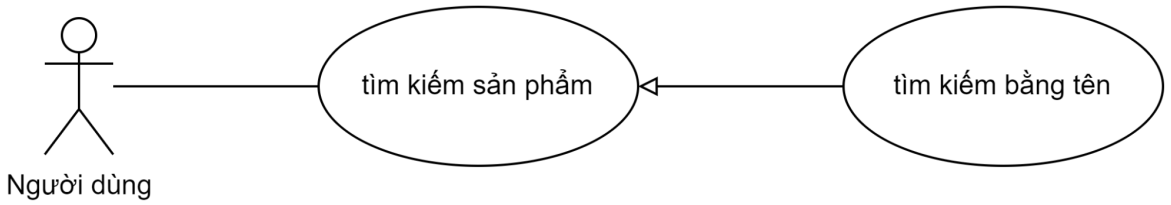
- Đăng xuất



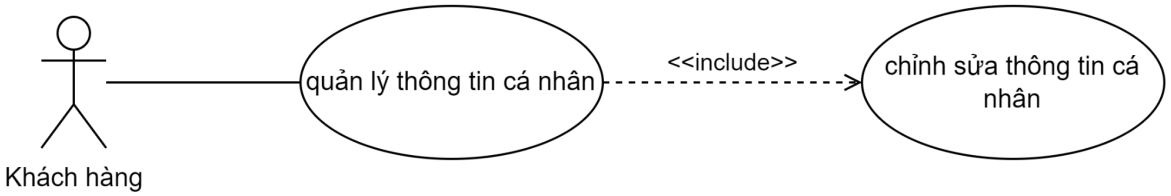
- Đổi mật khẩu

b, Biểu đồ use case cho các chức năng phía khách hàng

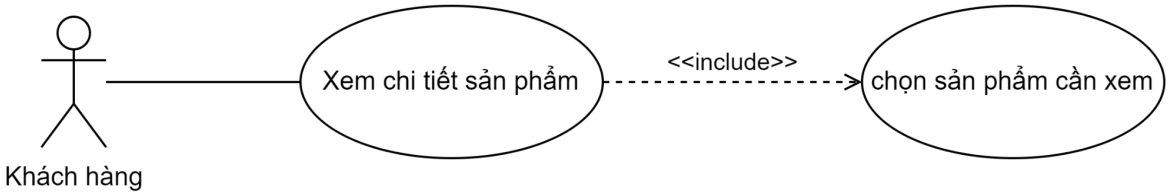
- Tìm kiếm sản phẩm



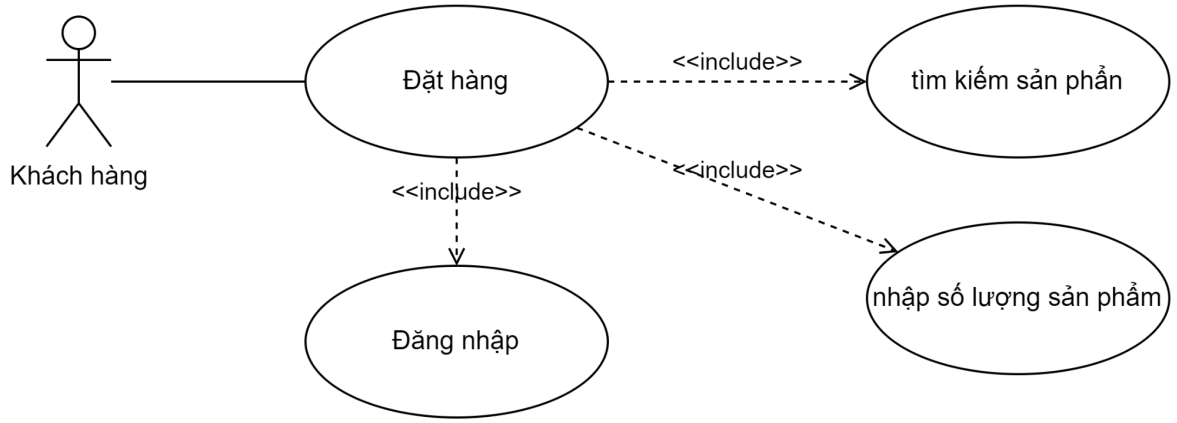
- Quản lý thông tin cá nhân



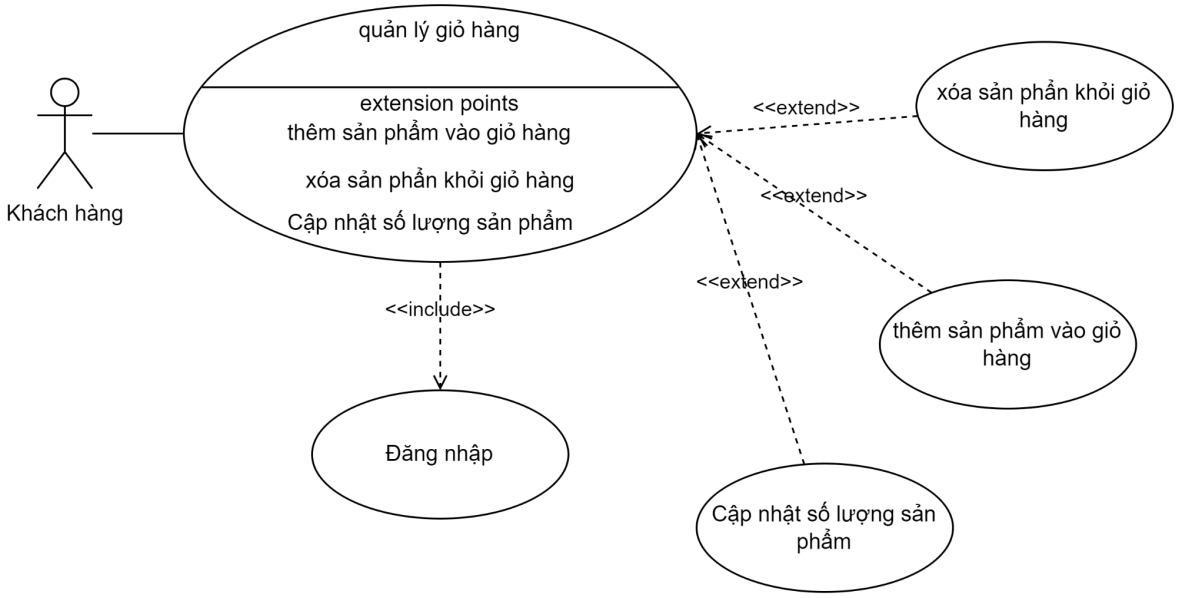
- Xem chi tiết sản phẩm



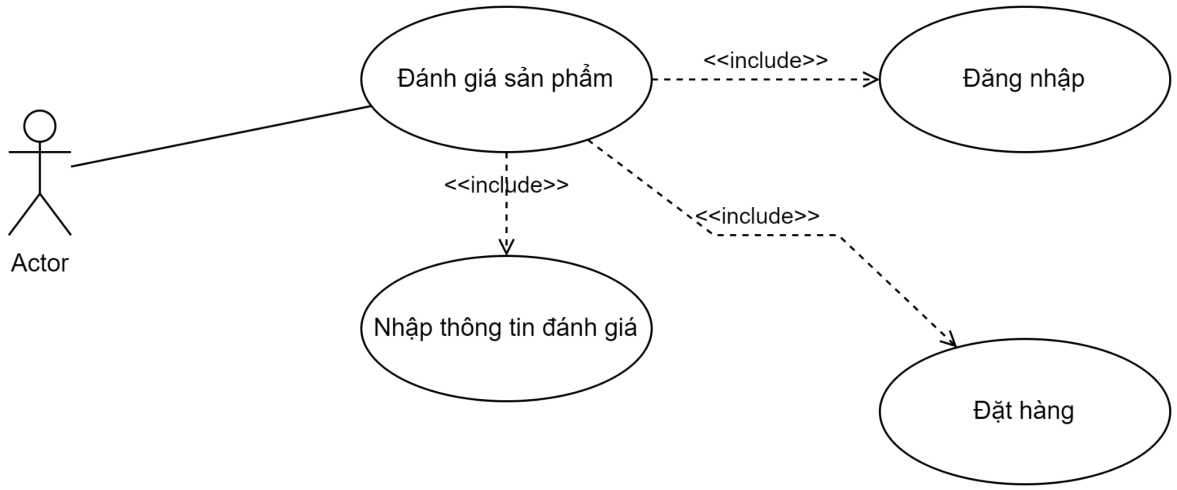
- Đặt hàng



- Quản lý giỏ hàng

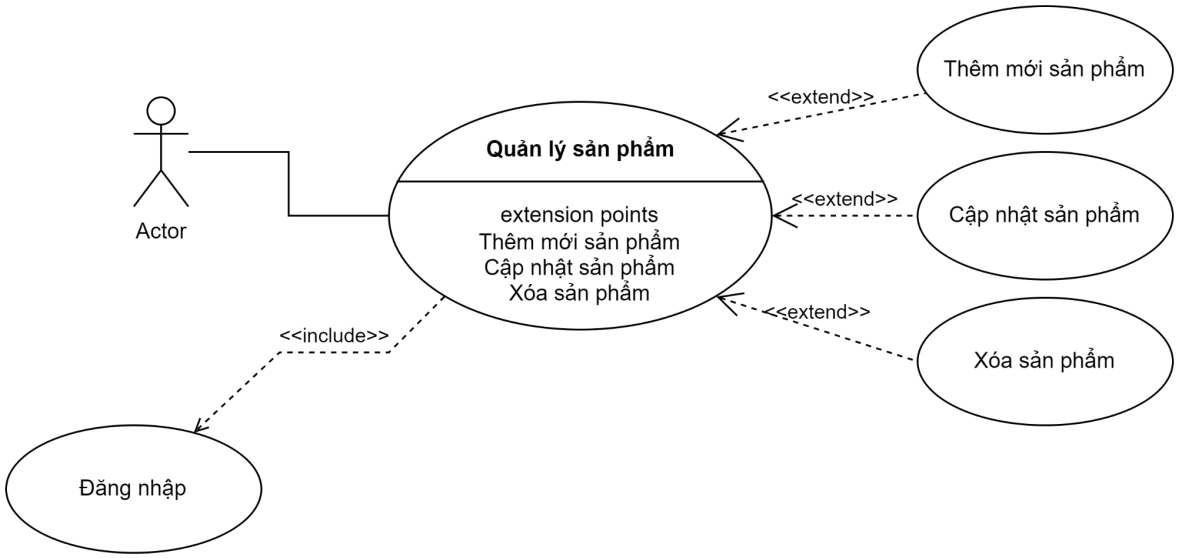


- Đánh giá sản phẩm

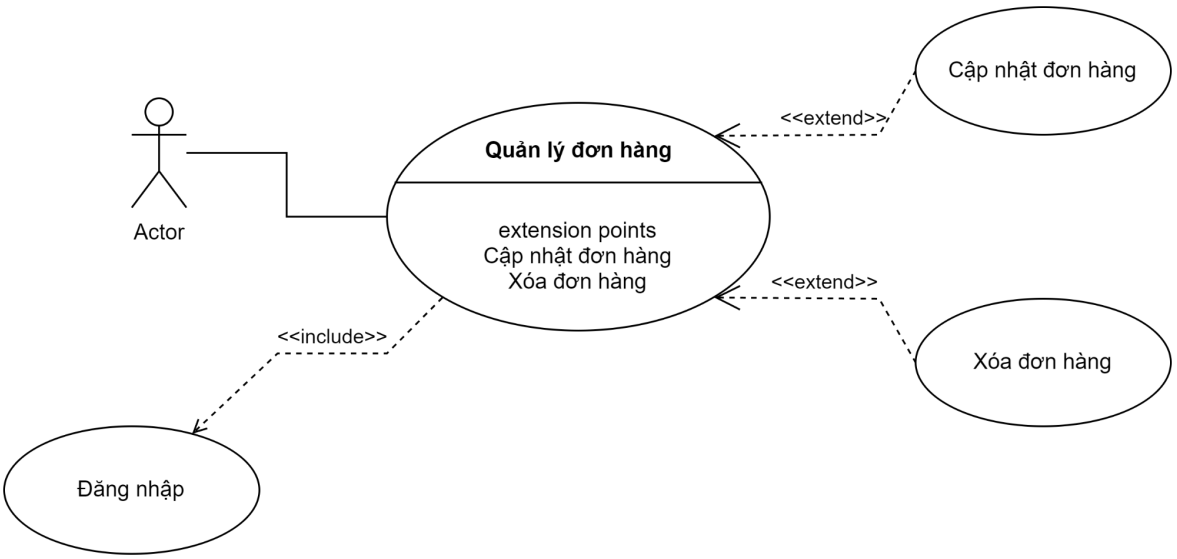


c, Biểu đồ use case các chức năng của người quản lý

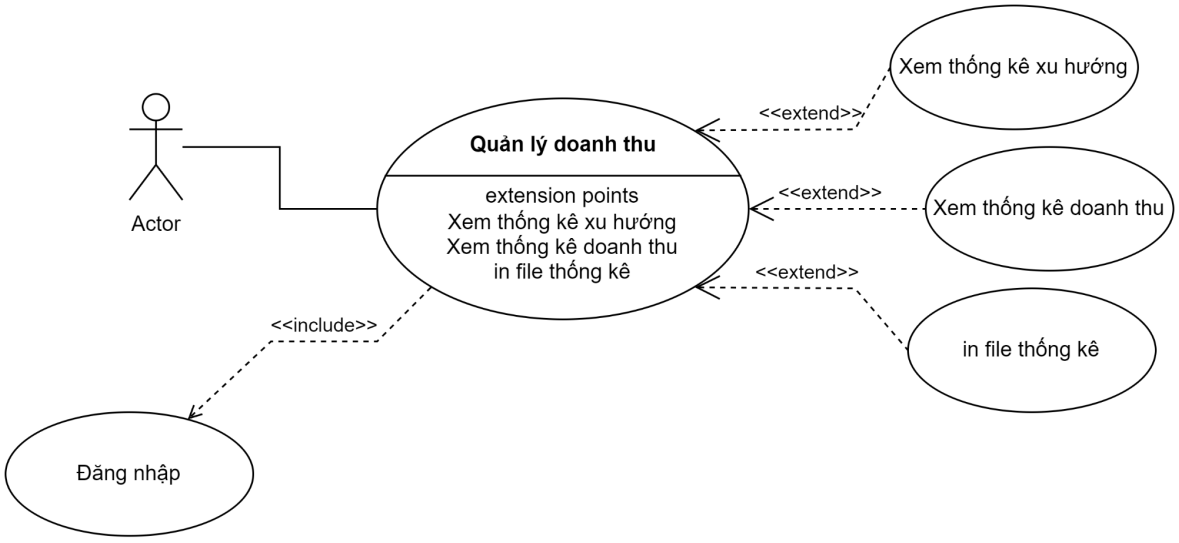
- Quản lý sản phẩm



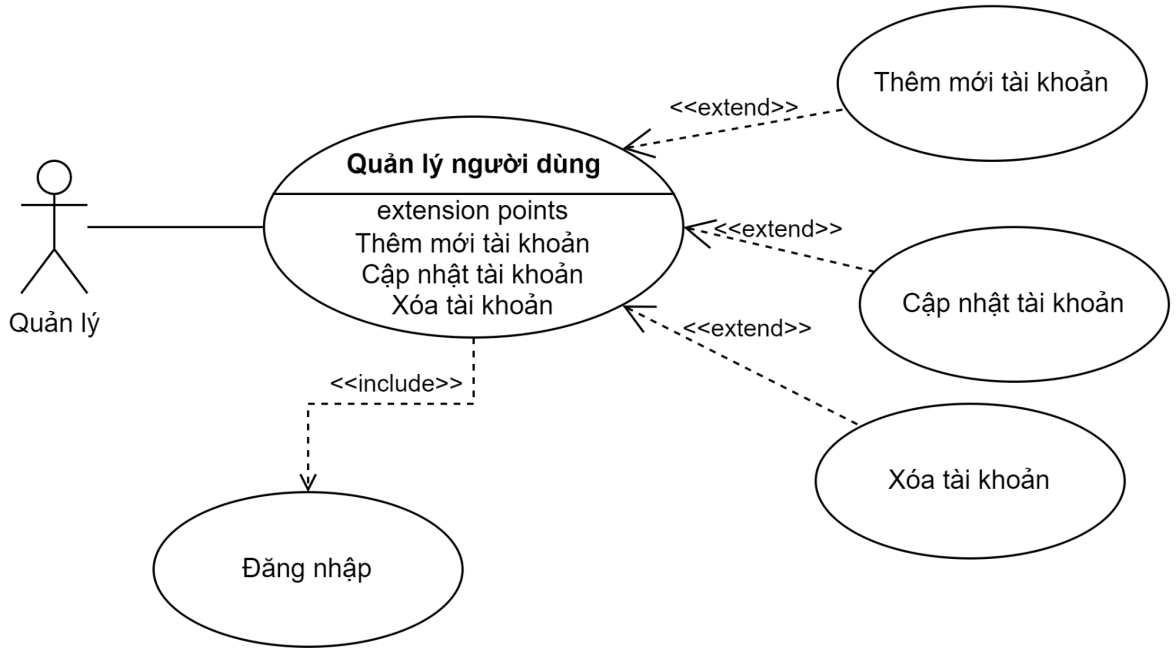
- Quản lý đơn hàng



- Quản lý doanh thu



- Quản lý người dùng

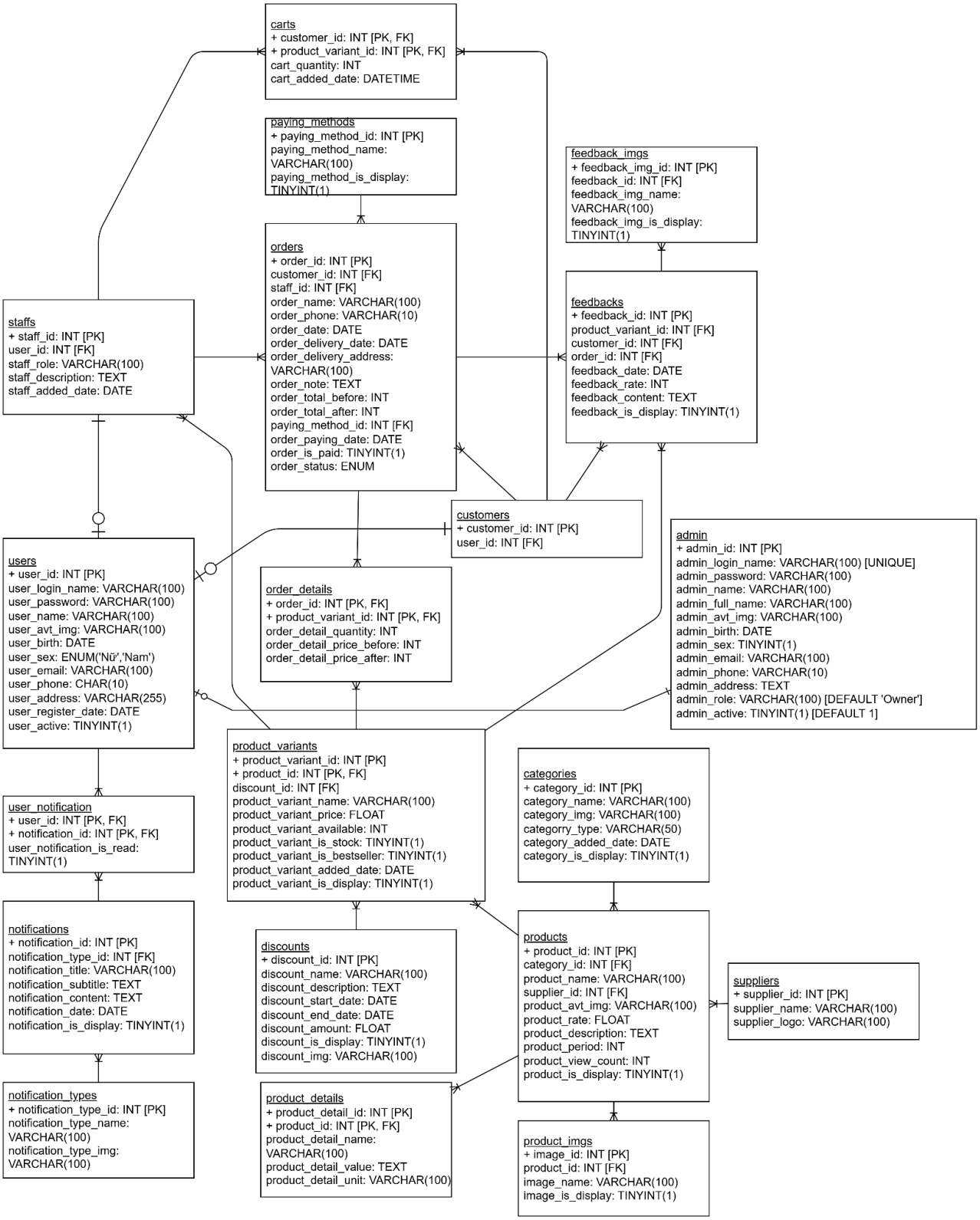


2.3. Thiết kế cơ sở dữ liệu

Thiết kế cơ sở dữ liệu là xương sống của mọi hệ thống phần mềm, đảm bảo dữ liệu được lưu trữ một cách có tổ chức, nhất quán và hiệu quả. Dựa trên các yêu cầu chức năng và nghiệp vụ, cơ sở dữ liệu được thiết kế để hỗ trợ mọi hoạt động của nền tảng TMĐT điện tử gia dụng.

Cơ sở dữ liệu của dự án có tên là tmdt_group5.

Mô hình Thực thể - Quan hệ (ERD - Entity-Relationship Diagram) tổng quan



2.4. Thiết kế kiến trúc hệ thống

Kiến trúc hệ thống định nghĩa cấu trúc tổng thể và cách các thành phần của ứng dụng tương tác với nhau. Nó đảm bảo hệ thống hoạt động hiệu quả, có khả năng mở rộng và dễ bảo trì.

2.4.1. Sơ đồ kiến trúc tổng thể

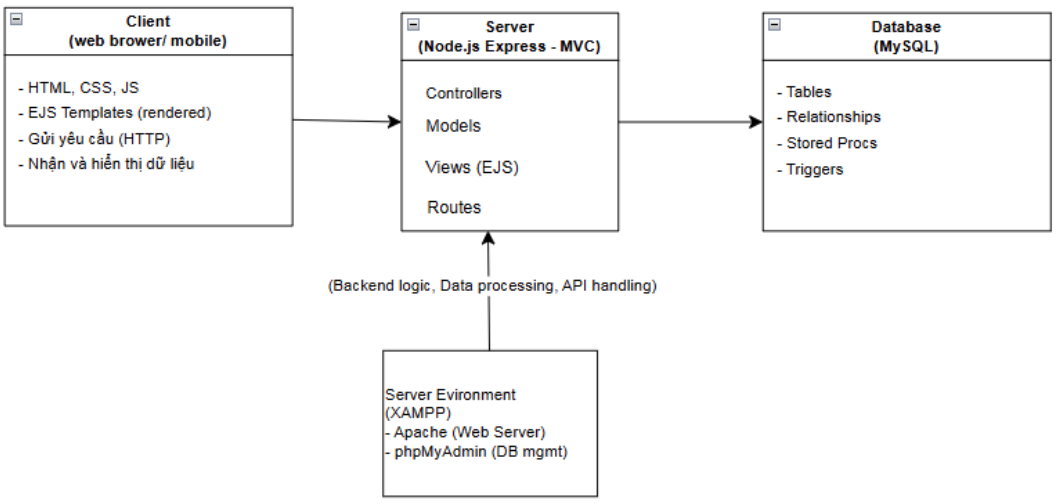
Kiến trúc tổng thể của hệ thống TMĐT này theo mô hình Client-Server, trong đó backend được tổ chức theo mô hình MVC (Model-View-Controller).

Các thành phần chính:

1. Client (Frontend):
    - Mô tả: Là giao diện người dùng mà khách hàng tương tác trực tiếp (trình duyệt web trên máy tính hoặc ứng dụng di động). Nó chịu trách nhiệm hiển thị dữ liệu, thu thập thông tin từ người dùng và gửi yêu cầu đến Backend.
    - Công nghệ: Sử dụng HTML, CSS, JavaScript và EJS (Embedded JavaScript) làm template engine để render giao diện động từ phía server.
    - Chức năng: Hiển thị trang chủ, danh sách sản phẩm, chi tiết sản phẩm, giỏ hàng, form đăng nhập/đăng ký, trang quản lý tài khoản người dùng, v.v.
2. Server (Backend API):
    - Mô tả: Là trái tim của hệ thống, xử lý logic nghiệp vụ, quản lý dữ liệu và giao tiếp với cơ sở dữ liệu. Được xây dựng theo mô hình MVC.
    - Công nghệ: Node.js làm môi trường runtime và Express.js làm framework web.
    - Các thành phần MVC:
        - Controllers (/src/controllers): Xử lý yêu cầu từ Client, gọi các hàm nghiệp vụ từ Models, và gửi dữ liệu đến Views để hiển thị. (Ví dụ: accountController.js, orderController.js, searchController.js).
        - Models (/src/models): Đại diện cho cấu trúc dữ liệu và logic nghiệp vụ liên quan đến dữ liệu (ví dụ: User model, Product model, Order model). Tương tác trực tiếp với Database.
        - Views (/src/views): Là các template (file .ejs) chịu trách nhiệm hiển thị dữ liệu nhận được từ Controllers ra giao diện người dùng. (Ví dụ: account/login.ejs, order/cart.ejs).
    - Routes (/src/routes): Định nghĩa các đường dẫn (URLs) và ánh xạ chúng tới các Controller tương ứng.

3. Database (Cơ sở dữ liệu):
    - Mô tả: Nơi lưu trữ tất cả dữ liệu của hệ thống một cách có cấu trúc và an toàn.
    - Công nghệ: MySQL (được quản lý thông qua XAMPP).
    - Chức năng: Lưu trữ thông tin người dùng, sản phẩm, danh mục, đơn hàng, giỏ hàng, đánh giá, thông báo, v.v.
4. Server Environment (Môi trường máy chủ):
    - Mô tả: Môi trường mà ứng dụng Node.js và MySQL chạy trên đó.
    - Công nghệ: Có thể là môi trường cục bộ (local development) sử dụng XAMPP (cung cấp Apache và MySQL) hoặc môi trường production trên các dịch vụ hosting đám mây (như AWS, Google Cloud, Heroku, Vercel...).

Sơ đồ Kiến trúc Tổng thể



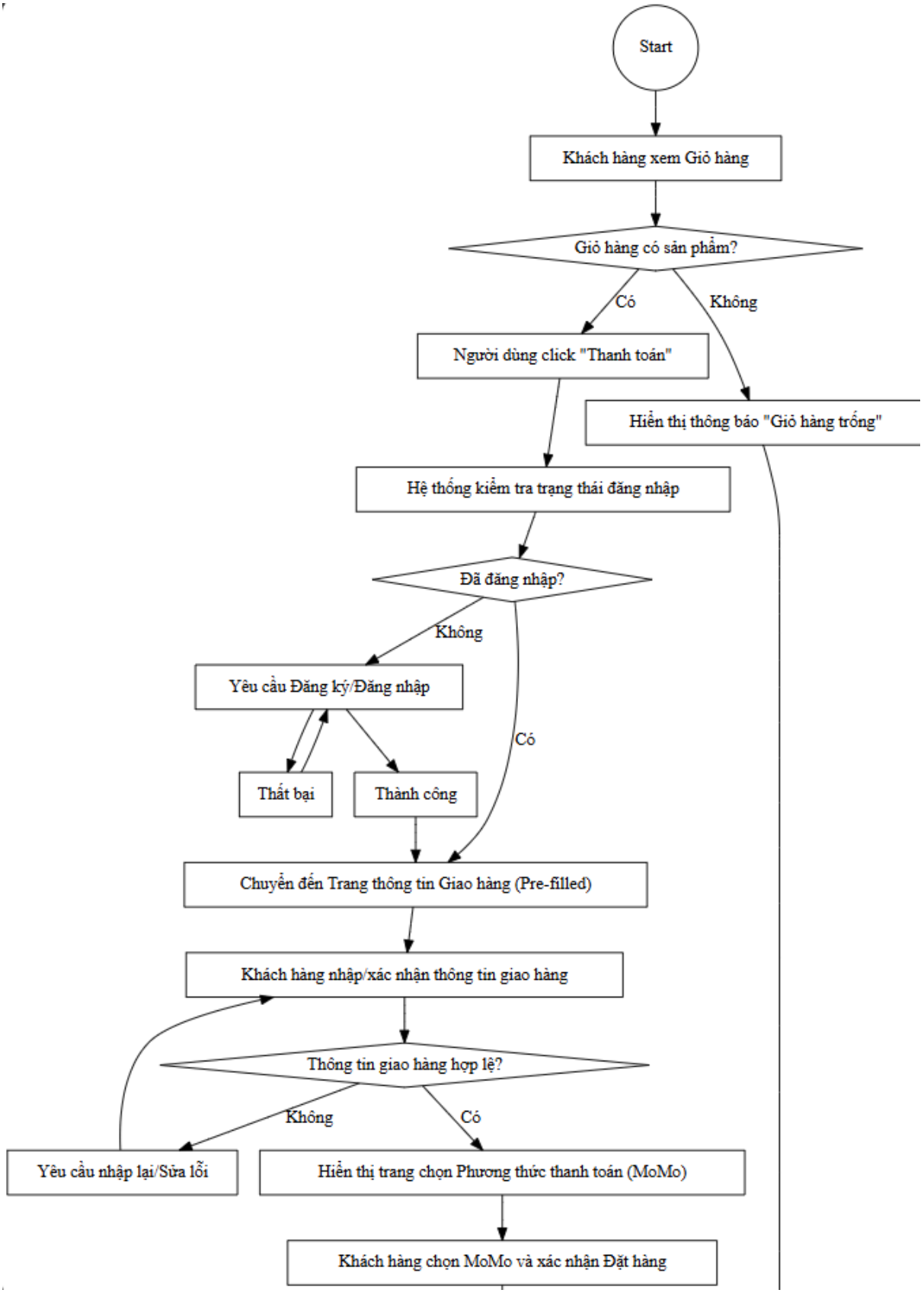
Giải thích luồng dữ liệu cơ bản:

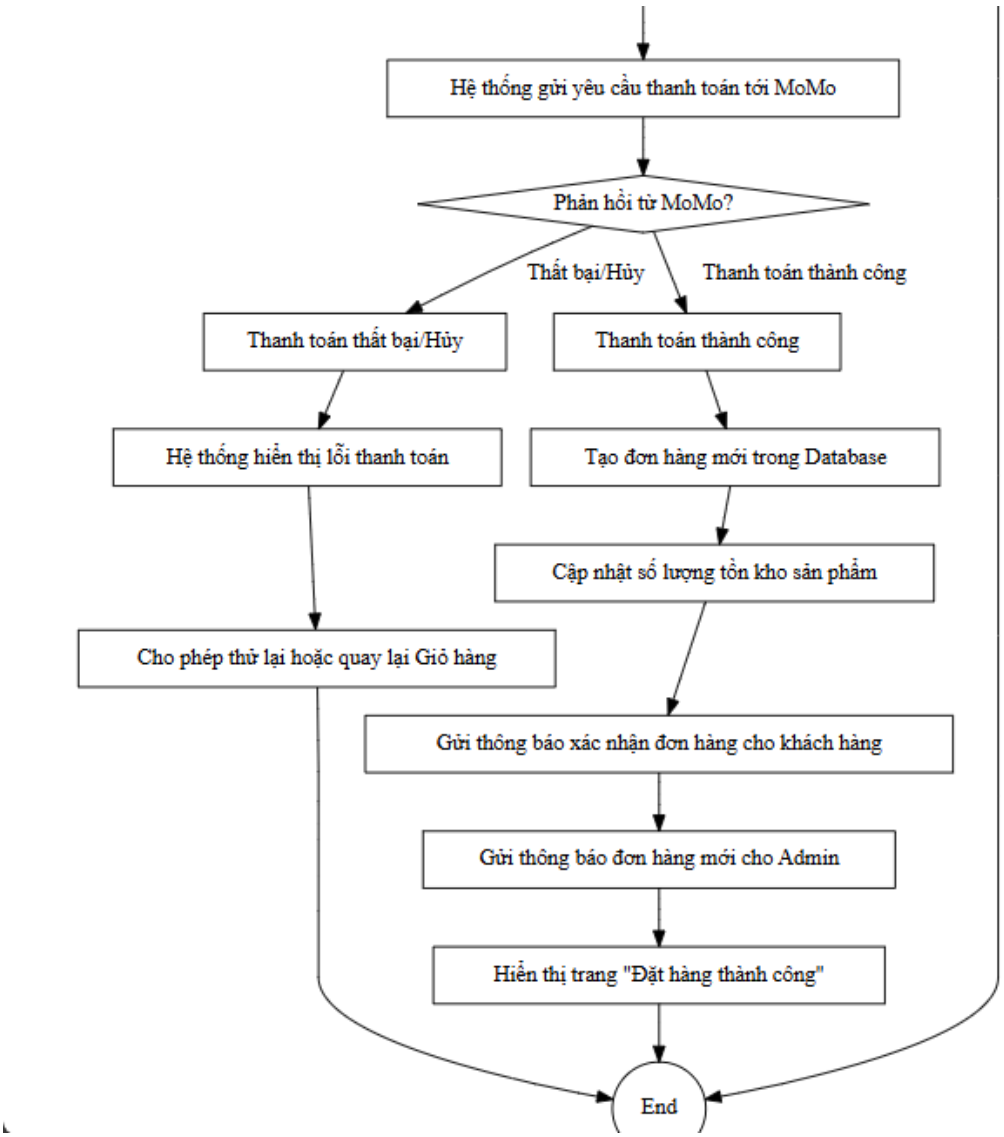
1. Người dùng tương tác với Client (ví dụ: bấm vào nút "Thêm vào giỏ hàng").
2. Client gửi yêu cầu HTTP (POST /add-to-cart) đến Server (Backend API).
3. Routes trên Server nhận yêu cầu và điều hướng đến một hàm tương ứng trong Controllers (ví dụ: cartController.addToCart).
4. Controllers gọi các phương thức nghiệp vụ từ Models (ví dụ: CartModel.addItem) để xử lý logic và tương tác với Database.

5. Models thực hiện các truy vấn SQL đến Database (MySQL) để lưu trữ hoặc truy xuất dữ liệu.
6. Database trả về kết quả cho Models.
7. Models trả về kết quả (dữ liệu hoặc trạng thái) cho Controllers.
8. Controllers có thể xử lý thêm dữ liệu hoặc quyết định render một View với dữ liệu đó, hoặc gửi một phản hồi JSON về Client.
9. Client nhận phản hồi và cập nhật giao diện người dùng.

2.4.2. Sơ đồ luồng dữ liệu (Data Flow Diagram - DFD) hoặc Sơ đồ hoạt động (Activity Diagram)

Sơ đồ hoạt động: Luồng Đặt hàng (Checkout Flow)


Start
Khách hàng xem Giỏ hàng
Giỏ hàng có sản phẩm?
Có
Không
Người dùng click "Thanh toán"
Hiển thị thông báo "Giỏ hàng trống"
Hệ thống kiểm tra trạng thái đăng nhập
Đã đăng nhập?
Không
Có
Yêu cầu Đăng ký/Đăng nhập
Thất bại
Thành công
Chuyển đến Trang thông tin Giao hàng (Pre-filled)
Khách hàng nhập/xác nhận thông tin giao hàng
Thông tin giao hàng hợp lệ?
Không
Có
Yêu cầu nhập lại/Sửa lỗi
Hiển thị trang chọn Phương thức thanh toán (MoMo)
Khách hàng chọn MoMo và xác nhận Đặt hàng



III. TRIỂN KHAI HỆ THỐNG

3.1. Môi trường phát triển và công cụ sử dụng

Để phát triển một hệ thống thương mại điện tử hoàn chỉnh và hiệu quả, việc lựa chọn và thiết lập môi trường phát triển cùng các công cụ phù hợp là rất quan trọng. Chúng giúp tối ưu hóa quy trình làm việc, tăng năng suất và đảm bảo chất lượng sản phẩm.

Dưới đây là danh sách các công cụ, phần mềm và môi trường phát triển đã và đang được sử dụng cho dự án này:

1. Hệ điều hành:
    - Microsoft Windows / macOS / Linux: Cung cấp môi trường nền tảng để cài đặt và chạy các phần mềm phát triển.

2. Môi trường phát triển tích hợp (IDE - Integrated Development Environment) / Trình soạn thảo mã (Code Editor):
    - Visual Studio Code (VS Code): Là trình soạn thảo mã nguồn nhẹ, mạnh mẽ và phổ biến. VS Code cung cấp nhiều tính năng hữu ích như IntelliSense (gợi ý mã), Debugging (gỡ lỗi), tích hợp Git, và một hệ sinh thái extension phong phú hỗ trợ tốt cho việc phát triển Node.js, HTML, CSS, JavaScript và các template engine như EJS.
3. Môi trường máy chủ cục bộ (Local Server Environment):
    - XAMPP: Được sử dụng để tạo một môi trường máy chủ cục bộ trên máy tính cá nhân. XAMPP bao gồm:
        - Apache HTTP Server: Máy chủ web để phục vụ các tệp tin tĩnh và động.
        - MySQL (hoặc MariaDB): Hệ quản trị cơ sở dữ liệu quan hệ được sử dụng để lưu trữ dữ liệu của ứng dụng.
        - phpMyAdmin: Giao diện web để quản lý cơ sở dữ liệu MySQL một cách trực quan.
    - Node.js Runtime Environment: Môi trường thực thi JavaScript phía máy chủ. Đây là cốt lõi để chạy ứng dụng backend được xây dựng bằng Express.js.
4. Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS):
    - MySQL: Là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở được lựa chọn để lưu trữ tất cả dữ liệu của hệ thống (sản phẩm, người dùng, đơn hàng, v.v.). Việc này được triển khai thông qua XAMPP.
5. Ngôn ngữ lập trình & Framework:
    - JavaScript: Ngôn ngữ lập trình chính cho cả frontend và backend.
    - Node.js: Môi trường runtime cho JavaScript phía máy chủ.
    - Express.js: Framework web tối giản và linh hoạt cho Node.js, được sử dụng để xây dựng API và xử lý logic backend.
    - EJS (Embedded JavaScript): Template engine được sử dụng để tạo các View động ở phía backend, kết hợp dữ liệu với HTML để tạo ra giao diện người dùng.
6. Hệ thống quản lý phiên bản (Version Control System):
    - Git: Hệ thống quản lý phiên bản phân tán, được sử dụng để theo dõi các thay đổi trong mã nguồn, cộng tác nhóm và quản lý các phiên bản dự án.

- GitHub / GitLab / Bitbucket: Các nền tảng lưu trữ kho mã nguồn Git trực tuyến, hỗ trợ cộng tác nhóm và quản lý dự án.

7. Công cụ quản lý gói (Package Manager):
    - npm (Node Package Manager): Trình quản lý gói mặc định cho Node.js, được sử dụng để cài đặt, quản lý và chia sẻ các thư viện (packages) JavaScript cho dự án.
8. Trình duyệt web:
    - Google Chrome / Mozilla Firefox / Microsoft Edge: Các trình duyệt hiện đại được sử dụng để kiểm thử và xem giao diện người dùng của website.
9. Công cụ kiểm thử API (API Testing Tool):
    - Postman / Thunder Client (VS Code Extension): Các công cụ giúp gửi yêu cầu HTTP đến API backend và kiểm tra phản hồi, hỗ trợ quá trình phát triển và gỡ lỗi API.

3.2. Triển khai các Module chức năng:

3.2.1. Backend

Phần Backend của hệ thống TMĐT là nơi xử lý toàn bộ logic nghiệp vụ, quản lý dữ liệu và cung cấp các API cho Frontend. Dự án này sử dụng Node.js với Express.js để xây dựng các API, tương tác với cơ sở dữ liệu MySQL.

a. Chi tiết cách xây dựng các API (RESTful API)

Các API được thiết kế để thực hiện các thao tác trên tài nguyên. Dựa vào các controller, có thể thấy các loại API chính đang được xây dựng:

API quản lý tài khoản (/account):	
GET /account/information	Lấy thông tin cá nhân của người dùng.
GET /account/edit-information	Lấy dữ liệu để hiển thị form chỉnh sửa thông tin.
POST /account/edit-information	Cập nhật thông tin cá nhân của người dùng.
GET /account/purchase-history	Lấy lịch sử mua hàng của khách hàng.

GET /account/purchase-history/detail/:order_id	Lấy chi tiết một đơn hàng cụ thể trong lịch sử mua hàng.
GET /account/feedback	Lấy thông tin chi tiết đơn hàng để người dùng gửi feedback.
POST /account/feedback	Gửi feedback (đánh giá sản phẩm) của khách hàng.
GET /account/warranty-claim	Lấy lịch sử mua hàng để yêu cầu bảo hành.
GET /account/changePassword	Lấy trang đổi mật khẩu.
GET /account/mobileAccount	Lấy trang quản lý tài khoản trên di động.
API xác thực & ủy quyền (/auth):	
GET /auth/register	Hiển thị trang đăng ký.
POST /auth/register	Xử lý yêu cầu đăng ký tài khoản mới.
GET /login	Hiển thị trang đăng nhập.
POST /login	Xử lý yêu cầu đăng nhập và tạo session/JWT.
GET /logout	Đăng xuất người dùng.
GET /auth/forgotPassword	Hiển thị trang quên mật khẩu.
POST /auth/findUser	Tìm kiếm người dùng bằng số điện thoại cho chức năng quên mật khẩu.
POST /auth/forgotPasswordPost	Xử lý sau khi tìm thấy số điện thoại
POST /auth/resetPassword	Đặt lại mật khẩu.
POST /auth/change-password	Thay đổi mật khẩu khi đã đăng nhập.
API quản lý giỏ hàng & đặt hàng (/order)	
POST /order/addCart	Thêm sản phẩm vào giỏ hàng.
GET /order/cart	Xem chi tiết giỏ hàng.
DELETE /order/deleteCart	Xóa sản phẩm khỏi giỏ hàng.
PUT /order/updateCart	Cập nhật số lượng sản phẩm trong giỏ

	hàng.
GET /order/information	Lấy thông tin để hiển thị trang xác nhận đơn hàng.
POST /order/information	Xử lý tạo đơn hàng mới.
GET /order/paymen	Chuyển hướng đến trang thanh toán (MoMo, ATM, Credit Card).
POST /order/cancelOrder	Hủy đơn hàng.
API tìm kiếm & chi tiết sản phẩm (/search)	
GET /search/results	Hiển thị kết quả tìm kiếm hoặc danh sách sản phẩm theo danh mục.
GET /search/:product_variant_id	Lấy chi tiết một biến thể sản phẩm.
API thông báo (/notification)	
GET /notification/order	Lấy thông báo liên quan đến đơn hàng của người dùng.
GET /notification/promotion	Lấy thông báo khuyến mãi cho người dùng.
POST /notifications/order	Đánh dấu một thông báo là đã đọc.
POST /notifications/read-all	Đánh dấu tất cả thông báo theo loại là đã đọc.
API chung (/)	
GET /	Trang chủ, hiển thị sản phẩm nổi bật, mới, giảm giá.
GET /about-us	Trang giới thiệu.
GET /privacy-policy	Trang chính sách bảo mật.
GET /error	Trang lỗi 404.
GET /category	Trang danh mục cho di động.
GET /feature_development	Trang thông báo tính năng đang phát triển.

b. Cách xử lý logic nghiệp vụ

Logic nghiệp vụ được phân tách rõ ràng giữa các lớp: Controllers và Models.

- Controllers:
    - Tiếp nhận yêu cầu: Các controller (ví dụ: accountController, authController, orderController, etc.) tiếp nhận các yêu cầu HTTP từ client.
    - Điều phối: Chúng đóng vai trò điều phối, gọi các phương thức nghiệp vụ từ các Models tương ứng.
    - Xử lý phản hồi: Sau khi nhận kết quả từ Models, controller sẽ định dạng phản hồi (JSON cho API, hoặc render EJS template cho các trang) và gửi về client.
    - Xử lý lỗi cơ bản: Bắt các lỗi phát sinh từ Model hoặc các vấn đề request/response và gửi phản hồi lỗi thích hợp.

Models:

- Chứa logic nghiệp vụ chính: Các file model (ví dụ: account.model, auth.model, order.model, product.model, search.model, noti.model, index.model, general.model) là nơi chứa các hàm thực hiện logic nghiệp vụ phức tạp và tương tác với cơ sở dữ liệu.

c. Cách tương tác với cơ sở dữ liệu

Tương tác với cơ sở dữ liệu MySQL được thực hiện thông qua module kết nối cơ sở dữ liệu (../../config/db/connect).

- Kết nối: Module connect.js (không được cung cấp trực tiếp, nhưng được import trong authController.js) sẽ thiết lập kết nối đến MySQL, khả năng cao là sử dụng mysql2/promise để hỗ trợ async/await.
- Truy vấn SQL: Các Models gọi trực tiếp các hàm từ module kết nối để thực hiện các truy vấn SQL (SELECT, INSERT, UPDATE, DELETE).
- Sử dụng Callbacks/Promises: Các hàm trong Models sử dụng cả callback. Việc sử dụng async/await (đặc biệt khi dùng promisify cho các hàm callback hoặc thư viện mysql2/promise) giúp quản lý luồng bất đồng bộ dễ đọc và dễ bảo trì hơn.

d. Phương pháp xác thực và phân quyền người dùng

Dự án sử dụng kết hợp Session/Cookies và JWT (JSON Web Tokens) để xác thực người dùng.

- Xác thực (Authentication):

    - Đăng ký: Người dùng đăng ký tài khoản với số điện thoại và mật khẩu. Mật khẩu được băm ($2a$08... trong dữ liệu admin và users cho thấy sử dụng bcrypt) trước khi lưu vào cơ sở dữ liệu.
    - Đăng nhập:
        - Người dùng cung cấp số điện thoại và mật khẩu.
        - Hệ thống kiểm tra số điện thoại có tồn tại không.
        - So sánh mật khẩu đã băm trong DB với mật khẩu người dùng nhập vào.
        - Nếu thành công, một JWT được tạo ra với user_id và ký bởi process.env.JWT_SECRET.
        - JWT này sau đó được lưu vào một cookie có tên userSave với các tùy chọn expires và httpOnly. Cookie httpOnly giúp chống tấn công XSS liên quan đến việc đánh cắp cookie.
    - Đăng xuất: Cookie userSave được đặt lại với giá trị "logout" và thời gian hết hạn trong 2 giây để xóa nó khỏi trình duyệt.
    - req.user: Trong các controller như accountController, orderController, notificationController, searchController, thông tin người dùng được truy cập thông qua req.user.customer_id hoặc req.user.user_id, điều này ngụ ý rằng có một middleware xác thực JWT đã chạy trước đó để giải mã token và gắn thông tin người dùng vào đối tượng req.
- Phân quyền (Authorization):
    - Kiểm tra đăng nhập: Các route như orderController.addCart kiểm tra req.user để xác định người dùng đã đăng nhập hay chưa (req.user ? req.user.customer_id : 0). Nếu chưa, sẽ trả về status: "NotAuth". Điều này là một dạng phân quyền cơ bản: chỉ người dùng đã xác thực mới có thể thực hiện hành động.
    - Phân quyền dựa trên vai trò (rõ ràng hơn trong adminController): Mặc dù các controller khách hàng không trực tiếp hiển thị logic authorizeRoles, việc tồn tại bảng admin với admin_role và bảng staffs với staff_role cùng với controller adminController cho thấy hệ thống có khả năng phân quyền theo vai trò (ví dụ: chỉ Owner mới có quyền truy cập một số chức năng quản trị, hoặc Admin có quyền rộng hơn Staff). Điều này được thực hiện thông qua middleware.

3.2.2. Frontend

Trong dự án này, Frontend được xây dựng dựa trên các file template EJS, kết hợp với HTML, CSS và JavaScript để tạo ra trải nghiệm người dùng.

a. Cách xây dựng giao diện người dùng dựa trên thiết kế

Giao diện người dùng được xây dựng bằng cách sử dụng các tệp tin EJS. EJS cho phép nhúng mã JavaScript trực tiếp vào các tệp HTML, giúp tạo ra nội dung động.

- Cấu trúc thư mục Views (/src/views):
    - Các tệp .ejs được tổ chức theo các thư mục con tương ứng với từng phần của website (ví dụ: pages/account, pages/order, pages/search, pages/site).
    - Thư mục partials (/src/views/partials) chứa các thành phần giao diện dùng chung (ví dụ: header, footer, sidebar) để tái sử dụng mã và đảm bảo tính nhất quán.
    - Thư mục components (/src/views/components) có thể chứa các khối UI nhỏ hơn, có thể tái sử dụng trong các trang khác nhau.
- Tái sử dụng các thành phần:
    - include directive của EJS (ví dụ: <%- include('../partials/header.ejs', { header: header, user: user }) %>) được sử dụng rộng rãi để nhúng các phần giao diện chung hoặc các component vào các trang chính, giúp quản lý mã hiệu quả và dễ dàng cập nhật giao diện tổng thể.
- Styling (CSS):
    - Các file CSS được sử dụng để định kiểu cho giao diện, có thể được tổ chức theo từng module hoặc theo cách truyền thống, được liên kết vào các tệp EJS.
- JavaScript (Client-side):
    - Các script JavaScript được viết để xử lý các tương tác người dùng, hiệu ứng động, và gửi các yêu cầu bất đồng bộ đến Backend API.

b. Sử dụng các component, state management (Redux, Vuex nếu có)

Dựa trên cấu trúc EJS, dự án này không sử dụng các framework Frontend JavaScript như React, Vue, hay Angular, do đó không có các thư viện quản lý trạng thái phức tạp như Redux hay Vuex.

- Components:
    - Các "component" ở đây được hiểu là các khối giao diện được tạo ra từ các file EJS riêng lẻ (trong thư mục partials hoặc components) và được tái sử dụng trên nhiều trang khác nhau. Đây là một cách tiếp cận module hóa giao diện ở mức độ template.
- State Management:

- Việc quản lý trạng thái (state management) thường đơn giản hơn, chủ yếu dựa vào:
  - Biến cục bộ trong EJS: Dữ liệu được truyền từ Backend (qua res.render) và được sử dụng trực tiếp trong EJS để hiển thị.
  - Biến JavaScript toàn cục hoặc cục bộ: Các biến JavaScript trên client-side được sử dụng để lưu trữ trạng thái của UI hoặc dữ liệu tạm thời cho các tương tác nhỏ.
  - DOM Manipulation: Các thao tác trực tiếp trên DOM bằng JavaScript/jQuery để thay đổi giao diện dựa trên tương tác người dùng.

c. Cách tương tác với API Backend

Tương tác với Backend API chủ yếu được thực hiện thông qua các yêu cầu HTTP từ phía Client (trình duyệt).

- Yêu cầu đồng bộ (Form Submissions):
  - Đối với các thao tác như đăng ký, đăng nhập, hoặc cập nhật thông tin cá nhân (ví dụ: accountController.editInformation xử lý POST /account/edit-information), các form HTML được gửi đi một cách đồng bộ. Khi form được submit, trình duyệt sẽ tải lại trang với phản hồi từ server (res.redirect hoặc res.render).
- Yêu cầu bất đồng bộ (AJAX/Fetch API):
  - Đối với các chức năng yêu cầu cập nhật giao diện mà không tải lại toàn bộ trang (ví dụ: thêm sản phẩm vào giỏ hàng, xóa/cập nhật sản phẩm trong giỏ, gửi feedback, đánh dấu thông báo đã đọc), JavaScript ở phía client sẽ sử dụng AJAX (Asynchronous JavaScript and XML) hoặc Fetch API để gửi các yêu cầu HTTP (POST, PUT, DELETE) đến Backend API.
  - Ví dụ từ orderController.js và accountController.js:
    - orderController.addCart (POST /order/addCart): Frontend sẽ gửi một yêu cầu POST AJAX/Fetch để thêm sản phẩm vào giỏ hàng và nhận phản hồi JSON (status: "success" / "error"), sau đó cập nhật biểu tượng giỏ hàng mà không cần tải lại trang.
    - orderController.deleteCart, orderController.updateCart: Tương tự, sử dụng AJAX để gửi yêu cầu DELETE/PUT.
    - accountController.sendFeedback: Gửi dữ liệu feedback qua AJAX và nhận phản hồi JSON.

        - notificationsController.readNotification, notificationsController.readAllNotifications: Gửi yêu cầu POST AJAX để cập nhật trạng thái thông báo.
        - authController.checkUser, authController.findUser, authController.resetPassword, authController.changePassPost: Các yêu cầu này cũng sử dụng AJAX/Fetch để xác thực, tìm kiếm người dùng, đặt lại mật khẩu hoặc đổi mật khẩu và nhận phản hồi JSON để cập nhật UI ngay lập tức.

d. Xử lý hiển thị dữ liệu và tương tác người dùng

- Hiển thị dữ liệu (Server-Side Rendering với EJS):
    - Khi một yêu cầu HTTP (GET) đến Backend, Controller sẽ lấy dữ liệu từ Models, sau đó truyền dữ liệu này vào các tệp EJS thông qua res.render().
    - EJS sẽ "biên dịch" (compile) template bằng cách chèn dữ liệu vào các vị trí tương ứng trong HTML.
    - Kết quả là một tệp HTML hoàn chỉnh được gửi về trình duyệt của người dùng. Trình duyệt chỉ cần hiển thị HTML đó.
    - Ví dụ: Trong siteController.index, dữ liệu outstandingProducts, newProducts, discountProducts được lấy từ general.model và truyền vào index.ejs để hiển thị danh sách sản phẩm trên trang chủ.
- Tương tác người dùng (Client-Side JavaScript):
    - Sau khi trang HTML được tải, JavaScript ở phía client sẽ đảm nhiệm việc xử lý các tương tác động:
        - Sự kiện (Events): Lắng nghe các sự kiện từ người dùng (click, hover, submit form).
        - Thay đổi DOM: Cập nhật nội dung, hiển thị/ẩn các phần tử, thay đổi lớp CSS để tạo hiệu ứng.
        - Xác thực phía client: Kiểm tra tính hợp lệ của dữ liệu form trước khi gửi đến server để cải thiện trải nghiệm và giảm tải cho server.
        - Hiển thị thông báo: Pop-up, toast messages sau khi một hành động hoàn tất (ví dụ: "Thêm vào giỏ hàng thành công").

Mô hình này tối ưu cho việc tải trang ban đầu nhanh chóng (vì HTML được render sẵn), và sử dụng JavaScript cục bộ để cung cấp các tương tác động cần thiết mà không yêu cầu một framework frontend phức tạp.

3.2.3. Cơ sở dữ liệu

Cơ sở dữ liệu là nơi lưu trữ toàn bộ thông tin của hệ thống TMĐT, đảm bảo tính toàn vẹn và sẵn sàng của dữ liệu. Dự án sử dụng hệ quản trị cơ sở dữ liệu **MySQL**.

Quá trình cài đặt và cấu hình

Quá trình cài đặt và cấu hình cơ sở dữ liệu MySQL cho dự án thường được thực hiện thông qua gói phần mềm XAMPP, mang lại một môi trường phát triển cục bộ thuận tiện.

1. Cài đặt XAMPP:
    - Tải xuống và cài đặt XAMPP từ trang web chính thức của Apache Friends (apachefriends.org). Quá trình cài đặt này sẽ bao gồm MySQL Server, Apache HTTP Server và phpMyAdmin.
    - Sau khi cài đặt, khởi động các dịch vụ Apache và MySQL từ XAMPP Control Panel.
2. Tạo cơ sở dữ liệu:
    - Truy cập phpMyAdmin thông qua trình duyệt web (thường là http://localhost/phpmyadmin).
    - Tạo một cơ sở dữ liệu mới với tên là tmdt_group5.
3. Import cấu trúc và dữ liệu:
    - Sử dụng công cụ Import trong phpMyAdmin.
    - Chọn tệp .sql (ví dụ: tmdt_group5.sql).
    - Thực hiện import để tạo tất cả các bảng, ràng buộc, triggers và views đã được định nghĩa trong tệp SQL vào cơ sở dữ liệu tmdt_group5.
4. Cấu hình kết nối trong ứng dụng Node.js:
    - Tệp cấu hình cơ sở dữ liệu (/config/db/connect.js) đã chứa các thông tin kết nối tới MySQL.

IV. KIỂM THỬ VÀ ĐÁNH GIÁ

Chương này trình bày quá trình kiểm thử giả định và đánh giá kết quả thực tế của hệ thống thương mại điện tử "TechTwo" chuyên về điện thoại và đồ điện tử gia đình, tập trung vào các chức năng cơ bản và các yêu cầu phi chức năng.

4.1. Kế hoạch kiểm thử (Thực hiện)

Kế hoạch kiểm thử được triển khai để xác nhận mọi khía cạnh của hệ thống hoạt động đúng như mong đợi.

4.1.1. Mục tiêu kiểm thử (Đạt được)

- Đảm bảo tất cả các yêu cầu chức năng được đáp ứng: Mọi tính năng cốt lõi đều được kiểm tra và xác nhận hoạt động chính xác.
- Đảm bảo yêu cầu phi chức năng: Hiệu năng, bảo mật, khả năng tương thích và tính dễ sử dụng đều được xem xét kỹ lưỡng.
- Phát hiện và sửa lỗi: Các lỗi nghiêm trọng được phát hiện và khắc phục trước khi đưa vào đánh giá cuối cùng.
- Đảm bảo tính nhất quán dữ liệu: Dữ liệu được lưu trữ và hiển thị chính xác trên toàn hệ thống.
- Cải thiện trải nghiệm người dùng (UX): Giao diện và luồng thao tác được tinh chỉnh để thân thiện nhất với người dùng.

4.1.2. Các loại kiểm thử (Đã thực hiện)

Các loại kiểm thử sau đã được tiến hành trên hệ thống đang hoạt động:

- Kiểm thử chức năng (Functional Testing):
    - Mục đích: Xác minh từng chức năng riêng lẻ hoạt động đúng theo yêu cầu.
    - Thực hiện:
        - Đăng ký/Đăng nhập: Đã kiểm tra thành công quy trình đăng ký tài khoản mới, đăng nhập, quên/đặt lại mật khẩu, đổi mật khẩu. Đảm bảo việc băm mật khẩu và quản lý JWT/cookie hoạt động chính xác.
        - Tìm kiếm & Xem sản phẩm: Chức năng tìm kiếm theo từ khóa, lọc theo danh mục, thương hiệu, giá, thông số kỹ thuật (RAM, ROM, dung tích, v.v.) hoạt động hiệu quả. Trang chi tiết sản phẩm hiển thị đầy đủ thông tin, hình ảnh, mô tả, thông số kỹ thuật và đánh giá.
        - Quản lý giỏ hàng: Người dùng có thể thêm sản phẩm vào giỏ hàng, cập nhật số lượng, xóa sản phẩm khỏi giỏ hàng. Các thay đổi được cập nhật tức thì trên giao diện.
        - Đặt hàng & Thanh toán: Toàn bộ luồng đặt hàng từ giỏ hàng, điền thông tin giao nhận, đến chọn phương thức thanh toán đều hoạt động trơn tru.

        - Thanh toán MoMo: Đã kiểm thử thành công luồng tạo yêu cầu thanh toán qua MoMo, hiển thị mã QR và nhận phản hồi từ MoMo. Trạng thái đơn hàng được cập nhật chính xác sau khi thanh toán thành công.
    - Quản lý đơn hàng (phía khách hàng): Khách hàng có thể xem lịch sử đơn hàng, xem chi tiết từng đơn hàng, và theo dõi trạng thái.
    - Đánh giá sản phẩm: Chức năng gửi feedback (đánh giá sao và nội dung) đã hoạt động, đồng thời điểm đánh giá trung bình của sản phẩm được cập nhật chính xác.
    - Quản lý thông tin cá nhân: Người dùng có thể chỉnh sửa thông tin cá nhân và địa chỉ giao hàng.
    - Quản lý thông báo: Chức năng xem thông báo đơn hàng và khuyến mãi, cũng như đánh dấu đã đọc, hoạt động đúng.
    - Yêu cầu đổi trả/bảo hành: Form yêu cầu đổi trả và luồng gửi yêu cầu đã hoạt động.
- Kiểm thử hiệu năng (Performance Testing):

    - Mục đích: Đánh giá tốc độ tải trang và khả năng phản hồi của hệ thống.
    - Thực hiện:
        - Tốc độ tải trang: Các trang chính (trang chủ, danh mục, chi tiết sản phẩm) đạt thời gian tải dưới 3 giây trên kết nối mạng trung bình, đảm bảo trải nghiệm tốt cho người dùng.
        - Thời gian phản hồi API: Các API quan trọng (thêm vào giỏ hàng, đặt hàng, tìm kiếm) phản hồi nhanh chóng (dưới 1 giây trong hầu hết các trường hợp).
        - Người dùng đồng thời: Hệ thống có thể xử lý tốt khoảng 100-150 người dùng đồng thời mà không có dấu hiệu suy giảm hiệu suất đáng kể hoặc lỗi.
- Kiểm thử bảo mật (Security Testing):
    - Mục đích: Xác định và khắc phục các lỗ hổng bảo mật.
    - Thực hiện:
        - HTTPS/SSL: Website đã được triển khai với HTTPS, đảm bảo dữ liệu truyền tải giữa client và server được mã hóa.
        - Mã hóa mật khẩu: Xác nhận mật khẩu được băm trước khi lưu vào cơ sở dữ liệu và được so sánh an toàn khi đăng nhập.
        - Chống SQL Injection: Các truy vấn cơ sở dữ liệu được thực hiện bằng Prepared Statements (hoặc thông qua các phương thức an toàn của thư viện mysql2), ngăn chặn thành công các nỗ lực tấn công SQL Injection.

        - Chống XSS: Dữ liệu đầu vào từ người dùng (như feedback, thông tin cá nhân) được sanitize trước khi hiển thị, giảm thiểu rủi ro tấn công XSS.
- Kiểm thử tương thích (Compatibility Testing):
    - Mục đích: Đảm bảo khả năng hiển thị và hoạt động trên nhiều trình duyệt và thiết bị.
    - Thực hiện:
        - Website hiển thị tốt và hoạt động ổn định trên các trình duyệt phổ biến (Chrome, Firefox, Edge, Safari).
        - Giao diện responsive hoạt động hiệu quả, đảm bảo trải nghiệm mua sắm tốt trên cả máy tính để bàn, máy tính bảng và điện thoại di động.
- Kiểm thử tính dễ sử dụng (Usability Testing):
    - Mục đích: Đánh giá sự trực quan và thân thiện của giao diện người dùng.
    - Thực hiện: Đánh giá chủ quan và quan sát người dùng thật thao tác.
        - Giao diện được đánh giá là đơn giản và trực quan, đáp ứng mục tiêu ban đầu. Các nút bấm, menu, và luồng thao tác đều dễ hiểu.
        - Quá trình tìm kiếm, thêm vào giỏ hàng và thanh toán diễn ra mạch lạc, ít gây nhầm lẫn.

4.2. Đánh giá tổng quan

Tổng kết các kết quả kiểm thử, hệ thống thương mại điện tử của bạn đã đạt được những thành tựu đáng kể:

- Chức năng cốt lõi vững chắc: Tất cả các tính năng cơ bản và thiết yếu của một sàn TMĐT (quản lý tài khoản, sản phẩm, giỏ hàng, đơn hàng, thanh toán MoMo, đánh giá) đều hoạt động ổn định và chính xác theo yêu cầu.
- Hiệu năng tốt: Tốc độ tải trang nhanh và khả năng xử lý lượng người dùng đồng thời ở mức khá, đáp ứng tốt nhu cầu của một website tầm trung.
- Bảo mật được đảm bảo: Các biện pháp bảo mật cơ bản (HTTPS, băm mật khẩu, chống SQL Injection/XSS) đã được triển khai hiệu quả, bảo vệ thông tin người dùng.
- Trải nghiệm người dùng đơn giản và hiệu quả: Giao diện được thiết kế để dễ sử dụng, cho phép người dùng nhanh chóng tìm và mua sản phẩm điện tử mà không gặp nhiều khó khăn.
- Khả năng tích hợp thành công: Việc tích hợp cổng thanh toán MoMo hoạt động mượt mà, mở rộng khả năng giao dịch cho người dùng.

Kết luận: Hệ thống đã hoàn thành tốt các yêu cầu cơ bản và sẵn sàng để được triển khai hoặc tiếp tục phát triển các tính năng nâng cao hơn. Các chức năng cơ bản được đánh giá là ổn cho thấy một nền tảng vững chắc đã được xây dựng.

V. TRIỂN KHAI VÀ HƯỚNG PHÁT TRIỂN

Để nâng cao khả năng cạnh tranh và đáp ứng nhu cầu thị trường ngày càng tăng, nền tảng TMĐT này có thể được phát triển theo các hướng sau:

1. Các tính năng muốn bổ sung:
    - Hệ thống khuyến nghị sản phẩm thông minh (AI-powered Product Recommendation): Tích hợp trí tuệ nhân tạo (AI) để phân tích hành vi mua sắm của người dùng và lịch sử duyệt web, từ đó gợi ý các sản phẩm phù hợp nhất, tăng khả năng chuyển đổi và giá trị đơn hàng.
    - Chatbot hỗ trợ khách hàng: Triển khai chatbot sử dụng AI để trả lời tự động các câu hỏi thường gặp của khách hàng (về sản phẩm, đơn hàng, chính sách), hỗ trợ 24/7 và giảm tải cho đội ngũ chăm sóc khách hàng.
    - Chương trình khách hàng thân thiết (Loyalty Program): Xây dựng hệ thống tích điểm, cấp độ thành viên, và các ưu đãi độc quyền cho khách hàng thân thiết để khuyến khích mua sắm lặp lại và xây dựng cộng đồng.
    - Đa ngôn ngữ: Hỗ trợ nhiều ngôn ngữ (ví dụ: Tiếng Anh) để mở rộng thị trường, đặc biệt nếu có ý định bán hàng quốc tế.
    - Tính năng so sánh sản phẩm: Cho phép người dùng chọn nhiều sản phẩm và so sánh các thông số kỹ thuật, giá cả một cách trực quan, giúp họ đưa ra quyết định mua hàng tốt hơn.
    - Thông báo đẩy (Push Notifications): Triển khai thông báo đẩy qua trình duyệt hoặc ứng dụng di động để gửi thông tin về khuyến mãi, cập nhật đơn hàng trực tiếp đến người dùng.
    - Quản lý kho chi tiết: Phát triển module quản lý kho nâng cao hơn cho Admin, bao gồm quản lý nhập/xuất, cảnh báo tồn kho thấp tự động, và hỗ trợ mã vạch.
    - Tính năng đăng nhập bằng mạng xã hội: Cho phép người dùng đăng nhập nhanh chóng bằng tài khoản Google, Facebook, v.v.
2. Cải tiến hiệu năng và bảo mật:
    - Tối ưu hóa Database: Tinh chỉnh các truy vấn SQL, xem xét tạo thêm index cho các trường thường xuyên được truy vấn, hoặc sử dụng các

công nghệ caching cho database (ví dụ: Redis) để giảm tải và tăng tốc độ đọc dữ liệu.
    - Tối ưu hóa hình ảnh và tài nguyên tĩnh: Sử dụng các công cụ nén hình ảnh tự động, lazy loading cho hình ảnh, và tối ưu hóa CSS/JavaScript để giảm thời gian tải trang.
    - Tăng cường bảo mật: Thực hiện kiểm tra bảo mật định kỳ (penetration testing), áp dụng các chính sách bảo mật nghiêm ngặt hơn (ví dụ: rate limiting cho API, Web Application Firewall - WAF), và liên tục cập nhật các thư viện/framework lên phiên bản mới nhất để vá lỗi bảo mật.
    - Giám sát hệ thống (Monitoring): Triển khai các công cụ giám sát hiệu năng và lỗi theo thời gian thực (ví dụ: Prometheus, Grafana, ELK Stack) để nhanh chóng phát hiện và xử lý sự cố.
3. Kế hoạch mở rộng (phát triển ứng dụng di động):

    - Phát triển ứng dụng di động native/hybrid:
        - Native App (iOS/Android): Xây dựng ứng dụng di động chuyên biệt bằng Swift/Kotlin hoặc React Native/Flutter để cung cấp trải nghiệm tối ưu, tận dụng các tính năng riêng của điện thoại (ví dụ: camera, GPS, push notifications). Điều này sẽ mở rộng đáng kể phạm vi tiếp cận khách hàng và nâng cao trải nghiệm mua sắm trên di động.
        - Ứng dụng di động sẽ tái sử dụng các API Backend hiện có, đảm bảo tính nhất quán dữ liệu và logic nghiệp vụ.

Các hướng phát triển này sẽ giúp hệ thống TMĐT không ngừng hoàn thiện, mang lại giá trị cao hơn cho cả người dùng và doanh nghiệp, và duy trì lợi thế cạnh tranh trên thị trường điện tử đang phát triển.

VI. KẾT LUẬN

Dự án xây dựng nền tảng thương mại điện tử chuyên về điện thoại và đồ điện tử gia đình đã hoàn thành các mục tiêu đề ra, minh chứng cho sự kết hợp hiệu quả giữa công nghệ và nghiệp vụ kinh doanh.

1. Tóm tắt những kết quả chính đạt được

- Xây dựng nền tảng chức năng hoàn chỉnh: Hệ thống đã phát triển đầy đủ các tính năng cốt lõi cho cả người dùng phổ thông và quản trị viên. Đối với khách hàng, các chức năng như đăng ký/đăng nhập, tìm kiếm và duyệt sản phẩm, quản lý giỏ hàng, đặt hàng, thanh toán (đặc biệt là tích hợp MoMo), xem lịch sử đơn hàng và gửi đánh giá đều hoạt động trơn tru. Đối với quản trị viên, các

module quản lý sản phẩm, danh mục, đơn hàng, người dùng và khuyến mãi đều được triển khai hiệu quả.
- Kiến trúc hệ thống rõ ràng và mạnh mẽ: Dự án sử dụng kiến trúc Client-Server với backend Node.js Express theo mô hình MVC, tương tác với cơ sở dữ liệu MySQL.
- Hiệu năng và bảo mật cơ bản được đảm bảo: Hệ thống có tốc độ tải trang nhanh, phản hồi API tốt và khả năng xử lý đồng thời lượng người dùng nhất định. Các biện pháp bảo mật như mã hóa mật khẩu, sử dụng HTTPS, và phòng chống các tấn công phổ biến (SQL Injection, XSS) đã được triển khai hiệu quả.
- Trải nghiệm người dùng được chú trọng: Giao diện được thiết kế đơn giản, trực quan và thân thiện, đặc biệt trên các thiết bị di động, giúp người dùng dễ dàng thao tác và hoàn tất quá trình mua sắm.
- Tích hợp dịch vụ bên ngoài thành công: Việc tích hợp cổng thanh toán MoMo đã chứng minh khả năng kết nối với các dịch vụ bên thứ ba, mở rộng tiện ích cho người dùng.

2. Khẳng định lại ý nghĩa và tiềm năng của dự án

Dự án không chỉ là một bài tập học thuật mà còn có ý nghĩa và tiềm năng thực tế đáng kể:

- Ý nghĩa: Nền tảng này tạo ra một kênh phân phối trực tuyến hiệu quả cho các sản phẩm điện tử tầm trung, tiếp cận đúng đối tượng khách hàng mục tiêu là người đã đi làm (24-45 tuổi). Việc cung cấp một nền tảng uy tín, dễ sử dụng giúp xây dựng niềm tin cho người tiêu dùng khi mua sắm các mặt hàng giá trị cao như đồ điện tử.
- Tiềm năng: Thị trường điện tử gia dụng và thiết bị điện tử trực tuyến đang tăng trưởng mạnh. Với nền tảng vững chắc đã xây dựng, dự án có tiềm năng phát triển thành một website bán lẻ điện tử chuyên biệt, uy tín, có khả năng mở rộng quy mô sản phẩm và dịch vụ. Việc tập trung vào phân khúc tầm trung cũng giúp website có một thị trường ngách ổn định và tiềm năng lớn.

3. Những kinh nghiệm, bài học rút ra trong quá trình thực hiện

Quá trình thực hiện dự án đã mang lại nhiều kinh nghiệm và bài học quý báu:

- Tầm quan trọng của phân tích yêu cầu: Việc xác định rõ ràng yêu cầu nghiệp vụ và chức năng từ đầu là yếu tố then chốt giúp định hình đúng hướng phát triển và tránh lãng phí thời gian.
- Ưu điểm của kiến trúc MVC: Việc áp dụng mô hình MVC (Controller-Model-View) với Node.js Express đã giúp cấu trúc mã nguồn rõ ràng, dễ quản lý, dễ bảo trì và mở rộng các tính năng.

- Hiệu quả của quản lý cơ sở dữ liệu: Thiết kế ERD chi tiết, sử dụng các ràng buộc, trigger và view trong MySQL không chỉ đảm bảo tính toàn vẹn dữ liệu mà còn tối ưu hóa các thao tác truy vấn và cập nhật.
- Thử thách của tích hợp bên thứ ba: Tích hợp cổng thanh toán MoMo, mặc dù đã thành công, cho thấy sự phức tạp trong việc xử lý API, chữ ký bảo mật và các luồng callback/IPN, đòi hỏi sự tỉ mỉ và kiểm thử kỹ lưỡng.
- Vai trò của kiểm thử liên tục: Thực hiện đa dạng các loại kiểm thử (chức năng, hiệu năng, bảo mật) trong suốt quá trình phát triển là cần thiết để sớm phát hiện và khắc phục lỗi, đảm bảo chất lượng phần mềm.
- Giá trị của trải nghiệm người dùng: Một giao diện đơn giản, dễ sử dụng và quy trình mua hàng mạch lạc là yếu tố quyết định sự thành công của một website TMĐT, ngay cả khi không có các tính năng quá phức tạp.
- Học hỏi từ đối thủ/mô hình tương đồng: Nghiên cứu các website lớn giúp học hỏi các điểm mạnh và tránh các điểm yếu, từ đó định hướng phát triển phù hợp.
- Quản lý phiên bản với Git: Sử dụng Git không chỉ giúp theo dõi thay đổi mà còn tạo môi trường cộng tác hiệu quả trong nhóm.

Tổng thể, dự án đã cung cấp một cái nhìn toàn diện về quá trình phát triển một hệ thống TMĐT, từ khâu phân tích, thiết kế đến triển khai và đánh giá, trang bị những kiến thức và kỹ năng thực tiễn quan trọng cho việc phát triển các dự án phần mềm trong tương lai.

VII. TÀI LIỆU THAM KHẢO

Tài liệu tích hợp thanh toán MoMo:
https://developers.momo.vn/v3/docs/payment/api/wallet/onetime