ỦY BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC SÀI GÒN

ĐỒ ÁN

KIỂM THỬ PHẦN MỀM

ĐỀ TÀI: Website buôn bán đồ nội thất

Giảng viên hướng dẫn: Đỗ Như Tài

Nhóm sinh viên thực hiện :

Lê Duy Tín – 3122411211

Nguyễn Hữu Anh Khoa - 3122411098

Đỗ Tấn Lộc - 3122411115

Võ Văn Luân – 3122411120

Lớp: DCT122C3

Thành Phố Hồ Chí Minh – 2025

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU VỀ ĐỀ TÀI

1. Tổng quan về đề tài

GoShop là một ứng dụng web thương mại điện tử được xây dựng với mục tiêu mô phỏng toàn diện hoạt động của một cửa hàng kinh doanh sản phẩm nội thất trong môi trường trực tuyến. Ứng dụng hướng đến việc mang lại trải nghiệm mua sắm thuận tiện, trực quan và dễ sử dụng cho người tiêu dùng, đặc biệt trong bối cảnh thương mại điện tử ngày càng phát triển mạnh mẽ và dần thay đổi thói quen mua sắm truyền thống. Thông qua GoShop, khách hàng có thể dễ dàng tiếp cận nhiều loại sản phẩm nội thất khác nhau, từ đó lựa chọn, đặt mua và thanh toán mà không cần trực tiếp đến cửa hàng, giúp tiết kiệm thời gian và công sức.

2. Phân tích yêu cầu

2.1. Khảo sát hiện trạng

Hiện tại khách hàng có nhu cầu mua sản phẩm nội thất của GoShop đều phải trực tiếp đến cửa hàng, để lựa chọn và mua sản phẩm. Điều này gây ra rất nhiều bất lợi cho khách hàng vì phải tốn một khoảng thời gian để đến cửa hàng nhưng không chắc rằng có thể tìm kiếm được sản phẩm mà mình muốn. Cùng với đó việc quản lý tài chính của cửa hàng hiện tại đang được thực hiện hoàn toàn thủ công, gây bất tiện trong công tác quản lý tài chính. Việc quản lý sản phẩm được thực hiện hoàn toàn bằng sổ sách khiến cho nhưng chi phí không đáng có phát sinh làm ảnh hưởng đến doanh thu và các dự kiến phát triển trong tương lai của cửa hàng.

2.2. Bối cảnh nghiệp vụ

GoShop Website là một website thương mại điện tử được xây dựng nhằm đáp ứng nhu cầu nhu cầu mua sắm nội thất trực tuyến. Nên GoShop được xây dựng với các kịch bản kinh doanh cơ bản về: Danh mục sản phẩm, giỏ hàng, quy trình thanh toán, hàng tồn kho, quản lý truy cập, kiểm tra thanh toán, xem đơn hàng , quản lý tài khoản và thống kê chi tiết đơn hàng. Với 2 người dùng chính là người mua và người quản trị.

Vai trò chính của người mua hàng là thực hiện các hành động mua sắm trực tuyến. Người mua có thể là khách hàng chưa đăng ký

hoặc đã đăng ký tài khoản. Trong đó các hành động chính mà người mua có thể thực hiện là: Duyệt và Tìm Kiếm sản phẩm, Xem chi tiết sản phẩm, quản lý giỏ hàng, thực hiện thanh toán, quản lí thông tin tài khoản, kiểm tra thanh toán.

Vai trò chính của người quản trị là quản lý và vận hành toàn bộ hệ thống GoShop để đảm bảo hoạt động kình doanh trôi chảy và hiệu quả. Trong đó các thao tác chính mà người quản trị có thể thực hiện là: quản lý sản phẩm, quản lý danh mục sản phẩm, quản lý đơn hàng, quản lý tài khoản, thống kê doanh thu.

Đối với Danh mục sản phẩm, người mua có thể duyệt danh sách sản phẩm với các chức năng lọc theo danh mục sản phẩm, sắp xếp theo giá, tìm kiếm sản phẩm theo tên. Người dùng có thể xem chi tiết sản phẩm trên danh sách bằng cách nhấp vào sản phẩm trên danh sách, khi này người dùng sẽ được nhìn thấy hiện thống hiển thị thông tin chi tiết bao gồm: chiều cao, chiều rộng, brand, vật liệu, …. Người quản trị có thể quản lí sân phẩm, quản lí danh mục sản phẩm và cập nhật số lượng tồn kho.

Đối với giỏ hàng, người mua có thể thêm một hoặc nhiều sản phẩm bất kì nào vào giỏ hàng thông qua nút Add to Cart. Ngoài ra, người mua cũng có thể thêm sản phẩm vào giỏ hàng thông qua trang chi tiết sản phẩm. Người mua cũng có thể xem lại các sản phẩm đã chọn, cập nhật số lượng hoặc xóa sản phẩm ra khỏi giỏ hàng.Hệ thống tự động tính tổng tiền, phí vận chuyển, khuyến mãi nếu có và cập nhật lại số lượng tồn kho của sản phẩm đó có trong kho.

Đối với quy trình thanh toán, sau khi người mua đã ưng ý với những gì mình đặt hàng thì có thể người mua khi xác nhận đặt hàng, hệ thống sẽ kiểm tra lại thông tin sản phẩm có trong giỏ hàng, địa chỉ giao hàng và phương thức thanh toán, xử lý thanh toán thông qua Stripe. Sau đó hệ thống cập nhật lại trạng thái thanh toán và gửi thông báo xác nhận thông qua email.

Đối với quản lý truy cập, người dùng có thể đăng ký, đăng nhập bằng email/password hoặc sử dụng tài khoản Google. Người quản trị có quyền truy cập vào trang quản trị để quản lý toàn bộ hệ thống. Khi người dùng website tiến hành đăng nhập vào tài khoản của mình đã tạo từ trước đó, hệ thống có nhiệm vụ phải kiểm tra vai trò của tài khoản để có thể hiện thị đúng trang mà người dùng cần.

Đối với quản lý tài khoản, người mua, người quản trị có thể quản lý thông tin cá nhân cho tài khoản của mình. Người quản trị có khả năng quản lý tài khoản người mua để đảm bảo an toàn hệ thống.

Đối với quản lý đơn hàng, người mua có thể xem lại các lịch sử đơn hàng mà mình đã mua. Người quản trị có thể xem được tất cả đơn hàng có trong hệ thống

Đối với thống kê đơn hàng, người quản trị có thể thực hiện các thao tác thống kê trên đơn hàng để có thể đưa ra được đánh giá về tình hình kinh doanh.

2.3. Yêu cầu nghiệp vụ

2.3.1. Yêu cầu chức năng

Nghiệp vụ	Mã Yêu cầu	Yêu cầu Chức năng	Mức Thiết Yếu
Quản Lý Truy Cập	BR1.1	Cho phép **Đăng ký** tài khoản mới bằng email/password hoặc tài khoản Google.	Cao
	BR1.2	Cho phép **Đăng nhập** bằng email/password hoặc tài khoản Google.	Cao
	BR1.3	**Kiểm tra và xác định Vai trò** của người dùng để chuyển hướng đến trang (Mua hàng/Quản trị) phù hợp.	Cao

Danh mục và Sản Phẩm	BR2.1	Cung cấp chức năng **Duyệt, Lọc (theo danh mục) và Sắp xếp (theo giá)** sản phẩm.	Cao
	BR2.2	Cung cấp chức năng **Tìm kiếm sản phẩm theo tên**.	Cao
	BR2.3	Cho phép **Xem chi tiết sản phẩm** (bao gồm thông tin, giá, mô tả).	Cao
	BR2.4	Cho phép **Quản lý (thêm/sửa/xóa) Danh mục** sản phẩm.	Cao
	BR2.5	Cho phép **Quản lý (thêm/sửa/xóa) Sản phẩm** và **cập nhật số lượng Tồn kho**.	Cao
Giỏ hàng	BR3.1	Cho phép **Thêm sản phẩm vào Giỏ hàng** từ bất kỳ trang nào.	Cao
	BR3.2	Cho phép **Xem, Cập nhật số lượng, hoặc Xóa** sản phẩm trong Giỏ hàng.	Cao

	BR3.3	**Tự động tính toán** Tổng tiền, Phí vận chuyển, và Khuyến mãi (nếu có) trong Giỏ hàng.	Cao
Thanh toán	BR4.1	Thu thập và xác nhận **Thông tin giao hàng** (địa chỉ).	Cao
	BR4.2	Tích hợp và xử lý thanh toán thông qua **Stripe**.	Cao
	BR4.3	**Cập nhật trạng thái Thanh toán** sau khi giao dịch hoàn tất.	Cao
	BR4.4	**Cập nhật (giảm) số lượng Tồn kho thực tế** ngay sau khi đơn hàng được xác nhận thanh toán.	Cao
	BR4.5	**Gửi Email xác nhận** đơn hàng cho người mua.	Cao
Đơn Hàng và Tài Khoản	BR5.1	Cho phép **Người mua xem lại các Đơn hàng đã mua** và trạng thái thanh toán.	Cao

	BR5.2	Cho phép **Người quản trị theo dõi tất cả Đơn hàng** trong hệ thống.	Trung bình
	BR5.3	Cho phép Người mua và Người quản trị **Quản lý và cập nhật thông tin cá nhân**.	Trung bình
Thông kê	BR6.1	Cung cấp chức năng **Thống kê chi tiết đơn hàng** (doanh thu, số lượng, v.v.).	Trung bình

2.3.2. Yêu cầu phi chức năng

Thuộc tính	Mã	Yêu cầu Phi Chức năng	Mức thiết yếu
Giao diện(UI/UX)	NBR1.1	Giao diện phải đơn giản, dễ sử dụng	Cao
	NBR1.2	Với mỗi đối tượng thì sẽ có một giao diện trình bày phù hợp và trực quan.	Cao

Hiệu Năng	NBR2.1	Trang hiển thị sản phẩm phải tải trong thời gian ngắn, đảm bảo trải nghiệm mượt mà cho người dùng	Cao
	NBR2.2	Hệ thống phải đảm bảo được thời gian phản hồi khi có nhiều lượt truy cập	Trung bình
Bảo mật	NBR3.1	Phân quyền rõ vai trò tài khoản người mua và người quản trị	Cao
	NBR3.2	Thông tin người dùng phải được bảo mật dưới cơ sở dữ liệu	Cao
Khả năng mở rộng	NBR4.1	Hệ thống cho phép mở rộng thêm chức năng và quy mô người dùng trong tương lai.	Trung bình

3. Mục tiêu đề tài

3.1. Mục tiêu phát triển

Mục tiêu phát triển của dự án GoShop là xây dựng một ứng dụng web thương mại điện tử chuyên kinh doanh sản phẩm nội thất, đáp ứng các nhu cầu cơ bản của người dùng trong việc mua sắm trực tuyến cũng như hỗ trợ hiệu quả cho công tác quản lý của cửa hàng. Hệ thống hướng đến sự ổn định, dễ sử dụng và khả năng mở rộng trong tương lai. Bên cạnh đó, dự án còn nhằm giúp nhóm sinh viên áp dụng kiến thức đã học vào thực tế, rèn luyện kỹ năng làm việc nhóm, phân tích yêu cầu, thiết kế hệ thống và triển khai phần mềm theo quy trình chuyên nghiệp.

3.2. Kiểm thử phần mềm

Trong đề tài này, kiểm thử phần mềm được xác định là trọng tâm chính nhằm đảm bảo chất lượng và độ ổn định của hệ thống GoShop. Các hoạt động kiểm thử được triển khai ở nhiều mức độ khác nhau để đánh giá toàn diện hệ thống. Cụ thể, các cách kiểm thử được áp dụng trong dự án bao gồm:

Unit Test: Kiểm tra các logic xử lý nhỏ trong backend, tập trung vào controller và service.

Integration Test: Kiểm tra sự tương tác giữa các module trong hệ thống, đặc biệt là giữa API và cơ sở dữ liệu.

System Test: Kiểm tra toàn bộ chức năng của hệ thống từ đầu đến cuối.

User Acceptance Test (UAT): Kiểm thử chấp nhận dựa trên các quy trình nghiệp vụ và các tiêu chí chấp nhận đã xác định.

Automation Test: Sử dụng Selenium để thực hiện kiểm thử giao diện người dùng một cách tự động.

Performance Test: Sử dụng JMeter để kiểm thử hiệu năng đối với các API quan trọng của hệ thống.

Việc áp dụng đa dạng các cách kiểm thử giúp dự án GoShop đánh giá chất lượng phần mềm một cách toàn diện và phù hợp với mục tiêu của đề tài.

3.3. Quy trình kiểm thử Agile, CI/CD

Trong dự án GoShop, quy trình kiểm thử được tổ chức và triển khai song song với quá trình phát triển phần mềm nhằm đảm bảo chất lượng hệ thống một cách liên tục. Nhóm lựa chọn kết hợp mô hình Agile với quy trình CI/CD để kiểm thử không chỉ diễn ra ở giai đoạn cuối mà được lồng ghép xuyên suốt vòng đời phát triển phần mềm. Cách tiếp cận này giúp phát hiện lỗi sớm, giảm chi phí sửa lỗi và đảm bảo hệ thống luôn ở trạng thái ổn định trong suốt quá trình thực hiện đề tài.

3.3.1. Quy trình Agile

Dự án GoShop được phát triển theo phương pháp Agile với các vòng lặp ngắn (Sprint), trong đó mỗi Sprint tập trung vào việc hoàn thiện một nhóm chức năng cụ thể. Ngay từ giai đoạn phân tích yêu cầu, các kịch bản kiểm thử đã được xác định song song với việc thiết kế và phát triển chức năng. Điều này giúp nhóm định hướng rõ ràng các tiêu chí kiểm thử trước khi tiến hành cài đặt.

Trong mỗi Sprint, sau khi hoàn thành việc phát triển chức năng, nhóm tiến hành kiểm thử để xác minh tính đúng đắn và mức độ đáp ứng yêu cầu của hệ thống. Các lỗi được ghi nhận, phân loại và ưu tiên xử lý ngay trong Sprint hiện tại hoặc Sprint tiếp theo. Việc kiểm thử lặp lại theo từng vòng phát triển giúp hệ thống được cải thiện liên tục, hạn chế tình trạng tích tụ lỗi và nâng cao chất lượng tổng thể của sản phẩm.

Ngoài ra, Agile còn tạo điều kiện thuận lợi cho việc phản hồi và điều chỉnh kịp thời. Khi có sự thay đổi về yêu cầu hoặc phát sinh vấn đề mới, nhóm có thể nhanh chóng cập nhật các kịch bản kiểm thử tương ứng, đảm bảo hệ thống luôn phù hợp với mục tiêu của đề tài và nhu cầu thực tế.

3.3.2. CI/CD

Bên cạnh Agile, dự án GoShop áp dụng quy trình CI/CD nhằm tự động hóa các bước xây dựng, kiểm thử và triển khai hệ thống. Mỗi khi có sự thay đổi trong mã nguồn, hệ thống CI sẽ tự động thực hiện quá trình tích hợp, kiểm tra và chạy các ca kiểm thử đã được xây dựng trước đó. Điều này giúp phát hiện sớm các lỗi phát sinh do thay đổi mã nguồn, đồng thời đảm bảo tính nhất quán của hệ thống.

Qui trình Continuous Integration (CI) trong GoShop tập trung vào việc tích hợp và kiểm thử mã nguồn một cách liên tục. Mỗi khi có thay đổi hoặc cập nhật mã nguồn, hệ thống CI sẽ tự động thực hiện các bước như xây dựng ứng dụng, kiểm tra lỗi cú pháp và chạy các kịch bản kiểm thử đã được thiết kế sẵn. Nhờ đó, các lỗi phát sinh trong quá trình phát triển được phát hiện sớm, đặc biệt là các lỗi ảnh hưởng đến chức năng chính như quản lý sản phẩm, xử lý đơn hàng và xác thực người dùng. CI giúp đảm bảo rằng mỗi thay đổi đều không làm ảnh hưởng tiêu cực đến các chức năng đã ổn định trước đó.

Qui trình Continuous Delivery/Deployment (CD) trong GoShop đóng vai trò đảm bảo khả năng triển khai hệ thống một cách an toàn và nhất quán. Sau khi các bước kiểm thử trong CI được thực hiện thành công, quy trình CD cho phép triển khai phiên bản mới của hệ thống sang môi trường tiếp theo (ví dụ: môi trường kiểm thử hoặc mô phỏng triển khai thực tế). CD giúp nhóm đánh giá hệ thống trong điều kiện gần với môi trường sử dụng thật, đồng thời giảm thiểu rủi ro khi cập nhật hoặc mở rộng chức năng. Việc triển khai chỉ được thực hiện khi hệ thống đáp ứng đầy đủ các tiêu chí kiểm thử đã đề ra, từ đó đảm bảo độ ổn định và độ tin cậy của GoShop.

3.4. Công nghệ sử dụng

Công nghệ	Loại	Mô tả
TypeScript	Ngôn ngữ lập trình	Là một **tập hợp siêu hợp lý (superset) của JavaScript** có bổ sung thêm các tính năng về kiểu dữ liệu tĩnh. Nó giúp phát triển các ứng dụng lớn, bảo trì dễ hơn và giảm thiểu lỗi trong quá trình phát triển.
ReactJS	Thư viện Frontend	Là một **thư viện JavaScript** do Facebook phát triển, dùng để xây dựng **giao diện**

		người dùng (UI), đặc biệt hiệu quả trong việc tạo ra các ứng dụng một trang (Single Page Applications - SPA) với khả năng tái sử dụng component cao.
NodeJS	Môi trường Runtime Backend	Là một **môi trường runtime JavaScript mã nguồn mở**, đa nền tảng, cho phép chạy mã JavaScript ở phía **máy chủ (server-side)**. NodeJS nổi tiếng với mô hình I/O không chặn (non-blocking I/O) và hiệu suất cao.
ExpressJS	Framework Backend	Là một **framework web tối giản và linh hoạt** dành cho NodeJS, cung cấp các tính năng cần thiết để xây dựng các ứng dụng web và API mạnh mẽ.
Stripe	Công nghệ bên thứ 3	Là một **nền tảng xử lý thanh toán trực tuyến** toàn diện, cho phép các doanh nghiệp chấp nhận thanh toán qua internet, với các API và công cụ dễ dàng tích hợp.
MySQL	Hệ quản trị CSDL (SQL)	Là một **hệ quản trị cơ sở dữ liệu quan hệ (RDBMS)** mã nguồn mở, phổ biến, sử

		dụng ngôn ngữ SQL để quản lý dữ liệu.
Firebase	Nền tảng Backend-as-a-Service	Là một **nền tảng phát triển ứng dụng di động và web** của Google, cung cấp nhiều dịch vụ như cơ sở dữ liệu thời gian thực (Realtime Database), xác thực (Authentication), lưu trữ đám mây (Cloud Storage),...
Prisma	ORM	Là một **Object-Relational Mapper (ORM)** thế hệ mới, giúp nhà phát triển cơ sở dữ liệu tương tác với cơ sở dữ liệu thông qua TypeScript/JavaScript, với tính năng an toàn kiểu dữ liệu (type-safe) và hiệu quả.

4. Kế hoạch triển khai

4.1. Kế hoạch phát triển

Sprint	Thời gian	Mục tiêu chính	Công việc thực hiện	Hoạt động kiểm thử
Sprint 1	Tuần 1–2	Phân tích yêu cầu & thiết kế tổng quan	Xác định yêu cầu nghiệp vụ, phân tích Busines	Xây dựng test plan, xác định test case sơ bộ

			s Context, thiết kế kiến trúc hệ thống	
Sprint 2	Tuần 3–4	Xây dựng chức năng nền tảng	Cài đặt đăng ký, đăng nhập, phân quyền người dùng	Unit Test cho controller, service
Sprint 3	Tuần 5–6	Quản lý sản phẩm	Thêm, sửa, xóa, hiển thị sản phẩm nội thất	Unit Test, Integration Test (API – Database)
Sprint 4	Tuần 7–8	Giỏ hàng & đặt hàng	Xử lý giỏ hàng, tạo đơn hàng	Integration Test, System Test
Sprint 5	Tuần 9–10	Thanh toán & xử lý đơn	Tích hợp thanh toán, cập nhật trạng thái đơn hàng	System Test, UAT
Sprint 6	Tuần 11–12	Giao diện & trải nghiệm người dùng	Hoàn thiện giao diện, tối ưu luồng sử dụng	Automation Test (Selenium)
Sprint 7	Tuần 13–	Tối ưu &	Tối ưu API, xử	Performanc e Test

	14	kiểm thử hiệu năng	lý tải	(JMeter)
Sprint 8	Tuần 15	Hoàn thiện & tổng kết	Sửa lỗi, hoàn thiện tài liệu, chuẩn bị báo cáo	Regression Test, kiểm thử tổng thể

4.2. Kế hoạch kiểm thử

4.2.1. Mục tiêu kiểm thử

Mục tiêu chính của hoạt động kiểm thử trong dự án GoShop là đảm bảo chất lượng phần mềm thông qua việc kiểm tra tính đúng đắn của nghiệp vụ, hiệu năng và độ tin cậy của hệ thống. Cụ thể, các mục tiêu kiểm thử bao gồm:

Xác minh chức năng: Đảm bảo 100% các yêu cầu chức năng của Khách hàng và Quản trị viên được thực hiện đúng theo đặc tả và đáp ứng đầy đủ các tiêu chí chấp nhận đã đề ra ở phần yêu cầu nghiệp vụ.

Đảm bảo chất lượng kỹ thuật: Xác nhận các thành phần trong kiến trúc hệ thống tương tác ổn định, luồng xử lý nghiệp vụ hoạt động chính xác và nhất quán trong quá trình tích hợp.

Đánh giá hiệu năng và độ tin cậy: Kiểm tra khả năng xử lý, thời gian phản hồi và mức độ ổn định của các chức năng quan trọng, đảm bảo hệ thống đáp ứng yêu cầu vận hành trong điều kiện thực tế.

Tích hợp sớm và kiểm soát lỗi: Phát hiện và khắc phục lỗi ở giai đoạn sớm thông qua việc tự động hóa kiểm thử và tích hợp liên tục (CI), hạn chế rủi ro phát sinh khi mở rộng hoặc triển khai hệ thống.

4.2.2. Các loại hình kiểm thử dự kiến

Dựa trên mục tiêu và phạm vi của đề tài, dự án GoShop dự kiến áp dụng các loại hình kiểm thử sau nhằm đánh giá chất lượng phần mềm một cách toàn diện:

Unit Test: Kiểm thử các đơn vị chức năng nhỏ trong hệ thống, tập trung vào logic xử lý của các thành phần backend như controller và service, nhằm đảm bảo từng thành phần hoạt động đúng theo thiết kế.

Integration Test: Kiểm tra sự tương tác và trao đổi dữ liệu giữa các module trong hệ thống, đặc biệt là giữa các API và cơ sở dữ liệu, qua đó phát hiện các lỗi phát sinh trong quá trình tích hợp.

System Test: Kiểm thử toàn bộ hệ thống GoShop từ đầu đến cuối, bao gồm các chức năng dành cho người dùng và người quản trị, nhằm xác minh hệ thống đáp ứng đầy đủ các yêu cầu nghiệp vụ.

User Acceptance Test (UAT): Thực hiện kiểm thử chấp nhận dựa trên các quy trình nghiệp vụ thực tế và các tiêu chí chấp nhận đã xác định, nhằm đánh giá mức độ phù hợp của hệ thống đối với nhu cầu và kỳ vọng của người dùng cuối.

Automation Test: Áp dụng kiểm thử tự động cho giao diện người dùng bằng công cụ Selenium, giúp kiểm tra các luồng thao tác chính, giảm thời gian kiểm thử và hỗ trợ kiểm thử lặp lại trong quá trình phát triển.

Performance Test: Thực hiện kiểm thử hiệu năng bằng công cụ JMeter đối với các API quan trọng, nhằm đánh giá thời gian phản hồi, khả năng chịu tải và độ ổn định của hệ thống trong các điều kiện khác nhau.

4.2.3. Phương pháp thực hiện

Trong dự án GoShop, kiểm thử phần mềm được triển khai như một hoạt động trung tâm trong mỗi Sprint và được tự động hóa thông qua quy trình CI/CD nhằm kiểm soát chất lượng hệ thống một cách liên tục. Ở giai đoạn CI, mỗi lần thay đổi mã nguồn đều kích hoạt pipeline kiểm thử, trong đó các bài kiểm thử đơn vị và kiểm thử tích hợp được thực thi tự động để xác minh tính đúng đắn của logic nghiệp vụ và sự tương tác giữa các thành phần hệ thống.

Khi Sprint kết thúc, các chức năng đã hoàn thành tiếp tục được đánh giá thông qua kiểm thử hệ thống và các kịch bản kiểm thử tự động nhằm đảm bảo toàn bộ luồng nghiệp vụ hoạt động ổn định và không phát sinh lỗi hồi quy. Các kết quả kiểm thử trong giai đoạn này là cơ sở quan trọng để quyết định việc chuyển sang bước tiếp theo trong pipeline.

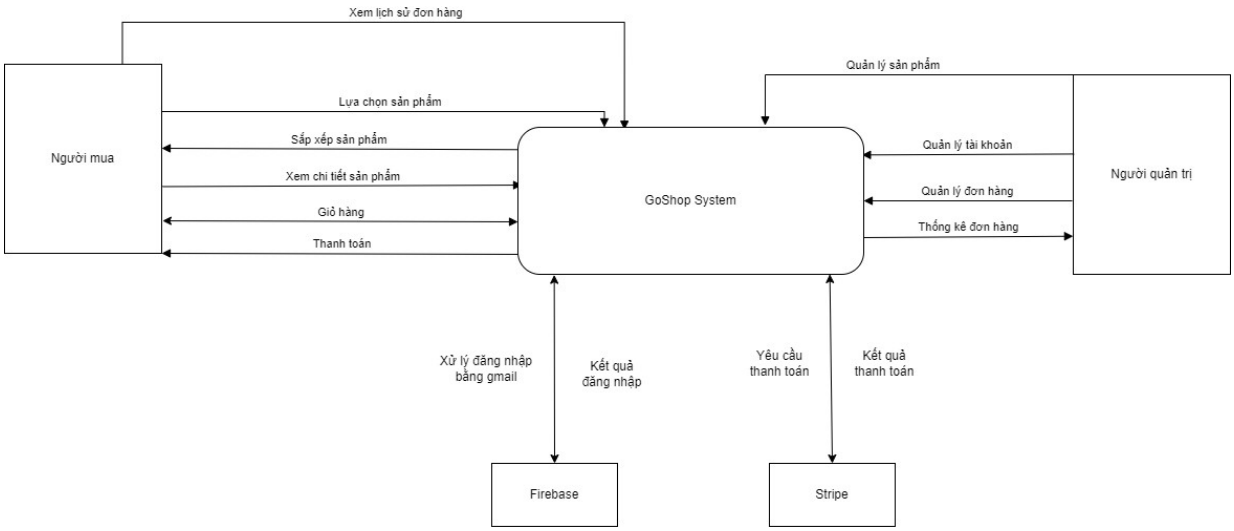
Trong giai đoạn CD, việc triển khai chỉ được cho phép khi toàn bộ các bài kiểm thử đều đạt yêu cầu. Phiên bản được triển khai lên môi trường Railway phục vụ cho kiểm thử chấp nhận người dùng và đánh giá Sprint, đồng thời hỗ trợ thực hiện các bài kiểm thử hiệu năng định kỳ. Cách tiếp cận này giúp hoạt động kiểm thử trong GoShop mang tính chủ động, liên tục và đóng vai trò then chốt trong việc đảm bảo chất lượng và độ tin cậy của hệ thống.

CHƯƠNG 2: THIẾT KẾ HỆ THỐNG VÀ THIẾT KẾ KIẾN TRÚC

1. Thiết kế hệ thống

1.1. Phân tích chức năng

1.1.1. Sơ đồ ngữ cảnh



Hệ thống GoShop gồm hai đối tượng chính là người mua và người quản trị.

Người mua có thể Đăng Ký/Đăng Nhập vào GoShop để thực hiện tìm kiếm sản phẩm, đặt hàng, thanh toán, và xem lịch sử đơn hàng.

Người quản trị có thể Đăng Ký/Đăng Nhập vào GoShop nhưng khác với giao diện của Người mua, người quản trị có thể thực hiện quản lý tài khoản, quản lý đơn hàng và thống kê đơn hàng.

GoShop còn sử dụng các dịch vụ của bên thứ 3 như:

Firebase để thực hiện việc đăng nhập bằng gmail cho khách hàng có nhu cầu.

Stripe dùng để thực hiện chức năng thanh toán.

1.1.2. Quy trình nghiệp vụ

1.1.2.1. Quy trình khám phá sản phẩm

Quy trình khám phá sản phẩm nội thất có nhu cầu tìm kiếm sản phẩm. Khách hàng, tiến hành chọn các mặt hàng nội thất mong muốn được bày trí sẵn ở trang danh mục sản phẩm của cửa hàng. Nếu khách hàng có nhu cầu tìm kiếm sản phẩm thì có thể sản dụng thanh tìm kiếm được bố trí ở phía trên cùng của trang web, và thực hiện tìm kiếm bằng cách gõ tên sản phẩm mình muốn tìm kiếm vào đó, khách hàng có thể thực hiện tìm kiếm khi đang ở trang danh mục sản phẩm . Kết quả tìm kiếm là một danh sách các sản phẩm thỏa mãn yêu cầu tìm kiếmm của người mua sẽ được hiển thị trong một trang duy nhất với các sản phẩm đang ở trạng thái bán có trong cơ sở dữ liệu hiện tại. Nếu khách hàng muốn thực hiện lọc sản phẩm theo danh mục thì việc này chỉ có thể thực hiện được ở trang danh mục sản phẩm, kết quả khi này vẫn là một danh sách các sản phẩm trùng khớp với danh mục mà người mua mong muốn. Nếu không có kết quả nào thỏa, thì cả 2 phương thức tìm kiếm thì trang tìm kiếm sẽ trả về một dòng thông báo không có sản phẩm phù hợp. Người dùng có thể không cần phải đăng nhập để thực hiện 2 hành động này. Người mua cũng có thể sắp xếp sản phẩm bằng một nút bấm được bố trí sẵn ở trang danh mục sản phẩm, hoặc người mua cũng có thể bấm vào một sản phẩm để có thể xem chi tiết về sản phẩm đó.

1.1.2.2. Quy trình mua sản phẩm

Quy trình Mua Sản phẩm Nội thất bắt đầu khi người mua có nhu cầu muốn hàng với sản phẩm. Người mua, sau khi đăng nhập và truy cập, tiến hành chọn các mặt hàng nội thất mong muốn (ví dụ: ghế sofa, tủ kệ) và thêm các tùy chọn liên quan (như chất liệu, màu sắc) vào Giỏ hàng, mỗi lần khách hàng thêm sản phẩm vào giỏ hàng thì chỉ được thêm một món hàng. Ngay lập tức, một công cụ tính toán sẽ xác định và hiển thị Tổng chi phí của Giỏ hàng (Tổng giá trị sản phẩm). Đồng thời, dựa trên địa chỉ giao hàng, hệ thống sẽ ước tính và hiển thị Chi phí Vận chuyển dựa trên chính sách vận tải cho hàng hóa cồng kềnh. Nếu người mua muốn điều chỉnh số

lượng thì khi đã thêm sản phẩm vào giỏ hàng sẽ có một chỗ để 2 nút bấm có khả năng tăng giảm số lượng sản phẩm, khi người dùng thực hiện thay đổi thì giỏ hàng sẽ ngay lập tức được cập nhật.

Khi khách hàng chuyển sang khâu Tạo Đơn Hàng, thông tin chi tiết về đơn hàng được Khách hàng xác nhận lần cuối (bao gồm địa chỉ giao hàng và tổng chi phí cuối cùng). Bước tiếp theo là Thanh Toán, nơi khách hàng lựa chọn phương thức thanh toán phù hợp (trực tuyến hoặc thanh toán khi nhận hàng/chuyển khoản). Sau khi thanh toán thì số lượng sản phẩm của đơn hàng này sẽ trừ vào số lượng sản phẩm đang có trong cơ sở dữ liệu.

1.1.2.3. Quy trình thanh toán

Quy trình này được kích hoạt khi khách hàng xác nhận các thông tin trong đơn hàng nội thất và lựa chọn phương thức thanh toán trực tuyến.

Khi khách hàng chọn thanh toán, hệ thống sẽ Chuyển hướng đến Cổng thanh toán Stripe. Khách hàng nhập thông tin thẻ (hoặc chọn ví điện tử đã liên kết) trên giao diện bảo mật của Stripe. Stripe thực hiện bước Xử lý Giao dịch, bao gồm việc mã hóa dữ liệu, gửi yêu cầu ủy quyền đến ngân hàng phát hành thẻ, và chờ phản hồi về tính hợp lệ của giao dịch.

Nếu Giao dịch thành công: Stripe gửi lại một phản hồi thành công (Webhook/API response) cho hệ thống bán hàng. Hệ thống ngay lập tức Cập nhật trạng thái Đơn hàng thành "Đã Thanh Toán", tạo hóa đơn điện tử. Khách hàng nhận được xác nhận thanh toán thành công.

Nếu Giao dịch không thành công: Stripe gửi mã lỗi về hệ thống. Khách hàng nhận được thông báo rõ ràng về việc giao dịch thất bại (ví dụ: thẻ bị từ chối, hết hạn) và được yêu cầu thử lại hoặc chọn phương thức thanh toán khác.

1.1.2.4. Quy trình quản lý hệ thống

Quy trình quản lý hệ thống bắt đầu khi người quản trị có thể thực hiện quản lý đơn hàng bằng cách cập nhật tình trạng đơn hàng, xem lịch sử đơn hàng và thực hiện thống kê đơn hàng. Người quản trị còn có thể quản lý sản phẩm hiện có trên hệ thống của cửa hàng: thêm, xóa, cập nhật thông tin sản phẩm. Người quản trị cũng có thể quản lý thông tin tài khoản của khách hàng.

1.1.2.4. Quy trình phân quyền

Quy trình đăng nhập và phân quyền bắt đầu khi người dùng truy cập vào hệ thống và nhập thông tin đăng nhập, bao gồm tên đăng nhập và mật khẩu. Hệ thống tiếp nhận dữ liệu, tiến hành kiểm tra tính hợp lệ của thông tin đầu vào, sau đó đối chiếu với dữ liệu tài khoản đã được lưu trữ. Nếu thông tin đăng nhập hợp lệ, hệ thống xác thực người dùng và tạo phiên làm việc tương ứng.

Sau khi xác thực thành công, hệ thống tiến hành kiểm tra vai trò (role) và quyền hạn của tài khoản, chẳng hạn như quản trị viên, nhân viên hoặc người dùng thông thường. Dựa trên quyền đã được phân quyền, hệ thống cho phép người dùng truy cập vào các chức năng và tài nguyên phù hợp, đồng thời hạn chế các thao tác không được phép. Trường hợp thông tin đăng nhập không hợp lệ hoặc tài khoản bị khóa, hệ thống sẽ từ chối truy cập và hiển thị thông báo lỗi cho người dùng nhằm đảm bảo an toàn và bảo mật cho hệ thống.

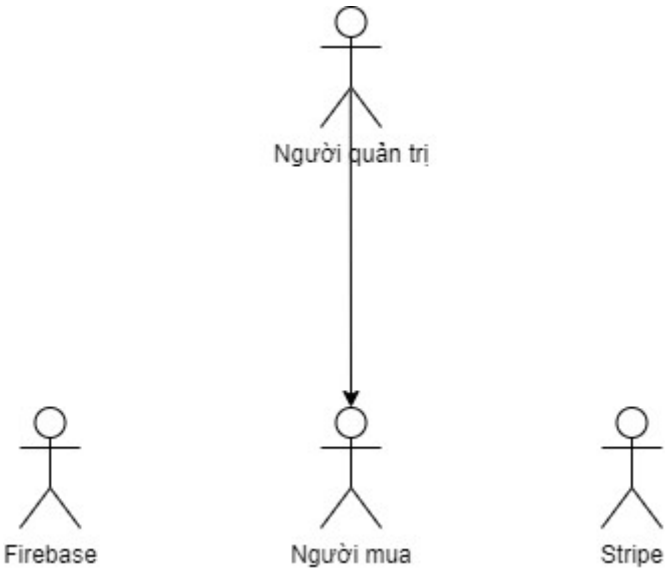
1.1.3. Các tác nhân tham gia vào hệ thống GoShop

Hệ thống GoShop hiện có hai tác nhân chính sử dụng hệ thống GoShop là

Người mua sử dụng hệ thống GoShop để thực hiện mua hàng.

Người quản trị là người có quyền thực hiện các hành động quản trị trong hệ thống.

Ngoài ra GoShop còn có 2 tác nhân từ dịch vụ thứ 3 là Firebase nhằm mục tiêu thực hiện nhu cầu đăng nhập bằng tài khoản gmail và Stripe là cổng thanh toán tích hợp để thực hiện các giao dịch với khách hàng.



1.1.4. Use Case

1.1.4.1. Use Case Summary

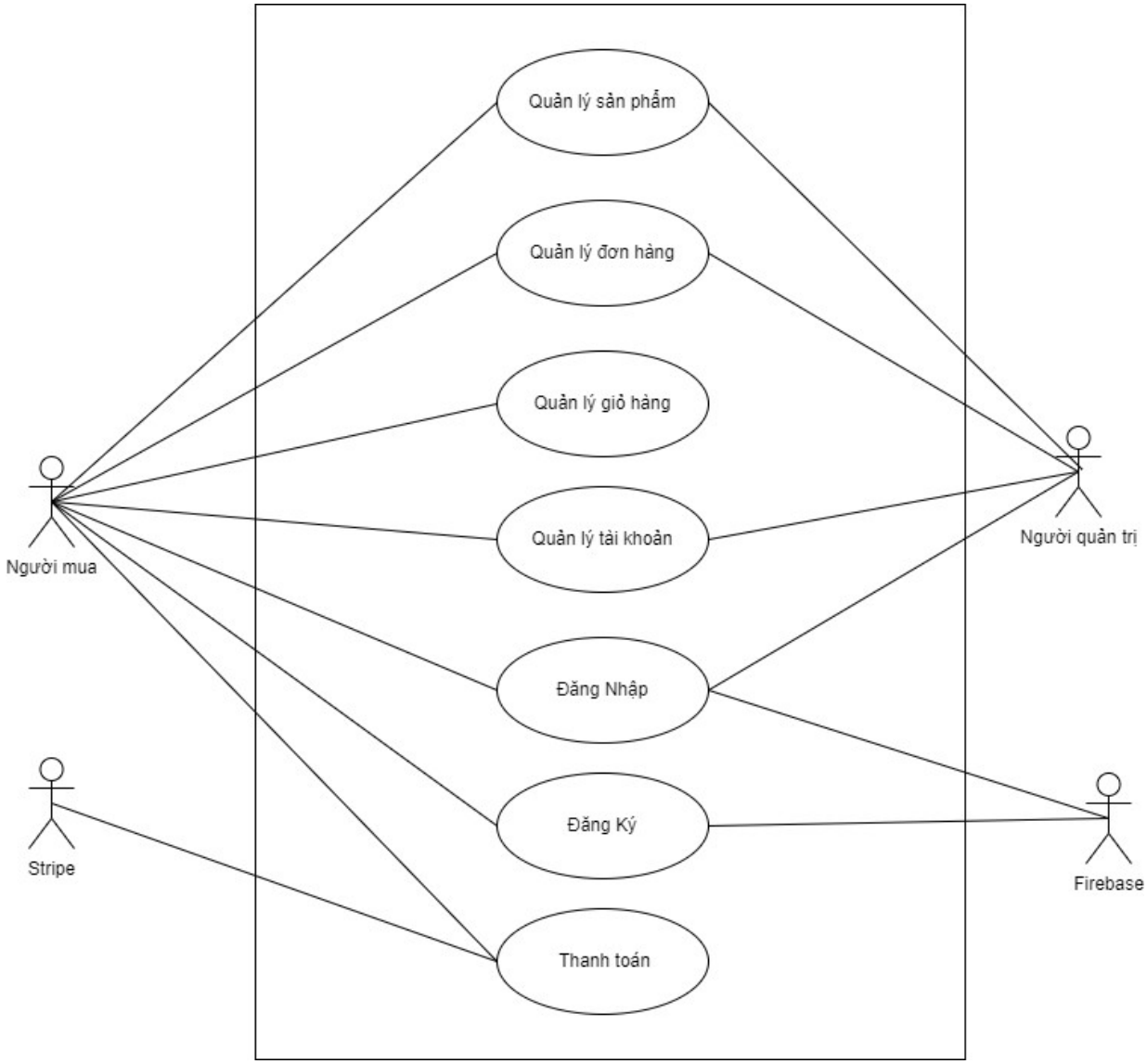
Người mua có thể thực hiện các chức năng sau: Quản lý sản phẩm, quản lý giỏ hàng, quản lý thông tin cá nhân của tài khoản, quản lý đơn hàng và quản lý truy cập.

Người quản trị có thể thực hiện các chức năng: Quản lý sản phẩm, quản lý đơn hàng, quản lý tài khoản và quản lý truy cập.

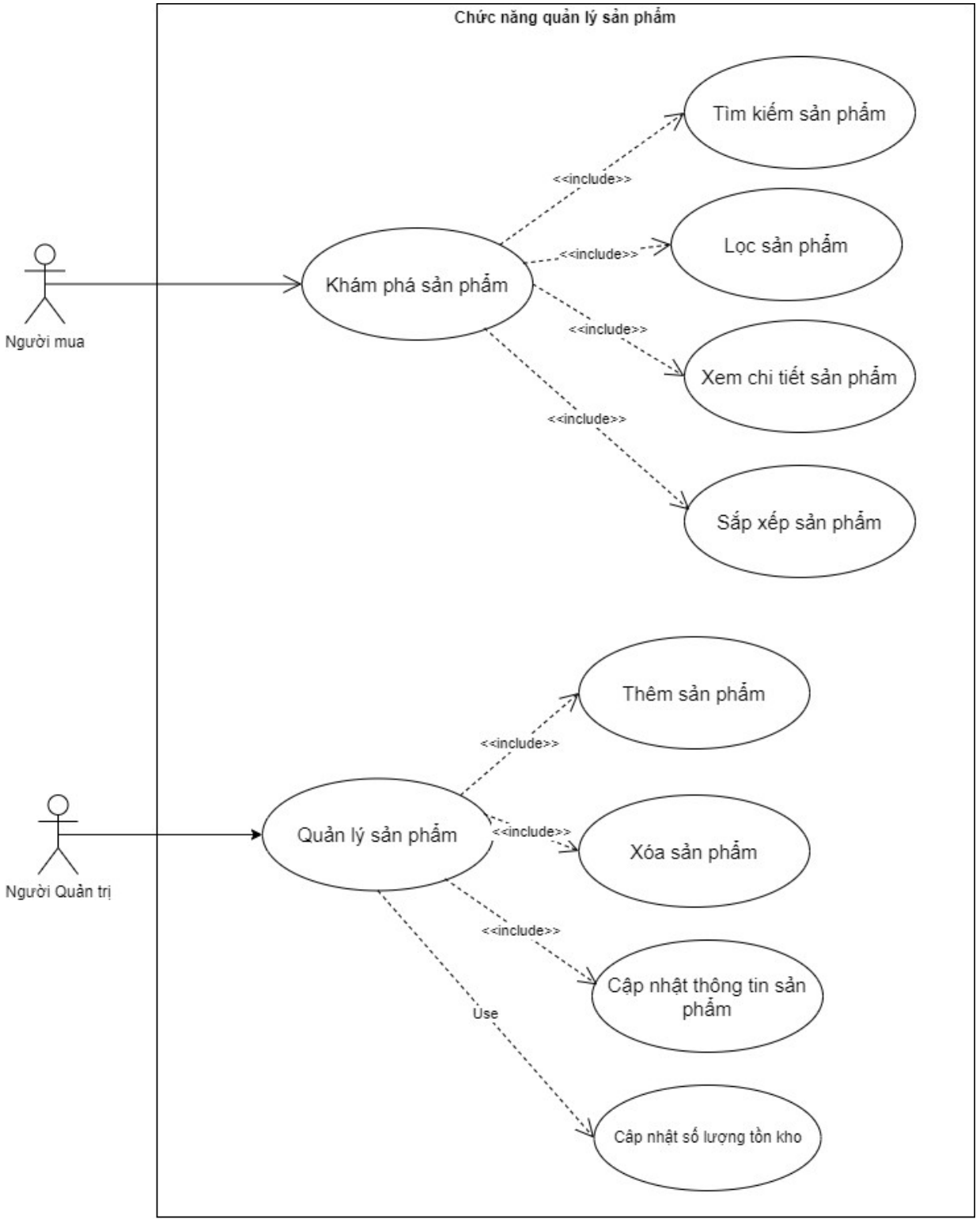
Công thanh toán Stripe tham gia vào quá trình xử lý thanh toán.

Firebase tham gia vào quy trình quản lý truy cập.

Sơ đồ Use Case tổng thể



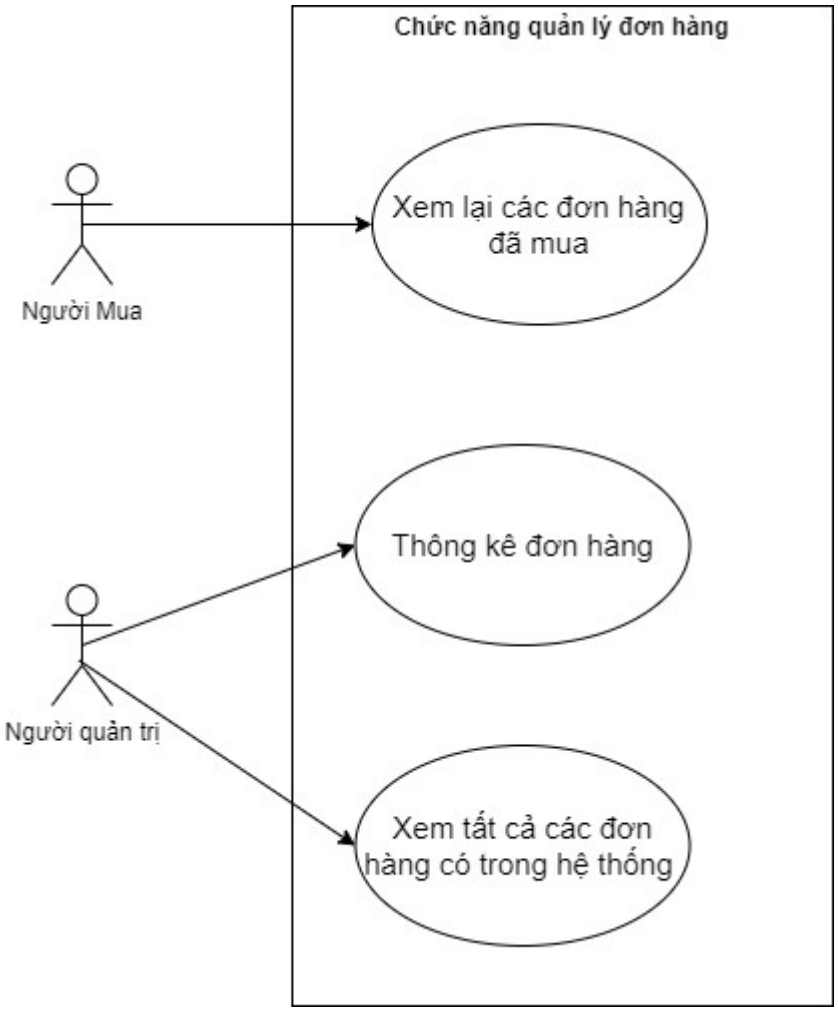
1.1.4.2. Use Case Chức năng Quản lý sản phẩm



Người mua trong chức năng quản lý sản phẩm có thể thể thực hiện tìm kiếm sản phẩm, lọc sản phẩm, sắp xếp và xem chi tiết sản phẩm.

Người quản trị có thể thực hiện các hành động thêm sản phẩm, xóa sản phẩm, cập nhật thông tin sản phẩm và thực hiện cập nhật tồn kho.

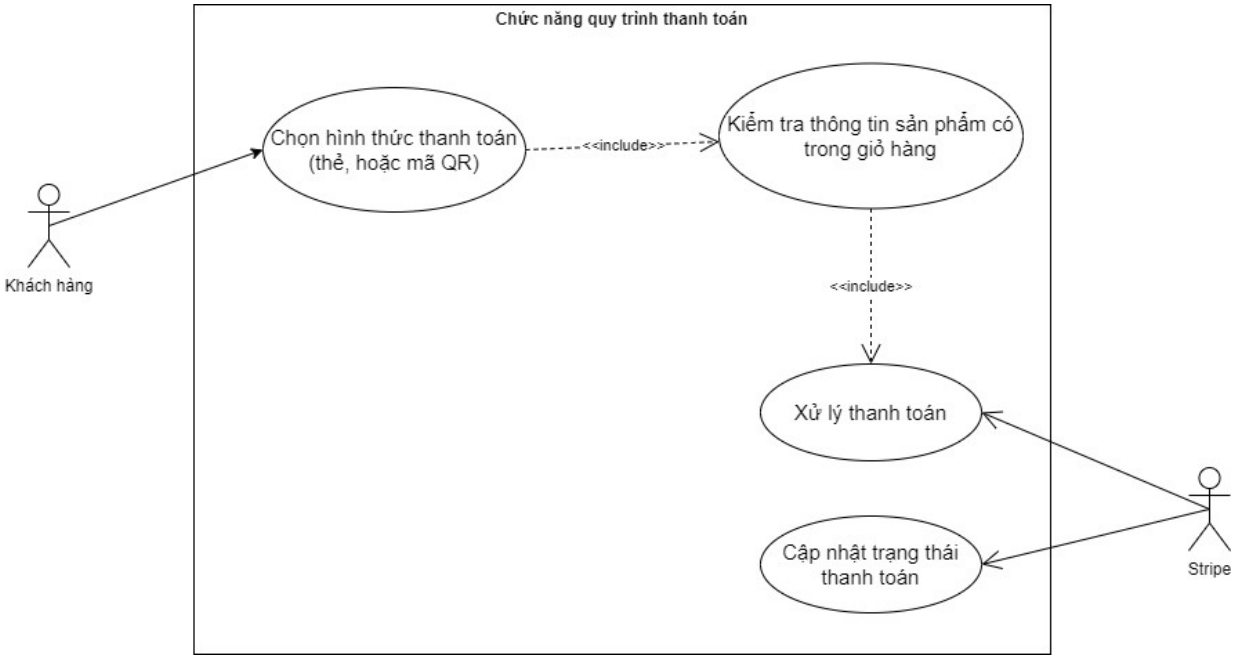
1.1.4.3. Use Case Chức năng Quản lý đơn hàng



Người Mua có thể xem lại các đơn hàng mà mình đã mua

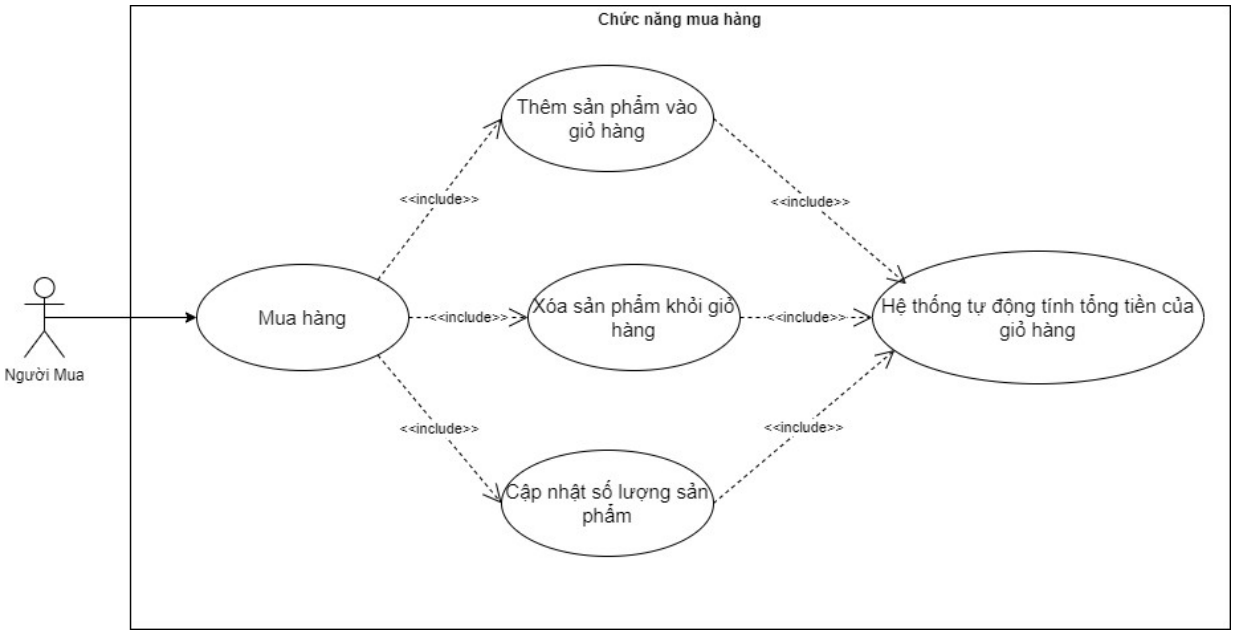
Người quản trị có thể thực hiện xem tất cả các đơn hàng có trong hệ thống và thực có thể thực hiện thống kê trên những đơn hàng đó.

1.1.4.4. Use Case Chức năng thanh toán



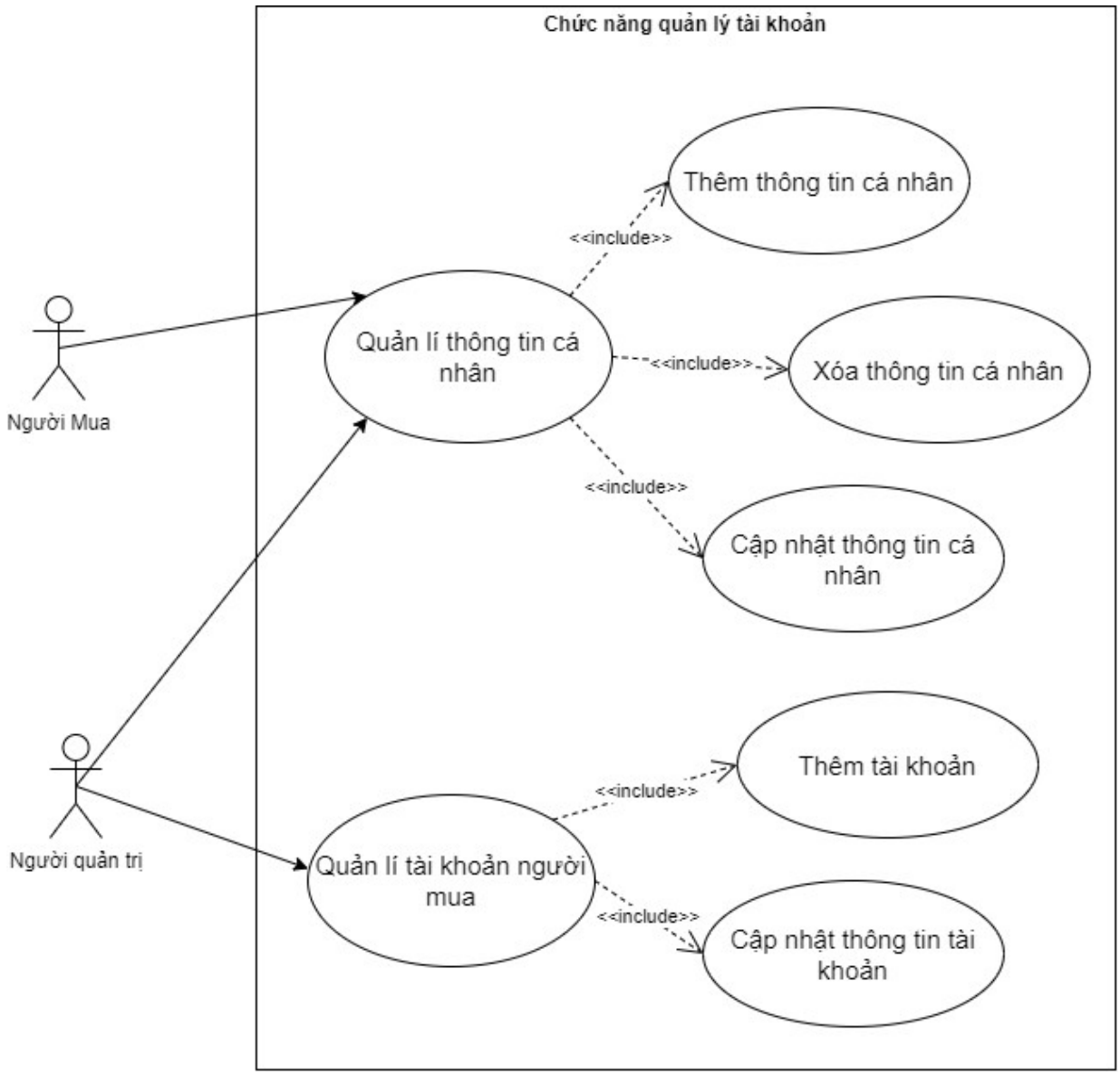
Người mua chọn một trong hai hình thức thanh toán là thanh toán bằng
thẻ ngân hàng hoặc mã QR.
Cổng thanh toán Stripe được gọi để có thể thực hiện giao dịch.

1.1.4.5. Use Case Chức năng quản lý giỏ hàng



Người mua có thể thực hiện thêm, sửa, xóa hoặc cập nhật số lượng sản phẩm trong giỏ hàng.

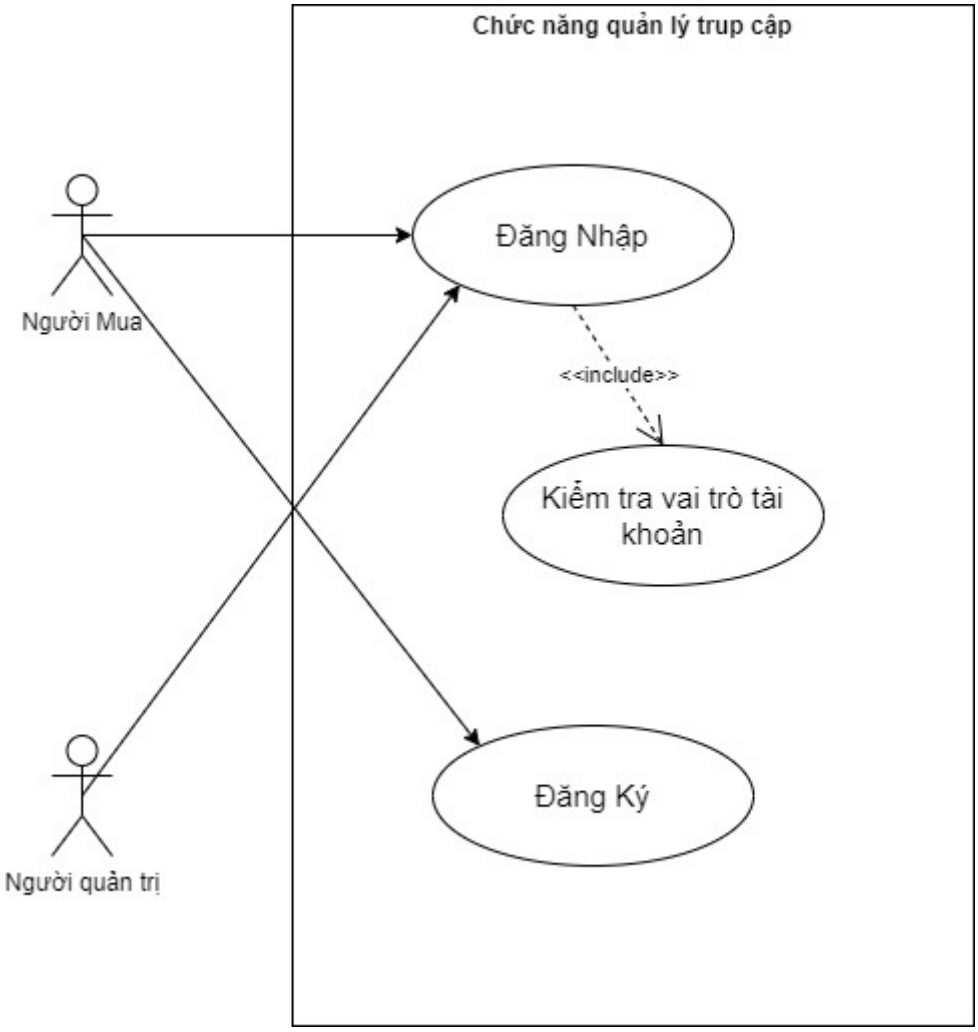
1.1.4.6. Use Case Chức năng quản lý tài khoản



Người mua có thể quản lý thông tin tài khoản cá nhân của mình

Người quản trị có thể thêm, sửa, xóa tài khoản của người mua, đồng thời cũng có thể quản lý thông tin tài khoản quản trị.

1.1.4.7. Use Case Chức năng quản lý truy cập



Người mua có thể đăng nhập và đăng kí tài khoản để có thể sử dụng hệ thống.

Người quản trị chỉ có thể đăng nhập vào hệ thống.

1.1.5. User Story

User Story Quản lý sản phẩm

– Là người mua hàng tôi muốn được nhìn thấy danh sách sản phẩm với giao diện ưa nhìn, đầy đủ thông tin, hình ảnh sản phẩm với độ phân giải tốt. Các chức năng như lọc, sắp xếp theo tên hoặc giá phải hoạt động nhanh và ổn định.

Khi thực hiện lọc sản phẩm theo giá hoặc tên sản phẩm nào, danh sách sản phẩm cần được thu hẹp lại với các sản phẩm phù hợp.

Khi sắp xếp theo thứ tự giảm dần hoặc tăng dần theo giá hoặc tên sản phẩm, thì danh sách sản phẩm cần được sắp xếp theo thứ tự này.

Khi cả lọc và sắp xếp đều được thực hiện, danh sách sản phẩm sẽ có hiệu lực theo cả hai mô tả trên.

– Là Người mua, tôi muốn được điều hướng đến trang chi tiết sản phẩm với các thuộc tính cơ bản như tên, mô tả, số lượng tồn kho, chất liệu, hình ảnh sản phẩm.

– Là Người quản trị, tôi muốn quản lí sản phẩm (CRUD) và cập nhật số lượng tồn kho của sản phẩm.

User Story Quản lý đơn hàng

– Là Người mua, tôi muốn được xem lịch sử đơn hàng của mình.

– Là Người quản trị, tôi muốn được xem tất cả đơn hàng có trong hệ thống.

– Là người quản trị, tôi muốn có được thống kê doanh thu các đơn hàng để từ đó có cái nhìn toàn cảnh về tình hình kinh doanh.

User Story thanh toán

Là người mua, tôi muốn thực hiện thanh toán cho các sản phẩm đã chọn, để hoàn tất việc mua hàng.

Là người mua, khi tôi bắt đầu quá trình thanh toán, tôi muốn hệ thống tự động xác thực thông tin sản phẩm và xử lý thanh toán, để đảm bảo giao dịch được thực hiện chính xác.
Là người mua, nếu thông tin sản phẩm không hợp lệ, tôi muốn quá trình thanh toán bị hủy và nhận được thông báo, để biết rằng giao dịch không thành công.

Là người mua, khi thanh toán thành công, tôi muốn hệ thống ghi nhận giao dịch thành một đơn hàng và gửi email xác nhận, để xác nhận rằng giao dịch của tôi đã hoàn tất thành công.

User Story Quản lý giỏ hàng

– Là Người mua, tôi có thể thêm bất kỳ sản phẩm nào đang hiển thị trên trang danh mục sản phẩm vào trong giỏ hàng của mình. (Khi sản phẩm này được thêm vào giỏ hàng, mặc định sẽ có một sản phẩm được thêm vào giỏ hàng).

– Là Người mua, tôi có thể thêm xem chi tiết sản phẩm và mua sản phẩm này nếu muốn. (Khi sản phẩm này được thêm vào giỏ hàng, mặc định sẽ có một sản phẩm được thêm vào giỏ hàng).

– Là Người mua, tôi muốn xem danh sách sản phẩm tôi vừa thêm vào giỏ hàng hiện tại và muốn xem bảng thông tin cho giỏ hàng hiện tại như tổng chi phí giỏ hàng hiện tại, chi phí khuyến mãi nếu có.

– Là Người mua, tôi muốn cập nhật số lượng sản phẩm trong giỏ hàng. Bất cứ khi nào số lượng sản phẩm trong giỏ hàng được cập nhật, bảng thông tin tóm được cập nhật theo số lượng thay đổi.

– Là Người mua, tôi muốn xóa bất kỳ sản phẩm nào có trong danh sách sản phẩm trong giỏ hàng mà tôi không muốn mua nữa.

– Là Người mua, khi thực hiện thanh toán cho giỏ hàng của mình tôi muốn nó phải diễn ra nhanh chóng, trường hợp có vấn đề phát sinh thì tôi mong muốn rằng mình sẽ nhận được sự trợ giúp từ phía cửa hàng.

User Story Quản lí tài khoản

– Là Người mua/Người quản trị, tôi muốn quản lý thông tin cá nhân trong tài khoản của mình.

– Là Người quản trị, tôi muốn quản lý tài khoản người mua.

User Story Quản lý truy cập

– Mỗi Người Mua/Người quản trị là một người dùng (User).

– Là Người Mua/Ngươi quản trị, tôi muốn đăng ký vào hệ thống.

– Là Người Mua/Ngươi quản trị, tôi muốn đăng nhập vào hệ thống.

– Bất cứ khi nào người dùng có vai trò là Người Mua đăng nhập vào hệ thống, người dùng sẽ được chuyển đến trang danh mục sản phẩm.

– Bất cứ khi nào người dùng có vai trò là Người quản trị đăng nhập vào hệ thống, người dùng sẽ được chuyển đến trang Dashboard.

– Là Người mua/Người quản trị, tôi muốn đăng xuất khỏi hệ thống.

1.1.6. Domian Driven Desgin

1.1.6.1. Các Bối Cảnh Giới Hạn (Bounded Contexts)

Hệ thống GoShop có thể được chia thành 4 Bối cảnh Giới hạn chính, dựa trên các kịch bản kinh doanh và vai trò người dùng:

A. Bối Cảnh TÀI KHOẢN & TRUY CẬP (Access & Account)

Mục đích: Quản lý thông tin người dùng, xác thực (Authentication) và ủy quyền (Authorization) truy cập.

Ngôn ngữ Chung (Ubiquitous Language): Người Dùng, Vai trò (Người Mua/Quản Trị), Thông tin Tài khoản, Đăng nhập, Kiểm tra Vai trò.

Thực thể Gốc (Aggregate Root): Tài Khoản Người Dùng (User Account).

B. Bối Cảnh DANH MỤC & SẢN PHẨM (Catalog)

Mục đích: Quản lý nội dung sản phẩm (thông tin chi tiết, danh mục) và theo dõi số lượng tồn kho.

Ngôn ngữ Chung: Sản phẩm, Danh mục, Thông tin Chi tiết Sản phẩm (chiều cao, brand, vật liệu), số lượng tồn Kho.

Thực thể Gốc: Sản Phẩm (Product).

C. Bối Cảnh MUA SẮM & GIỎ HÀNG (Shopping & Cart)

Mục đích: Quản lý trải nghiệm mua sắm của Người Mua, bao gồm duyệt, tìm kiếm và quản lý giỏ hàng.

Ngôn ngữ Chung: Giỏ Hàng, Mục Giỏ Hàng, Tính Tổng Tiền, Tìm kiếm, Lọc, Sắp xếp.

Thực thể Gốc: Giỏ Hàng (Shopping Cart).

D. Bối Cảnh QUẢN LÝ ĐƠN HÀNG & THANH TOÁN (Order & Payment)

Mục đích: Quản lý quy trình đặt hàng, xử lý thanh toán và theo dõi lịch sử đơn hàng/thống kê.

Ngôn ngữ Chung: Đơn Hàng, Mục Đơn Hàng, Trạng thái Đơn hàng, Thanh toán, Kiểm tra Thanh toán, Thống kê Doanh thu.

Thực thể Gốc: Đơn Hàng (Order).

1.1.6.2. Các Thực Thể Gốc (Aggregate Roots)

Bối Cảnh	Thực Thể Gốc	Vai trò/Trách nhiệm

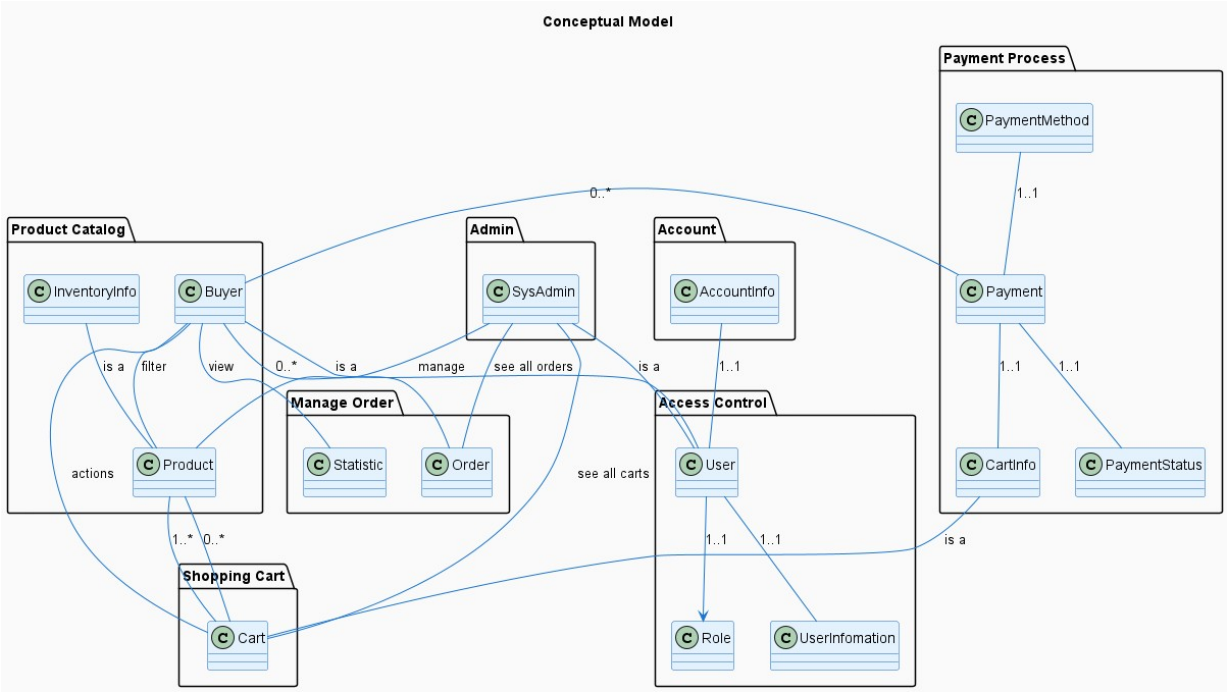
Giới Hạn	(Aggregate Root)	Chính
Tài Khoản & Truy Cập	User Account	Đảm bảo tính nhất quán của role và trạng thái truy cập.
Danh Mục & Sản Phẩm	Product	Chứa thông tin chi tiết và quan trọng nhất là số lượng Tồn Kho ().
Mua Sắm & Giỏ Hàng	Shopping Cart	Quản lý toàn bộ các mục sản phẩm được chọn và tính tổng tiền.
Quản Lý Đơn Hàng & Thanh Toán	Order	Quản lý trạng thái đơn hàng, trạng thái thanh toán và thực hiện tương tác với Stripe.

1.1.6.3. Mối Quan Hệ Giữa Các Bối Cảnh (Context Mapping)

Tương Tác	Bối Cảnh Nguồn	Bối Cảnh Đích	Mô Tả Mối Quan Hệ (Quy Tắc)
Thêm vào Giỏ hàng	Mua Sắm & Giỏ Hàng	Danh Mục & Sản Phẩm	Giỏ hàng cần **kiểm tra** Tồn Kho của Sản phẩm trước khi thêm
Đặt hàng	Quản Lý Đơn Hàng	Tài Khoản & Truy Cập	Cần thông tin Địa chỉ giao hàng từ Tài Khoản Người

			Mua.
Hoàn tất Thanh toán	Quản Lý Đơn Hàng	Danh Mục & Sản Phẩm	Sau khi thanh toán thành công, Đơn Hàng **Gửi Lệnh** để Trừ Tồn Kho
Quản lý Đơn hàng	Quản Lý Đơn Hàng	Tài Khoản & Truy Cập	Người Mua **xem lịch sử** đơn hàng của mình

1.1.7. Mô hình ý niệm



Hệ thống được thiết kế theo kiến trúc phân lớp với các package chức năng rõ ràng, bao gồm: Admin để quản lý toàn bộ hệ thống; Product Catalog chịu trách nhiệm quản lý sản phẩm, tồn kho và thông tin người mua; Shopping Cart xử lý các nghiệp vụ liên quan đến giỏ hàng; Access Control quản lý người dùng và vai trò; Account quản lý thông

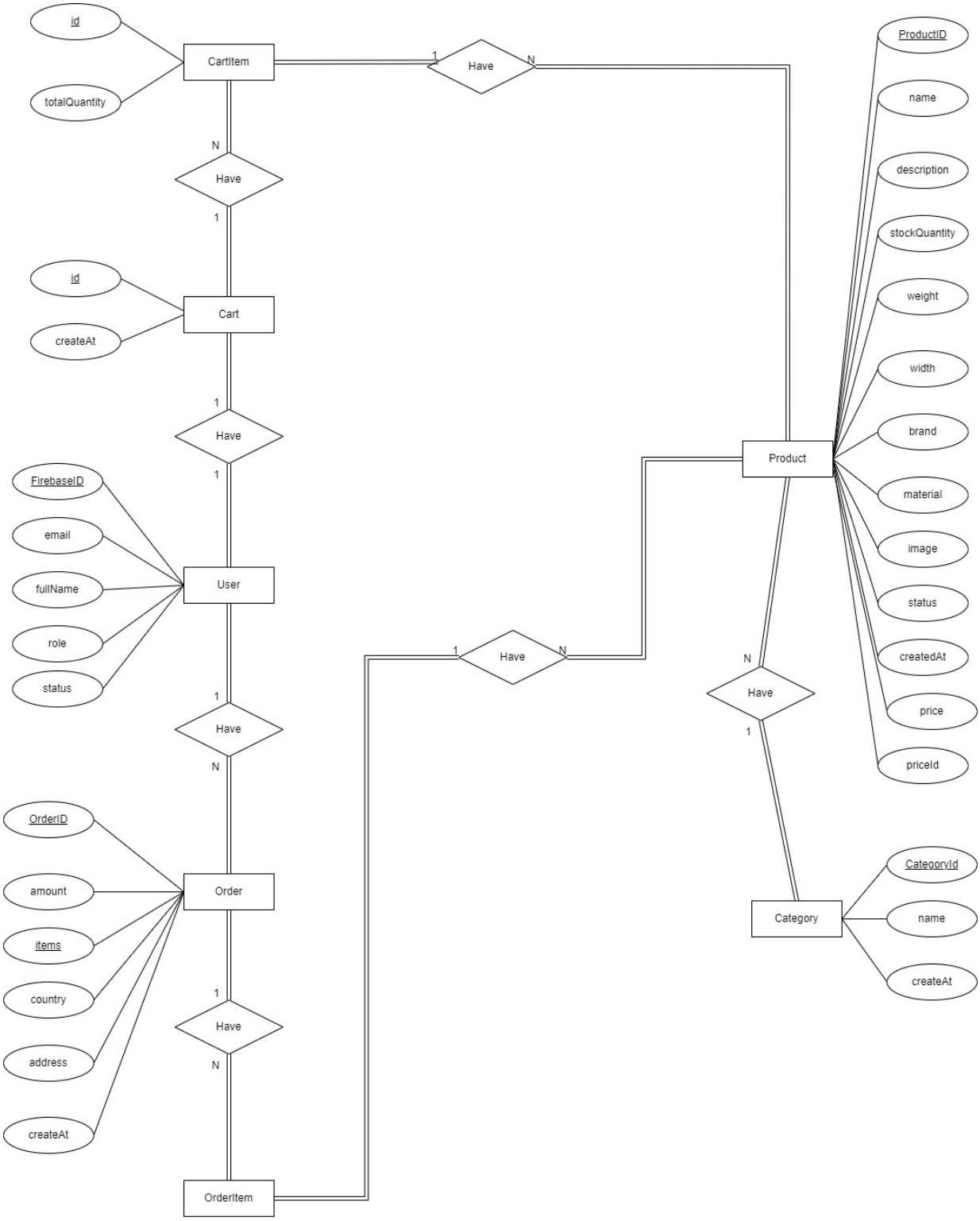
tin đăng nhập; Payment Process thực hiện thanh toán; và Manage Order để quản lý đơn hàng.

Trong mô hình dữ liệu, User là thực thể gốc và có quan hệ một–một với UserInformation, Role và AccountInfo. Từ User, hai thực thể Buyer và SysAdmin được kế thừa. Buyer có thể xem và lọc sản phẩm, thao tác trên giỏ hàng, tạo đơn hàng và thực hiện thanh toán. Ngược lại, SysAdmin có quyền quản lý sản phẩm, đồng thời theo dõi toàn bộ giỏ hàng và đơn hàng trong hệ thống. Product được liên kết với InventoryInfo để xác định số lượng tồn kho. Cart có quan hệ nhiều–nhiều với Product, và CartInfo là phần mở rộng của Cart, phục vụ cho quá trình thanh toán. Payment có quan hệ một–một với PaymentMethod, PaymentStatus và CartInfo, trong khi Order thuộc về Buyer và được SysAdmin giám sát trong suốt vòng đời xử lý.

1.2. Phân tích dữ liệu

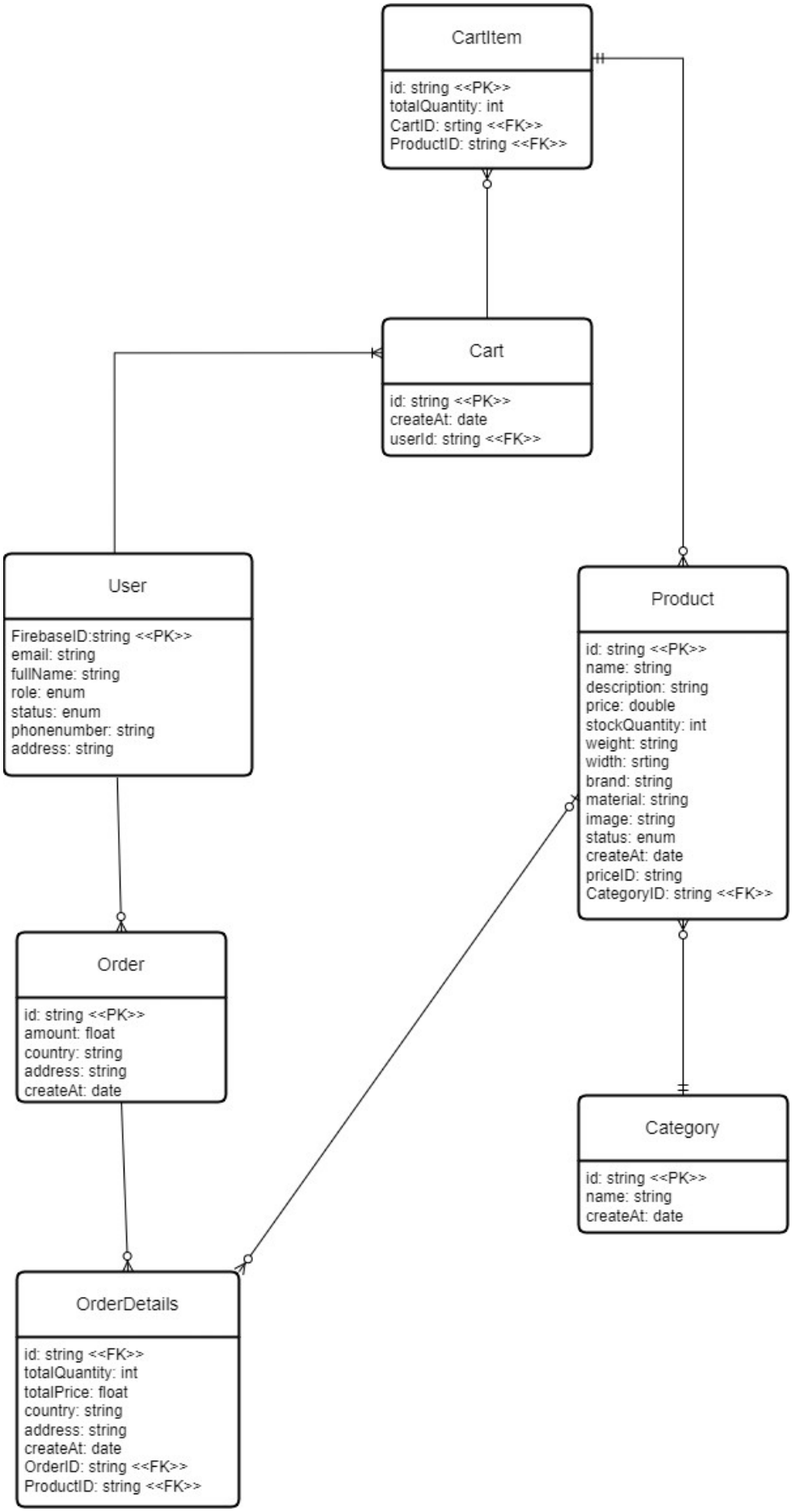
1.2.1. Mức khái niệm

Mô hình thực thể kết hợp mức khái niệm



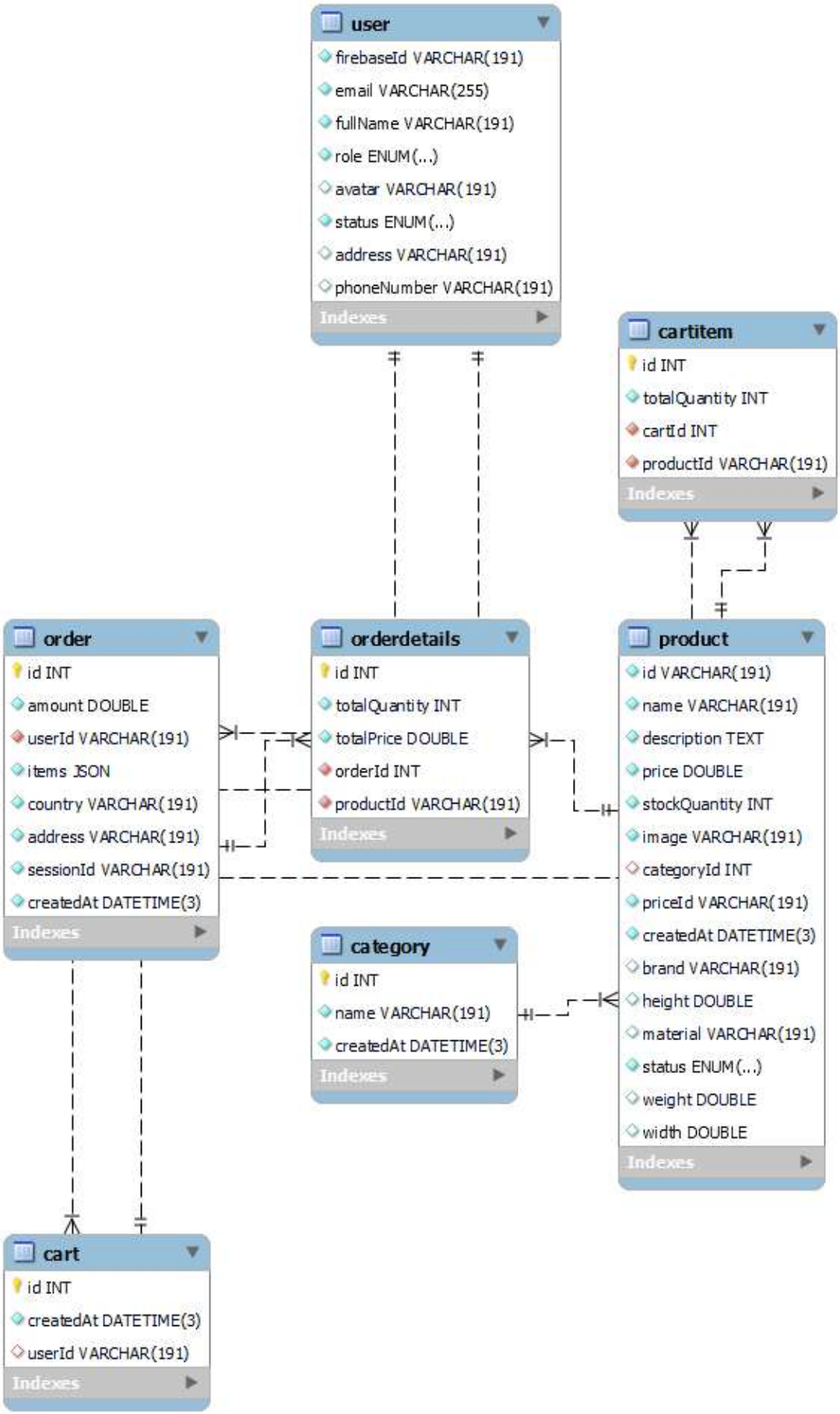
Mô hình thực thể kết hợp mức ý niệm này mô tả các thực thể chính gồm User, Product, Category và Order, OrderItem, Cart và CartItem, thông qua quan hệ Have.

1.2.2. Mức logic


CartItem
id: string <<PK>>
totalQuantity: int
CartID: srting <<FK>>
ProductID: string <<FK>>
Cart
id: string <<PK>>
createAt: date
userId: string <<FK>>
User
FirebaseID:string <<PK>>
email: string
fullName: string
role: enum
status: enum
phonenumber: string
address: string
Product
id: string <<PK>>
name: string
description: string
price: double
stockQuantity: int
weight: string
width: srting
brand: string
material: string
image: string
status: enum
createAt: date
priceID: string
CategoryID: string <<FK>>
Order
id: string <<PK>>
amount: float
country: string
address: string
createAt: date
Category
id: string <<PK>>
name: string
createAt: date
OrderDetails
id: string <<FK>>
totalQuantity: int
totalPrice: float
country: string
address: string
createAt: date
OrderID: string <<FK>>
ProductID: string <<FK>>

Mô hình mức logic được tiếp tục xây dựng từ mô hình khái niệm. Trong mô hình này mỗi thực thể từ mô hình ở mức ý nhiệm đã được chuyển hóa thành một bảng dữ liệu với khóa chính và khóa ngoại đầy đủ.

1.2.3. Mức vật lý

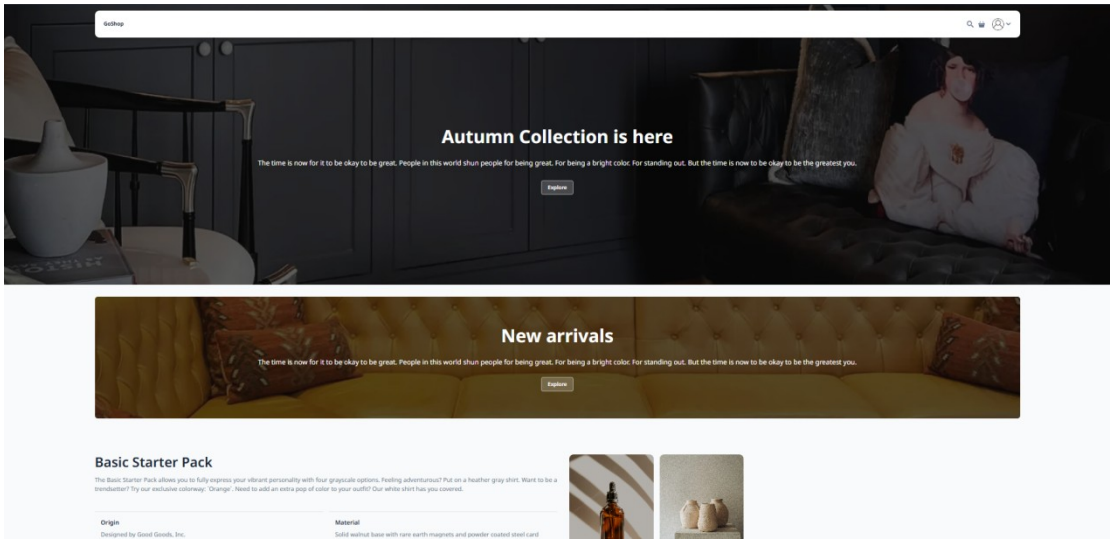

user
firebaseId VARCHAR(191)
email VARCHAR(255)
fullName VARCHAR(191)
role ENUM(...)
avatar VARCHAR(191)
status ENUM(...)
address VARCHAR(191)
phoneNumber VARCHAR(191)
Indexes
cartitem
id INT
totalQuantity INT
cartId INT
productId VARCHAR(191)
Indexes
order
id INT
amount DOUBLE
userId VARCHAR(191)
items JSON
country VARCHAR(191)
address VARCHAR(191)
sessionId VARCHAR(191)
createdAt DATETIME(3)
Indexes
orderdetails
id INT
totalQuantity INT
totalPrice DOUBLE
orderId INT
productId VARCHAR(191)
Indexes
product
id VARCHAR(191)
name VARCHAR(191)
description TEXT
price DOUBLE
stockQuantity INT
image VARCHAR(191)
categoryId INT
priceId VARCHAR(191)
createdAt DATETIME(3)
brand VARCHAR(191)
height DOUBLE
material VARCHAR(191)
status ENUM(...)
weight DOUBLE
width DOUBLE
Indexes
category
id INT
name VARCHAR(191)
createdAt DATETIME(3)
Indexes
cart
id INT
createdAt DATETIME(3)
userId VARCHAR(191)
Indexes

Mô hình thực thể kết hợp ở mức vật lý thể hiện cấu trúc dữ liệu thực tế được tổ chức bên trong cơ sở dữ liệu của hệ thống.

1.3. Phân tích giao diện

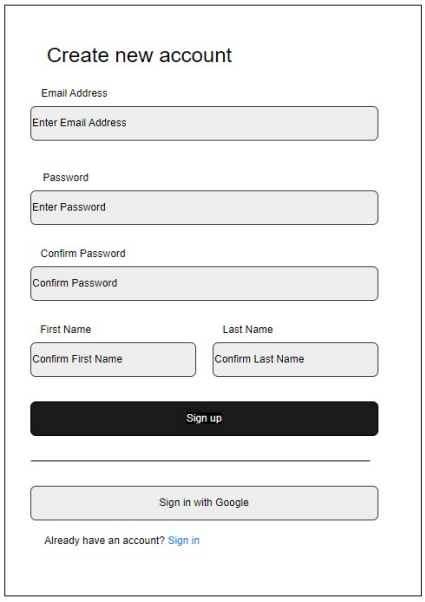
1.3.1. Giao diện dành cho người mua hàng

1.3.1.1. Giao diện trang chủ



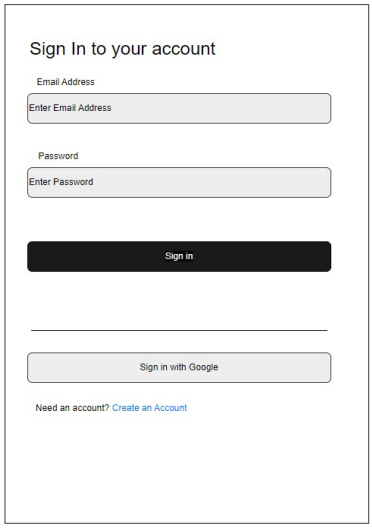
Giao diện trang chủ khi người dùng vừa truy cập vào trang web GoShop.

1.3.1.2. Giao diện đăng ký



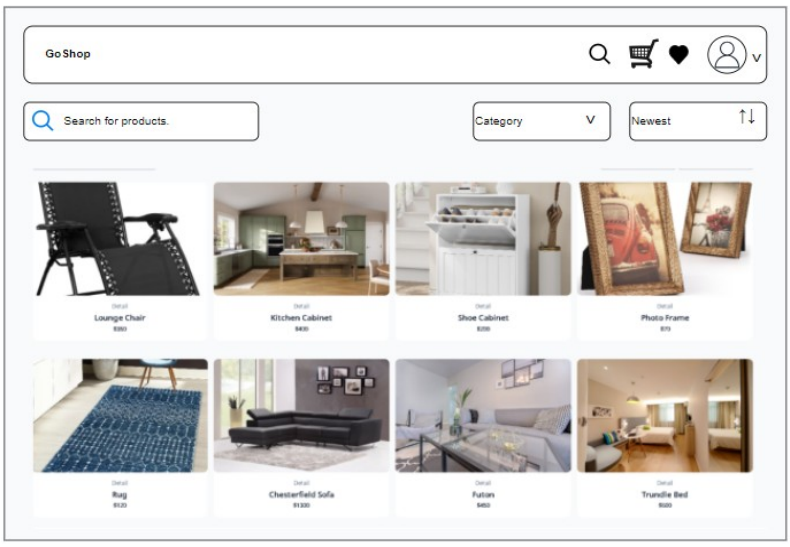
Giao diện đăng ký tài khoản danh cho người mua có nhu cầu mua hàng của GoShop

1.3.1.3. Giao diện đăng nhập



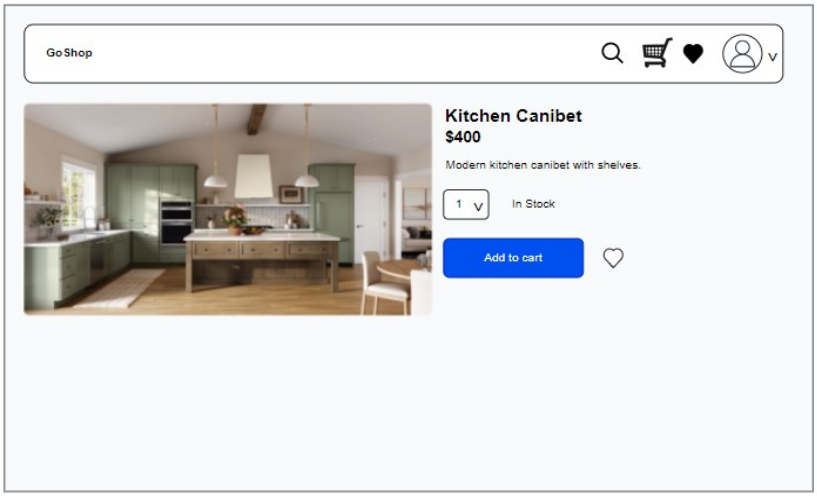
Giao diện đăng nhập dành cho người mua nếu đã đăng ký tài khoản.

1.3.1.4. Giao diện danh mục sản phẩm



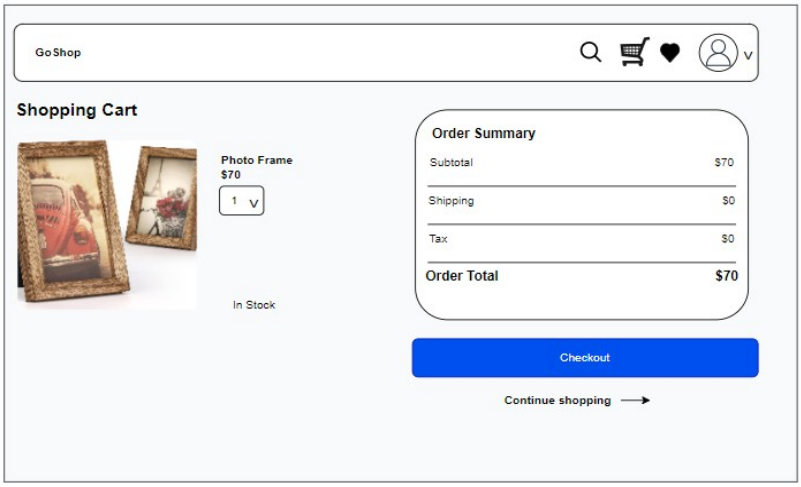
Giao diện trang danh mục sản phẩm nơi trưng bày những sản phẩm nội thất được bày bán ttrong GoShop. Ở phía trên bên trái là thanh tìm kiếm còn bên phải theo thứ tự từ ngoài vào lần lượt là bộ lọc sản phẩm theo danh mục và ô dù để kích hoạt chức năng sắp xếp sản phẩm theo giá tiền.

1.3.1.5. Giao diện chi tiết sản phẩm



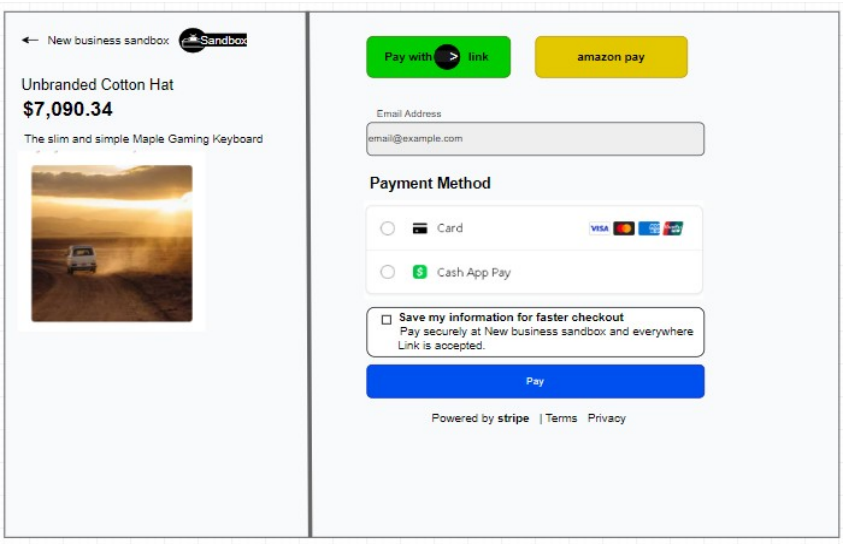
Giao diện chi tiết sản phẩm nhằm mục tiêu hiển thị chi tiết thông tin sản phẩm đến khách hàng.

1.3.1.6. Giao diện giỏ hàng



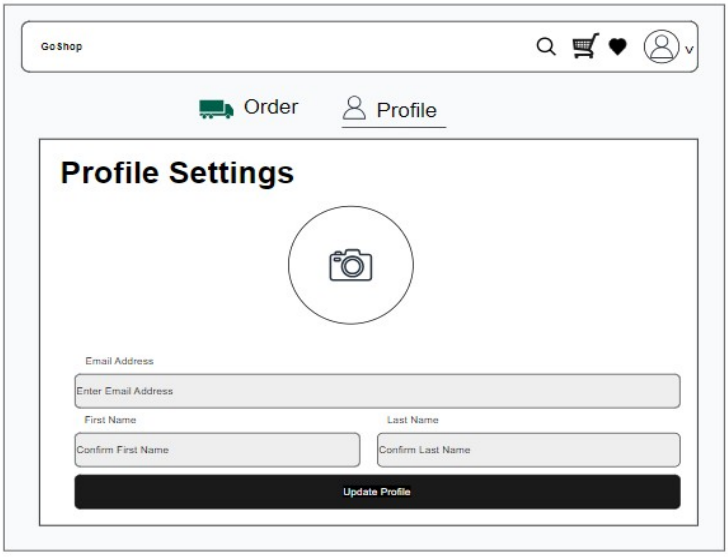
Giao diện giỏ hàng nhằm mục tiêu giúp người mua tương tác với giỏ hàng nhằm mục tiêu tăng cường trải nghiệm mua sắm.

1.3.1.7. Giao diện thanh toán



Giao diện thanh toán nhằm giúp người dùng có thể thực hiện giao dịch ngay tại trang web sau khi đã lựa chọn được món hàng mà mình ưng ý.

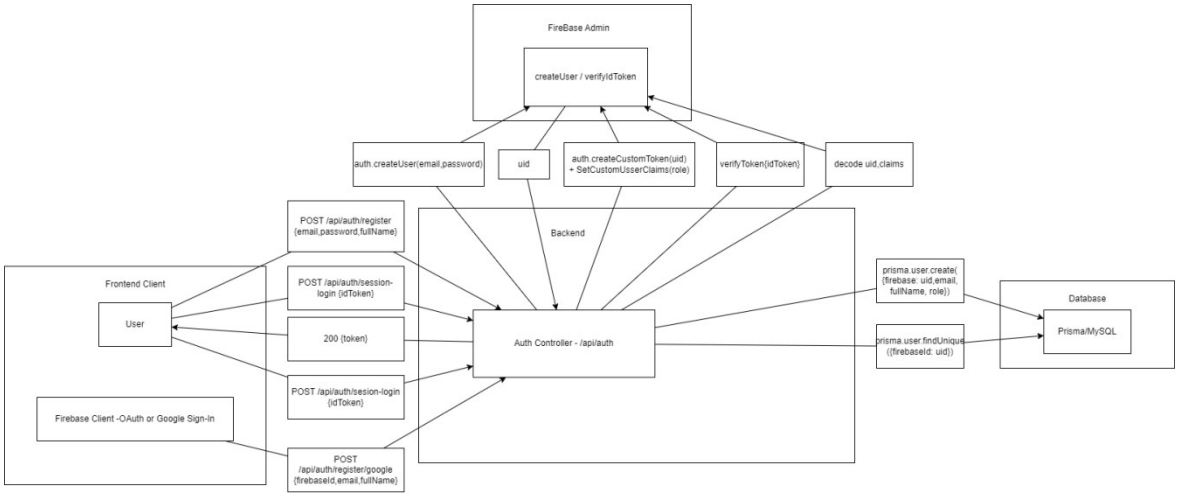
1.3.1.8. Giao diện chỉnh sửa thông tin tài khoản



Giao diện chỉnh sửa thông tin tài khoản nhằm mục tiêu giúp người dùng có thể điều chỉnh thông tin tài khoản của mình một cách tùy ý.

2. Thiết kế hệ thống

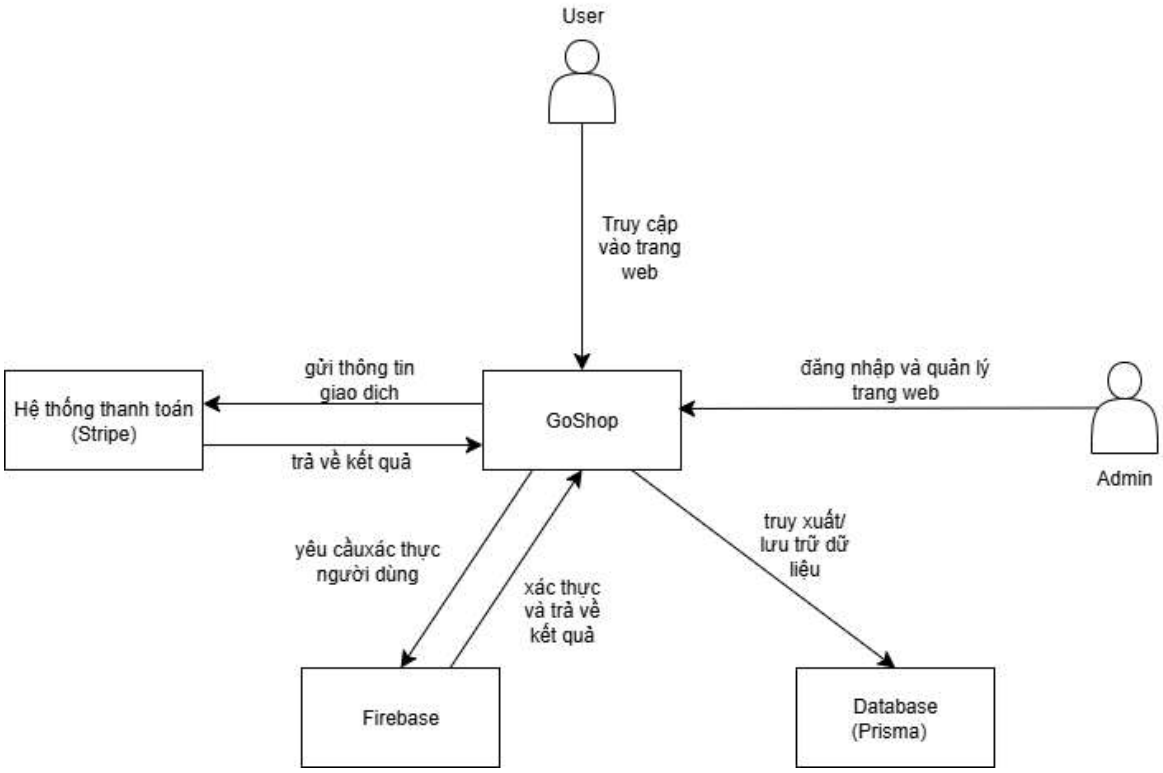
2.1. Sơ đồ khối



Sơ đồ thể hiện quy trình xác thực người dùng trong hệ thống theo mô hình tách biệt giữa BRontend, backend và Firebase. Người dùng thực hiện đăng ký hoặc đăng nhập từ phía BRontend bằng email–password hoặc Google Sign-In. Firebase Authentication chịu trách nhiệm xác thực danh tính và cấp ID Token. Token này được gửi về backend, nơi Auth Controller sử dụng Firebase Admin để xác minh, giải mã token và lấy thông tin người dùng. Sau đó, backend lưu hoặc đối chiếu dữ liệu người dùng với cơ sở dữ liệu MySQL thông qua Prisma. Cách tiếp cận này giúp hệ thống đảm bảo an toàn xác thực, đồng thời giữ toàn bộ logic nghiệp vụ và dữ liệu người dùng tập trung tại backend.

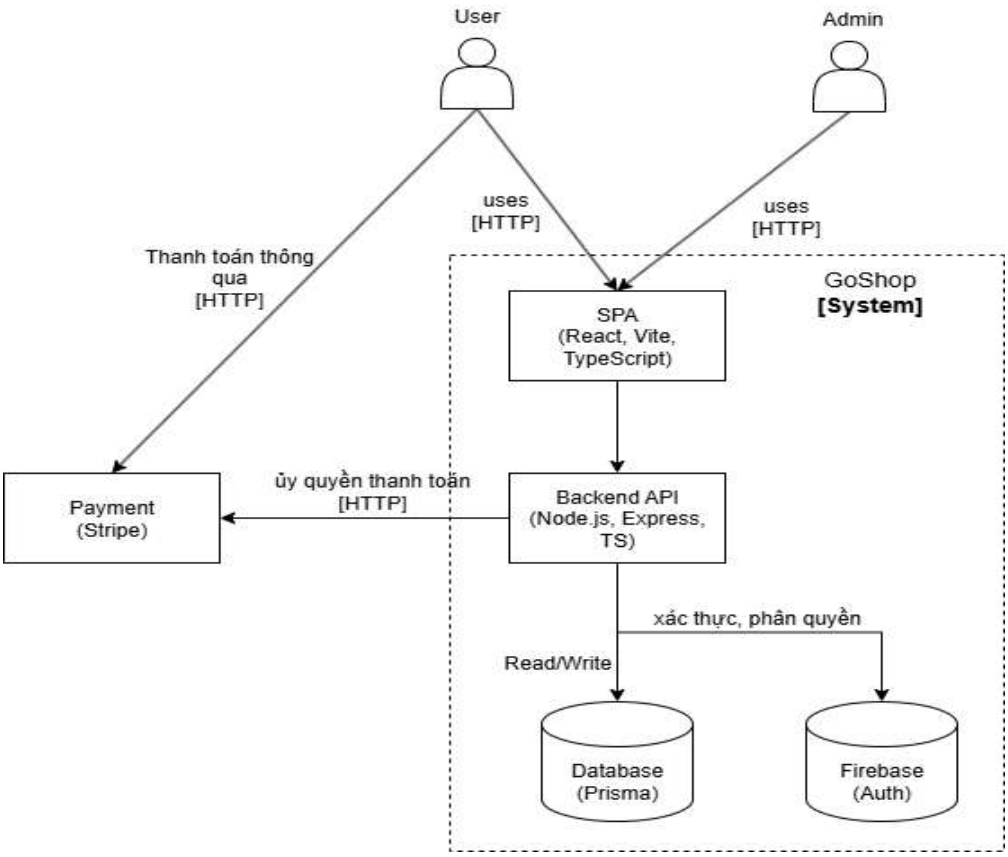
2.2. Decomposition View

2.2.1. C1



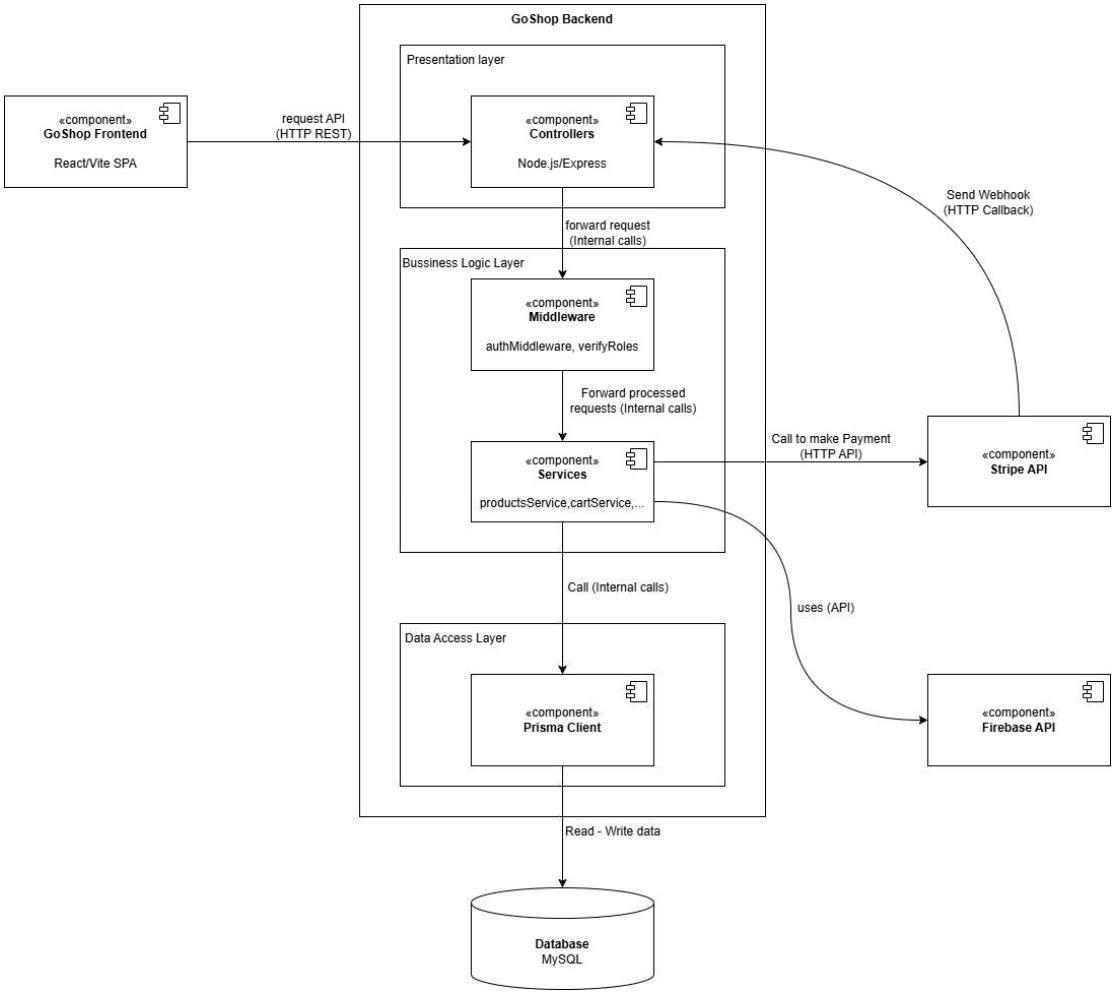
GoShop là hệ thống phần mềm trung tâm đang được xây dựng, đóng vai trò xử lý toàn bộ nghiệp vụ của ứng dụng thương mại điện tử. Người dùng và Admin là các thực thể bên ngoài, lần lượt tương tác với hệ thống thông qua việc truy cập, đăng nhập và quản lý các chức năng trên website. GoShop tích hợp với Stripe để xử lý các giao dịch thanh toán, trong đó hệ thống gửi thông tin giao dịch và nhận lại kết quả thành công hoặc thất bại. Đồng thời, Firebase được sử dụng như một dịch vụ bên ngoài để thực hiện xác thực người dùng và trả về kết quả xác thực cho hệ thống. Dữ liệu của GoShop được lưu trữ và truy xuất thông qua Database, với Prisma đóng vai trò lớp trung gian để kết nối và thao tác với cơ sở dữ liệu vật lý.

2.2.2. C2



Sơ đồ GoShop ở mức Container (Level 2) mô tả kiến trúc cấp cao của hệ thống bằng cách chia GoShop thành các thành phần có thể triển khai độc lập. Hệ thống bao gồm SPA được xây dựng bằng React, Vite và TypeScript, đóng vai trò giao diện người dùng cho cả User và Admin, tiếp nhận tương tác và gửi yêu cầu đến Backend API. Backend API, phát triển bằng Node.js, Express và TypeScript, là trung tâm xử lý logic nghiệp vụ, điều phối quy trình thanh toán, giao tiếp với cơ sở dữ liệu thông qua Prisma và thực hiện xác thực, phân quyền thông qua Firebase Auth. Database đảm nhiệm việc lưu trữ dữ liệu nghiệp vụ như sản phẩm, đơn hàng và thông tin người dùng, trong khi Firebase chịu trách nhiệm quản lý danh tính và token xác thực. Ngoài ra, hệ thống tích hợp Stripe để xử lý thanh toán, trong đó Backend tạo và quản lý giao dịch, còn người dùng trực tiếp tương tác với giao diện thanh toán do Stripe cung cấp.

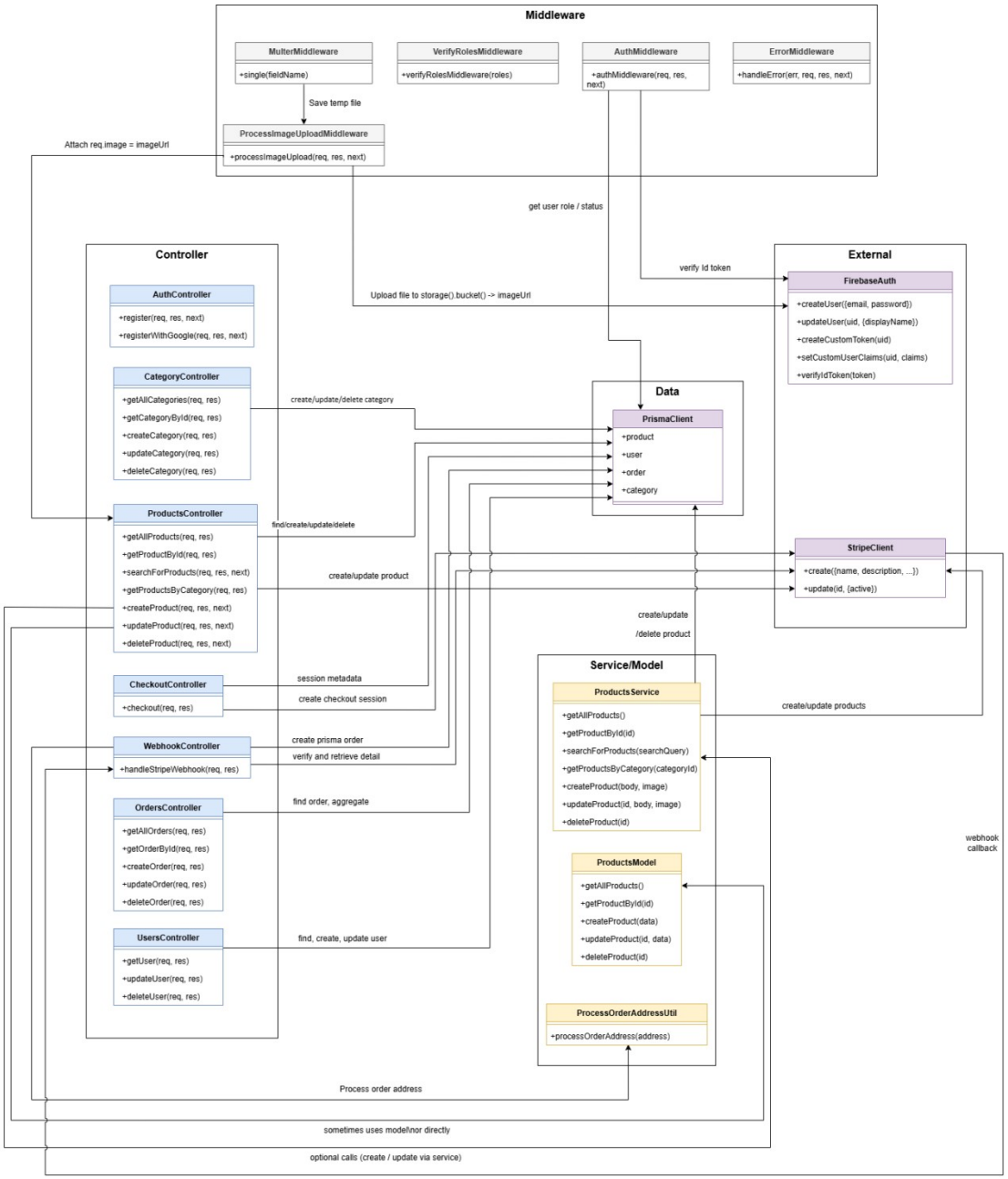
2.2.3. C3 (module-level)



Sơ đồ Cấp độ 3 (Component Diagram) của GoShop mô tả kiến trúc bên trong Backend bằng cách phân tách Container Backend thành các thành phần chức năng theo kiến trúc phân lớp. Backend được tổ chức thành ba lớp chính gồm Presentation Layer, Business Logic Layer và Data Access Layer. Presentation Layer bao gồm các Controller, đóng vai trò là điểm vào của các API, tiếp nhận yêu cầu từ BRontend và định tuyến chúng vào hệ thống. Business Logic Layer gồm Middleware và Services, trong đó Middleware xử lý các chức năng dùng chung như xác thực và phân quyền, còn Services thực hiện logic nghiệp vụ cốt lõi, đồng thời giao tiếp với các hệ thống bên ngoài như Stripe để xử lý thanh toán và Firebase

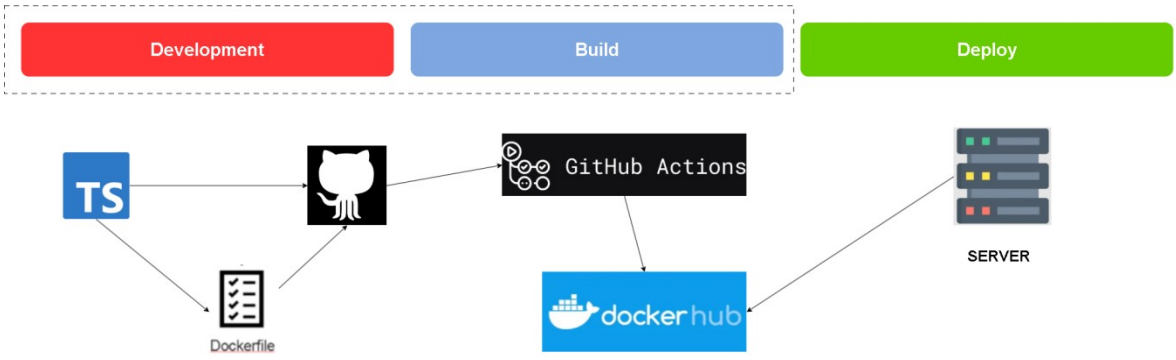
để kiểm tra thông tin người dùng. Data Access Layer được đại diện bởi Prisma Client, chịu trách nhiệm truy xuất và lưu trữ dữ liệu vào cơ sở dữ liệu MySQL. Ngoài ra, Backend còn nhận webhook từ Stripe để cập nhật trạng thái giao dịch, đảm bảo tính nhất quán cho đơn hàng trong hệ thống.

2.2.4 C4



Sơ đồ GoShop Backend ở Cấp độ 4 (Code Diagram) mô tả chi tiết cấu trúc mã nguồn bên trong các Component của Backend và mối quan hệ gọi hàm giữa chúng. Luồng xử lý bắt đầu khi request đi vào hệ thống và lần lượt được xử lý qua chuỗi Middleware để xác thực token, kiểm tra quyền, kiểm tra schema dữ liệu, xử lý ảnh và quản lý lỗi tập trung. Sau đó, request được chuyển đến Controller tương ứng theo từng nghiệp vụ như xác thực, sản phẩm, danh mục, đơn hàng, thanh toán hay webhook. Các Controller đóng vai trò điều phối, nhận request đã được xử lý sơ bộ và gọi các Service Model để thực hiện logic nghiệp vụ cốt lõi. Tại tầng Service, hệ thống xử lý các quy trình phức tạp như quản lý sản phẩm, đơn hàng, địa chỉ giao hàng và thanh toán, đồng thời thao tác dữ liệu thông qua PrismaClient hoặc gọi các dịch vụ bên ngoài như FirebaseAuth và StripeClient. Cách tổ chức này giúp mã nguồn rõ ràng, tách biệt trách nhiệm và dễ bảo trì, mở rộng.

2.3. Deloyment View



Website GoShop được viết bằng ngôn ngữ typescript, backend sẽ complie nó thành javascript để sử dụng, tạo dockerfile cũng như nginx để serve static files cho react rồi tất cả sẽ được đẩy lên github. Sau khi đẩy lên github thì github actions sẽ được trigger, github actions sẽ tự động build

Docker image khi push code, rồi push image lên DockerHub. Sau đó sẽ setup server, pull và run image và tự động hóa deploy.

CHƯƠNG 3: KẾ HOẠCH KIỂM THỬ

3.1. Giới thiệu

Dự án GoShop là một hệ thống thương mại điện tử full- stack: backend TypeScript/Node.js (Prisma, Express/Koa hoặc tương tự), BRontend khách hàng và admin sử dụng Vite + React/TS. Mục tiêu chính là cung cấp nền tảng quản lý sản phẩm, giỏ hàng, thanh toán và quản trị đơn hàng cho cửa hàng trực tuyến.

3.1.1. Mục đích

Xây dựng nền tảng bán hàng trực tuyến ổn định, mở rộng được và dễ vận hành. Hỗ trợ quy trình bán hàng: quản lý danh mục, sản phẩm, đơn hàng, thanh toán, thống kê. Cung cấp giao diện admin để quản trị và xử lý đơn hàng nhanh chóng.

3.1.2. Bối cảnh

Dự án Goshop là một hệ thống ứng dụng web (Web Application) phục vụ nhu cầu thương mại điện tử. Hệ thống được phát triển chủ yếu dựa trên ngôn ngữ TypeScript (76.1%), kết hợp với HTML và JavaScript để xây dựng giao diện người dùng và logic xử lý.

Đặc điểm kỹ thuật của dự án đòi hỏi việc kiểm thử phải chú trọng vào tính tương thích của trình duyệt (do sử dụng HTML/JS) và tính chặt chẽ của dữ liệu (do đặc thù của TypeScript) trong các luồng giao dịch mua sắm.

3.1.3. Phạm vi

Phạm vi kiểm thử của dự án GoShop tập trung vào việc xác minh và đảm bảo các chức năng chính của hệ thống thương mại điện tử hoạt động đúng theo yêu cầu nghiệp vụ và mang lại trải nghiệm người dùng ổn định.

Chức năng người dùng (BRontend):

Đăng ký và đăng nhập tài khoản.

Xem danh mục sản phẩm và chi tiết sản phẩm.

Tìm kiếm và lọc sản phẩm theo tiêu chí.

Thêm, chỉnh sửa, hoặc xóa sản phẩm trong giỏ hàng.

Thực hiện quy trình thanh toán (checkout).

Xem lịch sử và trạng thái đơn hàng.

Chức năng quản trị (Admin):

Quản lý danh mục và sản phẩm.

Quản lý người dùng, đơn hàng

Kiểm thử tích hợp (Integration Test):

Kiểm tra luồng dữ liệu giữa BRontend – Backend – Cơ sở dữ liệu.

Kiểm thử các API chính như đăng nhập, thanh toán, và truy xuất sản phẩm.

Kiểm thử giao diện và khả năng sử dụng (UI/UX Test):

Đảm bảo bố cục, màu sắc, và trải nghiệm người dùng nhất quán.

Kiểm thử hồi quy (Regression Test):

Đảm bảo các chức năng cũ vẫn hoạt động bình thường sau khi cập nhật hoặc thêm mới tính năng.

3.1.4. Danh sách rủi ro

STT	Rủi ro tiềm ẩn	Mức độ ảnh hưởng	Biện pháp giảm thiểu
1	Môi trường kiểm thử chưa được thiết lập đúng cách	**Cao**	Kiểm tra kỹ cấu hình (configuration) trước mỗi lần build.
2	Dữ liệu dùng để kiểm thử thiếu hoặc chưa đầy đủ	**Trung bình**	Chuẩn bị và rà soát bộ dữ liệu mẫu (mock data) đầy đủ hơn.
3	Lỗi tích hợp cổng thanh toán Stripe	**Cao**	Kiểm tra lại cấu hình và sandbox của cổng thanh toán.
4	Các module Tìm Kiếm hoặc Lọc hoạt động chậm	**Trung bình**	Sử dụng bộ dữ liệu test có kích thước vừa đủ để tối ưu hiệu năng.
5	Giao diện hiển thị sai lệch giữa các trình duyệt/thiết bị	**Cao**	Thực hiện kiểm thử tương thích (Cross-browser/device testing) trên nhiều môi trường.

3.2. Hạng mục được kiểm thử

3.2.1. Chức năng

Tập trung kiểm tra các luồng nghiệp vụ cốt lõi của hệ thống thương mại điện tử GoShop để đảm bảo hoạt động đúng theo yêu cầu thiết kế.

Quản lý tài khoản (Auth): Đăng ký, Đăng nhập (Email/Password, Social Login nếu có), Quên mật khẩu, Đổi mật khẩu, Cập nhật thông tin cá nhân, Quản lý địa chỉ giao hàng.

Quản lý sản phẩm (Product): Hiển thị danh sách sản phẩm, Tìm kiếm, Lọc sản phẩm (theo giá, danh mục, thương hiệu), Xem chi tiết sản phẩm, Sản phẩm liên quan/gợi ý.

Giỏ hàng (Cart): Thêm sản phẩm, Cập nhật số lượng, Xóa sản phẩm, Tính toán tổng tiền tạm tính.

Đặt hàng và Thanh toán (Checkout): Chọn địa chỉ, Chọn phương thức vận chuyển, Chọn phương thức thanh toán, Áp dụng mã giảm giá, Hoàn tất đơn hàng.

Quản lý đơn hàng (Order): Xem lịch sử đơn hàng, Theo dõi trạng thái đơn hàng (Pending, Shipping, Delivered, Cancelled).

Admin Dashboard: Quản lý người dùng, CRUD sản phẩm, Quản lý danh mục, Xử lý đơn hàng.

3.2.2. Khả năng sử dụng

Đánh giá trải nghiệm người dùng trên giao diện web (HTML/CSS/JS) để đảm bảo tính dễ sử dụng và thân thiện.

Điều hướng (Navigation): Kiểm tra tính logic của menu, breadcrumb, và các liên kết chuyển trang. Người dùng có dễ dàng tìm thấy sản phẩm không?

Thông báo hệ thống: Kiểm tra tính rõ ràng của các thông báo lỗi (validation forms), thông báo thành công (thêm giỏ hàng, đặt hàng), và các tooltip hướng dẫn.

Trải nghiệm thao tác: Đánh giá số bước cần thiết để hoàn thành việc mua hàng (tối ưu hóa checkout flow).

Độ thẩm mỹ: Bố cục layout, font chữ, màu sắc có đồng nhất và dễ nhìn không.

3.2.3. Hiệu năng

Đánh giá khả năng chịu tải của hệ thống GoShop, đặc biệt là phía Backend (API).

Load Testing: Kiểm tra tốc độ tải trang chủ, trang chi tiết sản phẩm khi có 10, 50, 100 người dùng truy cập đồng thời.

Response Time: Đo thời gian phản hồi của các API quan trọng (Search, Add to Cart, Checkout). Mục tiêu < 2 giây.

Stress Testing: Tìm điểm giới hạn của hệ thống khi lượng truy cập tăng đột biến (ví dụ mô phỏng ngày sale).

3.2.4. Tương thích

Do dự án sử dụng TypeScript/JavaScript và HTML (Web application), cần kiểm thử trên nhiều môi trường khác nhau.

Trình duyệt (Browser): Chrome, Firefox, Safari, Edge (các phiên bản mới nhất và phiên bản cũ hơn 1-2 đời).

Hệ điều hành (OS): Windows, macOS, Linux (cho phía server/dev), iOS, Android (trên trình duyệt mobile).

Thiết bị: Laptop/Desktop (các độ phân giải màn hình khác nhau), Tablet, Mobile (Responsive Design).

3.3. Hạng mục không được kiểm thử

3.3.1. Các chức năng thuộc hệ thống bên thứ ba

Nhóm không thực hiện kiểm thử các chức năng xử lý nội bộ của cổng thanh toán bên thứ 3. Chỉ đảm bảo tính ổn định, hiệu năng khi kết nối với công thanh toán.

3.3.2. Kiểm thử bảo mật nâng cao

Không thực hiện kiểm thử xâm nhập (Penetration Testing) và các bài kiểm thử bảo mật chuyên sâu.

Nhóm chỉ thực hiện kiểm thử bảo mật ở mức cơ bản, tập trung vào kiểm soát truy cập và phân quyền người dùng.

3.3.3. Kiểm thử khả năng sử dụng chuyên sâu

Khảo sát người dùng thực tế (User Acceptance Survey): Không tổ chức các phiên trải nghiệm thực tế với nhóm người dùng mẫu (Beta testers) trong giai đoạn này (trừ khi có yêu cầu UAT riêng biệt).

Tiêu chuẩn tiếp cận (Accessibility Compliance): Không kiểm thử sự tuân thủ nghiêm ngặt các tiêu chuẩn WCAG (Web Content Accessibility Guidelines) dành cho người khuyết tật, trừ các yêu cầu cơ bản về độ tương phản và điều hướng bằng bàn phím.

3.4. Tiêu chí kiểm thử chấp nhận

3.4.1. Độ bao phủ kiểm thử

Tiêu chí chấp nhận kiểm thử – GoShop

Bao phủ kiểm thử ≥ 95% chức năng.

Hiệu năng tải trang ≤ 3 giây.

Mật khẩu mã hóa, không lộ dữ liệu người dùng.

Trưởng nhóm xác nhận trước chuyển sang UAT.

3.4.2. Tỷ lệ trường hợp kiêm thử đạt

Tỷ lệ test case pass ≥ 90%.

Không có lỗi Critical/High severity..

3.4.3. Số lượng trường hợp kiểm thử

Toàn bộ test case ở các cấp độ sau phải được thực thi 100%:

Kiểm thử đơn vị (Unit Test)

Kiểm thử tích hợp (Integration Test)

Kiểm thử hệ thống (System Test)

3.4.4. Số lượng lỗi

Lỗi Critical (Nghiêm trọng): 0 lỗi. Hệ thống không được phép dính lỗi gây crash, mất dữ liệu, hoặc chặn luồng thanh toán tại thời điểm bàn giao.

Lỗi Major (Lớn): Tối đa 02 lỗi. Các lỗi này phải có phương án khắc phục tạm thời (Workaround) được Product Owner chấp thuận và có kế hoạch sửa chữa (Hotfix) ngay sau khi release.

Lỗi Minor/Cosmetic (Nhỏ/Giao diện): Cho phép tồn tại một số lượng nhỏ nhưng không được ảnh hưởng đến trải nghiệm người dùng chính và phải được ghi nhận vào backlog để xử lý ở Sprint kế tiếp.

3.4.5. Mật độ kiểm thử đơn vị

Mật độ kiểm thử phải đạt tối thiểu 80 Unit Test Case trên 1000 dòng mã nguồn (80 UTC/KLOC) đối với các module xử lý logic nghiệp vụ (Backend Services, Utility Functions).

3.4.6. Độ bao phủ mã nguồn

Statement coverage ≥ 90%, Branch ≥ 95%.

100% các hàm xử lý thanh toán và tính toán giá trị đơn hàng phải được kiểm thử.

3.4.7. Quy trình CI/CD

Quy trình GitHub Actions CI/CD phải chạy hoàn tất với trạng thái thành công.

Không được tồn tại bất kỳ lỗi kiểm thử nào trong pipeline tự động.

3.5. Chiến lược kiểm thử

3.5.1. Phương pháp kiểm thử

Để đảm bảo tính chặt chẽ và chất lượng xuyên suốt vòng đời phần mềm, dự án tuân thủ mô hình **V-Model**. Tại đây, quy trình kiểm thử được tiến hành song song và đối xứng với các giai đoạn phát triển. Hệ thống được kiểm soát qua 4 cấp độ: Unit Test, Integration Test, System Test và Acceptance Test, với các Test Case được xây dựng bám sát yêu cầu chức năng và Use Case. Ngoài ra, quy trình CI/CD được thiết lập bằng **GitHub Actions** giúp tự động hóa toàn bộ chuỗi Build, Test và Deploy, nâng cao hiệu suất làm việc.

3.5.2. Loại kiểm thử

Activity	Vai trò thực hiện
Functionally Testing	Được thực hiện bởi nhóm kiểm thử trong giai đoạn Integration Testing hoặc System Testing để đáp ứng yêu cầu chức năng đã thống nhất của ứng dụng website GoShop. Các chức năng sau sẽ được kiểm tra: 1. Đăng Ký 2. Đăng Nhập 3. Giỏ Hàng 4. Thanh Toán 5. Danh Mục Sản Phẩm 6. Sản Phẩm Yêu Thích 7 Quản Lý Sản Phẩm 8 Kiểm Tra Toàn Bộ Đơn Hàng 9 Quản Lý Tài Khoản Người Dùng. Tất cả các tính năng được kiểm thử sẽ được mô tả chi tiết

	trong tài liệu về Test Scenario và Test Case .
Performance và Load	Được thực hiện trong giai đoạn System Testing nhằm đo lường hiệu suất hệ thống - tốc độ, khả năng chịu tải (load), và khối lượng dữ liệu (volume).
Code Testing	Được thực hiện trong giai đoạn Unit Testing nhằm đảm bảo từng phần mã (method, function) hoạt động đúng.

3.5.3. Cấp độ kiểm thử

Loại kiểm thử	**Mục đích**	**Kỹ thuật**	**Phương pháp**
Kiểm thử đơn vị	Kiểm tra tính đúng đắn của các hàm/phương thức	Hộp trắng	Kiểm thử tự động bằng JUnit và Mockito ở môi trường Development/CI
Kiểm thử tích hợp	Kiểm tra sự tương tác giữa các module chức năng	Hộp đen	Kiểm thử tự động bằng Spring Boot Test, MockMvc và H2 ở môi trường Development/CI

Kiểm thử hệ thống	Kiểm tra các luồng nghiệp vụ và giao diện	Hộp đen	Kiểm thử tự động bằng Selenium (UI, E2E) và K6 (hiệu năng) trong môi trường Development
Kiểm thử chấp nhận	Kiểm tra và nghiệm thu sản phẩm	Hộp đen	Kiểm thử thủ công ở môi trường Production

Type of Tests	**Stage of Test**			
	Unit	**Integration**	**System**	**Acceptance**
Function Test	X	X	X	X
User Interface test	X		X	
Performance Tests (Performance profiles of individual components)	X	X		
Load, Stress, Volume test			X	X
Security test	X		X	
Database test		X	X	
Adhoc test		X	X	

3.6. Kế hoạch thực thi kiểm thử

3.6.1. Nguồn nhân lực

Người thực hiện	Vai Trò	Trách nhiệm chính
Võ Văn Luân	Kiểm thử viên/Lập trình viên	Lập kế hoạch kiểm thử, phân bổ nguồn lực, kiểm soát tiến độ và chất lượng đầu ra.
Đỗ Tấn Lộc	Kiểm thử viên/Lập trình viên	Thiết kế Test Case, thực hiện kiểm thử các chức năng BRontend (Auth, Cart, Admin), log bug và verify bug.
Nguyễn Hưu Anh Khoa	Kiểm thử viên/Lập trình viên	Viết Unit Test , cấu hình pipeline CI/CD trên GitHub Actions, Chịu trách nhiệm review kết quả Test CI/CD.
Lê Duy Tín	Kiểm thử viên/Lập trình viên	Hỗ trợ làm rõ yêu cầu nghiệp vụ, thực hiện UAT để nghiệm thu sản phẩm.

3.7. Môi trường kiểm thử

3.7.1. Phần cứng

Mục	Thành phần	Mục đích

Development	Máy tính cá nhân	Lập trình, chạy kiểm thử đơn vị và kiểm thử tích hợp
Thiết bị	Desktop, Android, iOS	Kiểm thử giao diện và kiểm thử tương thích

3.7.2. Phần mềm

Mục	Thành phần	Mục đích
Development	VSCode, ReactJS, Local MySQL	Lập trình, chạy kiểm thử đơn vị và kiểm thử tích hợp
Trình duyệt	Chrome, Firefox	Kiểm thử giao diện và kiểm thử tương thích

3.7.3. Hạ tầng

Mục	Thành phần	Mục đích
Production	MySQL, BRontend và Backend deploy trên Railway	Triển khai phần mềm thực tế, thực hiện kiểm thử hệ thống và chấp nhận
CI/CD	GitHub Actions	Tự động hóa quy trình Build, Test và Deploy

3.8. Tài liệu bàn giao

Loại hạng mục	Định dạng	Chi tiết nội dung
Tài liệu dự án	Word (.docx)	Architecture Design & Database Design Test Plan & Bug Report Báo cáo tổng kết dự án
Tài liệu kiểm thử & Chi tiết	Excel (.xlsx)	Danh sách giao diện & Đặc tả Use Case Review Checklists Test Design, Test Case, Test Report
Mã nguồn & Tự động hóa	GitHub	Source Code Test Scripts (Unit, Integration, System) CI/CD Pipeline (.github/workflows/ci.yml , CD ở railway)

CHƯƠNG 4: PHƯƠNG PHÁP KIỂM THỬ

4.1. Giới thiệu

Chương này trình bày quy trình và kết quả của việc thiết kế các trường hợp kiểm thử (Test Case) chi tiết dựa trên chiến lược V-Model

4.2. Quy trình thiết kế kiểm thử theo V-Model

Quy trình kiểm thử của dự án được xây dựng dựa trên cấu trúc V-Model, một phương pháp luận nhấn mạnh tính đối xứng và mối quan hệ mật thiết giữa các giai đoạn phát triển và các cấp độ kiểm thử tương ứng. Thay vì kiểm thử chỉ diễn ra ở giai đoạn cuối, mô hình Chữ V cho phép nhóm thực hiện các hoạt động chuẩn bị kiểm thử song song với quá trình phân tích và thiết kế, giúp tối ưu hóa khả năng phát hiện lỗi sớm.

Cụ thể, sự tương hỗ trong quy trình được thể hiện qua các cấp độ đối ứng sau:

Thu thập yêu cầu (Requirement Gathering) & Kiểm thử chấp nhận (Acceptance Testing): Giai đoạn khởi đầu tập trung vào việc xác định nhu cầu của khách hàng. Đây là căn cứ duy nhất để xây dựng bộ kiểm thử chấp nhận (UAT), nhằm xác nhận liệu phần mềm có đáp ứng đúng mục tiêu kinh doanh và tiêu chí nghiệm thu của người dùng hay không.

Phân tích yêu cầu & Kiểm thử hệ thống (System Testing): Các yêu cầu nghiệp vụ chi tiết được chuyển hóa thành các kịch bản kiểm thử hệ thống (End-to-End). Giai đoạn này đảm bảo phần mềm vận hành ổn định trên môi trường thực tế, bao gồm cả giao diện (UI) và hiệu năng.
Thiết kế hệ thống & Kiểm thử tích hợp (Integration Testing): Cấu trúc kiến trúc và sự tương tác giữa các module được đối chứng thông qua các bài kiểm thử tích hợp, xác minh tính chính xác của luồng dữ liệu khi truyền tải giữa API, Backend và Database.

Thiết kế chi tiết & Kiểm thử đơn vị (Unit Testing): Các logic xử lý bên trong từng hàm (Function) được kiểm chứng trực tiếp bằng các bài kiểm thử đơn vị, đảm bảo mọi nhánh logic (Branch Coverage) đều hoạt động đúng theo đặc tả kỹ thuật.

Việc áp dụng V-Model không chỉ giúp nhóm duy trì tính Traceability (khả năng truy xuất nguồn gốc) – nơi mỗi yêu cầu đều có ít nhất một trường hợp kiểm thử tương ứng – mà còn đảm bảo chất lượng hệ thống được kiểm soát chặt chẽ từ những đơn vị mã nguồn nhỏ nhất đến toàn bộ luồng nghiệp vụ phức tạp.

4.2.1. Xác định Yêu cầu và Tiêu chí Nghiệm thu (Requirements & Acceptance)

Ở giai đoạn khởi đầu, nhóm tập trung vào việc làm rõ các luồng nghiệp vụ thực tế của hệ thống bán sách giấy trực tuyến. Nhóm ưu tiên

việc xác định các hành vi của người dùng và các quy tắc nghiệp vụ cần thiết để hệ thống vận hành trơn tru.

Thông qua quá trình phân tích luồng công việc (Workflow Analysis), nhóm đã định hình hệ thống dựa trên sự tương tác của hai đối tượng chính:

Khách hàng: Tập trung vào trải nghiệm tìm kiếm, lựa chọn sản phẩm, quản lý giỏ hàng và hoàn tất quy trình thanh toán một cách an toàn.

Quản trị viên (Admin): Tập trung vào khả năng kiểm soát dữ liệu đơn hàng, quản lý danh mục, tài khoản người dùng và khai thác các báo cáo thống kê lợi nhuận.

Hệ thống được phân rã thành 07 phân hệ chức năng cốt lõi: Authentication (Xác thực), Danh mục sản phẩm, Giỏ hàng, Quản lý đơn hàng, Thống kê đơn hàng, Quản lý tài khoản và Quy trình thanh toán.

Toàn bộ các yêu cầu này được chuyển hóa trực tiếp thành các Tiêu chí nghiệm thu. Đây là căn cứ thực tế để đối chiếu ở giai đoạn cuối của quy trình (Acceptance Testing). Việc xác định rõ tiêu chí ngay từ đầu giúp nhóm đảm bảo rằng mọi chức năng được phát triển đều bám sát nhu cầu thực tế của người dùng và có thể đo lường, xác minh độ thành công một cách khách quan.

4.2.2. Phân tích Hệ thống và Hoạch định Kiểm thử Hệ thống

Sau khi các tiêu chí nghiệm thu được xác lập, nhóm tiến hành giai đoạn Phân tích Hệ thống (System Analysis). Đây là bước chuyển tiếp quan trọng từ việc hiểu "người dùng cần gì" sang việc xác định "hệ thống phải vận hành như thế nào" một cách tổng quát nhất. Nhóm tập trung vào việc mô hình hóa các luồng nghiệp vụ liên hoàn (End-to-End Workflows) và cấu trúc tương tác giữa các thành phần.

Nội dung phân tích trọng tâm bao gồm:

1. Phân tích luồng nghiệp vụ (Business Flow Analysis): Nhóm sử dụng sơ đồ Activity Diagram để chi tiết hóa các lộ trình nghiệp vụ phức tạp. Việc phân tích này giúp nhận diện các điểm quyết định (Decision points) và các kịch bản ngoại lệ trong quá trình người dùng sử dụng ứng dụng.

2. Xác định các kịch bản sử dụng (Use Case Analysis): Thông qua sơ đồ Use Case, nhóm làm rõ mối quan hệ giữa các tác nhân (Khách hàng, Admin) và các dịch vụ mà hệ thống cung cấp, đảm bảo không có tính năng nào nằm ngoài phạm vi kiểm soát.

Kết quả của giai đoạn phân tích hệ thống là "đầu vào" duy nhất và trực tiếp cho việc xây dựng bộ kịch bản Kiểm thử hệ thống (System Testing). Đặc biệt, để tối ưu hóa hiệu suất kiểm tra, nhóm đã áp dụng phương pháp Kiểm thử tự động (Automation Testing) với công cụ Selenium WebDriver:

1. Mô phỏng hành vi người dùng (User Simulation): Các luồng nghiệp vụ sau khi được phân tích sẽ được chuyển hóa thành các đoạn mã lệnh Selenium. Driver sẽ điều khiển trình duyệt thực hiện các thao tác click, nhập liệu, điều hướng hoàn toàn giống với hành vi của một khách hàng thực tế.

2. Xác minh trạng thái cuối (End-state Verification): Dựa trên phân tích hệ thống, nhóm xác định các điểm kiểm chứng (Assertions). Selenium sẽ thực hiện so sánh các thành phần hiển thị thực tế trên giao diện với các trạng thái mong đợi đã được hoạch định trong giai đoạn phân tích.
3. Kiểm thử hồi quy tự động (Automated Regression Testing): Việc phân tích hệ thống bài bản cho phép nhóm xây dựng các kịch bản Selenium có tính bền vững, giúp nhanh chóng kiểm tra lại toàn bộ chức năng hệ thống mỗi khi có sự thay đổi về mã nguồn mà không cần thực hiện thủ công.

Việc kết hợp giữa phân tích hệ thống chi tiết và kiểm thử tự động bằng Selenium đảm bảo rằng phần mềm không chỉ hoạt động đúng về mặt kỹ thuật mà còn đáp ứng trọn vẹn trải nghiệm người dùng trên môi trường thực tế.

4.2.3. Thiết kế Kiến trúc và Kiểm thử Tích hợp

Giai đoạn thiết kế kiến trúc đóng vai trò then chốt trong việc định hình khung xương của hệ thống. Nhóm đã áp dụng mô hình Kiến trúc C4 (Context, Container, Component, Code) để phân rã hệ thống một cách có hệ thống, từ cái nhìn tổng quan về ngữ cảnh đến chi tiết các thành phần kỹ thuật. Mục tiêu trọng tâm là xác định rõ ràng sự phân tách trách nhiệm (Separation of Concerns), cấu trúc các tầng và đặc biệt là các giao diện giao tiếp (Interfaces) giữa các thành phần trong hệ thống.

Các bản thiết kế từ mức Container (Backend, BRontend, Database) và Component là cơ sở dữ liệu duy nhất để xây dựng kịch bản cho Kiểm thử tích hợp (Integration Testing). Giai đoạn này tập trung vào việc xác minh tính chính xác của các luồng dữ liệu truyền tải qua lại giữa các điểm giao thoa (Integration Points) đã được xác lập trong kiến trúc.

Để thực hiện kiểm thử tích hợp một cách hiệu quả và tự động hóa, nhóm đã thực hiện kiểm thử tích hợp mức API bằng cách dùng Jest kết hợp với Supertest để giả lập các HTTP request và kiểm tra toàn bộ luồng xử lý từ Controller → Service → Repository mà không cần khởi chạy server thực tế. Trong các test, ứng dụng Express được import/khởi tạo trong môi trường test và Supertest gửi request trực tiếp tới instance đó.

Về dữ liệu, nhóm sử dụng Prisma (với mock hoặc test DB được cấu hình trong jest.setup.js) — các test thực hiện seed dữ liệu trước khi chạy và dọn dẹp (ví dụ prisma.user.create, prisma.user.deleteMany) trong các hook (beforeEach/afterEach) để đảm bảo tính cô lập và không để lại side-effect giữa các ca test. Các SDK/ngoại vi (Stripe, Firebase, Cloudinary, v.v.) được mock trong môi trường test để tránh gọi dịch vụ thật và giữ tests nhanh, ổn định.

Ví dụ về kiểm thử tích hợp có trong dự án:

Tên	Mục tiêu	Đầu vào	Kết quả mong đợi	Giải thích
updateUser	Kiểm tra luồng API khi chủ tài khoản cập nhật profile — endpoint trả về thông tin user đã cập nhật và token.	Có user test trong DB: firebaseId = 'acc-user-1' email = 'acc@example.com' fullName = 'Old Name' role = 'USER', status = 'ACTIVE'	HTTP 200; response chứa user với fullName đã đổi sang "New Name"; response chứa token.	Test này là API- level integration test (tương đương MockMvc flow): mock auth middleware cung cấp uid, request đi qua Controller → Service

		.		→ (gọi Firebase mock) → Repository (Prisma) và trả về payload. Seed + cleanup đảm bảo tính cô lập giữa các test.
lockUser	Kiểm tra khi admin khóa tài khoản (status = HIDDEN) thì user đó không thể truy cập endpoint bảo vệ (ví dụ /cart).	Tạo user test: firebaseId = 'lock-user', email = 'lock@example.com', fullName = 'Lock', role = 'USER'.	Yêu cầu của user bị khóa trả 401 hoặc 403.	Test này kiểm tra chính sách truy cập dựa trên status của user: khi status = HIDDEN, middleware/ controller phải chặn truy cập vào các route bảo vệ. Mô hình test dùng seed/update trực tiếp vào Prisma để đưa ứng dụng về trạng thái mong muốn, rồi gửi request thực tế để xác minh toàn bộ

				luồng.

4.2.4. Thiết kế Chi tiết và Kiểm thử Đơn vị

Giai đoạn thiết kế chi tiết (còn gọi là thiết kế Module) là cấp độ thấp nhất trong nhánh trái của mô hình V-Model. Tại đây, nhóm tập trung vào việc định nghĩa cấu trúc nội tại của từng thành phần phần mềm. Công cụ chủ đạo được sử dụng là Sơ đồ lớp (Class Diagram), giúp mô tả tường tận các thuộc tính, phương thức, kiểu dữ liệu và các mối quan hệ (Dependency, Association, Inheritance) giữa các đối tượng trong mã nguồn.

Sự tương hỗ giữa Thiết kế Chi tiết và Kiểm thử Đơn vị:

Kiến trúc chi tiết của các lớp và phương thức là cơ sở trực tiếp để xây dựng bộ kịch bản Kiểm thử đơn vị (Unit Testing). Đây là quy trình kiểm thử hộp trắng (White-box testing) tập trung vào việc xác minh tính đúng đắn của các đơn vị mã nguồn nhỏ nhất trước khi chúng được tích hợp vào các phân hệ lớn hơn.

Chiến lược kiểm thử đơn vị của nhóm:

Công cụ chính: dùng Jest (ts-jest) cho TypeScript nhóm đã dùng jest.mock/jest.fn() để giả lập (mock) phụ thuộc.

Cô lập logic với mock: Mọi phụ thuộc bên ngoài (Prisma client, Firebase, Stripe, Cloudinary, v.v.) phải được mock trong unit test để cô lập hoàn toàn logic hàm/ lớp. Tránh gọi DB/HTTP thật trong unit tests — những trường hợp cần DB sẽ là integration test.

Tối ưu coverage có trọng tâm: Viết test để bao phủ các nhánh quan trọng (if/else), đường xử lý lỗi (try/catch) và các edge case. Sử dụng Jest + Istanbul (built-in coverage) để đo Statement/Branch/Function/Line coverage. Tuy nhiên ưu tiên test các luồng nghiệp vụ quan trọng, không chase % một cách mù quáng.

Việc thực hiện kiểm thử đơn vị một cách bài bản dựa trên thiết kế chi tiết không chỉ giúp nâng cao chất lượng mã nguồn mà còn tạo ra một

"mạng lưới an toàn", cho phép nhóm tự tin thực hiện các thay đổi hoặc tối ưu hóa mã (Refactoring) mà không làm phá vỡ các chức năng hiện có.

Tên	Mô tả	Đầu vào	Kết quả mong đợi
UT-CART-001	Trả về 401 khi không có uid trong request.	req = {}	HTTP 401 Unauthorized
UT-CAT-006	Tạo category mới thành công.	body = { name: 'NewCat' }, mock prisma.category.create => { id:3, name:'NewCat' }	HTTP 201

4.3. Kỹ thuật thiết kế kiểm thử

4.3.1. Kiểm thử hộp đen

Kỹ thuật phân vùng tương đương (Equivalence Partitioning):

Trong bộ test case thuộc các module Authentication và Quản lý tài khoản, dữ liệu đầu vào được phân chia thành các lớp hợp lệ và không hợp lệ để tối ưu hóa số lượng kịch bản kiểm thử mà vẫn đảm bảo độ bao phủ.

Đối với chức năng Đăng nhập:

Lớp hợp lệ: Email và mật khẩu chính xác, tài khoản đang hoạt động.

Lớp không hợp lệ: Sai mật khẩu, email chưa đăng ký, hoặc tài khoản đang bị khóa.

Đối với chức năng Đăng ký / Cập nhật thông tin:

Lớp hợp lệ: Email mới chưa tồn tại, số điện thoại đúng định dạng (10-11 số).

Lớp không hợp lệ: Email đã tồn tại trong hệ thống, số điện thoại chứa ký tự đặc biệt/chữ cái, hoặc bỏ trống các trường bắt buộc.

=> Minh chứng Test Case:

AUT-001, AUT-009: Đăng nhập/Đăng ký hợp lệ.

AUT-002, AUT-003, AUT-010, AUT-011: Đăng nhập/Đăng ký không hợp lệ (sai pass, trùng email).

ACC-008, ACC-009: Cập nhật số điện thoại chứa ký tự không hợp lệ.

Kỹ thuật Chuyển trạng thái (State Transition Testing)

Nhóm áp dụng kỹ thuật này để kiểm chứng tính đúng đắn của các đối tượng có vòng đời phức tạp. Mục tiêu là đảm bảo rằng các hành động (Actions) chỉ có thể thực hiện khi đối tượng ở đúng trạng thái cho phép, và các chuyển đổi trạng thái diễn ra nhất quán sau mỗi sự kiện.

A. Quản lý trạng thái Tài khoản người dùng

Trạng thái của tài khoản quyết định quyền truy cập vào hệ thống. Các kịch bản được thiết kế để kiểm tra sự thay đổi quyền hạn khi Admin thực hiện thao tác khóa/mở khóa.

Các trạng thái: Hoạt động (Active), Bị khóa (Locked).

Sự kiện chuyển đổi: Admin thực hiện lệnh Lock hoặc Unlock.

Mã Test Case	Trạng thái bắt đầu	Sự kiện (Hành động)	Trạng thái đích	Kết quả kỳ vọng
ACC-002	Active	Admin thực hiện khóa	Locked	User đăng nhập sẽ báo lỗi "Account is locked".
ACC-004	Locked	Admin thực hiện mở khóa	Active	User có thể đăng nhập lại bình thường.

ACC-003	Active (Đang login)	Admin thực hiện khóa	Locked (Log out)	User đang thao tác sẽ bị đẩy ra trang Login ngay lập tức.

B. Quản lý trạng thái Đơn hàng (Order Workflow)

Vòng đời của một đơn hàng đi qua nhiều giai đoạn. Kỹ thuật này giúp xác minh rằng đơn hàng không bị nhảy cóc trạng thái (ví dụ: từ Chờ xử lý không thể nhảy thẳng sang Đã giao mà không qua bước Vận chuyển).

Các trạng thái: Chờ xử lý (Pending), Đang giao (Shipping), Đã giao (Delivered), Đã hủy (Cancelled).

Sự kiện: Admin cập nhật trạng thái đơn hàng.

Mã Test Case	Trạng thái bắt đầu	Sự kiện	Trạng thái đích	Xác minh (Assertion)
ODM-003	Pending	Admin cập nhật trạng thái	Accepted/Rejected	User kiểm tra mục "Orders" thấy trạng thái mới cập nhật.
CKT-004	Pending (Đang checkout)	Admin xóa sản phẩm/Hết hàng	Failure	Không cho phép tạo đơn hàng thành công.

C. Quản lý trạng thái Giỏ hàng (Cart State)

Trạng thái giỏ hàng phụ thuộc vào sự biến động của dữ liệu thực tế từ hệ thống quản trị (Admin).

Các trạng thái: Trống (Empty), Có sản phẩm (Populated), Cập nhật (Updated), Bị gỡ bỏ (Removed).

Mã Test Case	Trạng thái bắt đầu	Sự kiện từ phía Admin	Trạng thái đích	Kết quả thực tế
CART-005	Populated (Giá cũ)	Admin thay đổi giá sản phẩm	Updated (Giá mới)	Tổng tiền trong giỏ tự động thay đổi khi User reBResh.
CART-006	Populated	Admin xóa sản phẩm khỏi kho	Removed	Sản phẩm tự động biến mất khỏi giỏ hàng của User.

Bảng quyết định (Decision Table Testing)

Kỹ thuật này được áp dụng để kiểm thử các tổ hợp điều kiện phức tạp trong việc Lọc sản phẩm (Danh mục sản phẩm) và Phân quyền người dùng (Authentication & Quản lý tài khoản).

Bước 1: Xác định các điều kiện (Conditions)

Dựa trên yêu cầu của module lọc sản phẩm, nhóm xác định 3 điều kiện chính ảnh hưởng đến kết quả hiển thị:

C1: Người dùng có nhập từ khóa tìm kiếm (Search keyword).

C2: Người dùng có chọn danh mục sản phẩm (Category).

C3: Người dùng có chọn tiêu chí sắp xếp theo giá (Sort by price).

Bước 2: Xác định các hành động (Actions)

Tương ứng với các tổ hợp điều kiện, hệ thống sẽ thực hiện các hành động:

A1: Hiển thị danh sách sản phẩm theo tên (Search result).

A2: Lọc danh sách sản phẩm theo danh mục (Filtered result).

A3: Sắp xếp lại thứ tự danh sách hiển thị (Sorted result).

Bước 3: Tính số luật (Rules)

Mỗi điều kiện (C1, C2, C3) có 2 giá trị là Đúng (Yes) hoặc Sai (No).

Số luật tính toán: 2 * 2 * 2 = 8 luật (Rules).

Bước 4: Bảng quyết định đầy đủ

Điều kiện / Quy tắc	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**
C1: Có từ khóa	N	N	N	N	Y	Y	Y	Y
C2: Có danh mục	N	N	Y	Y	N	N	Y	Y
C3: Có sắp xếp	N	Y	N	Y	N	Y	N	Y
Hành động (Action)								
A1: Hiện theo tên					X	X	X	X
A2: Lọc theo DM			X	X			X	X
A3: Sắp xếp giá		X		X		X		X

Bước 5: Rút gọn bảng quyết định (Focus vào các case tích hợp)

Nhóm tập trung vào việc rút gọn bảng để xử lý các kịch bản kết hợp quan trọng nhất (Integration logic), nơi các kỹ thuật như Phân vùng tương đương đơn lẻ không bao phủ hết:

Quy tắc rút gọn	Mô tả kịch bản	Kết quả kỳ vọng	Test Case ID
Luật kết hợp 1 (R7)	Tìm kiếm + Chọn danh mục	Hiện sản phẩm chứa từ khóa và thuộc đúng danh mục đã chọn.	**CAT-007**
Luật kết hợp 2 (R6)	Tìm kiếm + Sắp xếp giá	Hiện sản phẩm theo tên và sắp xếp theo thứ tự giá yêu cầu.	**CAT-008**
Luật kết hợp 3 (R8)	Tìm kiếm + Danh mục + Sắp xếp	Kết quả phải thỏa mãn đồng thời cả 3 bộ lọc: Tên, Danh mục và Thứ tự giá.	**CAT-009**

Phân tích giá trị biên (Boundary Value Analysis - BVA)

Kỹ thuật này dùng để kiểm tra các "điểm nhạy cảm" nơi dữ liệu chuyển giao giữa đúng và sai.

A. Độ dài dữ liệu (Module Authentication & Quản lý tài khoản)

Mật khẩu (AUT-014): Quy định tối thiểu 6 ký tự.

Biên: 5 ký tự (Báo lỗi), 6 ký tự (Hợp lệ).

Số điện thoại (ACC-010, ACC-011): Quy định từ 10 - 11 số.

Biên dưới: 9 số (Lỗi), 10 số (Đúng).

Biên trên: 11 số (Đúng), 12 số (Lỗi).

B. Số lượng và Tồn kho (Module Giỏ hàng & Đơn hàng)

Số lượng thêm vào giỏ (CART-002):

Giả sử tồn kho = 10.

Biên: Thêm 10 sản phẩm (Thành công), Thêm 11 sản phẩm (Báo lỗi không đủ hàng).

Tìm kiếm ID đơn hàng (ODM-005): Kiểm tra với ID nhỏ nhất (1) hoặc ID không tồn tại (số âm/0).

4.3.2. Kiểm thử hộp trắng

Kiểm thử hộp trắng được nhóm thực hiện nhằm mục đích xác minh cấu trúc nội tại và logic xử lý chi tiết của mã nguồn. Thay vì chỉ tập trung vào đầu vào và đầu ra, phương pháp này cho phép kiểm soát toàn bộ luồng điều khiển và luồng dữ liệu bên trong hệ thống.

Để đánh giá hiệu quả của quá trình này, nhóm áp dụng ba tiêu chí bao phủ cốt lõi làm thước đo:

1. Độ bao phủ câu lệnh (Statement Coverage): Đây là tiêu chí cơ bản nhất, đo lường tỷ lệ phần trăm các câu lệnh thực thi trong mã nguồn đã được chạy bởi bộ kiểm thử. Mục tiêu của nhóm là đảm bảo không có mã "chết" (dead code) và mọi dòng lệnh quan trọng đều được thực hiện ít nhất một lần để tránh lỗi cú pháp hoặc logic tiềm ẩn.

2. Độ bao phủ nhánh (Decision/Branch Coverage): Tiêu chí này tập trung vào các điểm quyết định trong mã (như if-else, switch-case, while). Nhóm thiết kế các kịch bản để đảm bảo mọi nhánh rẽ của luồng điều khiển đều được thực thi (cả trường hợp Đúng và Sai). Điều này cực kỳ quan trọng để đảm bảo hệ thống phản ứng chính xác trong mọi điều kiện rẽ nhánh nghiệp vụ.

3. Độ bao phủ điều kiện (Condition Coverage): Đi sâu hơn vào các biểu thức logic phức tạp, tiêu chí này yêu cầu mỗi điều kiện đơn lẻ trong một biểu thức điều kiện (ví dụ: các toán tử AND, OR trong câu lệnh if) đều phải được kiểm tra với cả hai giá trị True và False. Việc này giúp loại bỏ các lỗi sai sót khi kết hợp nhiều điều kiện logic với nhau.

Kết quả đo lường độ bao phủ thực tế (Coverage Report)

Nhóm sử dụng công cụ Jest tích hợp để thu thập các chỉ số thực tế sau:

Thành phần (File/Folder)	% Statement	% Branch	% Function	% Line	Đánh giá
Tổng cộng (All files)	**79.18**	**54.21**	**85.29**	**79.74**	**Khá**
config/stripe.ts	100	100	100	100	Tuyệt đối
controllers/auth.ts	87.5	66.66	100	86.11	Tốt
controllers/cart.ts	83.51	79.16	100	82.08	Tốt
controllers/category.ts	95.74	72.72	100	94.59	Rất tốt
controllers/checkout.ts	79.16	38.00	100	79.48	Trung bình
controllers/orders.ts	69.93	58.73	73.33	72.32	Cần cải thiện

controllers/products.ts	87.59	68.22	90.47	87.25	Tốt
controllers/users.ts	76.14	63.79	85.71	74.46	Khá
controllers/webhook.ts	66.00	27.39	66.66	65.11	Thấp
services/productsService.ts	92.00	76.00	85.71	95.45	Rất tốt
utils/dbMetrics.ts	69.38	42.42	71.42	81.57	Trung bình
utils/metrics.ts	86.95	12.50	0.00	86.95	Cần lưu ý
utils/processOrderAddress.ts	28.57	0.00	0.00	16.66	Rất thấp

Dựa trên chỉ số Jest phía trên, nhóm có các nhận xét chuyên môn sau:

A. Những điểm tích cực (Strengths)

Độ bao phủ hàm (Function Coverage) cao: Đạt 85.29%, cho thấy hầu hết các hàm nghiệp vụ chính đã được gọi và thực thi trong quá trình test.

Tầng Controller & Service ổn định: Các file quan trọng như auth.ts, cart.ts, productsService.ts đều có Statement Coverage trên 80%.

Điều này đảm bảo các logic nghiệp vụ lõi (Core Logic) đã được kiểm chứng kỹ lưỡng.

Cấu hình an toàn: File stripe.ts đạt 100%, đảm bảo các thiết lập kết nối thanh toán không có sai sót.

B. Những điểm cần cải thiện (Weaknesses)

Độ bao phủ nhánh (Branch Coverage) thấp: Chỉ đạt 54.21%. Đây là dấu hiệu cho thấy các câu lệnh if-else, switch-case hoặc các nhánh xử lý lỗi (try-catch) chưa được test hết. Đặc biệt là file webhook.ts (27%) và checkout.ts (38%).

Module Webhook & Utils: File webhook.ts và processOrderAddress.ts có chỉ số rất thấp. Điều này tiềm ẩn rủi ro lớn vì Webhook là nơi nhận dữ liệu từ bên thứ ba (Stripe), nếu không test kỹ các nhánh lỗi sẽ dễ dẫn đến sập hệ thống khi dữ liệu không chuẩn.

Logic ảo (Dead Code): File metrics.ts có 0% Function Coverage dù Statement cao, có thể đây là các đoạn code khai báo nhưng chưa bao giờ được gọi tới.

4.4. Kiểm thử tự động và kiểm thử thủ công

4.4.1. Kiểm thử thủ công (Manual Testing)

Trong giai đoạn kiểm thử chấp nhận, nhóm triển khai phương pháp kiểm thử thủ công như một bước thẩm định cuối cùng nhằm đánh giá hệ thống dưới góc độ của người dùng cuối (End-user). Khác với các bài kiểm tra tự động vốn tập trung vào tính đúng đắn của mã nguồn, kiểm thử thủ công tập trung vào trải nghiệm thực tế, luồng nghiệp vụ phức tạp và các yếu tố cảm quan mà máy móc không thể thay thế.

Quy trình và phương pháp thực thi:

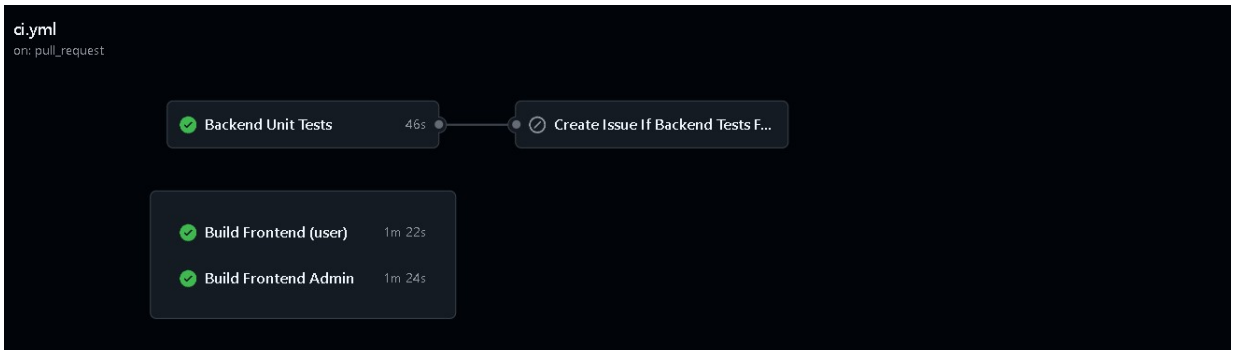
Xây dựng kịch bản kiểm thử (Test Scenario): Nhóm thiết kế các bộ kịch bản dựa trên các luồng nghiệp vụ thực tế (Business Workflows) thay vì các tính năng rời rạc. Điều này đảm bảo hệ thống vận hành trơn tru từ khi người dùng truy cập lần đầu cho đến khi hoàn tất quy trình thanh toán và nhận đơn hàng.

Kiểm thử dựa trên Checklist: Kiểm thử thủ công sẽ được thực hiện dựa trên Test Case và Review Checklist đã thiết kế, kết quả kiểm thử sẽ được ghi nhận và đối chiếu với kết quả mong đợi trước khi tiến hành nghiệm thu hệ thống.

4.4.2. Kiểm thử tự động (Automation Testing)

Dự án triển khai kiểm thử tự động thông qua việc tích hợp quy trình CI/CD trên nền tảng GitHub Actions. Giải pháp này giúp tối ưu hóa chu kỳ phát triển phần mềm bằng cách tự động hóa hoàn toàn các khâu từ xây dựng (build), kiểm thử đến triển khai hệ thống.

Cơ chế vận hành: Mỗi khi có mã nguồn mới được đẩy (push) lên kho lưu trữ GitHub hoặc thông qua các yêu cầu kéo (Pull Request), pipeline CI/CD sẽ được kích hoạt tự động. Trong dự án của nhóm có thiết lập luật bảo vệ nhánh (Branch protection rules) nên sẽ không cho phép đẩy mã nguồn mới trực diện lên nhánh main mà phải thông qua một nhánh khác rồi merge pull request, khi mà status check hết thì mới cho merge vào main không thì sẽ không cho merge vào main. Hệ thống tiến hành thiết lập môi trường, xây dựng dự án và thực thi toàn bộ bộ kịch bản kiểm thử đã thiết lập. Việc này đảm bảo mọi thay đổi về mã nguồn đều được kiểm chứng ngay lập tức, giúp phát hiện sớm các lỗi xung đột và duy trì tính ổn định liên tục cho hệ thống.



Đây là Pipeline CI của Github Actions

Nếu job Backend Unit Tests thất bại, pipeline sẽ tự động tạo Issue trên GitHub để ghi nhận lỗi phát sinh, hỗ trợ nhóm phát triển theo dõi và xử lý kịp thời.

CHƯƠNG 5: BÁO CÁO VÀ KẾT LUẬN

5.1. Tổng quan về quá trình kiểm thử (Testing Overview)

Sau khi hoàn tất giai đoạn thiết kế kiểm thử chi tiết tại Chương 4, nhóm đã tiến hành thực thi kiểm thử cho hệ thống GoShop theo đúng lộ trình và kế hoạch đã đề ra. Quá trình kiểm thử được triển khai, tuân thủ nghiêm ngặt mô hình V-Model với các cấp độ từ thấp đến cao nhằm đảm bảo tính toàn vẹn của phần mềm.

Các giai đoạn kiểm thử được thực hiện bao gồm:

Kiểm thử đơn vị (Unit Testing): Xác minh tính đúng đắn của các hàm và phương thức xử lý logic nội bộ.

Kiểm thử tích hợp (Integration Testing): Đảm bảo sự tương tác mượt mà giữa các module và các dịch vụ bên thứ ba (như Stripe Payment).

Kiểm thử hệ thống (System Testing): Đánh giá toàn bộ chức năng và hiệu năng của hệ thống trên môi trường thực tế.

Kiểm thử chấp nhận (Acceptance Testing): Kiểm chứng mức độ đáp ứng của hệ thống so với các yêu cầu nghiệp vụ từ góc độ người dùng.

Trong suốt quá trình này, nhóm tập trung tối đa vào các luồng nghiệp vụ cốt lõi như quản lý giỏ hàng, quy trình thanh toán và các tính năng quản trị hệ thống. Mỗi kịch bản kiểm thử (Test Case) đều được thực thi chi tiết, cẩn thận; mọi lỗi phát sinh (Defects) đều được phân loại, ghi lại đầy đủ và đối chiếu với kết quả mong đợi trong hồ sơ bàn giao. Đây là cơ sở quan trọng để nhóm đánh giá chất lượng phần mềm trước khi tiến hành nghiệm thu và vận hành chính thức.

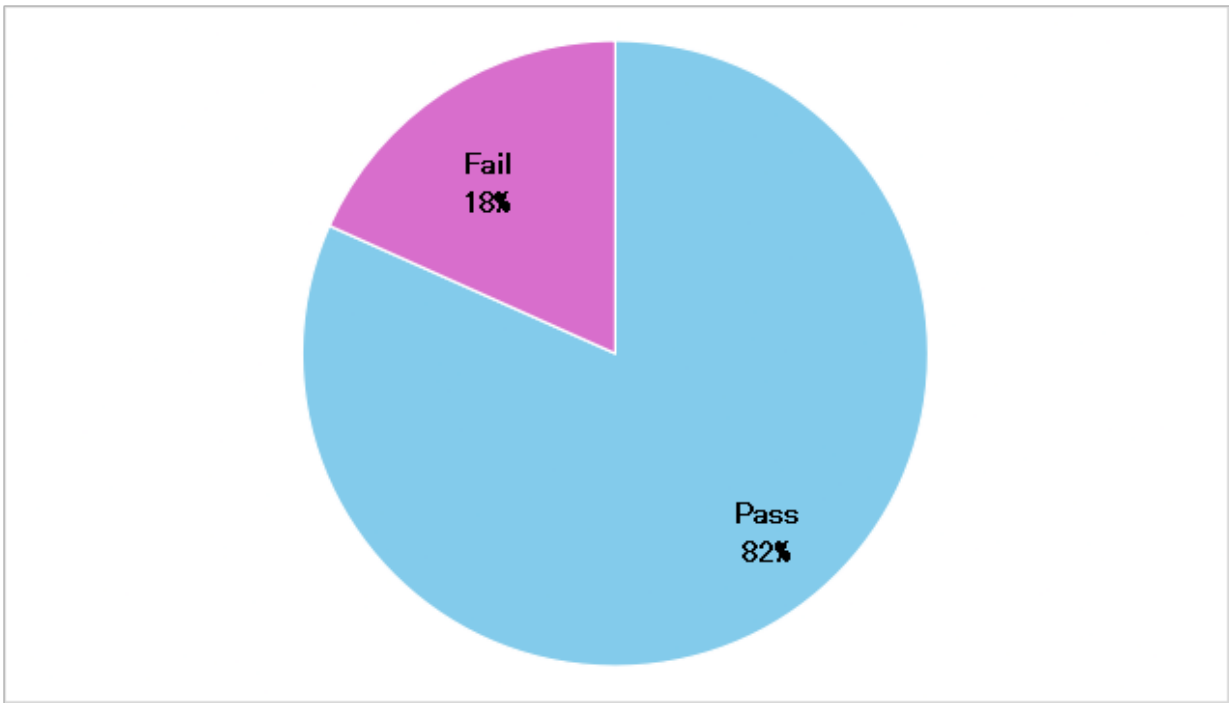
5.2. Báo cáo trường hợp kiểm thử

Các trường hợp kiểm thử này được phân loại thành hai nhóm Pass (kiểm tra hành vi đúng với mong đợi của hệ thống) và Fail (ngoại lệ và điều kiện không hợp lệ).

Bộ kiểm thử được xây dựng cho 7 module chức năng chính của hệ thống

STT	Tên Module	Mã Module	Mô tả chi tiết (Description)
1	**Xác thực người dùng**	AUT	Kiểm tra quy trình Đăng nhập (thường & Google), Đăng ký tài khoản mới và Đăng xuất. Đảm bảo tính bảo mật và đúng đắn của thông tin tài khoản.
2	**Danh mục sản phẩm**	CAT	Kiểm tra khả năng hiển thị danh sách, tìm kiếm theo từ khóa và các bộ lọc thông minh (theo danh mục, khoảng giá tăng/giảm).
3	**Quản lý Giỏ hàng**	CART	Kiểm tra các thao tác thêm/sửa/xóa sản phẩm, tính toán tổng tiền và khả năng đồng bộ dữ liệu tồn kho thực tế.
4	**Quản lý Đơn hàng**	ODM	Kiểm tra luồng xem danh sách đơn hàng của User và quyền quản trị của Admin (tìm kiếm đơn, lọc theo trạng thái/ngày tháng).
5	**Quy trình Thanh toán**	CKT	Kiểm tra tích hợp cổng thanh toán Stripe, xác thực thẻ test và luồng trừ tồn kho sau khi giao dịch thành công.
6	**Quản lý Tài khoản**	ACC	Kiểm tra tính năng cập nhật thông tin cá nhân (Profile) và các quyền quản trị nâng cao như Khóa/Mở khóa người dùng hoặc Đổi vai trò (Role).
7	**Thống kê đơn hàng**	STAT	Kiểm tra khả năng tổng hợp dữ liệu của hệ thống dưới dạng biểu đồ (lợi nhuận, top sản phẩm, tỷ lệ đơn hàng) và xuất báo cáo CSV.

Nhóm đã thực thi tổng cộng 71 kịch bản kiểm thử bao phủ 100% các chức năng nghiệp vụ trọng yếu của hệ thống GoShop. Kết quả ghi nhận 58 kịch bản Pass (81.69%) và 13 kịch bản Fail (18.31%). Mặc dù các chức năng phi chức năng (hiệu năng, bảo mật chuyên sâu) chưa nằm trong phạm vi kiểm thử chi tiết lần này, nhưng bộ kịch bản đã đảm bảo tính ổn định cho luồng vận hành chính.



5.3. Báo cáo lỗi

Trong quá trình thực thi 71 kịch bản kiểm thử cho hệ thống GoShop, nhóm đã ghi nhận tổng cộng 13 lỗi (Defects). Các lỗi này đã được phân loại theo mức độ nghiêm trọng và module để đội ngũ phát triển có kế hoạch khắc phục phù hợp.

5.3.1. Thống kê lỗi theo mức độ nghiêm trọng

Nhóm phân loại lỗi dựa trên tầm ảnh hưởng đến quy trình nghiệp vụ của người dùng:

Critical (Nghiêm trọng): Lỗi làm gián đoạn luồng nghiệp vụ chính, ảnh hưởng đến doanh thu hoặc tính nhất quán dữ liệu.

Major (Lớn): Lỗi chức năng không hoạt động đúng như mong đợi nhưng có cách xử lý tạm thời.

Minor (Nhỏ): Các lỗi về giao diện, thông báo hoặc trải nghiệm người dùng không ảnh hưởng đến logic hệ thống.

Mức độ	Số lượng	Tỷ lệ (%)	Đặc điểm
Critical	4	31%	Lỗi thanh toán, đăng nhập Google, tạo đơn hàng khi hết kho.
Major	6	46%	Lỗi giá trị biên SĐT, tìm kiếm đơn hàng, điều hướng (Redirect).
Minor	3	23%	Lỗi hiển thị đồng bộ giỏ hàng, cập nhật UI sau khi xóa.

5.3.2. Danh sách lỗi chi tiết (Defect Log)

Dưới đây là bảng thống kê toàn bộ 13 lỗi đã được phát hiện trong quá trình thực thi kiểm thử hệ thống GoShop. Các lỗi này đại diện cho những lỗ hổng logic và dữ liệu cần được khắc phục trước khi triển khai chính thức.

STT	**Mã TC**	**Module**	**Mô tả kịch bản lỗi**	**Kết quả thực tế (Actual Result)**	**Mức độ**
1	AUT-004	Auth	Đăng nhập Google lần đầu.	Không tự động tạo record User mới trong Database.	Critical

2	AUT-006	Auth	Điều hướng sau đăng nhập (Redirect).	Hệ thống đưa về trang chủ thay vì trang sản phẩm trước đó.	Major
3	AUT-008	Auth	Quên mật khẩu (Chức năng chưa hoàn thiện).	Không gửi được mail reset password hoặc lỗi token.	Major
4	AUT-016	Auth	Bảo mật phiên đăng xuất.	Nhấn back sau khi logout vẫn còn nhìn thấy dữ liệu cũ.	Major
5	CART-006	Giỏ hàng	Đồng bộ khi Admin xóa sản phẩm.	Sản phẩm vẫn tồn tại trong giỏ hàng User dù đã bị xóa khỏi hệ thống.	Minor
6	ODM-005	Đơn hàng	Tìm kiếm đơn hàng theo mã (ID).	Nhập ID chính xác nhưng hệ thống báo không tìm thấy đơn.	Major
7	STAT-008	Thống kê	Xuất dữ liệu báo cáo (Export).	File CSV xuất ra bị lỗi font hiển thị tiếng Việt (UTF-8).	Minor
8	ACC-010	Tài khoản	Kiểm tra biên độ dài SĐT (Biên trên).	Cho phép lưu số điện thoại dài 12 chữ số.	Major
9	ACC-011	Tài khoản	Kiểm tra biên độ dài SĐT (Biên dưới).	Cho phép lưu số điện thoại chỉ có 8 chữ số.	Major

10	ACC-013	Tài khoản	Thay đổi ảnh đại diện (Avatar).	Lỗi upload khi kích thước file ảnh vượt quá 2MB (chưa chặn).	Minor
11	CKT-004	Thanh toán	Kiểm tra tồn kho thời gian thực.	Cho phép thanh toán thành công khi Admin vừa set tồn kho về 0.	Critical
12	CKT-007	Thanh toán	Xử lý lỗi từ cổng Stripe.	Giao diện bị treo khi người dùng nhập sai mã CVC quá nhiều lần.	Critical
13	CKT-008	Thanh toán	Áp dụng mã giảm giá (Discount).	Mã giảm giá không được trừ trực tiếp vào tổng tiền thanh toán.	Critical

5.3.3. Phân tích nguyên nhân và Phương pháp xử lý lỗi chi tiết

Sau khi ghi nhận 13 lỗi trên, nhóm đã tiến hành họp phân tích kỹ thuật để tìm ra nguyên nhân gốc rễ (Root Cause Analysis) và thiết lập quy trình xử lý triệt để.

A. Phân tích nguyên nhân gốc rễ

Nhóm xác định các lỗi phát sinh chủ yếu đến từ ba nguyên nhân kỹ thuật chính:

Lỗi bất đồng bộ dữ liệu (Concurrency Issues): Điển hình là lỗi CKT-004 và CART-006. Hệ thống chưa kiểm tra lại trạng thái cuối cùng của dữ liệu (Final State) tại thời điểm thực hiện hành động. Điều này dẫn đến việc người dùng thao tác trên dữ liệu cũ (Stale Data) đã bị Admin thay đổi.

Thiếu hụt cơ chế Validation đa tầng: Các lỗi như ACC-010, ACC-011 xảy ra do nhóm quá chú trọng vào UI mà thiếu các ràng buộc (Constraints) chặt chẽ tại tầng DTO (Data Transfer Object) ở Backend. Việc thiếu kiểm soát maxLength và minLength dẫn đến dữ liệu rác được lưu vào Database.
Xử lý Logic ngoại lệ (Exception Handling) chưa bao quát: Lỗi AUT-004 (Google Login) và CKT-007 (Stripe) cho thấy các kịch bản lỗi từ bên thứ ba (Third-party API) chưa được bắt lỗi (catch) và xử lý một cách thân thiện, dẫn đến hệ thống bị treo hoặc không phản hồi đúng trạng thái.
B. Phương pháp và Quy trình xử lý lỗi (Defect Resolution Workflow)
Để giải quyết các vấn đề này, nhóm áp dụng quy trình 5 bước nghiêm ngặt:
Phân loại và Ưu tiên (Triage): Các lỗi Critical (Thanh toán, Đăng nhập) được đẩy lên mức ưu tiên cao nhất (P1), yêu cầu xử lý ngay trong 24h. Các lỗi Minor được xếp vào nhóm cải thiện UI (P3).
Cô lập và Tái hiện (Isolation): Lập trình viên sử dụng dữ liệu từ mục "Test Data" trong Test Case để tái hiện lại lỗi trên môi trường Local. Sử dụng Debugger để theo dõi luồng dữ liệu từ Controller xuống Repository.
Khắc phục lỗi kỹ thuật (Fixing):
Đối với lỗi tồn kho: Triển khai cơ chế Database Locking kết hợp xác thực lại tồn kho ngay tại thời điểm khởi tạo giao dịch Stripe để ngăn chặn tình trạng "Race Condition".
Đối với lỗi Validation: Áp dụng mô hình Defense in Depth bằng cách sử dụng thư viện class-validator tại Backend kết hợp đồng bộ hóa Regex tại BRontend.
Kiểm thử xác nhận (Confirmation Testing): Tester sau khi nhận bản build đã sửa sẽ thực hiện chạy lại đúng Test Case đã Fail. Nếu trạng thái là Pass, lỗi sẽ được chuyển sang "Closed".
Kiểm thử hồi quy (Regression Testing): Đây là bước cực kỳ quan trọng. Nhóm sẽ thực hiện chạy lại bộ Smoke Test gồm 15 kịch bản quan trọng

nhất để đảm bảo việc sửa lỗi "Thanh toán" không làm hỏng tính năng "Giỏ hàng" hay "Đơn hàng".

C. Đánh giá rủi ro tồn đọng

Mặc dù nhóm đã có phương pháp xử lý, nhưng vẫn còn rủi ro về độ trễ mạng khi giao tiếp với API Stripe hoặc Google OAuth. Nhóm đề xuất trong tương lai sẽ tích hợp thêm hệ thống Logging & Monitoring (như Sentry) để phát hiện và cảnh báo lỗi ngay lập tức khi hệ thống đi vào vận hành thực tế, giúp giảm thiểu thời gian gián đoạn dịch vụ (Downtime).

5.4. Kết luận chung (General Conclusion)

Trải qua quá trình kiểm thử toàn diện từ cấp độ đơn vị đến hệ thống, nhóm đã có cái nhìn sâu sắc và khách quan về chất lượng hiện tại của hệ thống GoShop. Dựa trên các con số thống kê và phân tích tại Chương 5, nhóm đưa ra các kết luận sau:

5.4.1. Về mức độ hoàn thiện của hệ thống

Hệ thống đã đạt được độ bao phủ kiểm thử (Test Coverage) rất cao, lên đến 100% các chức năng nghiệp vụ trọng yếu. Tỷ lệ 81.7% kịch bản đạt (Pass) cho thấy GoShop đã hình thành được một khung xương vững chắc, đáp ứng tốt các nhu cầu cơ bản của một nền tảng thương mại điện tử từ tìm kiếm, quản lý giỏ hàng đến thanh toán trực tuyến. Sự ổn định của các module Danh mục sản phẩm (CAT) và Thống kê (STAT) là điểm sáng, minh chứng cho việc xử lý dữ liệu và hiển thị đã đi vào quỹ đạo.

5.4.2. Về các thách thức tồn đọng

Mặc dù đạt kết quả khả quan, hệ thống vẫn tồn tại 13 lỗi, trong đó có 4 lỗi ở mức độ Critical. Những lỗi này tập trung chủ yếu vào các kịch bản bất đồng bộ dữ liệu và tích hợp API bên thứ ba. Điều này phản ánh một thực tế rằng trong môi trường mạng thực tế với độ trễ và các thao tác đồng thời từ nhiều người dùng, hệ thống cần có những cơ chế bảo vệ dữ liệu mạnh mẽ hơn (như Locking Database hoặc Re-validation tầng trung gian).

5.4.3. Đánh giá khả năng triển khai

Dựa trên các tiêu chí nghiệm thu (Acceptance Criteria):

Về mặt chức năng: Hệ thống Sẵn sàng (Ready) cho giai đoạn thử nghiệm hạn chế (Beta Testing).

Về mặt an toàn dữ liệu: Hệ thống cần Khắc phục ngay (Fix required) các lỗi liên quan đến tồn kho và xác thực Google trước khi triển khai thực tế rộng rãi.

5.4.4. Hướng phát triển và Hoàn thiện

Trong thời gian tới, nhóm sẽ tập trung vào các kế hoạch sau để tối ưu hóa GoShop:

Xử lý triệt để danh sách lỗi (Defect Log): Đặc biệt ưu tiên luồng Checkout và đồng bộ trạng thái đơn hàng.

Mở rộng phạm vi kiểm thử: Tăng cường các bài kiểm thử phi chức năng (Non-functional testing) như kiểm thử bảo mật (Security Testing) để chống các lỗi SQL Injection và kiểm thử hiệu năng (Performance Testing) với cường độ cao hơn bằng kịch bản k6 đã xây dựng.