

KHOA CÔNG NGHỆ THÔNG TIN

=====***=====



BÁO CÁO BÀI TẬP LỚN HỌC PHẦN: KIỂM THỦ PHẦN MỀM

NGHIÊN CỨU KỸ THUẬT KIỂM FRONTEND, BACKEND CHO HỆ THỐNG WEBSITE BÁN QUẦN ÁO

GVHD: Ts. Hoàng Quang Huy

Nhóm: 16

Lớp: 2024IT6084007

Sinh viên: Đỗ Trọng Hoàng - 2022604369

Nguyễn Huy Nhật - 2022605668

Nguyễn Quốc Thái - 2022606184

Lê Anh Tạo - 2022605301

Hà Nội – Năm 2024

MỤC LỤC

MỤC LỤC	2
PHẦN 1. MỞ ĐẦU	4
PHẦN 2. KẾT QUẢ NGHIÊN CỨU	5
2.1. CHƯƠNG 1: TỔNG QUAN.....	5
2.1.1. Khái niệm kiểm thử:	5
2.1.2. Mục tiêu của kiểm thử.	5
2.1.3. Tâm quan trọng của kiểm thử	5
2.1.4. Các giai đoạn kiểm thử	6
2.1.5. Các kĩ thuật kiểm thử.....	6
2.2. CHƯƠNG 2: TÌM HIỂU CÔNG CỤ KIỂM THỬ PHẦN MỀM	8
2.2.1. Giới thiệu công cụ.....	8
2.2.2. Đặc điểm	9
2.2.3. Cài đặt và sử dụng công cụ	12
2.3. CHƯƠNG 3: GIỚI THIỆU WEB SITE BÁN QUẦN ÁO.....	15
2.3.1. Giới thiệu	15
2.3.2. Tài liệu đặc tả yêu cầu phần mềm	21
2.4. Chương 4 - Kiểm thử phần mềm.....	23
2.4.1. Lập kế hoạch kiểm thử.....	23
2.4.2. Đỗ Trọng Hoàng – Thêm sản phẩm vào giỏ hàng.....	25
2.4.3. Lê Anh Tạo - Xóa sản phẩm.....	29
2.4.4. Nguyễn Huy Nhật - Chính sửa sản phẩm	34
2.4.5. Nguyễn Quốc Thái - Đánh giá sản phẩm.....	39
2.4.6. Lê Anh Tạo – Thêm sản phẩm vào giỏ hàng.....	43
2.4.7. Đỗ Trọng Hoàng – Thêm sản phẩm mới	52
2.4.8. Nguyễn Huy Nhật – Đăng ký	61
2.4.9. Nguyễn Quốc Thái – Đăng nhập	66
2.4.10. Báo cáo kiểm thử	72

PHẦN 3. KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM.....	73
3.1. Kiến thức và kỹ năng.....	73
3.2. Chuẩn đầu ra đạt được	74
3.3. Bài học kinh nghiệm	74
3.4. Hướng phát triển.....	74
Danh mục hình ảnh.....	75
Danh mục bảng biểu.....	79
TÀI LIỆU THAM KHẢO	80

PHẦN 1. MỞ ĐẦU

Trong thời đại công nghệ số hiện nay, các hệ thống thương mại điện tử ngày càng phát triển mạnh mẽ và đóng vai trò quan trọng trong việc kết nối doanh nghiệp với người tiêu dùng. Đặc biệt, các website bán hàng trực tuyến đã trở thành công cụ không thể thiếu, giúp doanh nghiệp mở rộng thị trường và nâng cao trải nghiệm mua sắm của khách hàng. Tuy nhiên, để đảm bảo một hệ thống website hoạt động hiệu quả, ổn định và đáp ứng được nhu cầu người dùng, việc kiểm thử phần mềm, bao gồm kiểm thử Frontend và Backend, là yếu tố then chốt không thể bỏ qua.

Đề tài "Nghiên cứu kỹ thuật kiểm thử Frontend, Backend cho hệ thống website bán quần áo" được thực hiện nhằm nghiên cứu và áp dụng các kỹ thuật kiểm thử phần mềm vào một hệ thống thực tế. Trong đó, Frontend tập trung vào các yếu tố giao diện, chức năng và trải nghiệm người dùng, còn Backend liên quan đến cơ sở dữ liệu, logic nghiệp vụ và xử lý máy chủ. Thông qua đề tài này, chúng em mong muốn làm rõ tầm quan trọng của từng kỹ thuật kiểm thử, từ đó đảm bảo chất lượng và hiệu suất cho hệ thống website bán quần áo.

Với kiến thức đã được học như kiểm thử hộp trắng, kiểm thử hộp đen, kiểm thử API, kiểm thử tương thích và cấu hình, chúng em đã phần nào hiểu được cách kiểm thử một hệ thống. Để có thể thực hiện tốt đề tài này, chúng em đã tìm hiểu thêm về các công cụ phục vụ cho việc kiểm thử và quan trọng nhất là kỹ năng làm việc nhóm.

Thông qua việc thực hiện đề tài này, chúng em đã tích lũy được nhiều kiến thức thực tiễn về kỹ thuật kiểm thử phần mềm, đặc biệt là trong môi trường phát triển và vận hành website thương mại điện tử. Ngoài ra, việc áp dụng các kỹ thuật kiểm thử giúp chúng em rèn luyện kỹ năng phân tích, đánh giá và xử lý vấn đề một cách hệ thống. Chúng em hy vọng kết quả nghiên cứu sẽ là tài liệu tham khảo hữu ích cho những ai quan tâm đến lĩnh vực kiểm thử và phát triển hệ thống thương mại điện tử.

PHẦN 2. KẾT QUẢ NGHIÊN CỨU

2.1. CHƯƠNG 1: TỔNG QUAN

2.1.1. Khái niệm kiểm thử:

- Theo Glenford Myers: Kiểm thử là quá trình vận hành chương trình để tìm ra lỗi

- Theo IEEE, Kiểm thử là:

+ Là quá trình vận hành hệ thống hoặc thành phần dưới những điều kiện xác định, quan sát hoặc ghi nhận kết quả và đưa ra đánh giá về hệ thống hoặc thành phần đó.

+ Là quá trình phân tích phần mềm để tìm ra sự khác biệt giữa điều kiện thực tế và điều kiện yêu cầu và dựa vào điểm khác biệt đó để đánh giá tính năng phần mềm

2.2.2. Mục tiêu của kiểm thử.

- Tìm ra được càng nhiều lỗi càng tốt trong điều kiện về thời gian đã định và nguồn lực sẵn có - Chứng minh rằng sản phẩm phần mềm phù hợp với các đặc tả của nó.

- Xác thực chất lượng kiểm thử phần mềm đã dùng chi phí và hoàn thành với nỗ lực tối thiểu.

- Thiết kế tài liệu kiểm thử một cách có hệ thống và thực hiện nó sao cho có hiệu quả, tiết kiệm được thời gian công sức.

2.2.3. Tầm quan trọng của kiểm thử

Những người phát triển phần mềm cho rằng:

- Kiểm thử chỉ để chứng minh chương trình không có lỗi
- Kiểm thử chỉ ra rằng chương trình đã thực hiện đúng các chức năng trong đặc tả.

- Kiểm thử để tìm ra lỗi và sửa chữa các lỗi đó nhằm tăng độ tin cậy cho phần mềm.

2.2.4. Các giai đoạn kiểm thử

Quy trình kiểm thử tổng quát có 5 giai đoạn:

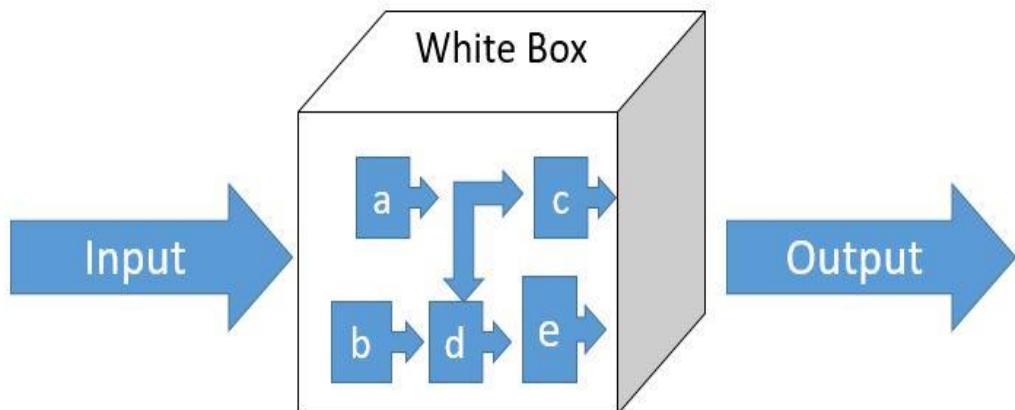
- Lập kế hoạch và kiểm soát kiểm thử
- Phân tích và thiết kế kiểm thử.
- Triển khai và thực hiện kiểm thử.
- Đánh giá tiêu chí kết thúc kiểm thử và lập báo cáo.
- Kết thúc kiểm thử.



Hình 2.1.1: Các giai đoạn kiểm thử

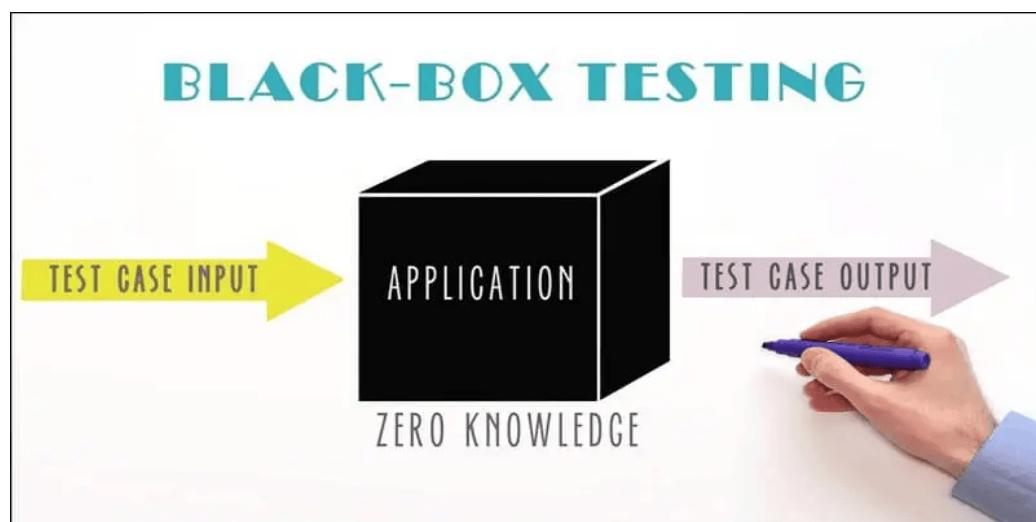
2.2.5. Các kỹ thuật kiểm thử

- Kiểm thử hộp trắng là các kỹ thuật tập trung vào phân tích cấu trúc/thiết kế bên trong của hệ thống phần mềm, các cấu trúc dữ liệu được sử dụng, cấu trúc mã và việc vận hành thay vì chỉ tập trung vào chức năng như trong kiểm thử hộp đen.



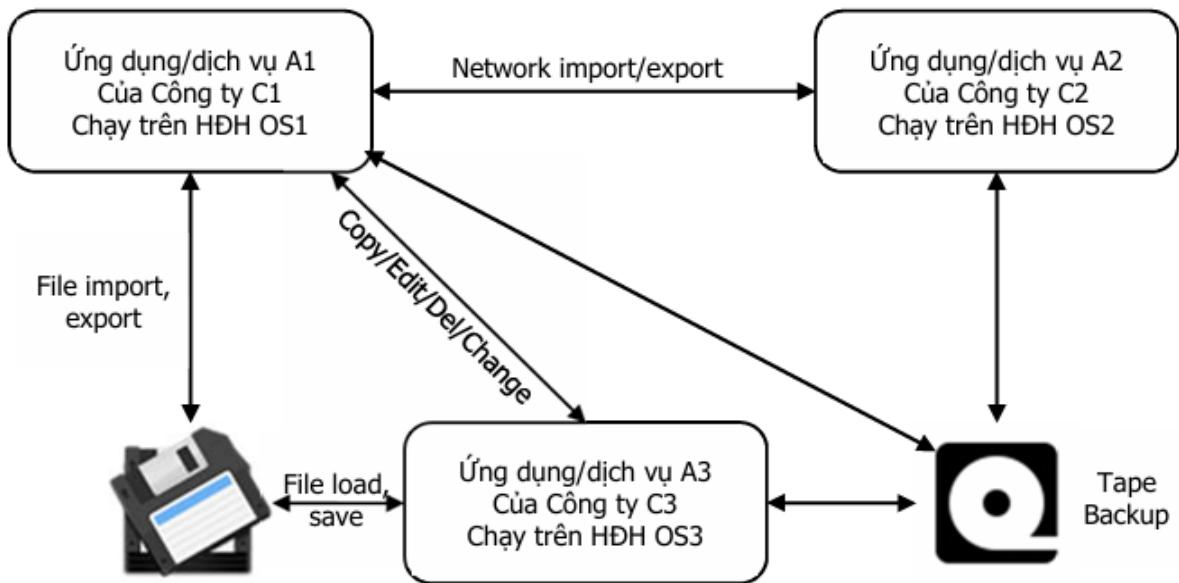
Hình 2.1.2: Minh họa kiểm thử hộp trắng

- Kiểm thử hộp đen là một kỹ thuật kiểm thử phần mềm trong đó người kiểm thử không quan tâm đến hiểu biết bên hoặc chi tiết cài đặt của phần mềm mà tập trung vào việc xác thực chức năng dựa trên các thông số kỹ thuật hoặc yêu cầu được cung cấp.



Hình 2.1.3: Minh họa kiểm thử hộp đen

- Kiểm thử khả năng tương thích (Compatibility testing) là kiểm thử thuộc thể loại kiểm thử phi chức năng (non functional testing) và được thực hiện trên một ứng dụng để kiểm tra khả năng tương thích (khả năng chạy) của nó trên các nền tảng/môi trường khác nhau.



Hình 2.1.4: Minh họa kiểm thử tương thích

2.2. CHƯƠNG 2: TÌM HIỂU CÔNG CỤ KIỂM THỬ PHẦN MỀM

2.2.1. Giới thiệu công cụ.

2.2.1.1. Công cụ kiểm thử Postman:

- Postman là công cụ kiểm thử và phát triển API mạnh mẽ và phổ biến, cho phép người dùng có thể thao tác với API, mà trong đó phổ biến nhất là REST. Với thử nghiệm API thì Postman là một trong những công cụ phổ biến vì được thực nghiệm nhiều nhất. Nhờ Postman lập trình viên có thể gọi Rest API mà không cần phải viết bất kỳ dòng code nào. Postman có khả năng hỗ trợ mọi phương thức HTTP bao gồm: POST, PUT, DELETE, PATCH, GET,...
- Vào năm 2012, Abhinav Asthana, một kỹ sư phần mềm Ấn Độ, tạo ra Postman như một công cụ nội bộ để đơn giản hóa việc phát triển và thử nghiệm API. Lúc đầu, Postman chỉ là một tiện ích mở rộng (Chrome Extension). Đến năm 2014, Postman phát triển thành một ứng dụng desktop độc lập để vượt qua các giới hạn của plugin trình duyệt. Công ty Postman Inc. được thành lập để mở rộng công cụ này. Từ 2016 đến 2019, các nhà phát triển đã bổ sung các tính năng như chia sẻ API, cộng tác nhóm, giám sát API và phân tích kết quả kiểm thử và đáp ứng nhu cầu của các doanh nghiệp lớn

với các tính năng quản lý, bảo mật và hỗ trợ API quy mô lớn. Mở rộng thêm khả năng CI/CD để tích hợp với các công cụ DevOps. Từ 2020 đến nay, Postman đã tiến hóa từ công cụ kiểm thử API đơn thuần thành nền tảng phát triển API toàn diện, hỗ trợ như: Thiết kế API, Mô phỏng (Mocking), Tự động hóa kiểm thử, Quản lý vòng đời API (API lifecycle management), Theo dõi hiệu suất API. Cung cấp phiên bản Postman Workspace để cộng tác theo thời gian thực.

2.2.1.2. Công cụ LambdaTest:

- LambdaTest là một nền tảng kiểm thử đám mây (cloud-based testing) mạnh mẽ, giúp các nhà phát triển và kiểm thử viên chạy các bài kiểm thử tự động và thủ công trên nhiều trình duyệt và thiết bị khác nhau. Công cụ này được thiết kế để tối ưu hóa quy trình kiểm thử giao diện người dùng, đảm bảo ứng dụng web và di động của bạn hoạt động mượt mà trên mọi môi trường.
- LambdaTest được thành lập vào năm 2017 bởi Asad Khan và Jay Singh, với mục tiêu cung cấp giải pháp kiểm thử đơn giản nhưng hiệu quả cho các nhà phát triển trên toàn thế giới. Năm 2017, LambdaTest ra đời với nền tảng kiểm thử đám mây, tập trung vào kiểm thử tương thích trình duyệt. Từ 2018 đến 2019, mở rộng hỗ trợ cho kiểm thử tự động với Selenium Grid và các framework kiểm thử tự động hóa khác. Năm 2020, phát triển mạnh mẽ với việc tích hợp kiểm thử ứng dụng di động và hỗ trợ nhiều môi trường trình duyệt và thiết bị thực tế. Năm 2021 – 2022, LambdaTest liên tục nâng cấp tính năng, bổ sung kiểm thử Visual Regression và các công cụ giám sát hiệu suất UI. Năm 2023, đẩy mạnh khả năng tự động hóa với công nghệ AI, tối ưu hóa quy trình kiểm thử, và mở rộng tích hợp với các CI/CD như Jenkins, GitLab, và các công cụ DevOps phổ biến.

2.2.2. Đặc điểm

2.2.2.1. Công cụ Postman

- Postman không phải mã nguồn mở
- Postman là công cụ chủ yếu phục vụ cho việc kiểm thử API, cụ thể:

- + Kiểm thử chức năng (Functional Testing): Đảm bảo các API hoạt động đúng như thiết kế.
- + Kiểm thử tích hợp (Integration Testing): Kiểm thử sự tương tác giữa các dịch vụ API khác nhau.
- + Kiểm thử hồi quy (Regression Testing): Kiểm tra lại các API sau khi có sự thay đổi hoặc cập nhật.
- + Kiểm thử tự động (Automated Testing): Sử dụng Scripts và Collections Runner để tự động hóa kiểm thử.
- + Kiểm thử hiệu suất (Performance Testing): Postman hỗ trợ giám sát API thông qua tích hợp với các công cụ khác để đo tải và thời gian phản hồi.
- Postman cho phép viết scripts kiểm thử thông qua JavaScript, đây là ngôn ngữ phổ biến và dễ sử dụng để tạo các điều kiện kiểm thử trước (Pre-request Script) và sau (Tests Script).
- Postman hỗ trợ cài đặt đa nền tảng:
 - + Hệ điều hành: Windows, macOS, Linux.
 - + Cài đặt dưới dạng ứng dụng desktop hoặc sử dụng phiên bản Postman for Web trên trình duyệt.
- Đặc điểm khác:
 - + Giao diện đồ họa thân thiện (GUI): Giúp người dùng dễ dàng thao tác mà không cần viết mã phức tạp.
 - + Mock Server: Tạo các API ảo để thử nghiệm mà không cần backend hoàn chỉnh.
 - + Tính năng cộng tác nhóm: Hỗ trợ chia sẻ collections, environments và API giữa các thành viên trong nhóm thông qua Workspaces.

- + Hỗ trợ nhiều phương thức HTTP: GET, POST, PUT, PATCH, DELETE, v.v.
- + Tích hợp CI/CD: Postman có thể tích hợp vào quy trình DevOps để tự động kiểm thử và giám sát API thông qua Postman CLI hoặc Newman (command-line collection runner).
- + Tích hợp API Network: Cho phép người dùng khám phá và chia sẻ các API công khai trên mạng lưới toàn cầu.

2.2.2.2. Công cụ LambdaTest

- LambdaTest là công cụ đóng được phát triển và cung cấp bởi công ty LambdaTest.
- LambdaTest hỗ trợ nhiều loại kiểm thử, bao gồm:
 - + Kiểm thử tương thích trình duyệt (Cross-Browser Testing): Đảm bảo ứng dụng hoạt động trên các trình duyệt khác nhau như Chrome, Safari, Firefox, Edge...
 - + Kiểm thử trên thiết bị di động (Mobile Testing): Kiểm thử trên nhiều thiết bị thật và giả lập chạy iOS, Android.
 - + Kiểm thử tự động (Automation Testing): Hỗ trợ các framework như Selenium, Cypress, Playwright, Appium, Puppeteer...
 - + Kiểm thử thủ công (Manual Testing): Giúp kiểm thử giao diện người dùng trong môi trường đám mây thời gian thực.
 - + Kiểm thử Visual Regression (Kiểm thử hồi quy giao diện): Phát hiện các thay đổi nhỏ trong giao diện so với thiết kế ban đầu.
 - + Kiểm thử hiệu suất (Performance Testing): Đánh giá tốc độ và khả năng tải của ứng dụng.
- LambdaTest hỗ trợ nhiều ngôn ngữ lập trình và framework kiểm thử phổ biến.

- **Đặc điểm khác:**
 - + Hỗ trợ CI/CD: LambdaTest dễ dàng tích hợp với các công cụ CI/CD như Jenkins, GitLab, CircleCI, Azure DevOps, giúp tự động hóa quy trình kiểm thử.
 - + Real Device Testing: Cung cấp các thiết bị thực tế để kiểm thử trải nghiệm người dùng trên môi trường thật.
 - + Tích hợp AI: Sử dụng trí tuệ nhân tạo để phân tích lỗi giao diện và cải thiện khả năng kiểm thử tự động.
 - + Báo cáo chi tiết: Tính năng báo cáo kết quả kiểm thử chi tiết, trực quan giúp phát hiện và khắc phục lỗi nhanh chóng.
 - + Tính năng chụp màn hình và quay video: Ghi lại quá trình kiểm thử để dễ dàng theo dõi và phân tích.

2.2.3. Cài đặt và sử dụng công cụ

2.2.3.1. Công cụ Postman

- Postman hỗ trợ đa nền tảng, có thể cài đặt trên các hệ điều hành phổ biến sau:
 - + Windows: Hỗ trợ từ Windows 7 trở lên (bao gồm Windows 10 và 11).
 - + macOS: Hỗ trợ từ macOS 10.11 (El Capitan) and later, with Intel or Apple silicon.
 - + Linux: Hỗ trợ các bản phân phối như Ubuntu 14.04 and later, Fedora 24, Debian 8 and later, with 64-bit.
 - + Ngoài ra, Postman còn cung cấp phiên bản Postman for Web sử dụng trực tiếp trên trình duyệt mà không cần cài đặt phần mềm.
- Yêu cầu phần cứng:
 - + Bộ xử lý (CPU): Tối thiểu Intel Core i3 hoặc tương đương.

- + Bộ nhớ RAM: Tối thiểu 4 GB (khuyến nghị 8 GB trở lên để chạy mượt mà).
- + Dung lượng lưu trữ: Tối thiểu 500 MB dung lượng trống trên ổ đĩa.
- + Độ phân giải màn hình: 1024x768 trở lên.
- Yêu cầu phần mềm:
 - + Trình duyệt web: Đối với phiên bản Postman for Web, cần trình duyệt hiện đại như Google Chrome, Firefox, Microsoft Edge hoặc Safari.
 - + Node.js (tùy chọn): Nếu sử dụng Newman (Postman CLI) để tự động hóa kiểm thử, cần cài đặt Node.js phiên bản 10 trở lên.
 - + Quyền cài đặt ứng dụng: Đối với hệ điều hành như Windows và macOS, người dùng cần có quyền quản trị viên để cài đặt phần mềm Postman.
- Yêu cầu khác:
 - + Kết nối Internet: Cần kết nối mạng để tải Postman về thiết bị và sử dụng các tính năng như đồng bộ hóa, cộng tác nhóm, và gửi yêu cầu API trực tuyến.
 - + Tài khoản Postman: Người dùng cần tạo tài khoản Postman để tận dụng các tính năng như lưu trữ collections, cộng tác nhóm và đồng bộ dữ liệu trên nhiều thiết bị.
 - + Quyền truy cập API: Đảm bảo rằng các API cần kiểm thử có quyền truy cập từ Postman. Một số API yêu cầu thông tin xác thực như API key, Bearer Token hoặc các phương thức OAuth.

2.2.3.2. Công cụ LambdaTest

- LambdaTest là một công cụ kiểm thử đám mây, vì vậy người dùng không cần phải cài đặt công cụ trực tiếp trên hệ điều hành của mình.
- LambdaTest hỗ trợ các hệ điều hành sau:

OPERATING SYSTEM	CHROME	FIREFOX	EDGE
macOS Ventura	66 and above (Except 82)	60 and above	80 and above (Except 82)
macOS Monterey	66 and above (Except 82)	60 and above	80 and above (Except 82)
macOS Big Sur	66 and above (Except 82)	60 and above	80 and above (Except 82)
macOS Mojave	66 and above (Except 82)	60 and above	80 and above (Except 82)
macOS Catalina	66 and above (Except 82)	60 and above	80 and above (Except 82)
Windows 11	66 and above (Except 82)	60 and above	80 and above (Except 82)
Windows 10	66 and above (Except 82)	60 and above	80 and above (Except 82)
Windows 8.1	66 and above (Except 82)	60 and above	80 and above (Except 82)
Windows 8	66 and above (Except 82)	60 and above	80 and above (Except 82)
Windows 7	66 and above (Except 82)	60 and above	80 and above (Except 82)

Hình 2.2.1: Các hệ điều hành LambdaTest hỗ trợ

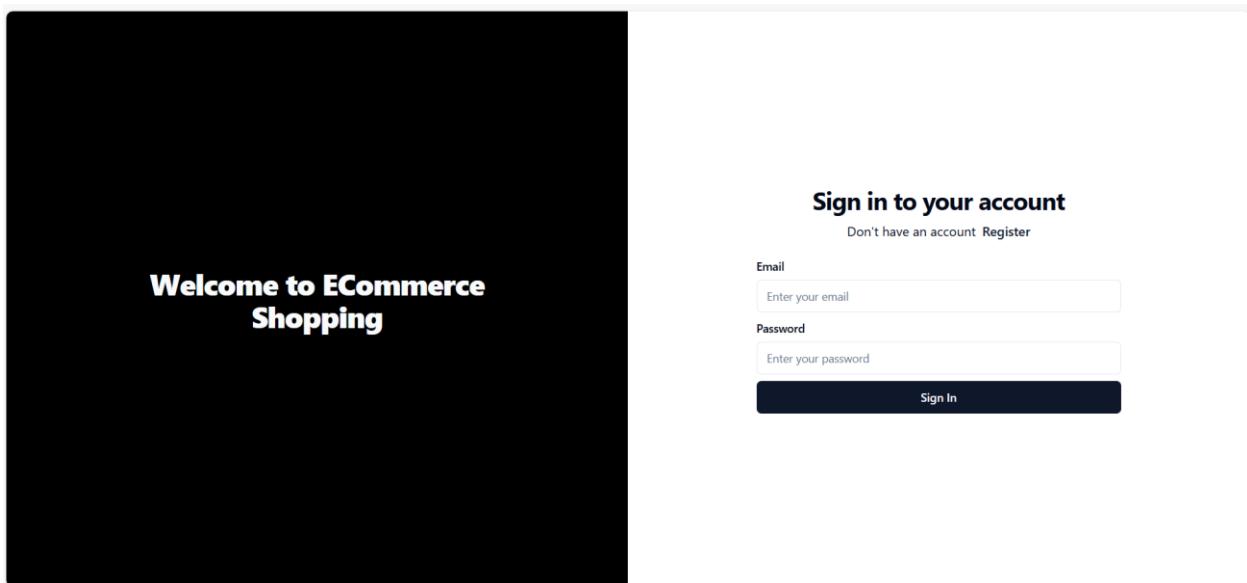
- Yêu cầu phần mềm:
 - + Trình duyệt web hiện đại (Chrome, Firefox, Safari, Edge) để truy cập vào giao diện LambdaTest.
 - + Để thực hiện kiểm thử tự động, người dùng cần cài đặt các framework kiểm thử như Selenium, Cypress, Appium, Playwright, tùy vào ngôn ngữ và nhu cầu kiểm thử của họ.
 - + Tích hợp CI/CD: Để kết nối LambdaTest với các hệ thống CI/CD như Jenkins, GitLab, CircleCI, người dùng cần cài đặt và cấu hình các công cụ này tương ứng.
- Yêu cầu khác:
 - + Kết nối Internet ổn định: Vì LambdaTest hoạt động trên nền tảng đám mây, việc có một kết nối Internet nhanh và ổn định là rất quan trọng để đảm bảo các phiên kiểm thử diễn ra suôn sẻ.

- + Tài khoản LambdaTest: Người dùng cần đăng ký tài khoản LambdaTest để sử dụng các tính năng kiểm thử. Dịch vụ cung cấp các gói miễn phí và trả phí, tùy theo nhu cầu sử dụng.
- + Quyền truy cập vào các công cụ phát triển (cho kiểm thử tự động): Nếu người dùng muốn chạy kiểm thử tự động với Selenium hoặc các framework khác, họ cần có quyền truy cập vào các công cụ phát triển phù hợp và các API của LambdaTest để cấu hình và chạy các bài kiểm thử tự động.

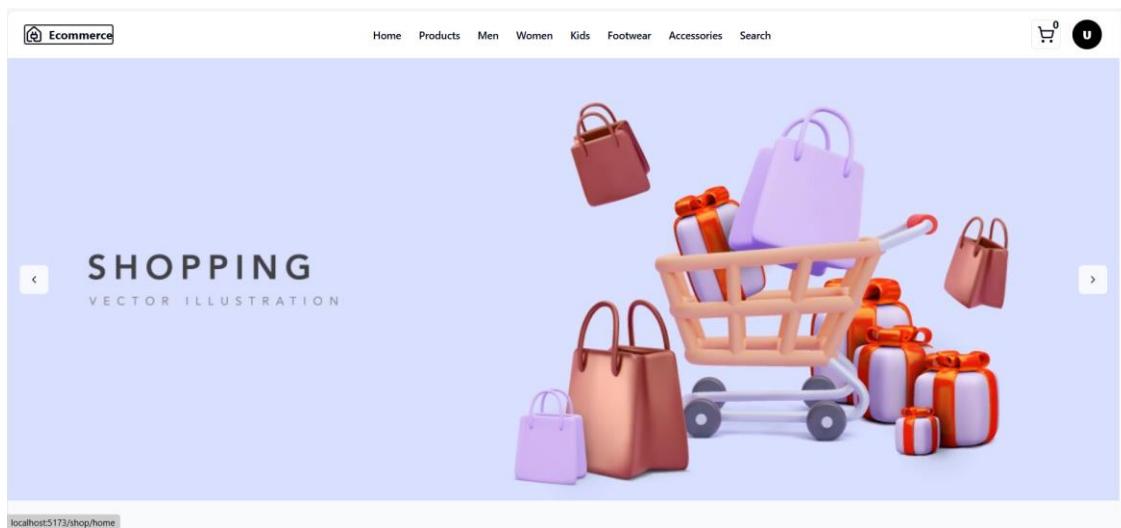
2.3. CHƯƠNG 3: GIỚI THIỆU WEB SITE BÁN QUẦN ÁO

2.3.1. Giới thiệu

- Ứng dụng web có tên là ‘ECommerce Shopping’ và được phát triển bởi nhóm chúng em trên các framework của Javascript: Reactjs cho Frontend và Nodejs cho Backend. Mục đích làm ứng dụng web này để nâng cao kỹ năng sử dụng ngôn ngữ học hỏi lẫn nhau và cũng mong muốn sản phẩm của mình được nhiều người quan tâm và biết đến.
- Một số hình ảnh về trang web:



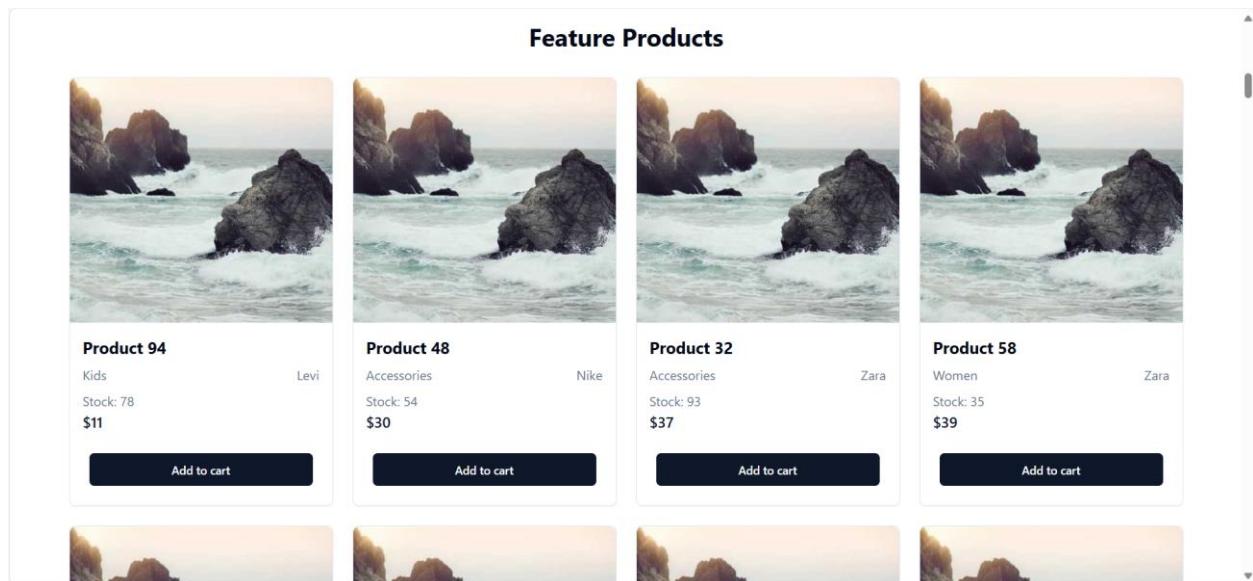
Hình 2.3.1: Trang đăng nhập



Hình 2.3.2: Trang chủ sau khi đăng nhập.

A screenshot of a website's product listing page. At the top, there is a section titled 'Shop by category' containing five categories: Men (represented by a t-shirt icon), Women (represented by a woman's head icon), Kids (represented by a smiling child icon), Accessories (represented by a clock icon), and Footwear (represented by a shoe icon). Below this is a section titled 'Shop by Brand' featuring six brand logos: Nike (t-shirt icon), Adidas (three stripes icon), Puma (basket icon), Levi's (tv screen icon), Zara (square icon with a circle), and H&M (hanger icon). At the very bottom, there is a section titled 'Feature Products' which is currently empty, indicated by a horizontal bar with five yellow segments.

Hình 2.3.3: Giao diện phân loại sản phẩm của trang web



Hình 2.3.4: Danh sách các sản phẩm.

Ecommerce

Home Products Men Women Kids Footwear Accessories Search

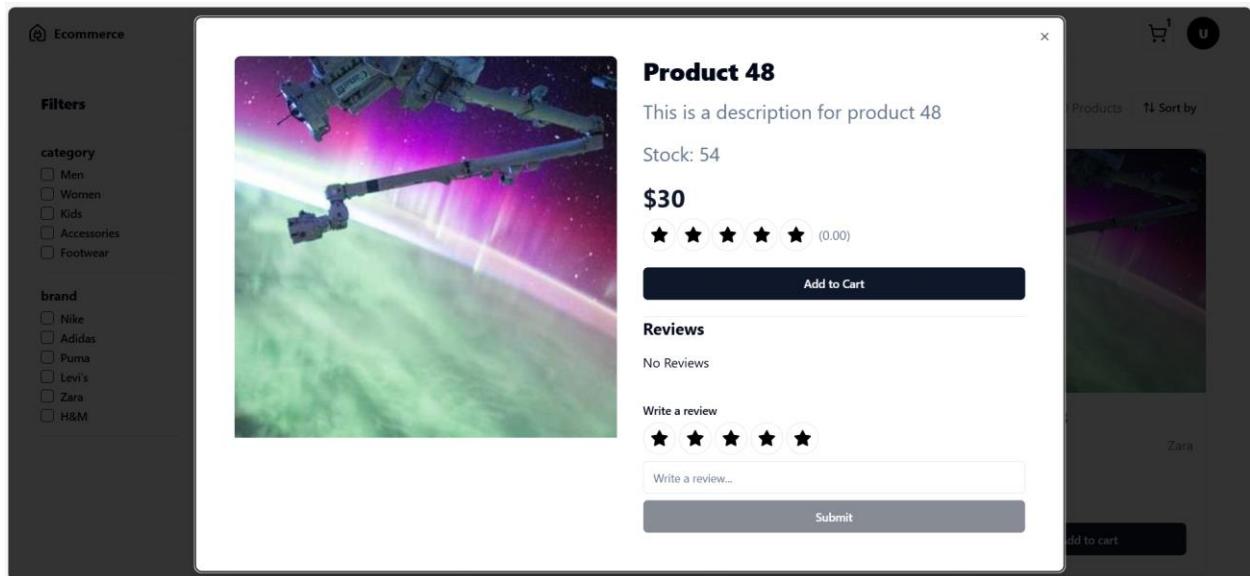
Filters

All Products

100 Products Sort by

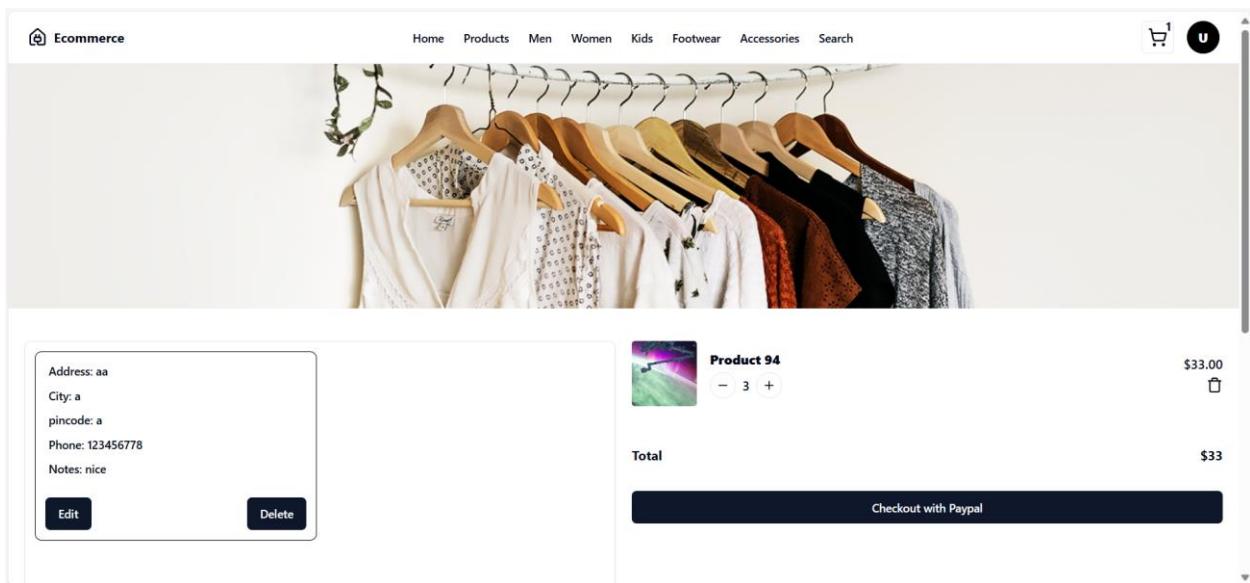
Product	Category	Stock	Price
Product 94	Kids	78	\$11
Product 48	Accessories	54	\$30
Product 32	Accessories	93	\$37
Product 58	Women	35	\$39

Hình 2.3.5: Danh sách các sản phẩm

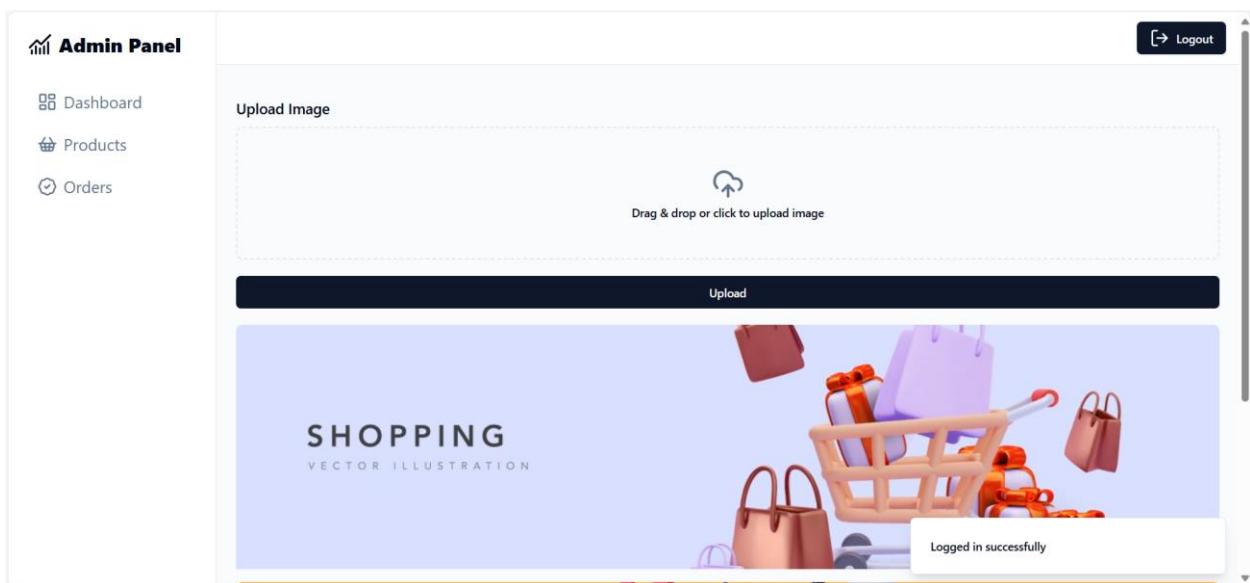


Hình 2.3.6: Chi tiết sản phẩm.

Hình 2.3.7: Giao diện giỏ hàng



Hình 2.3.8: Giao diện đặt hàng.



Hình 2.3.9: Giao diện trang quản trị

Admin Panel

Logout

Add New Product

Dashboard Products Orders

Product 0
Stock: 13
\$221 \$162
Edit Delete

Product 1
Stock: 94
\$258 \$184
Edit Delete

Product 2
Stock: 94
\$78 \$37
Edit Delete

Product 3
Stock: 31
\$658 \$812
Edit Delete

Hình 2.3.10: Giao diện trang quản trị sản phẩm

Admin Panel

Dashboard Products Orders

Add New Product

Upload Image
Drag & drop or click to upload image

Title
Enter product title

Description
Enter product description

Category
Category

Brand
Brand

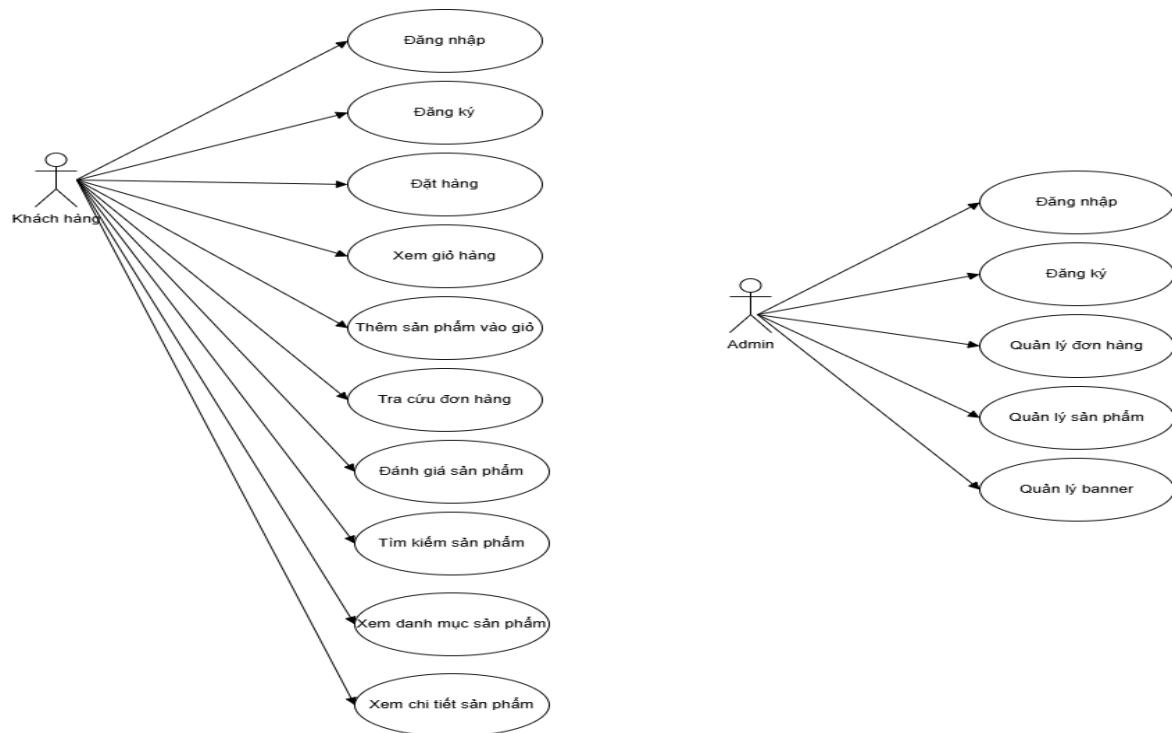
Price
Enter product price

Hình 2.3.11: Giao diện trang thêm sản phẩm mới

Hình 2.3.12: Giao diện trang quản trị đơn hàng.

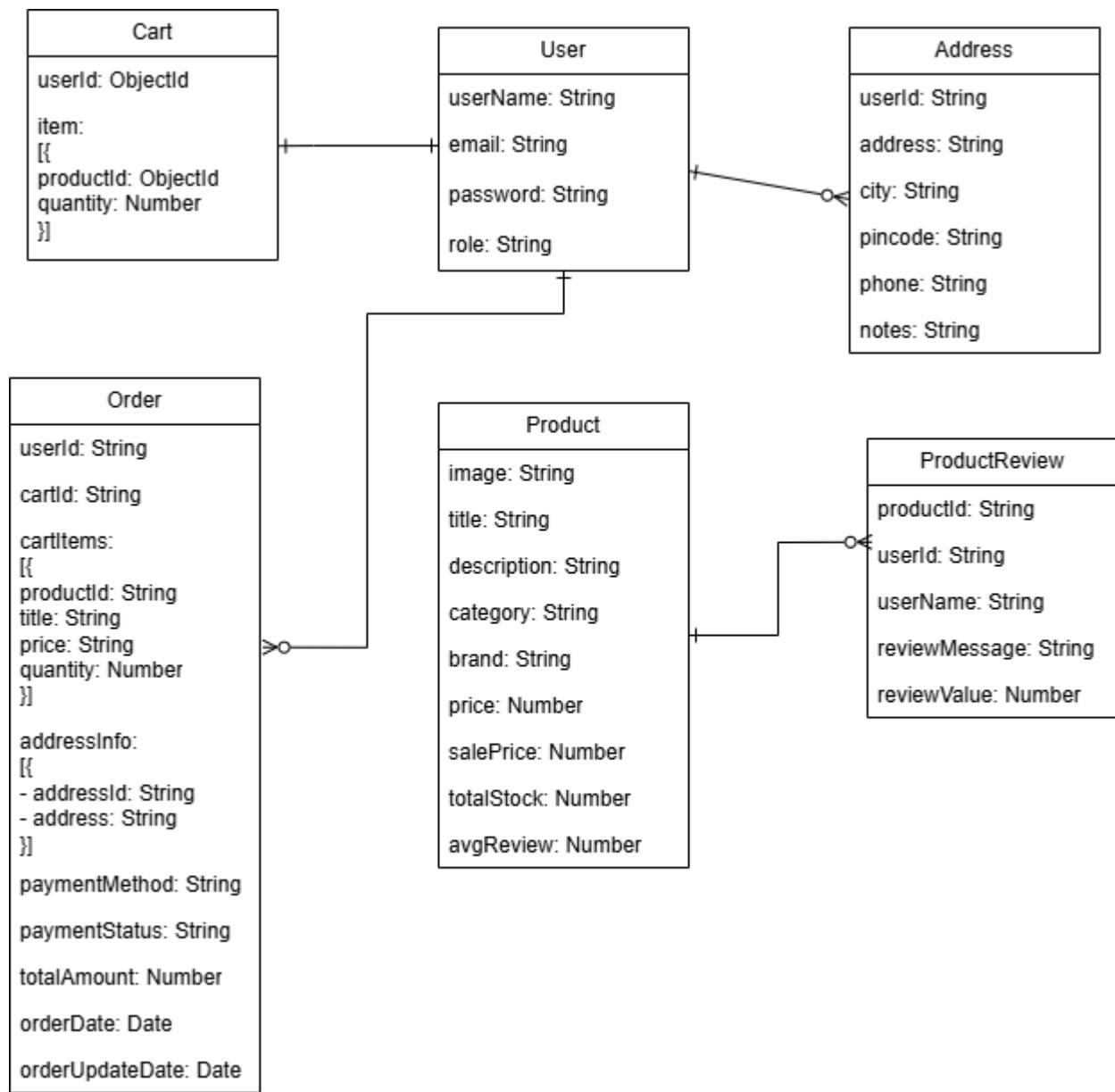
2.3.2. Tài liệu đặc tả yêu cầu phần mềm

2.3.2.1. Các usecase chính của hệ thống.



Hình 2.3.13: Các usecase chính của hệ thống.

2.3.2.2. Biểu đồ thực thể



Hình 2.3.14: Biểu đồ thực thể.

2.4. Chương 4 - Kiểm thử phần mềm

2.4.1. Lập kế hoạch kiểm thử

2.4.1.1. Mục đích của tài liệu kế hoạch kiểm thử

Tài liệu kế hoạch kiểm thử này đưa ra các mục đích sau:

- Xác định thông tin cơ bản về dự án và các thành phần chức năng được kiểm thử và không được kiểm thử.
- Liệt kê những yêu cầu cho việc kiểm thử (Test Requirements).
- Những chiến lược kiểm thử nên được sử dụng.
- Những tài liệu được lập sau khi hoàn thành việc kiểm thử.
- Các chức năng được thực hiện đúng trên các trình duyệt.
- API trả về đúng cho client.

2.4.1.2. Phạm vi kiểm thử

Phần mềm: Website bán hàng online.

- Sửa sản phẩm.
- Thêm đánh giá.
- Thêm sản phẩm vào giỏ hàng.
- Xóa sản phẩm.
- Đăng nhập.
- Đăng ký.
- Thêm sản phẩm mới.
- Thêm sản phẩm vào giỏ.

2.4.1.3. Phân công nhiệm vụ.

Bảng 2.4.1: Bảng phân công nhiệm vụ cụ thể

STT	Người phụ trách	Chức năng	Nhiệm vụ cụ thể
1	Nguyễn Huy Nhật	Sửa sản phẩm.	Thực hiện đăng nhập hệ thống với vai trò admin và sửa một sản phẩm bất kỳ.
2	Đỗ Trọng Hoàng	Thêm sản phẩm vào giỏ hàng	Thực hiện đăng nhập hệ thống và thêm một bất kỳ sản phẩm vào giỏ hàng rồi hiển thị giỏ hàng đã thêm.
3	Nguyễn Quốc Thái	Thêm Đánh giá	Thực hiện đăng nhập hệ thống và thêm đánh giá vào sản phẩm bất kỳ.
4	Lê Anh Tạo	Xóa sản phẩm.	Thực hiện đăng nhập hệ thống với vai trò admin và xóa 1 sản phẩm bất kỳ.
5	Nguyễn Huy Nhật	Đăng ký	Thực hiện kiểm tra dữ liệu từ client gửi lên có hợp lệ và trả về dữ liệu cho người dùng có đúng như mong muốn.
6	Nguyễn Quốc Thái	Đăng nhập	
7	Đỗ Trọng Hoàng	Thêm sản phẩm mới	
8	Lê Anh Tạo	Thêm sản phẩm vào giỏ hàng	

2.4.1.4. Phương pháp kiểm thử.

Bảng 2.4.2: Bảng phân công nhiệm vụ (theo công cụ)

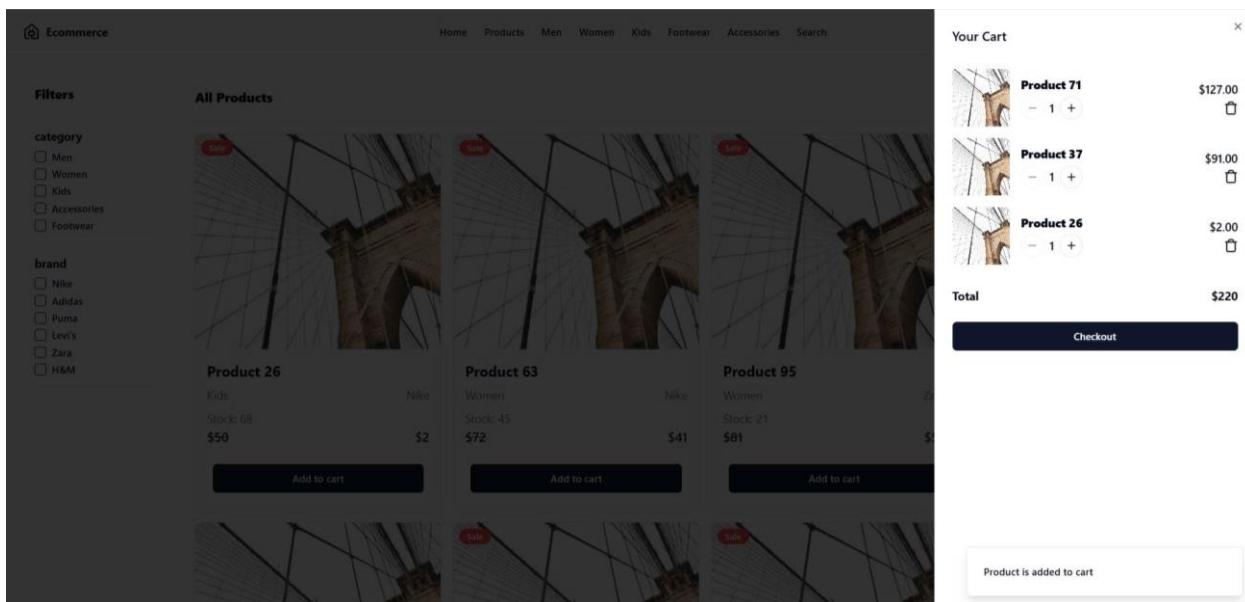
STT	Người phụ trách	Chức năng	Công cụ
1	Nguyễn Huy Nhật	Sửa sản phẩm.	Lambdatest để kiểm thử tương thích.
2	Đỗ Trọng Hoàng	Thêm sản phẩm vào giỏ hàng	
3	Nguyễn Quốc Thái	Thêm Đánh giá	
4	Lê Anh Tạo	Xóa sản phẩm.	
5	Nguyễn Huy Nhật	Đăng ký	Postman để kiểm thử API

6	Nguyễn Quốc Thái	Đăng nhập
7	Đỗ Trọng Hoàng	Thêm sản phẩm mới
8	Lê Anh Tạo	Thêm sản phẩm vào giỏ hàng

2.4.2. Đỗ Trọng Hoàng – Thêm sản phẩm vào giỏ hàng.

2.4.2.1. Phân tích thiết kế kiểm thử

- Giao diện sẽ hiện như sau khi test case thực hiện thành công:



Hình 2.4.1: Giao diện mong muốn khi kiểm thử tự động.

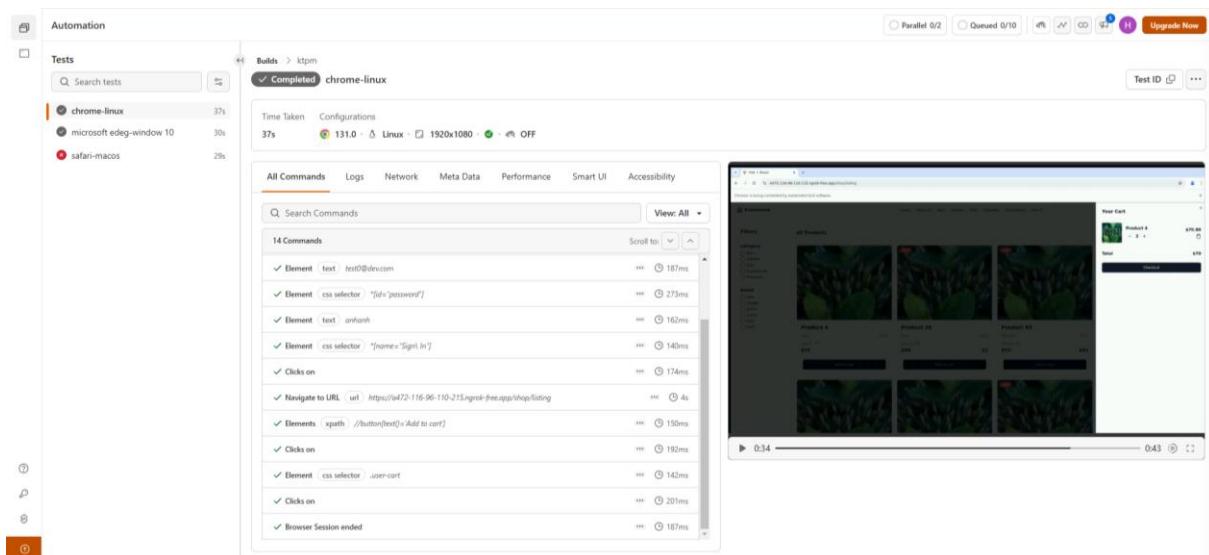
Bảng 2.4.3: Bảng testcase cho chức năng thêm sản phẩm vào giỏ hàng

STT	Chức năng	Môi trường	Bước thực hiện	Kết quả mong đợi	Ghi chú
1	Thêm sản phẩm vào giỏ hàng	macOS + Safari	1. Truy cập trang đăng nhập. 2. Đăng nhập tài khoản và mật khẩu	Sản phẩm được thêm vào giỏ hàng và hiển thị đúng thông tin.	Cơ chế tìm test case: Cross-Browser Testing (kiểm thử

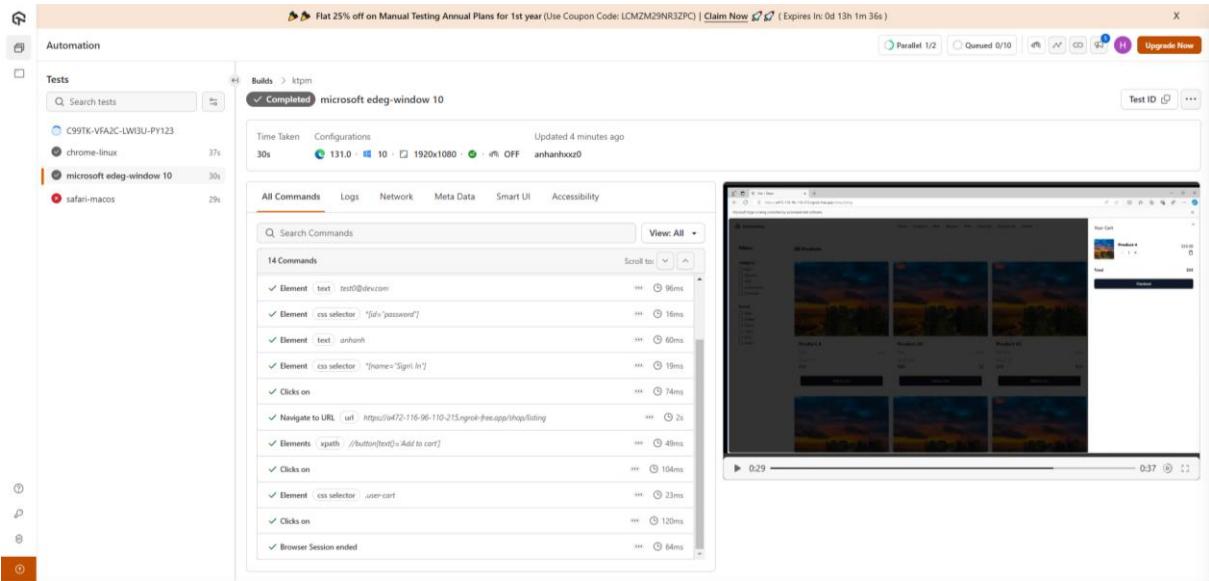
			<p>hợp lệ.</p> <p>3. Chọn sản phẩm đầu tiên và nhấn "Add to cart".</p> <p>4. Kiểm tra giỏ hàng.</p>		<p>trên các trình duyệt khác nhau).</p> <p>Độ bao phủ: Bao phủ chức năng "Thêm sản phẩm vào giỏ hàng" trên môi trường macOS + Safari.</p>
2		Linux + Chrome	Sản phẩm được thêm vào giỏ hàng và hiển thị đúng thông tin.	Cơ chế tìm test case: Cross-Browser Testing.	<p>Độ bao phủ: Bao phủ chức năng trên môi trường Linux + Chrome.</p>
3		Windows + Edge	Sản phẩm được thêm vào giỏ hàng và hiển thị đúng thông tin.	Cơ chế tìm test case: Cross-Browser Testing.	<p>Độ bao phủ: Bao phủ chức năng trên môi trường Windows + Edge.</p>
4		macOS + Chrome	Sản phẩm được thêm vào giỏ hàng và	Cơ chế tìm test case: Cross-Browser	

				hiển thị đúng thông tin.	Testing. Độ bao phủ: Bao phủ chức năng trên môi trường macOS + Chrome.
--	--	--	--	--------------------------	---

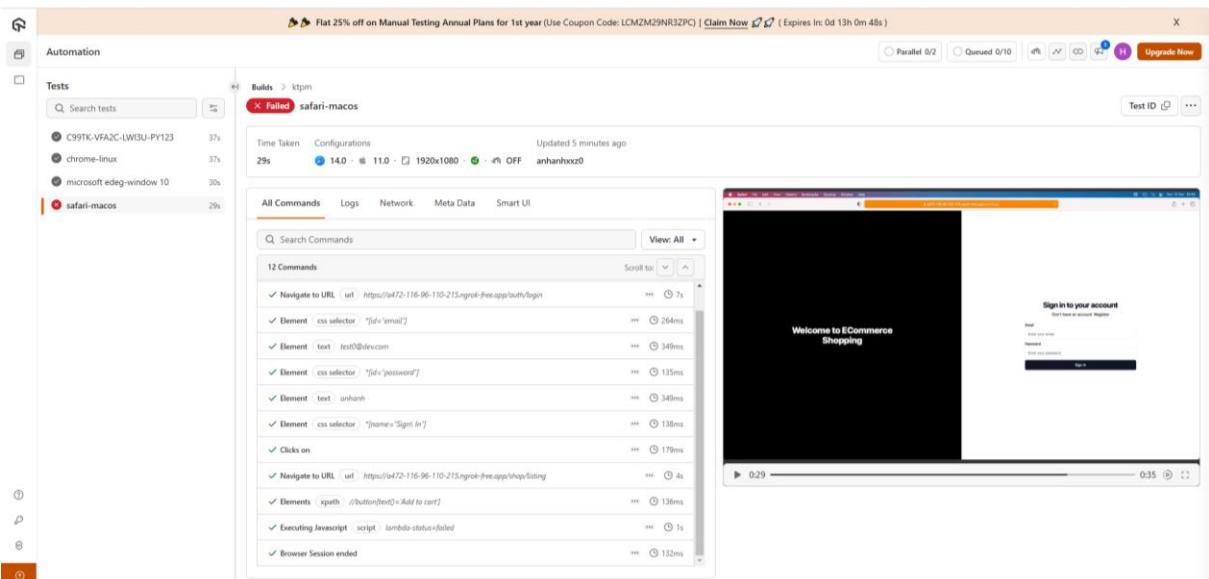
2.4.2.2. Thực hiện kiểm thử



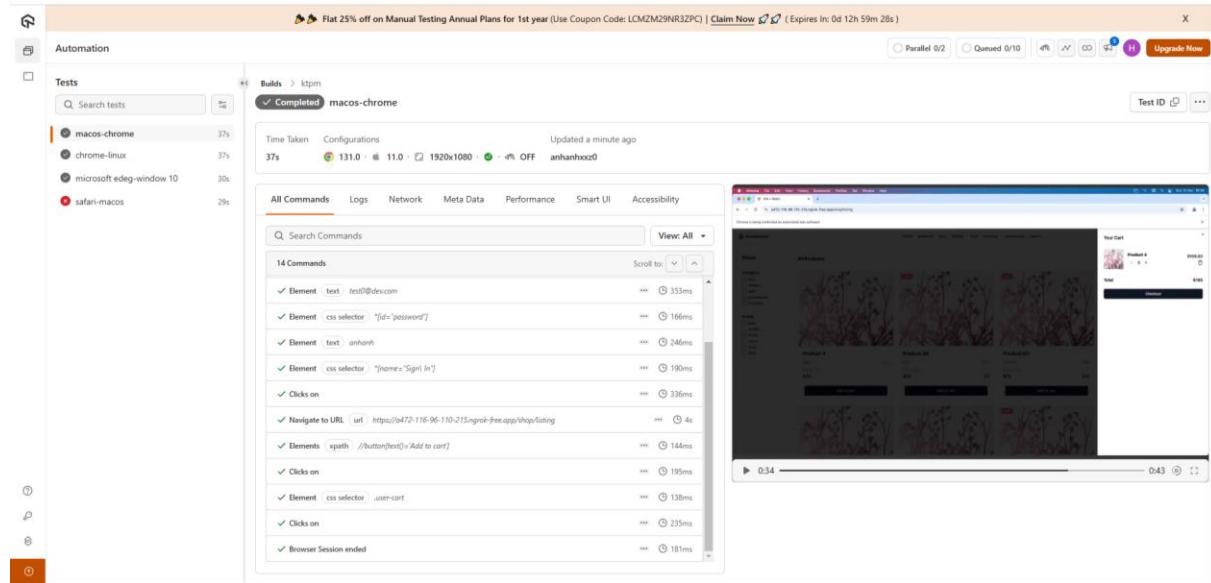
Hình 2.4.2: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.



Hình 2.4.3: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.



Hình 2.4.4: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.



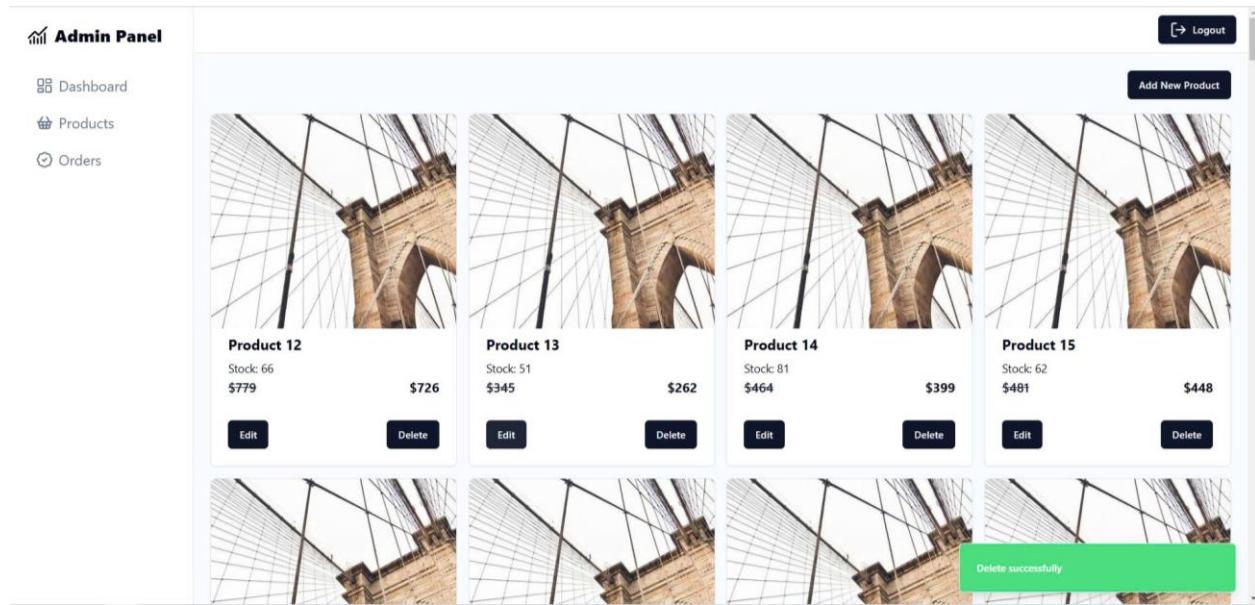
Hình 2.4.5: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.

- Đánh giá:
 - + Có 3 hành động thành công trên 2 trình duyệt là Chorme và Microsoft Edge trên 3 hệ điều hành macos,linux,window.
 - + Có 1 hành động thất bại trên trình duyệt safari trên hệ điều hành macos.
- Tổng kết:
 - + Ứng dụng có thể chạy tốt hành động trên 2 trình duyệt và thất bại trên trình duyệt safari.
 - + Lý do thất bại có thể do trình duyệt Safari kích hoạt các chính sách bảo mật nghiêm ngặt liên quan đến cookie gửi tới server dẫn đến thất bại.

2.4.3. Lê Anh Tạo - Xóa sản phẩm.

2.4.3.1. Phân tích thiết kế kiểm thử.

- Giao diện thành công:



Hình 2.4.6: Giao diện mong muốn khi xóa thành công

Bảng 2.4.4: Bảng testcase cho chức năng xóa sản phẩm

STT	Chức năng	Môi trường	Bước thực hiện	Kết quả mong đợi	Ghi chú
1	Xóa sản phẩm từ danh sách	macOS + Safari	1. Truy cập trang đăng nhập. 2. Đăng nhập tài khoản và mật khẩu hợp lệ. 3. Truy cập trang quản lý sản phẩm. 4. Chọn sản phẩm đầu tiên và nhấn "Delete". 5. Kiểm tra thông báo hệ thống.	Hệ thống thông báo thành công khi sản phẩm bị xóa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng "Xóa sản phẩm" trên môi trường macOS + Safari.
2		Linux + Chrome		Hệ thống thông báo thành công khi sản phẩm bị xóa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ

					chức năng trên môi trường Linux + Chrome.
3		Windows + Edge		Hệ thống thông báo thành công khi sản phẩm bị xóa.	Cơ chế tìm test case: Cross- Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường Windows + Edge.
4		macOS + Chrome		Hệ thống thông báo thành công khi sản phẩm bị xóa.	Cơ chế tìm test case: Cross- Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường macOS + Chrome.

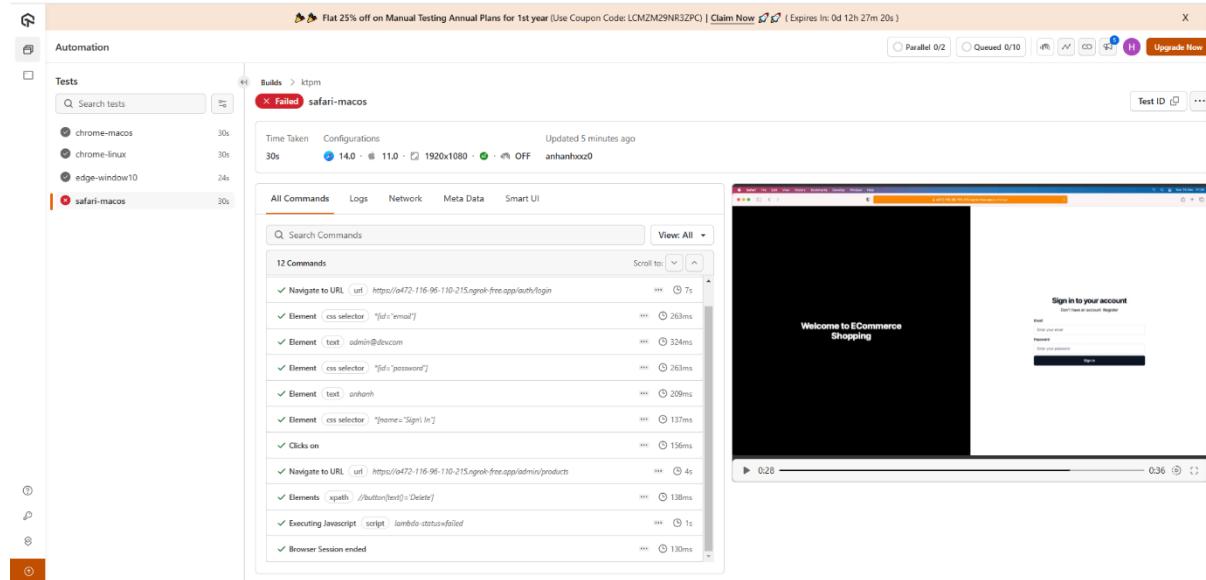
2.4.3.2.Thực hiện kiểm thử

The screenshot shows a test automation interface with a completed build for 'chrome-linux'. The build took 30s and was run on a Linux configuration (131.0). The log details 12 commands, including navigating to URLs and interacting with elements like email and password fields. On the right, a screenshot of a web application's admin panel is shown, displaying a grid of products with names and some numerical values.

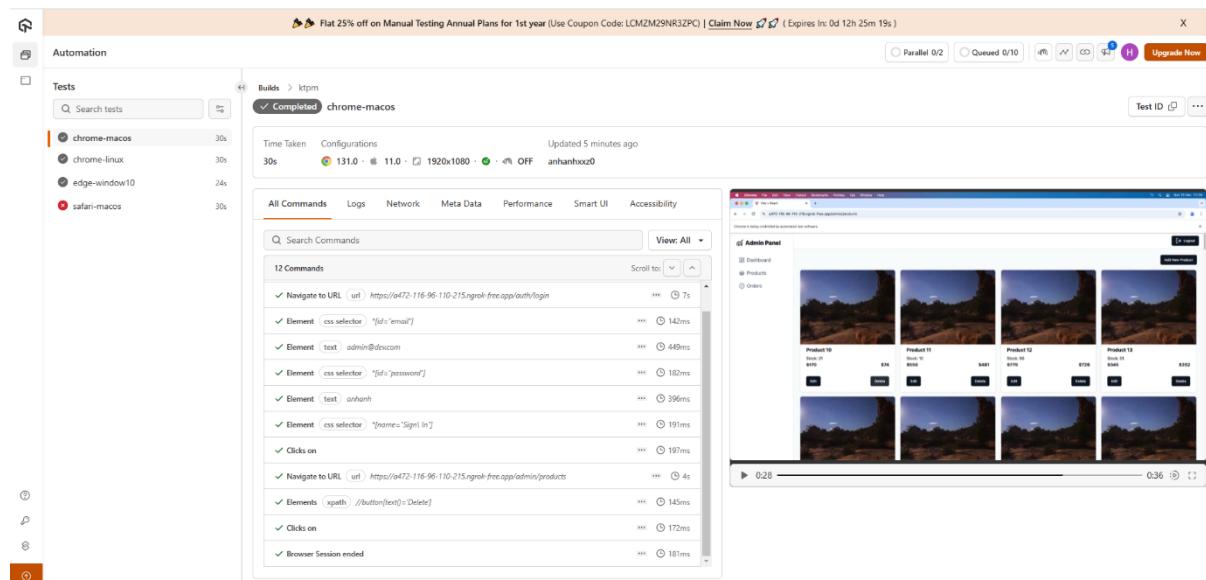
Hình 2.4.7: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.

The screenshot shows a test automation interface with a completed build for 'edge-windows10'. The build took 24s and was run on a Windows 10 configuration (131.0). The log details 12 commands, including navigating to URLs and interacting with elements like email and password fields. On the right, a screenshot of a web application's admin panel is shown, displaying a grid of products with names and some numerical values.

Hình 2.4.8: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.



Hình 2.4.9: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.



Hình 2.4.10: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.

- Đánh giá:
 - + Có 3 hành động thành công trên 2 trình duyệt là Chorme và Microsoft Edge trên 3 hệ điều hành macos,linux,window.
 - + Có 1 hành động thất bại trên trình duyệt safari trên hệ điều hành macos.

- Tổng kết:
 - + Ứng dụng có thể chạy tốt hành động trên 2 trình duyệt và thất bại trên trình duyệt safari.
 - + Lý do thất bại có thể do trình duyệt Safari kích hoạt các chính sách bảo mật nghiêm ngặt liên quan đến cookie gửi tới server dẫn đến thất bại.

2.4.4. Nguyễn Huy Nhật - Cảnh sửa sản phẩm

2.4.4.1. Phân tích thiết kế kiểm thử

The screenshot shows the Admin Panel interface. On the left, there's a sidebar with 'Admin Panel' at the top, followed by 'Dashboard', 'Products', and 'Orders'. The main area displays a grid of product cards. Each card contains a thumbnail image of a bridge, the product name (e.g., 'Product 12'), current stock (e.g., 'Stock: 66'), price (\$779), and two buttons: 'Edit' and 'Delete'. A modal window titled 'Edit Product' is open over the grid, specifically for 'Product 12'. The modal includes fields for 'Title' (set to 'Product 12'), 'Description' (set to 'This is a description for product 12'), 'Category' (set to 'Accessories'), 'Brand' (set to 'Nike'), 'Price' (\$779), 'Sale Price' (\$726), and 'Total Stock' (66). There are also 'Upload Image' and 'Save' buttons.

Hình 2.4.11: Giao diện trước khi sửa sản phẩm

This screenshot shows the Admin Panel after the changes made in the previous modal have been applied. The products now have updated values: Product 12 has a price of \$888 and stock of 66; Product 13 has a price of \$345 and stock of 51; Product 14 has a price of \$464 and stock of 81. The modal is no longer visible. Instead, a green success message 'Update successfully' is displayed in the bottom right corner of the product grid area. The rest of the interface remains the same, with the sidebar and other product cards visible.

Hình 2.4.12: Giao diện sau khi sửa thành công.

Bảng 2.4.5: Bảng testcase cho chức năng chỉnh sửa sản phẩm

STT	Chức năng	Môi trường	Bước thực hiện	Kết quả mong đợi	Ghi chú
1	Chỉnh sửa sản phẩm từ danh sách	macOS + Safari	1. Truy cập trang đăng nhập. 2. Đăng nhập tài khoản và mật khẩu hợp lệ. 3. Truy cập trang quản lý sản phẩm. 4. Chọn sản phẩm đầu tiên và nhấn "Edit".	Hệ thống thông báo thành công khi sản phẩm được chỉnh sửa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng "Chỉnh sửa sản phẩm" trên môi trường macOS + Safari.
2		Linux + Chrome	5. Nhập tên sản phẩm mới vào ô input và nhấn "Save Edit". 6. Kiểm tra thông báo hệ thống.	Hệ thống thông báo thành công khi sản phẩm được chỉnh sửa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường Linux + Chrome.
3		Windows + Edge		Hệ thống thông báo thành công khi sản phẩm được chỉnh sửa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi

					trường Windows + Edge.
4		macOS + Chrome		Hệ thống thông báo thành công khi sản phẩm được chỉnh sửa.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường macOS + Chrome.

2.4.4.2. Thực hiện kiểm thử

Thực thi kiểm thử:

Hình 2.4.13: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.

Automation

Builds > ktmp

Completed PSQIN-XSNRU-TVH43-5317B

Time Taken: 29s Configurations: Updated 2 minutes ago

All Commands Logs Network Meta Data Smart UI Accessibility

Search Commands View: All

16 Commands

- ✓ Element text: anhnh
- ✓ Element css selector: #name="Sign In"
- ✓ Clicks on
- ✓ Navigate to URL url: https://o472-116-96-110-215.ngrok-free.app/admin/products
- ✓ Elements xpath: //button[text()='Edit']
- ✓ Clicks on
- ✓ Element css selector: #id="title"
- ✓ Element text: Auto test 1004322430
- ✓ Element xpath: //button[text()='Save Edit']
- ✓ Clicks on
- ✓ Browser Session ended

Test ID: ...

Upgrade Now

Hình 2.4.14: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.

Automation

Builds > ktmp

Failed NDSFZ-TNRZN-7BAN0-GFNAE

Time Taken: 29s Configurations: Updated 2 minutes ago

All Commands Logs Network Meta Data Smart UI

Search Commands View: All

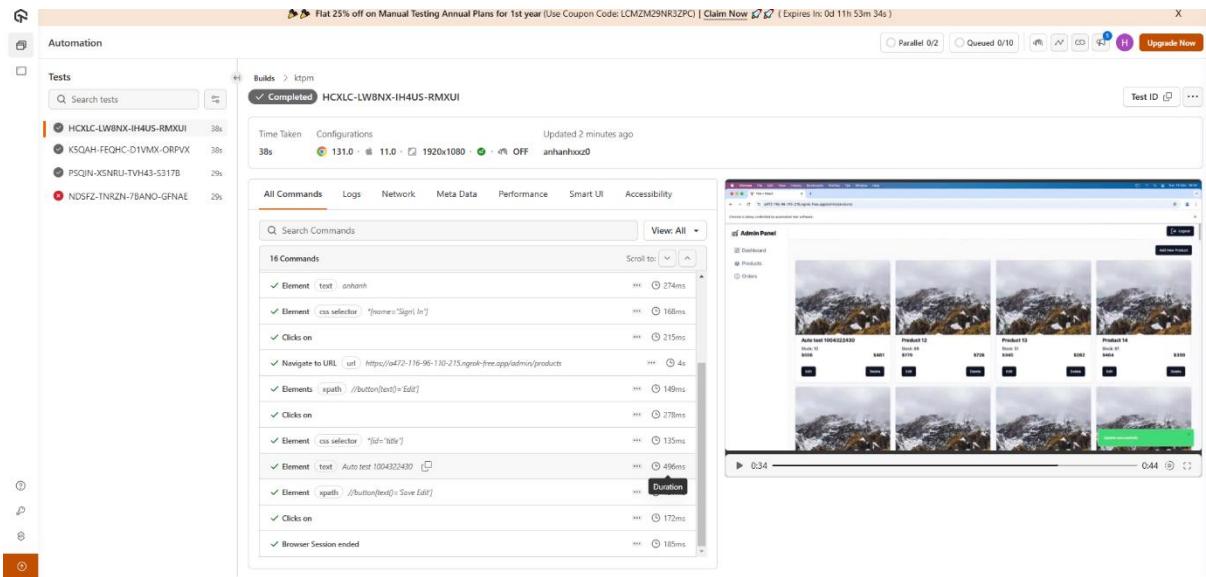
12 Commands

- ✓ Navigate to URL url: https://o472-116-96-110-215.ngrok-free.app/auth/login
- ✓ Element css selector: #id="email"
- ✓ Element text: admin@tescom
- ✓ Element css selector: #id="password"
- ✓ Element text: anhnh
- ✓ Element css selector: #name="Sign In"
- ✓ Clicks on
- ✓ Navigate to URL url: https://o472-116-96-110-215.ngrok-free.app/admin/products
- ✓ Elements xpath: //button[text()='Edit']
- ✓ Executing Javascript script: lambda-status=failed
- ✓ Browser Session ended

Test ID: ...

Upgrade Now

Hình 2.4.15: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.

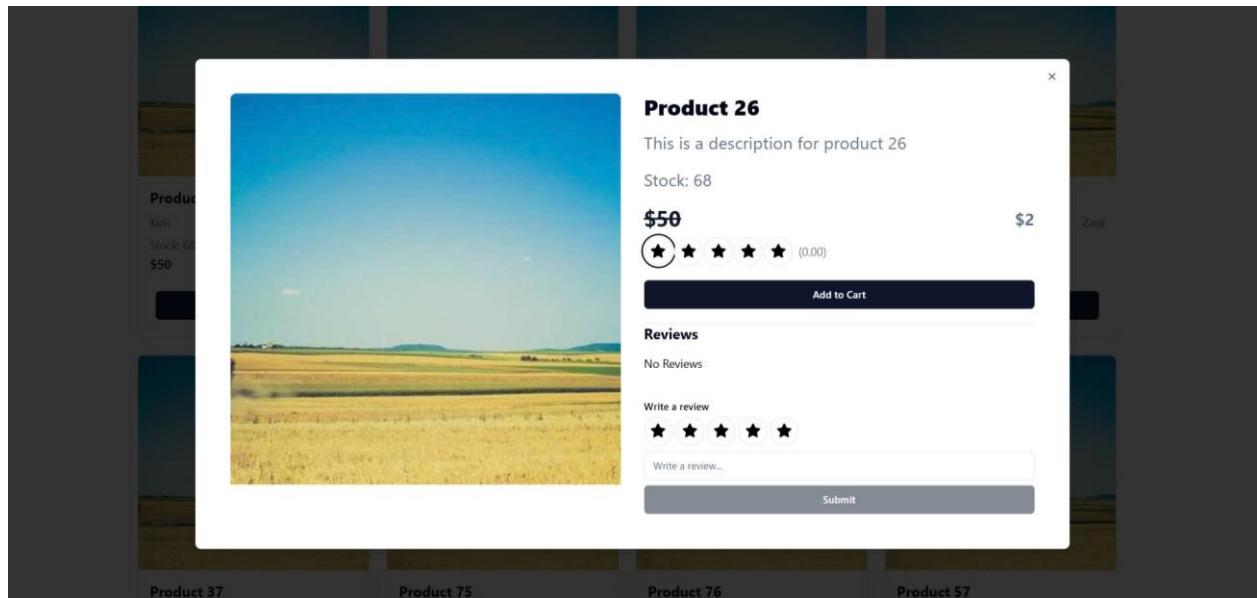


Hình 2.4.16: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.

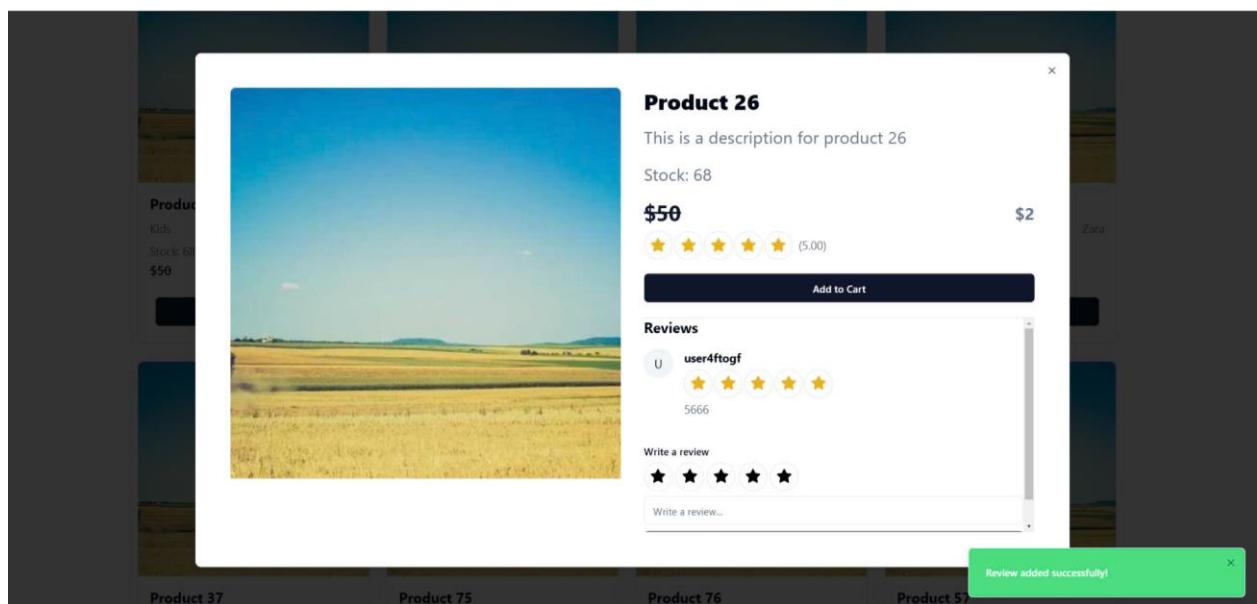
- Đánh giá:
 - + Có 3 hành động thành công trên 2 trình duyệt là Chorme và Microsoft Edge trên 3 hệ điều hành macos,linux,window.
 - + Có 1 hành động thất bại trên trình duyệt safari trên hệ điều hành macos.
- Tổng kết:
 - + Ứng dụng có thể chạy tốt hành động trên 2 trình duyệt và thất bại trên trình duyệt safari.
 - + Lý do thất bại có thể do trình duyệt Safari kích hoạt các chính sách bảo mật nghiêm ngặt liên quan đến cookie gửi tới server dẫn đến thất bại.

2.4.5. Nguyễn Quốc Thái - Đánh giá sản phẩm

2.4.5.1. Phân tích thiết kế kiểm thử



Hình 2.4.17: Giao diện trước khi đánh giá



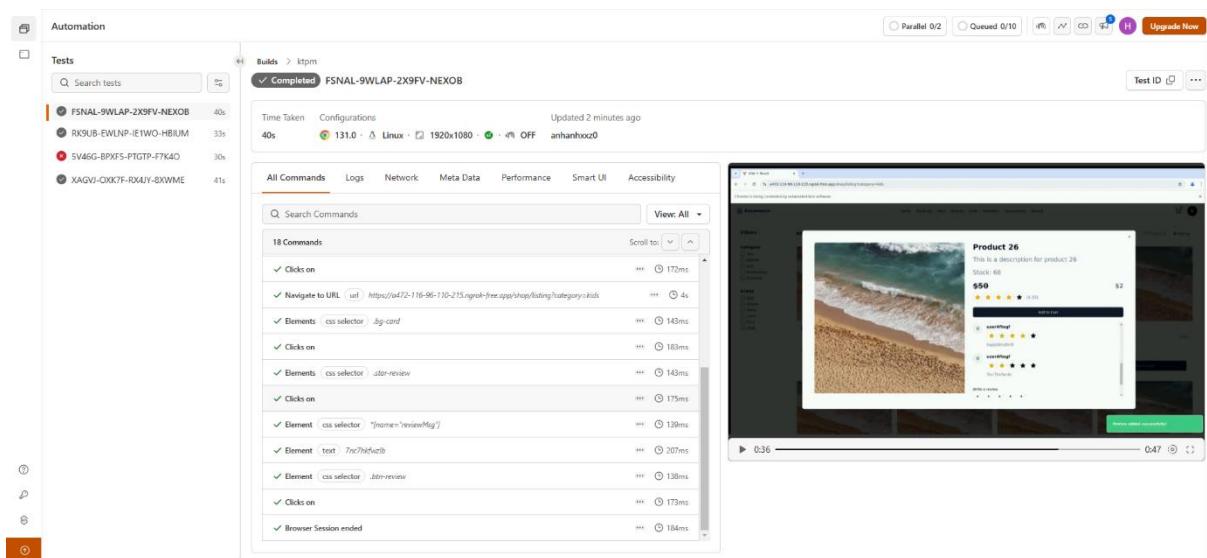
Hình 2.4.18: Giao diện sau khi sửa.

Bảng 2.4.6: Bảng testcase cho chức năng đánh giá sản phẩm

STT	Chức năng	Môi trường	Bước thực hiện	Kết quả mong đợi	Ghi chú
1	Đánh giá sản phẩm từ danh sách	macOS + Safari	1. Truy cập trang đăng nhập. 2. Đăng nhập tài khoản và mật khẩu hợp lệ. 3. Truy cập trang sản phẩm. 4. Chọn sản phẩm trong danh sách và nhấn vào sản phẩm. 5. Chọn số sao và nhập nội dung đánh giá. 6. Nhấn "Submit". 7. Kiểm tra thông báo thành công.	Hệ thống thông báo thành công khi đánh giá sản phẩm.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng "Đánh giá sản phẩm" trên môi trường macOS + Safari.
2		Linux + Chrome		Hệ thống thông báo thành công khi đánh giá sản phẩm.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường Linux + Chrome.
3		Windows + Edge		Hệ thống thông báo thành công khi đánh giá sản phẩm.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường

					Windows + Edge.
4		macOS + Chrome		Hệ thống thông báo thành công khi đánh giá sản phẩm.	Cơ chế tìm test case: Cross-Browser Testing. Độ bao phủ: Bao phủ chức năng trên môi trường macOS + Chrome.

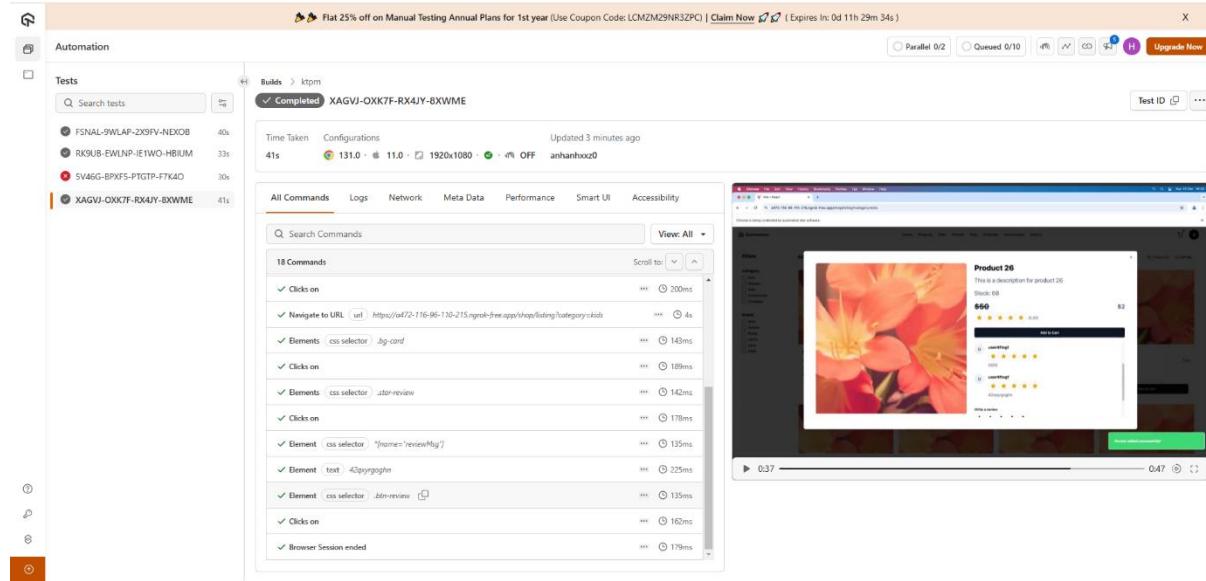
2.4.5.2. Thực hiện kiểm thử



Hình 2.4.19: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.

Hình 2.4.20: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.

Hình 2.4.21: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.



Hình 2.4.22: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.

- Đánh giá:
 - + Có 3 hành động thành công trên 2 trình duyệt là Chorme và Microsoft Edge trên 3 hệ điều hành macos,linux,window.
 - + Có 1 hành động thất bại trên trình duyệt safari trên hệ điều hành macos.
- Tổng kết:
 - + Ứng dụng có thể chạy tốt hành động trên 2 trình duyệt và thất bại trên trình duyệt safari.
 - + Lý do thất bại có thể do trình duyệt Safari kích hoạt các chính sách bảo mật nghiêm ngặt liên quan đến cookie gửi tới server dẫn đến thất bại.

2.4.6. Lê Anh Tạo – Thêm sản phẩm vào giỏ hàng.

2.4.6.1. Phân tích thiết kế kiểm thử

Bảng 2.4.7: Bảng testcase cho chức năng thêm sản phẩm vào giỏ hàng (Postman)

ID	Tên test case	Dữ liệu đầu vào	Kết quả mong đợi	Kết quả thực tế
-----------	----------------------	------------------------	-------------------------	------------------------

TC_01	Kiểm tra khi không gửi đầy đủ dữ liệu đầu vào	{ "userId": "", "productId": "", "quantity": 0 }	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"
TC_02	Kiểm tra khi userId bỏ trống	{ "userId": "", "productId": "6717bf3a96301709fa1931ed", "quantity": 2 }	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"
TC_03	Kiểm tra khi userId là một số thay vì chuỗi	{ "userId": "12345", "productId": "6717bf3a96301709fa1931ed", "quantity": 2 }	Trả về lỗi và hiển thị dòng thông báo:"UserId or ProductId is invalid"	Trả về lỗi và hiển thị dòng thông báo:"UserId or ProductId is invalid"
TC_04	Kiểm tra khi productId rỗng	{ "userId": "6717bf3a96301709fa193188", "productId": "", "quantity": 2 }	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"

TC_05	Kiểm tra khi productId đã tồn tại trong giỏ hàng	{ "userId": "6717bf3a96301709fa193188", "productId": "6717bf3a96301709fa1931ed", "quantity": 2 }	Thêm thành công và tăng số lượng sản phẩm trong giỏ hàng	Thêm thành công và tăng số lượng sản phẩm trong giỏ hàng
TC_06	Kiểm tra khi thêm sản phẩm có quantity = 0	{ "userId": "6717bf3a96301709fa193188", "productId": "6717bf3a96301709fa1931ec", "quantity": 0 }	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"
TC_07	Kiểm tra khi quantity là một chuỗi (thay vì số nguyên)	{ "userId": "6717bf3a96301709fa193188", "productId": "6717bf3a96301709fa1931ec", "quantity": "five" }	Trả về lỗi và hiển thị dòng thông báo:"Error"	Trả về lỗi và hiển thị dòng thông báo:"Error"

TC_08	Kiểm tra khi quantity là số âm	{ "userId": "6717bf3a96301709fa193188", "productId": "6717bf3a96301709fa1931ec", "quantity": -1 }	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"	Trả về lỗi và hiển thị dòng thông báo:"Invalid data provided!"
TC_09	Kiểm tra khi quantity vượt quá số lượng sản phẩm	{ "userId": "6717bf3a96301709fa193188", "productId": "6717bf3a96301709fa1931ec", "quantity": 20 }	Trả về lỗi và hiển thị dòng thông báo:"Product out of Stock"	Trả về lỗi và hiển thị dòng thông báo:"Product out of Stock"

2.4.6.2. Thực hiện kiểm thử

- TC_01: Kiểm tra khi không gửi đầy đủ dữ liệu đầu vào

The screenshot shows the Postman application interface. A new request is being prepared with the following details:

- Method:** POST
- URL:** http://localhost:5000/api/shop/cart/add
- Body:** (JSON) (radio button selected)
- Body Content:**

```

1 {
2   "userId": "",
3   "productId": "",
4   "quantity": 0
5 }
```

After sending the request, the results are displayed:

- Status:** 400 Bad Request
- Time:** 8 ms
- Size:** 1.04 KB
- Response:**

```

1 {
2   "success": false,
3   "message": "Invalid data provided!"
4 }
```

Hình 2.4.23: Hình minh họa test use case 1_Kiểm tra khi không gửi đầy đủ dữ liệu đầu vào

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
- TC_02: Kiểm tra khi userId bỏ trống

The screenshot shows the Postman interface with a collection named 'ktpm / add_to_cart'. A POST request is selected with the URL `http://localhost:5000/api/shop/cart/add`. The 'Body' tab is active, showing the following JSON payload:

```
1 {
2   "userId": "",
3   "productId": "6717bf3a96301709fa1931ec",
4   "quantity": 1
5 }
6 }
```

The response section shows a **400 Bad Request** status with a timestamp of 9 ms and a size of 1.04 KB. The response body is:

```
1 {
2   "success": false,
3   "message": "Invalid data provided!"
4 }
```

Hình 2.4.24: Hình minh họa test use case 2_Kiểm tra khi userId bỏ trống

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
- TC_03: Kiểm tra khi userId là một số thay vì chuỗi

The screenshot shows a Postman interface with a collection named 'add_to_cart'. A POST request is being made to `http://localhost:5000/api/shop/cart/add`. The request body is set to `JSON` and contains the following JSON:

```

1 {
2   "userId": "12345",
3   "productId": "6717bf3a96301709fa1931ec",
4   "quantity": 1
5 }

```

The response status is **400 Bad Request**, with a duration of 7 ms and a size of 1.04 KB. The response body is:

```

1 {
2   "success": false,
3   "message": "UserId or ProductId is invalid"
4 }

```

Hình 2.4.25: Hình minh họa test use case 3_Kiểm tra khi userId là một số thay vì chuỗi

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi userId hoặc productId không tồn tại
 - TC_04: Kiểm tra khi productId rỗng

The screenshot shows a Postman interface with a collection named 'add_to_cart'. A POST request is being made to `http://localhost:5000/api/shop/cart/add`. The request body is set to `JSON` and contains the following JSON:

```

1 {
2   "userId": "6717bf3a96301709fa193188",
3   "productId": "",
4   "quantity": 1
5 }

```

The response status is **400 Bad Request**, with a duration of 5 ms and a size of 1.04 KB. The response body is:

```

1 {
2   "success": false,
3   "message": "Invalid data provided!"
4 }

```

Hình 2.4.26: Hình minh họa test use case 4_Kiểm tra khi productId rỗng

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
 - TC_05: Kiểm tra khi productId đã tồn tại trong giỏ hàng

The screenshot shows a Postman interface with the following details:

- Request URL:** http://localhost:5000/api/shop/cart/add
- Method:** POST
- Body:** JSON (raw)
- Request Body Content:**

```

1  [
2   "userId": "6717bf3a96301709fa1931bb",
3   "productId": "6717bf3a96301709fa1931ed",
4   "quantity": 2
5 ]

```

- Response Status:** 200 OK
- Response Headers:** 17 ms - 1.42 KB
- Response Body (Pretty JSON):**

```

1  [
2   {
3     "productId": "6717bf3a96301709fa1931bb",
4     "quantity": 1,
5     "_id": "673857c05a276ce91af1b77"
6   },
7   {
8     "productId": "6717bf3a96301709fa1931ed",
9     "quantity": 1,
10    "_id": "6758a966c58d647351d48b39"
11  },
12  {
13    "createdAt": "2024-11-16T08:26:13.050Z"
14  }
15 ]

```

Hình 2.4.27: Hình minh họa test use case 5_Kiểm tra khi productId đã tồn tại trong giỏ hàng

- + Kết quả kiểm thử:
 - Sau khi kiểm thử xong, thông báo thêm sản phẩm vào giỏ thành công và tăng số lượng sản phẩm sau mỗi lần thêm
 - TC_06: Kiểm tra khi thêm sản phẩm có quantity = 0

The screenshot shows the Postman interface with a POST request to `http://localhost:5000/api/shop/cart/add`. The request body is:

```

1 {
2   "userId": "6717bf3a96301709fa193188",
3   "productId": "6717bf3a96301709fa1931ed",
4   "quantity": 0
5 }

```

The response status is 400 Bad Request, with the message:

```

1 {
2   "success": false,
3   "message": "Invalid data provided!"
4 }

```

Hình 2.4.28: Hình minh họa test use case 6_Kiểm tra khi thêm sản phẩm có quantity = 0

- Kết quả kiểm thử:
- + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
- TC_07: Kiểm tra khi **quantity** là một chuỗi (thay vì số nguyên)

The screenshot shows the Postman interface with a POST request to `http://localhost:5000/api/shop/cart/add`. The request body is:

```

1 {
2   "userId": "6717bf3a96301709fa193188",
3   "productId": "6717bf3a96301709fa1931ed",
4   "quantity": "five"
5 }

```

The response status is 500 Internal Server Error, with the message:

```

1 {
2   "success": false,
3   "message": "Error"
4 }

```

Hình 2.4.29: Hình minh họa test use case 7_Kiểm tra khi quantity là một chuỗi (thay vì số nguyên)

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
- TC_08: Kiểm tra khi **quantity** là số âm

```
POST http://localhost:5000/api/shop/cart/add
{
  "userId": "6717bf3a96301709fa193180",
  "productId": "6717bf3a96301709fa1931ed",
  "quantity": "-1"
}
```

Body Cookies (1) Headers (21) Test Results 400 Bad Request 4 ms 1.04 KB Save Response

```
1 {
2   "success": false,
3   "message": "Invalid data provided!"
4 }
```

Hình 2.4.30: Hình minh họa test use case 8_Kiểm tra khi quantity là số âm

- Kết quả kiểm thử:
 - + Sau khi kiểm thử xong, thông báo lỗi không thêm vào giỏ hàng được và trả về lỗi
- TC_09: Kiểm tra khi quantity vượt quá số lượng sản phẩm

The screenshot shows the Postman interface with a successful API call. The request URL is `http://localhost:5000/api/shop/cart/add`. The request body is a JSON object:

```

1 {
2   "userId": "6717bf3a96301709fa193180",
3   "productId": "6717bf23a96301709fa1931ec",
4   "quantity": 11
5 }

```

The response status is 200 OK, with a response time of 337 ms and a size of 1.42 KB. The response body is:

```

1 {
2   "success": true,
3   "data": [
4     {
5       "_id": "6738573b5a276ce91afbf16a",
6       "userId": "6717bf3a96301709fa193180",
7       "items": [
8         {
9           "productId": "6717bf3a96301709fa1931ec",
10          "quantity": 15,
11          "_id": "6738573b5a276ce91afbf16b"
12        },
13        {
14          "productId": "6717bf3a96301709fa1931f9"
15        }
16      ]
17    }
18  ]
19 }

```

Hình 2.4.31: Hình minh họa test use case 9_Kiểm tra khi quantity vượt quá số lượng sản phẩm

- Kết quả kiểm thử:
 - + Kiểm thử không thành công khi quantity đã vượt quá stock nhưng vẫn thêm sản phẩm vào trong giỏ hàng

Kết luận: Chức năng thêm giỏ hàng tồn tại một số lỗi.

2.4.7. Đỗ Trọng Hoàng – Thêm sản phẩm mới

2.4.7.1. Phân tích thiết kế kiểm thử

Bảng 2.4.8: Bảng testcase cho chức năng thêm sản phẩm mới (Postman)

STT	Tên test case	Nhập đầu vào	Kết quả mong đợi	Kết quả thực tế
1	Thêm thành công	{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes." }	Hiện thị sản phẩm được thêm mới	Hiện thị sản phẩm mới khi được thêm.

		<pre> "category": "footwear", "brand": "nike", "price": 100, "salePrice": 90, "totalStock": 10, "averageReview": 4.5 } </pre>		
2	Không có quyền truy cập	req.user.role = "user"	Hiển thị dòng thông báo: "Permission denied!"	Hiển thị dòng thông báo: "Permission denied!"
3	Thiếu một số trường quan trọng (image,description,category,brand,price,salePrice,totalStock,averageReview)	<pre> { "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "footwear", "brand": "nike", "price": 100, "salePrice": 90, "totalStock": 10, "averageReview": 4.5 } </pre> <p>Thiếu trường image</p>	Hiển thị dòng thông báo: " All fields are required"	Hiển thị dòng thông báo: "All fields are required"

4	Thể loại không hợp lệ.Không phải một trong các thuộc tính sau :men, women,kids,accessories,footwear.	<pre>{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "electronics", "brand": "nike", "price": 100, "salePrice": 90, "totalStock": 10, "averageReview": 4.5 }</pre>	Hiển thị dòng thông báo: "Category is invalid"	Hiển thị dòng thông báo: "Category is invalid"
5	Hãng không hợp lệ.Không phải một trong các thuộc tính sau :nike,adidas,puma,levi,h&m.	<pre>{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "footwear", "brand": "sony", "price": 100, "salePrice": 90, "totalStock": 10, "averageReview": 4.5 }</pre> <p>Brand khác brand được định nghĩa.</p>	Hiển thị dòng thông báo: "Brand is invalid"	Hiển thị dòng thông báo: "Brand is invalid"

6	Giá hoặc số lượng âm	<pre>{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "footwear", "brand": "nike", "price": -100, "salePrice": 90, "totalStock": -10, "averageReview": 4.5 }</pre> <p>Price âm</p>	Hiển thị dòng thông báo: " Price, Sale Price, Total Stock and Average Review must be greater than 0"	Hiển thị dòng thông báo: " Price, Sale Price, Total Stock and Average Review must be greater than 0"
7	Giá bán phải lớn hơn giá ưu đãi	<pre>{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "footwear", "brand": "nike", "price": 80, "salePrice": 90, "totalStock": 10, "averageReview": 4.5 }</pre> <p>Price = 80 < salePrice = 90</p>	Hiển thị dòng thông báo: " Price must be greater than Sale Price"	Hiển thị dòng thông báo: " Price must be greater than Sale Price"

8	Đánh giá trung bình vượt ngoài giới hạn từ 0-5	<pre>{ "image": "http://image-url.com", "title": "Running Shoes", "description": "Comfortable running shoes.", "category": "footwear", "brand": "nike", "price": 100, "salePrice": 90, "totalStock": 10, "averageReview": 6 } averageReview = 6 > 5</pre>	Hiển thị dòng thông báo: "Average Review must be between 0 and 5"	Hiển thị dòng thông báo: "Average Review must be between 0 and 5"
9	Database bị lỗi. Ví dụ database disconnect		Hiển thị dòng thông báo: "Some error occurred"	Hiển thị dòng thông báo: " Some error occurred"

2.4.7.2. Thực hiện kiểm thử

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspace' containing collections like 'New Collection', 'thuecanho', and 'thuecanho'. The main area shows a collection named 'thuecanho' with a single POST request named 'thêm sản phẩm mới'. The request URL is 'localhost:5000/api/admin/products/add'. The 'Body' tab is selected, showing a raw JSON payload:

```

{
  "image": "http://image-url.com",
  "title": "Running Shoes",
  "description": "Comfortable running shoes.",
  "category": "footwear",
  "brand": "nike",
  "price": 100,
  "salePrice": 90,
  "totalStock": 10,
  "averageReview": 4.5
}

```

Below the body, the response is shown in the 'Pretty' tab:

```

{
  "success": true,
  "data": {
    "image": "http://image-url.com",
    "title": "Running Shoes",
    "description": "Comfortable running shoes.",
    "category": "footwear",
    "brand": "nike",
    "price": 100,
    "salePrice": 90,
    "totalStock": 10,
    "averageReview": 4.5
  }
}

```

The status bar at the bottom indicates a '201 Created' response.

Hình 2.4.32: Hình minh họa test case 1_ Thêm thành công.

The screenshot shows the Postman interface with a collection named 'New Collection'. A POST request is being made to 'localhost:5000/api/admin/products/add'. The request body contains the following JSON:

```

1 {
2   "image": "http://image-url.com",
3   "title": "Running Shoes",
4 }

```

The response status is 403 Forbidden, and the message is "Permission denied!".

Hình 2.4.33: Hình minh họa test case 2_Không có quyền truy cập.

The screenshot shows the Postman interface with a collection named 'New Collection'. A POST request is being made to 'localhost:5000/api/admin/products/add'. The request body contains the following JSON:

```

1 {
2   "name": "Nike",
3   "price": 100,
4   "salePrice": 90,
5   "totalStock": 10,
6   "averageReview": 4.5
7 }

```

The response status is 400 Bad Request, and the message is "All fields are required".

Hình 2.4.34: Hình minh họa test case 3_ Thiếu một số trường quan trọng (image, description, category, brand, price, salePrice, totalStock, averageReview).

POST localhost:5000/api/admin/products/add

```

1 {
2   "success": false,
3   "message": "Category is invalid"
4 }

```

Hình 2.4.35: Hình minh họa test use case 4_ Thẻ loại không hợp lệ.Không phải một trong các thuộc tính sau :men,women,kids,accessories,footwear.

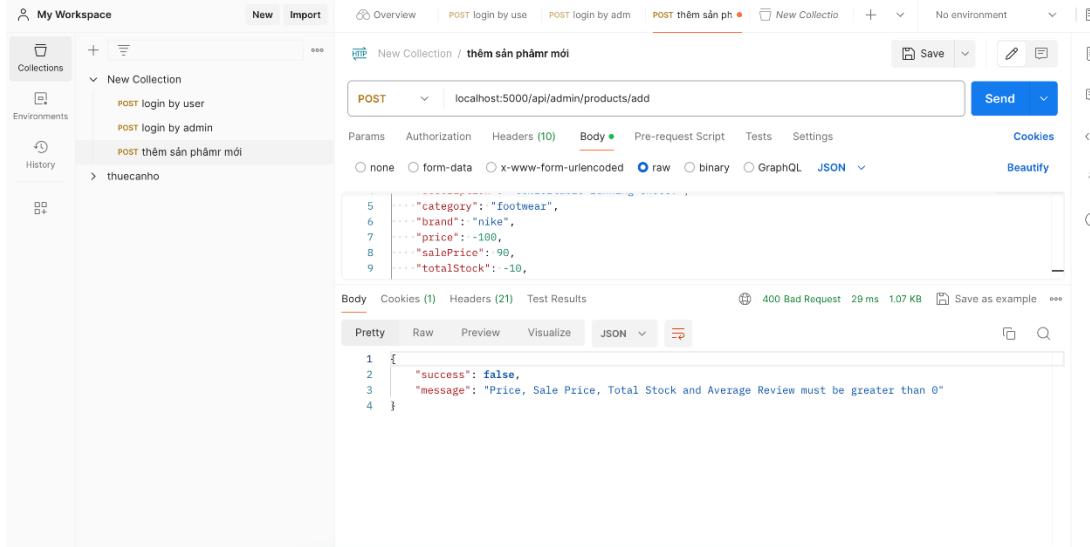
POST localhost:5000/api/admin/products/add

```

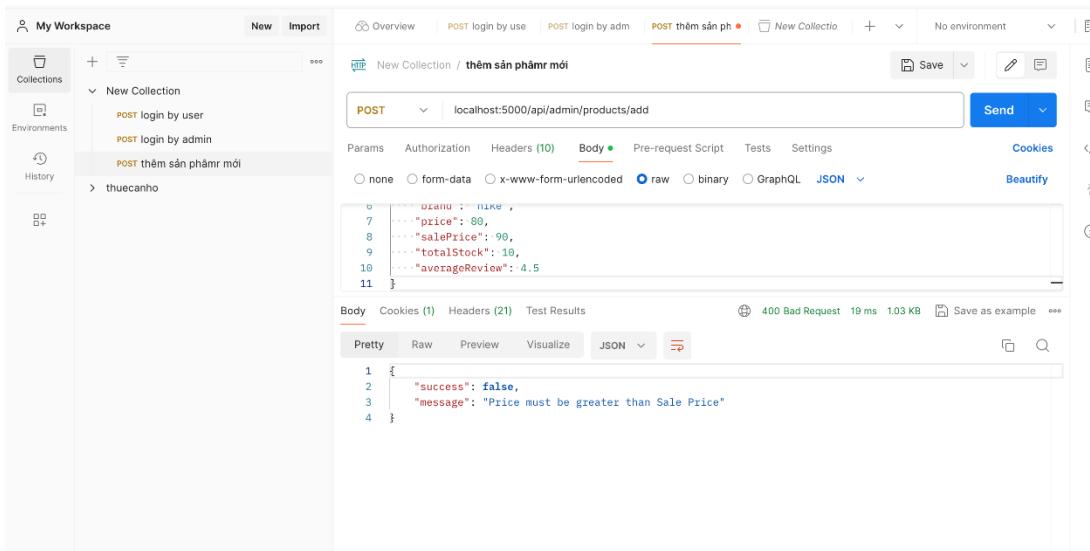
1 {
2   "success": false,
3   "message": "Brand is invalid"
4 }

```

Hình 2.4.36: Hình minh họa test use case 5_ Hàng không hợp lệ.Không phải một trong các thuộc tính sau:nike, adidas, puma, levi, h&m.



Hình 2.4.37: Hình minh họa test use case 6_Giá hoặc số lượng âm.



Hình 2.4.38: Hình minh họa test use case 7_ Giá bán phải lớn hơn giá ưu đãi.

The screenshot shows the Postman interface with a collection named "thuecanho". A POST request is being tested for adding a product. The URL is `localhost:5000/api/admin/products/add`. The request body is JSON:

```

1 {
2   "name": "nike",
3   "price": 100,
4   "salePrice": 90,
5   "totalStock": 10,
6   "averageReview": 6
7 }
  
```

The response status is 400 Bad Request, with the message: "success": false, "message": "Average Review must be between 0 and 5".

Hình 2.4.39: Hình minh họa test use case 8_ Đánh giá trung bình vượt ngoài giới hạn từ 0-5.

The screenshot shows the Postman interface with a collection named "thuecanho". A POST request is being tested for adding a product. The URL is `localhost:5000/api/admin/products/add`. The request body is JSON:

```

1 {
2   "name": "nike",
3   "price": 100,
4   "salePrice": 90,
5   "totalStock": 10,
6   "averageReview": 4.5
7 }
  
```

The response status is 500 Internal Server Error, with the message: "success": false, "message": "Some error occurred".

Hình 2.4.40: Hình minh họa test use case 9_ Database bị lỗi. Ví dụ database disconnect.

Kết luận: Chức năng đăng nhập của API hoạt động tốt

2.4.8. Nguyễn Huy Nhật – Đăng kí

2.4.8.1. Phân tích thiết kế kiểm thử

Bảng 2.4.9: Bảng testcase cho chức năng đăng kí (Postman)

STT	Dữ liệu đầu vào	Kết quả mong muốn	Kết quả thực tế
TC_01	{ "userName":"", "email":"", "password":"" }	{ "success": false "message": "Please provide both email and password" }	{ "success": false "message": "Please provide both email and password" }
TC_02	{ "userName":"testUser", "email":"invalidEmail", "password":"password123" }	{ "success": false "message": "Please provide a valid email address" }	{ "success": false "message": "Please provide a valid email address" }
TC_03	{ "userName":"testUser", "email":"huynhat0606@gmail.com", "password":"23@" }	{ "success": false "message": "Password must have at least 6 characters! Please try again" }	{ "success": false "message": "Password must have at least 6 characters! Please try again" }
TC_04	{ "userName":"testUser67", "email":"hai34456@example.com", }	{ "success": false "message": "Registration successful" }	{ "success": false "message": "Registration successful" }

	<pre>"password":"password123" }</pre>		
TC_05	<pre>{ "userNmae": "huynhat66", "email": "test0@dev.com", "password": "anhanh" }</pre>	<pre>{ "success": false "message": "Email already exist!" }</pre>	<pre>{ "success": false "message": "Email already exist!" }</pre>
TC_06	<pre>{ "userNmae": "huynhat66", "email": "abc12345@mail.com", "password": "anhanh" }</pre>	<pre>{ "success": false "message": "Some error occured" }</pre>	<pre>{ "success": false "message": "Some error occured" }</pre>

2.4.8.2. Thực hiện kiểm thử

- Testcase 01: Nhập thiếu dữ liệu đầu vào

The screenshot shows a POST request to `localhost:5000/api/auth/register`. The request body is JSON with fields `userName`, `email`, and `password` all set to empty strings. The response is a 200 OK status with a message indicating both email and password are required.

```

1
2   "userName": "",
3   "email": "",
4   "password": ""
      
```

Body Cookies Headers (21) Test Results 200 OK 45 ms 1.02 KB Save Response

Hình 2.4.41: Hình minh họa test use case 1_Nhập thiếu dữ liệu đầu vào

- Testcase 02: Kiểm tra định dạng của email khi đăng ký

The screenshot shows a POST request to `localhost:5000/api/auth/register`. The request body is JSON with `userName` set to `testUser`, `email` set to `invalidEmail`, and `password` set to `password123`. The response is a 400 Bad Request status with a message indicating a valid email address is required.

```

1
2   "userName": "testUser",
3   "email": "invalidEmail",
4   "password": "password123"
      
```

Body Cookies Headers (21) Test Results 400 Bad Request 11 ms 1.03 KB Save Response

Hình 2.4.42: Hình minh họa test use case 2_Kiểm tra định dạng của email

- Testcase 03: Kiểm tra độ dài mật khẩu khi đăng ký

HTTP TX1 / test2

POST localhost:5000/api/auth/register

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

```

1 {
2   "userName": "testUser",
3   "email": "huynhat0606@gmail.com",
4   "password": "23@"
5 }
```

Body Cookies Headers (21) Test Results

401 Unauthorized • 15 ms • 1.05 KB • Save Response ⚙️

Pretty Raw Preview Visualize JSON

```

1 {
2   "success": false,
3   "message": "Password must have at least 6 characters! Please try again"
4 }
```

Hình 2.4.43: Hình minh họa test use case 3_Kiểm tra độ dài mật khẩu

- Testcase 04: Kiểm tra đăng ký thành công

HTTP TX1 / test2

POST localhost:5000/api/auth/register

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

```

1 {
2   "userName": "testUser67",
3   "email": "hai34456@example.com",
4   "password": "password123"
5 }
```

Body Cookies Headers (21) Test Results

200 OK • 563 ms • 1.01 KB • Save Response ⚙️

Pretty Raw Preview Visualize JSON

```

1 {
2   "success": true,
3   "message": "Registration successful"
4 }
```

Hình 2.4.44: Hình minh họa test use case 4 _đăng kí thành công

- Testcase 05: Kiểm tra email đã tồn tại hay chưa khi đăng ký

HTTP TX1 / test2

POST localhost:5000/api/auth/register

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

```

1 {
2   "email": "test0@dev.com",
3   "userName": "huynhat66",
4   "password": "anhanh"
5 }
```

Body Cookies Headers (21) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "success": false,
3   "message": "Email already exist!"
4 }
```

401 Unauthorized 19 ms 1.02 KB Save Response

Hình 2.4.45: Hình minh họa test use case 6_Kiểm tra email đã tồn tại

- Testcase 06: Kiểm tra khi nhập trùng username

HTTP TX1 / test2

POST localhost:5000/api/auth/register

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

```

1 {
2   "email": "abc12345@gmail.com",
3   "userName": "huynhat66",
4   "password": "anhanh"
5 }
```

Body Cookies Headers (21) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "success": false,
3   "message": "Some error occurred"
4 }
```

500 Internal Server Error 540 ms 1.02 KB Save Response

Hình 2.4.46: Hình minh họa test use case 6_Kiểm tra khi nhập trùng username

Kết luận: Chức năng đăng ký của website hoạt động tốt

2.4.9. Nguyễn Quốc Thái – Đăng nhập

2.4.9.1. Phân tích thiết kế kiểm thử

Bảng 2.4.10: Bảng testcase cho chức năng đăng nhập (Postman)

STT	Tên test case	Nhập đầu vào	Kết quả mong đợi	Kết quả thực tế
1	Để trống tài khoản và mật khẩu	{ "email":"", "password":"" }	Hiển thị dòng thông báo: "Please provide both email and password "	Hiển thị dòng thông báo: "Please provide both email and password "
2	Để trống tài khoản, nhập mật khẩu	{ "email":"", "password":"anhanh" }	Hiển thị dòng thông báo: "Please provide both email and password "	Hiển thị dòng thông báo: "Please provide both email and password "
3	Nhập tài khoản, bỏ trống mật khẩu	{ "email":"admin@ev.com", "password":"" }	Hiển thị dòng thông báo: "Please provide both email and password "	Hiển thị dòng thông báo: "Please provide both email and password "
4	Nhập tài khoản thiếu "@"	{ "email":"adminnev.co", "password":"anhanh" }	Hiển thị dòng thông báo: "Please provide a valid email "	Hiển thị dòng thông báo: "Please provide a valid email "

5	Nhập đúng tài khoản và mật khẩu	<pre>{ "email": "admin@dev.com", "password": "anhanh" }</pre>	<p>Hiển thị dòng thông báo:</p> <p>"Logged in successfully" và khai báo ra "user": {</p> <p> "email": "admin@dev.com",</p> <p> "role": "admin",</p> <p> "id": "6717d947ff7ca805b899fe48",</p> <p> "userName": "user1sxlnp"</p> <p>}</p>	<p>Hiển thị dòng thông báo:</p> <p>"Logged in successfully" và khai báo ra "user": {</p> <p> "email": "admin@dev.com",</p> <p> "role": "admin",</p> <p> "id": "6717d947ff7ca805b899fe48",</p> <p> "userName": "user1sxlnp"</p> <p>}</p>
6	Nhập sai tài khoản và nhập mật khẩu	<pre>{ "email": "admin1@dev.com", "password": "anhanh1" }</pre>	<p>Hiển thị dòng thông báo: "User doesn't exists! Please register first"</p>	<p>Hiển thị dòng thông báo: "User doesn't exists! Please register first"</p>
7	Nhập đúng tài khoản và nhập sai mật khẩu	<pre>{ "email": "admin@dev.com", "password": "anhanh1" }</pre>	<p>Hiển thị dòng thông báo: "Incorrect password! Please try again"</p>	<p>Hiển thị dòng thông báo: "Incorrect password! Please try again"</p>

8	Nhập sai tài khoản và nhập sai mật khẩu	{ "email": "admin1@dev.com", "password": "anhanh1" }	Hiển thị dòng thông báo: "User doesn't exists! Please register first"	Hiển thị dòng thông báo: "User doesn't exists! Please register first"
---	---	--	---	---

2.4.9.2. Thực hiện kiểm thử

- Testcase 01: Để trống tài khoản và mật khẩu

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** http://localhost:5000/api/auth/login
- Body:** Raw JSON (Pretty) - Contains the following payload:


```

1 {
2   "email": "",
3   "password": ""
4 }
```
- Response Headers:** 200 OK, 14 ms, 1.02 KB
- Response Body:** (Pretty) - Contains the following JSON:


```

1 {
2   "success": false,
3   "message": "Please provide both email and password"
4 }
```

Hình 2.4.47: Hình minh họa test use case 1 _Để trống email, mật khẩu

- Testcase 02: Để trống tài khoản, nhập mật khẩu

The screenshot shows the Postman interface. A POST request is being sent to `http://localhost:5000/api/auth/login`. The request body is a JSON object with two fields: `"email": ""` and `"password": "anhanh"`. The response status is `200 OK`, and the response body is a JSON object with `"success": false` and `"message": "Please provide both email and password"`.

```

1 {
2   "email": "",
3   "password": "anhanh"
4 }
    
```

```

1 {
2   "success": false,
3   "message": "Please provide both email and password"
4 }
    
```

Hình 2.4.48: Hình minh họa test use case 2 _Để trống email, nhập mật khẩu

- Testcase 03: Nhập tài khoản, bỏ trống mật khẩu

The screenshot shows the Postman interface. A POST request is being sent to `http://localhost:5000/api/auth/login`. The request body is a JSON object with `"email": "admin@dev.com"` and an empty `"password": ""`. The response status is `200 OK`, and the response body is a JSON object with `"success": false` and `"message": "Please provide both email and password"`.

```

1 {
2   "email": "admin@dev.com",
3   "password": ""
4 }
    
```

```

1 {
2   "success": false,
3   "message": "Please provide both email and password"
4 }
    
```

Hình 2.4.49: Hình minh họa test use case 3 _Nhập email, để trống mật khẩu

- Testcase 04: Nhập tài khoản thiếu “@”

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The request body is a JSON object with `"email": "admindev.com"` and `"password": "anhanh"`. The response status is `200 OK`, but the response body indicates failure with the message `"Please provide a valid email"`.

```

1 {
2   "email": "admindev.com",
3   "password": "anhanh"
4 }
    
```

```

1 {
2   "success": false,
3   "message": "Please provide a valid email"
4 }
    
```

Hình 2.4.50: Hình minh họa test use case 4_Nhập email thiếu @

- Testcase 05: Nhập đúng tài khoản và mật khẩu

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The request body is a JSON object with `"email": "admin@dev.com"` and `"password": "anhanh"`. The response status is `200 OK`, and the response body indicates success with a message and user details.

```

1 {
2   "email": "admin@dev.com",
3   "password": "anhanh"
4 }
    
```

```

1 {
2   "success": true,
3   "message": "Logged in successfully",
4   "user": {
5     "email": "admin@dev.com",
6     "role": "admin",
7     "id": "6717d947ff7ca805b899fe48",
8     "userName": "user1sxlnp"
9   }
10 }
    
```

Hình 2.4.51: Hình minh họa test use case 5_Nhập đúng email, mật khẩu

- Testcase 06: Nhập sai tài khoản và nhập mật khẩu

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The request body is JSON:

```

1 {
2   "email": "admin1@dev.com",
3   "password": "anhanh1"
4 }

```

The response status is `200 OK`, time `16 ms`, size `1.03 KB`. The response body is:

```

1 {
2   "success": false,
3   "message": "User doesn't exists! Please register first"
4 }

```

Hình 2.4.52: Hình minh họa test use case 6_Nhập sai email, mật khẩu

- Testcase 07: Nhập đúng tài khoản và nhập sai mật khẩu

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The request body is JSON:

```

1 {
2   "email": "admin@dev.com",
3   "password": "anhanh1"
4 }

```

The response status is `200 OK`, time `1.26 s`, size `1.02 KB`. The response body is:

```

1 {
2   "success": false,
3   "message": "Incorrect password! Please try again"
4 }

```

Hình 2.4.53: Hình minh họa test use case 7_Nhập đúng email, sai mật khẩu

- Testcase 08: Nhập sai tài khoản và nhập sai mật khẩu

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The request body is a JSON object:

```

1  {
2    "email": "admin1@dev.com",
3    "password": "anhanh"
4  }

```

The response is a 200 OK status with the following JSON data:

```

1  {
2    "success": false,
3    "message": "User doesn't exist! Please register first"
4  }

```

Hình 2.4.54: Hình minh họa test use case 8_Nhập sai tài khoản đúng mật khẩu

Kết luận: Chức năng đăng nhập của API hoạt động tốt

2.4.10. Báo cáo kiểm thử

Bảng 2.4.11: Bảng kết quả kiểm thử

STT	Chức năng	Trạng thái	Lỗi	Ghi chú
1	Thêm sản phẩm vào giỏ hàng	Passed 3/4	Tại testcase 1 lỗi không thêm được sản phẩm	- Lý do thất bại có thể do trình duyệt Safari kích hoạt các chính sách bảo mật nghiêm ngặt liên quan đến
2	Xóa sản phẩm	Passed 3/4	Tại testcase 1 lỗi không xóa được sản phẩm	
3	Chỉnh sửa sản phẩm	Passed 3/4	Tại testcase 1 lỗi không sửa được sản phẩm	

4	Đánh giá sản phẩm	Passed 3/4	Tại testcase 1 lỗi không đánh giá được sản phẩm	cookie gửi tới server dẫn đến thất bại.
5	Thêm sản phẩm vào giỏ hàng	Passed 9/9	Không	Không
6	Thêm sản phẩm mới	Passed 9/9	Không	Không
7	Đăng kí	Passed 6/6	Không	Không
8	Đăng nhập	Passed 8/8	Không	Không

Kết luận:

- Chức năng sẵn sàng bàn giao: thêm sản phẩm vào giỏ hàng, thêm sản phẩm mới, đăng kí, đăng nhập.
- Chức năng cần sửa lỗi: thêm sản phẩm vào giỏ hàng (phía client), xóa sản phẩm(phía client), chỉnh sửa sản phẩm(phía client) và đánh giá sản phẩm(phía client).

Đánh giá tổng quát: Trang web hiện tại đã đáp ứng được 90% yêu cầu chức năng trên các trình duyệt chrome, edge. Cần sớm khắc phục các chức năng trên trình duyệt safari trước khi bàn giao.

PHẦN 3. KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM

3.1. Kiến thức và kỹ năng

- Sau khi kết thúc học phần “Kiểm thử phần mềm”, nhóm 16 đã đạt được những kiến thức và kỹ năng quý báu sau:
 - + Hiểu rõ các khái niệm cơ bản về kiểm thử phần mềm, bao gồm các loại kiểm thử khác nhau như kiểm thử API, kiểm thử tương thích,...
 - + Hiểu biết về quy trình kiểm thử phần mềm và vai trò của kiểm thử trong vòng đời phát triển phần mềm.

- + Kỹ năng lập kế hoạch kiểm thử, bao gồm việc xác định phạm vi kiểm thử, lập lịch trình kiểm thử, và phân bổ nguồn lực.
- + Kỹ năng phân tích kết quả kiểm thử để đánh giá chất lượng phần mềm, bao gồm việc xác định các lỗi, đánh giá mức độ nghiêm trọng của lỗi, và đề xuất các biện pháp cải thiện.

3.2. Chuẩn đầu ra đạt được

- CDR L1: Lựa chọn quy trình và lập kế hoạch Kiểm thử phần mềm.
- CDR L2: Lựa chọn các kỹ thuật, phương pháp và chiến lược kiểm thử phần mềm hiện đại vào bài toán thực tiễn.
- CDR L3: Đánh giá hoạt động nhóm và giải pháp sử dụng công cụ trong kiểm thử phần mềm tự động hiệu quả.

3.3. Bài học kinh nghiệm

- Xác định các yêu cầu chức năng của website và tạo các ca kiểm thử tương ứng.
- Trau dồi kiến thức về cách xây dựng và hoạt động của một website.
- Nâng cao kỹ năng làm việc nhóm.

3.4. Hướng phát triển

Ngay khi được tìm hiểu, học tập, nghiên cứu về kiểm thử, chúng em đã có một cái nhìn sâu sắc về tầm quan trọng của kiểm thử phần mềm. Sự áp dụng kiến thức đã học, tìm hiểu vào bài tập lớn này chỉ là một phần nhỏ của việc kiểm thử phần mềm. Chính vì thế, chúng em có một số hướng phát triển là:

- Tìm hiểu và nghiên cứu nhiều kỹ thuật và công cụ kiểm thử hơn
- Tận dụng trí tuệ nhân tạo và machine learning vào kiểm thử

Danh mục hình ảnh

Hình 2.1.1: Các giai đoạn kiểm thử.....	6
Hình 2.1.2: Minh họa kiểm thử hộp trắng	7
Hình 2.1.3: Minh họa kiểm thử hộp đen.....	7
Hình 2.1.4: Minh họa kiểm thử tương thích.....	8
Hình 2.2.1: Các hệ điều hành LambdaTest hỗ trợ.....	14
Hình 2.3.1: Trang đăng nhập	15
Hình 2.3.2: Trang chủ sau khi đăng nhập.....	16
Hình 2.3.3: Giao diện phân loại sản phẩm của trang web.....	16
Hình 2.3.4: Danh sách các sản phẩm.....	17
Hình 2.3.5: Danh sách các sản phẩm.....	17
Hình 2.3.6: Chi tiết sản phẩm	18
Hình 2.3.7: Giao diện giỏ hàng.....	18
Hình 2.3.8: Giao diện đặt hàng.....	19
Hình 2.3.9: Giao diện trang quản trị.....	19
Hình 2.3.10: Giao diện trang quản trị sản phẩm.....	20
Hình 2.3.11: Giao diện trang thêm sản phẩm mới.....	20
Hình 2.3.12: Giao diện trang quản trị đơn hàng.....	21
Hình 2.3.13: Các usecase chính của hệ thống.	21
Hình 2.3.14: Biểu đồ thực thể.....	22
Hình 2.4.1: Giao diện mong muốn khi kiểm thử tự động.....	25
Hình 2.4.2: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.	27
Hình 2.4.3: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.	28
Hình 2.4.4: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.	28
Hình 2.4.5: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.	29
Hình 2.4.6: Giao diện mong muốn khi xóa thành công.....	30
Hình 2.4.7: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.	32

Hình 2.4.8: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.	32
Hình 2.4.9: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari.	33
Hình 2.4.10: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.	33
Hình 2.4.11: Giao diện trước khi sửa sản phẩm	34
Hình 2.4.12: Giao diện sau khi sửa thành công.....	35
Hình 2.4.13: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.	36
Hình 2.4.14: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.	37
Hình 2.4.15: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari. .	37
Hình 2.4.16: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.	38
Hình 2.4.17: Giao diện trước khi đánh giá	39
Hình 2.4.18: Giao diện sau khi sửa.....	39
Hình 2.4.19: Thực thi thành công trên hệ điều hành linux và trình duyệt chrome.	41
Hình 2.4.20: Thực thi thành công trên hệ điều hành window và trình duyệt Microsoft Edge.	42
Hình 2.4.21: Thực thi thất bại trên hệ điều hành macos và trình duyệt safari. .	42
Hình 2.4.22: Thực thi thành công trên hệ điều hành macos và trình duyệt chrome.	43
Hình 2.4.23: Hình minh họa test use case 1_Kiểm tra khi không gửi đầy đủ dữ liệu đầu vào	47
Hình 2.4.24: Hình minh họa test use case 2_Kiểm tra khi userId bỏ trống	47
Hình 2.4.25: Hình minh họa test use case 3_Kiểm tra khi userId là một số thay vì chuỗi.....	48
Hình 2.4.26: Hình minh họa test use case 4_Kiểm tra khi productId rỗng	48
Hình 2.4.27: Hình minh họa test use case 5_Kiểm tra khi productId đã tồn tại trong giỏ hàng	49
Hình 2.4.28: Hình minh họa test use case 6_Kiểm tra khi thêm sản phẩm có quantity = 0	50

Hình 2.4.29: Hình minh họa test use case 7_Kiểm tra khi quantity là một chuỗi (thay vì số nguyên)	51
Hình 2.4.30: Hình minh họa test use case 8_Kiểm tra khi quantity là số âm ...	51
Hình 2.4.31: Hình minh họa test use case 9_Kiểm tra khi quantity vượt quá số lượng sản phẩm.....	52
Hình 2.4.32: Hình minh họa test case 1_ Thêm thành công.....	56
Hình 2.4.33: Hình minh họa test case 2_Không có quyền truy cập.	57
Hình 2.4.34: Hình minh họa test case 3_ Thiếu một số trường quan trọng (image, description, category, brand, price, salePrice, totalStock, averageReview).....	57
Hình 2.4.35: Hình minh họa test use case 4_ Thể loại không hợp lệ.Không phải một trong các thuộc tính sau :men,women,kids,accessories,footwear.....	58
Hình 2.4.36: Hình minh họa test use case 5_ Hàng không hợp lệ.Không phải một trong các thuộc tính sau:nike, adidas, puma, levi, h&m.....	58
Hình 2.4.37: Hình minh họa test use case 6_ Giá hoặc số lượng âm.	59
Hình 2.4.38: Hình minh họa test use case 7_ Giá bán phải lớn hơn giá ưu đãi.	59
Hình 2.4.39: Hình minh họa test use case 8_ Đánh giá trung bình vượt ngoài giới hạn từ 0-5.	60
Hình 2.4.40: Hình minh họa test use case 9_ Database bị lỗi. Ví dụ database disconnect.	60
Hình 2.4.41: Hình minh họa test use case 1_Nhập thiếu dữ liệu đầu vào.....	63
Hình 2.4.42: Hình minh họa test use case 2_ Kiểm tra định dạng của email....	63
Hình 2.4.43: Hình minh họa test use case 3_Kiểm tra độ dài mật khẩu	64
Hình 2.4.44: Hình minh họa test use case 4_đăng ký thành công.....	64
Hình 2.4.45: Hình minh họa test use case 6_Kiểm tra email đã tồn tại	65
Hình 2.4.46: Hình minh họa test use case 6_Kiểm tra khi nhập trùng username	65
Hình 2.4.47: Hình minh họa test use case 1_Để trống email, mật khẩu	68
Hình 2.4.48: Hình minh họa test use case 2_Để trống email, nhập mật khẩu...69	69
Hình 2.4.49: Hình minh họa test use case 3_Nhập email, để trống mật khẩu...69	69
Hình 2.4.50: Hình minh họa test use case 4_Nhập email thiếu @	70
Hình 2.4.51: Hình minh họa test use case 5_Nhập đúng email, mật khẩu.....70	70
Hình 2.4.52: Hình minh họa test use case 6_Nhập sai email, mật khẩu	71

Hình 2.4.53: Hình minh họa test use case 7_Nhập đúng email, sai mật khẩu ..71
Hình 2.4.54: Hình minh họa test use case 8_Nhập sai tài khoản đúng mật khẩu72

Danh mục bảng biểu

Bảng 2.4.1: Bảng phân công nhiệm vụ cụ thể	24
Bảng 2.4.2: Bảng phân công nhiệm vụ (theo công cụ)	24
Bảng 2.4.3: Bảng testcase cho chức năng thêm sản phẩm vào giỏ hàng	25
Bảng 2.4.4: Bảng testcase cho chức năng xóa sản phẩm	30
Bảng 2.4.5: Bảng testcase cho chức năng chỉnh sửa sản phẩm.....	35
Bảng 2.4.6: Bảng testcase cho chức năng đánh giá sản phẩm	40
Bảng 2.4.7: Bảng testcase cho chức năng thêm sản phẩm vào giỏ hàng (Postman)	43
Bảng 2.4.8: Bảng testcase cho chức năng thêm sản phẩm mới (Postman)	52
Bảng 2.4.9: Bảng testcase cho chức năng đăng kí (Postman)	61
Bảng 2.4.10: Bảng testcase cho chức năng đăng nhập (Postman)	66
Bảng 2.4.11: Bảng kết quả kiểm thử	72

TÀI LIỆU THAM KHẢO

- [1] <https://www.guru99.com/what-is-backend-testing.html> - Truy cập lần cuối 18/12/2024.
- [2] <https://www.guru99.com/frontend-testing.html> - Truy cập lần cuối 18/12/2024.
- [3] <https://qlht.hau.edu.vn/> - Truy cập lần cuối 18/12/2024.