Vietnam National University- Ho Chi Minh City

University of Science

Faculty of Information Technology

HW04. GUI USABILITY TESTING

Course Software Testing

Class 22KTPM3

Group Group10

Student **20127233 – Huỳnh Thế Long**
22127225 – Trần Thị Thiên Kim
22127312 – Nguyễn Thị Yến Nhi
22127316 – Nguyễn Ngô Ngọc Như

Ho Chi Minh City, 2025

Mục lục

1 Phân chia chức năng

MSSV	Sinh Viên	Chức năng	Màn hình
20127233	Huỳnh Thế Long	Product Details & Related Products	Product List Screen
22127225	Trần Thị Thiên Kim	Shopping Cart Operation	Cart Screen
22127312	Nguyễn Thị Yến Nhi	Contact Form with File Upload	Contact Form Screen
22127316	Nguyễn Ngô Như Ngọc	User Registration	Registration Screen

2 Giới thiệu

Trong quá trình phát triển phần mềm, việc kiểm thử giao diện người dùng (GUI Testing) đóng vai trò quan trọng nhằm đảm bảo rằng sản phẩm cuối cùng thân thiện, dễ sử dụng và không xảy ra lỗi hiển thị hay chức năng. Tuy nhiên, việc kiểm thử giao diện thủ công thường tốn nhiều thời gian, dễ bỏ sót lỗi và phụ thuộc nhiều vào kinh nghiệm cá nhân.

Nhờ sự phát triển của trí tuệ nhân tạo (AI), đặc biệt là các mô hình sinh ngôn ngữ như Gemini, quá trình kiểm thử GUI có thể được hỗ trợ hiệu quả hơn. Các công cụ AI có thể nhanh chóng phân tích ảnh chụp giao diện, checklist và gợi ý các lỗi giao diện, các điểm không nhất quán hoặc những chỗ chưa tuân thủ tiêu chuẩn thiết kế.

Trong phần này, công cụ **Gemini** sẽ được dùng để hỗ trợ kiểm thử giao diện màn hình **Trang thanh toán giỏ hàng (Checkout Screen)** thuộc phần mềm *Toolshop*. Dựa trên checklist đã chuẩn bị và ảnh chụp màn hình thực tế, nhóm đã tiến hành đối chiếu, phát hiện lỗi, bổ sung test case và ghi nhận kết quả kiểm thử.

Quy trình kiểm thử kết hợp AI và kiểm thử thủ công giúp đảm bảo chất lượng giao diện, phát hiện lỗi chính xác hơn, đồng thời tiết kiệm thời gian và công sức so với kiểm thử hoàn toàn thủ công.

3 Công cụ sử dụng: Gemini 2.5 Flash

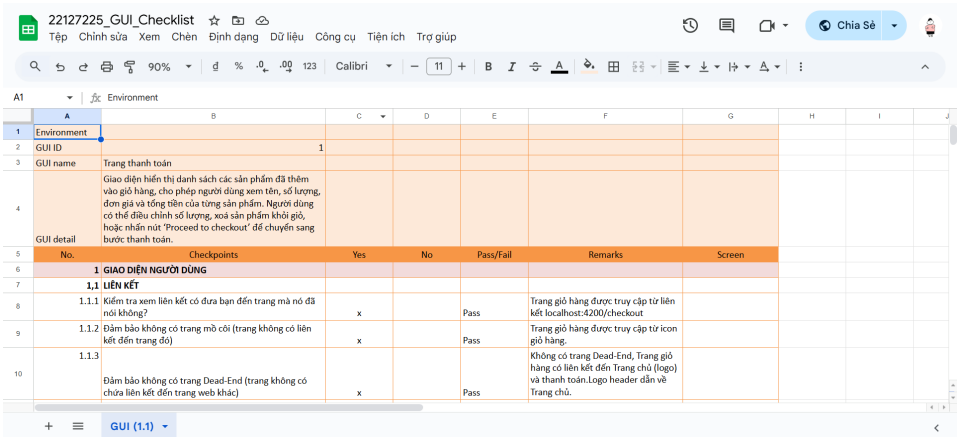
Trong quá trình kiểm thử giao diện có hỗ trợ AI, công cụ **Gemini 2.5 Flash** được sủ dụng – một phiên bản thuộc dòng mô hình ngôn ngữ lớn (LLM) do Google phát triển, nằm trong hệ sinh thái của Google Bard (hiện nay gọi là Gemini).

Gemini 2.5 Flash là một biến thể nhẹ, được tối ưu hóa cho tốc độ phản hồi nhanh, phù hợp để xử lý các tác vụ phân tích văn bản và hỗ trợ đánh giá checklist hiệu quả trong thời gian ngắn. Mặc dù nhẹ hơn so với các bản Gemini Pro, Gemini Flash vẫn có khả năng hiểu ngữ cảnh sâu, suy luận logic, và đưa ra đề xuất cải tiến hợp lý khi được cung cấp dữ liệu kiểm thử phù hợp.

- **Ưu điểm:** Phản hồi nhanh, dễ tích hợp trong quy trình kiểm thử giao diện, giao diện thân thiện (tương thích với tài khoản Google).
- **Vai trò trong báo cáo này:** Gemini được dùng để **review checklist thủ công do nhóm tự tạo**, đưa ra nhận xét, cảnh báo các mục không rõ ràng, gợi ý bổ sung test case còn thiếu và xác nhận logic hợp lý của checklist.

4 Dữ liệu đầu vào cung cấp cho AI

- Checklist kiểm thử GUI (22127225_GUI_Checklist.xlsx).



22127225_GUI_Checklist

Tệp Chỉnh sửa Xem Chèn Định dạng Dữ liệu Công cụ Tiện ích Trợ giúp

Chia Sẻ

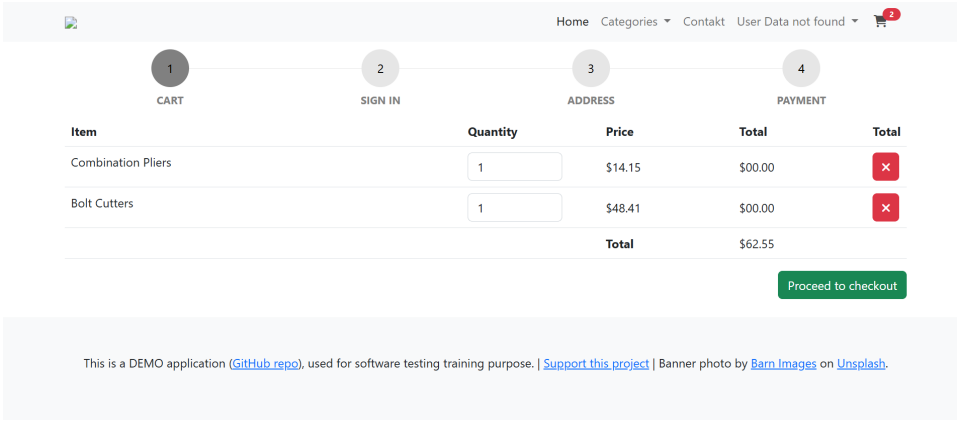
A1 Environment

	A	B	C	D	E	F	G
1	Environment						
2	GUI ID	1					
3	GUI name	Trang thanh toán					
4	GUI detail	Giao diện hiển thị danh sách các sản phẩm đã thêm vào giỏ hàng, cho phép người dùng xem tên, số lượng, đơn giá và tổng tiền của từng sản phẩm. Người dùng có thể điều chỉnh số lượng, xoá sản phẩm khỏi giỏ, hoặc nhấn nút 'Proceed to checkout' để chuyển sang bước thanh toán.					
5	No.	Checkpoints	Yes	No	Pass/Fail	Remarks	Screen
6	1	GIAO DIỆN NGƯỜI DÙNG					
7	1,1	LIÊN KẾT					
8	1.1.1	Kiểm tra xem liên kết có đưa bạn đến trang mà nó đã nói không?	x		Pass	Trang giỏ hàng được truy cập từ liên kết localhost:4200/checkout	
9	1.1.2	Đảm bảo không có trang mồ côi (trang không có liên kết đến trang đó)	x		Pass	Trang giỏ hàng được truy cập từ icon giỏ hàng.	
10	1.1.3	Đảm bảo không có trang Dead-End (trang không có chứa liên kết đến trang web khác)	x		Pass	Không có trang Dead-End, Trang giỏ hàng có liên kết đến Trang chủ (logo) và thanh toán.Logo header dẫn về Trang chủ.	

GUI (1.1)

Hình 1: Checklist kiểm thử GUI

- Ảnh chụp màn hình của giao diện thanh toán (checkout) từ hệ thống Toolshop.



Hình 2: Giao diện Trang thanh toán trong giỏ hàng

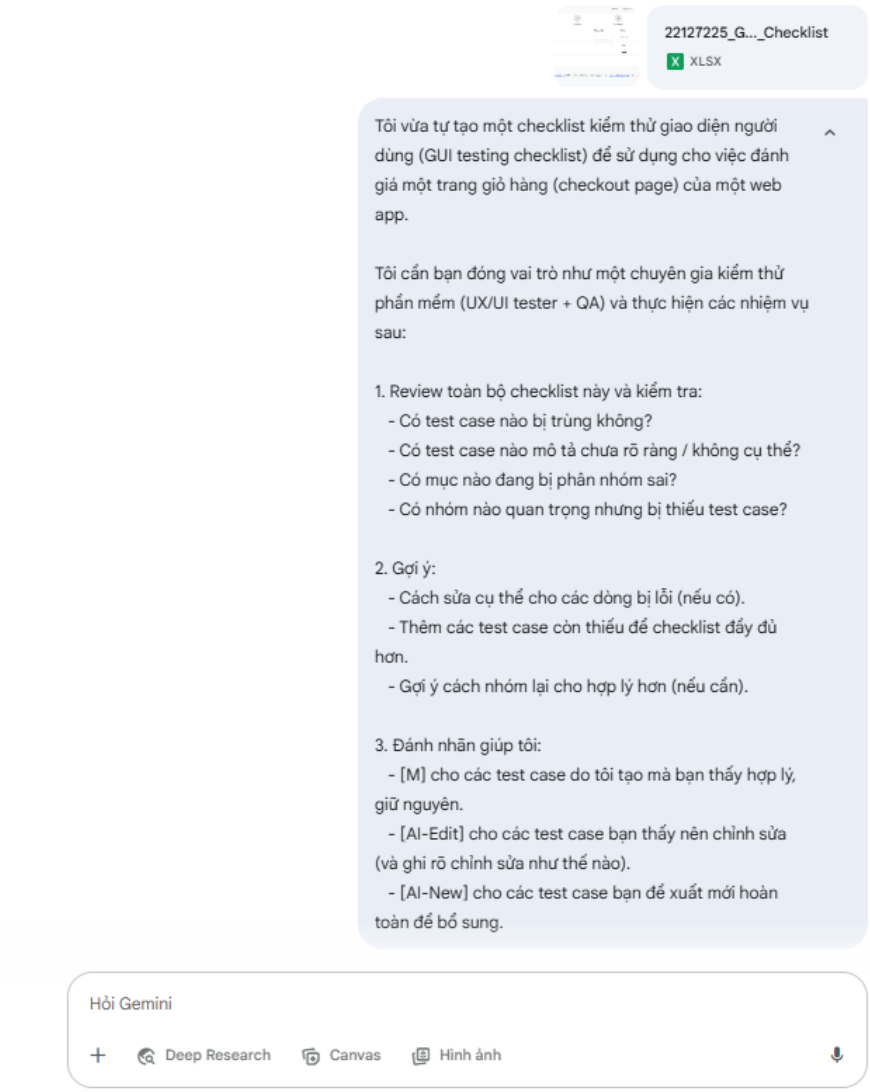
5 Prompt được sử dụng

Để đảm bảo chất lượng của checklist kiểm thử giao diện người dùng (GUI testing checklist) do nhóm tự tạo, một prompt đã được thiết kế nhằm yêu cầu công cụ AI **Gemini 2.5 Flash** thực hiện việc rà soát và đề xuất cải tiến toàn diện. Prompt cụ thể như sau:

> Tôi vừa tự tạo một checklist kiểm thử giao diện người dùng (GUI testing checklist) để sử dụng cho việc đánh giá một trang giỏ hàng (checkout page) của một web app.
>
> Tôi cần bạn đóng vai trò như một chuyên gia kiểm thử phần mềm (UX/UI tester + QA) và thực hiện các nhiệm vụ sau:
>
> 1. **Review toàn bộ checklist này và kiểm tra:**

- Có test case nào bị trùng không?
- Có test case nào mô tả chưa rõ ràng / không cụ thể?
- Có mục nào đang bị phân nhóm sai?
- Có nhóm nào quan trọng nhưng bị thiếu test case?

2. **Gợi ý:**
   - Cách sửa cụ thể cho các dòng bị lỗi (nếu có).
   - Thêm các test case còn thiếu để checklist đầy đủ hơn.
   - Gợi ý cách nhóm lại cho hợp lý hơn (nếu cần).
3. **Đánh nhãn giúp tôi:**
   - `[M]` cho các test case do tôi tạo mà bạn thấy hợp lý, giữ nguyên.
   - `[AI-Edit]` cho các test case bạn thấy nên chỉnh sửa (và ghi rõ chỉnh sửa như thế nào).
   - `[AI-New]` cho các test case bạn đề xuất mới hoàn toàn để bổ sung.



Hình 3: Prompt được sử dụng

Prompt trên được thiết kế nhằm đạt các mục tiêu:

- Khai thác khả năng tổng hợp, phân tích và đưa ra nhận xét có cơ sở của AI.
- Giữ nguyên tính chủ động của người dùng bằng cách phân loại rõ các loại test case.
- Tối ưu quy trình cộng tác giữa người dùng và công cụ AI trong thiết kế checklist kiểm thử.

6 Cách kiểm chứng và tinh chỉnh kết quả AI đưa ra

6.1 Kết quả sau khi prompt:

Sau khi sử dụng mô hình ngôn ngữ Gemini 2.5 Flash để đánh giá checklist kiểm thử giao diện người dùng do nhóm tự xây dựng, hệ thống AI đã phản hồi đầy đủ và chi tiết theo đúng yêu cầu được nêu trong prompt.

AI thực hiện các bước đánh giá chính như sau:

- Xác nhận các test case hợp lý và cần giữ nguyên, đánh nhãn `[M]`.
- Gợi ý chỉnh sửa cho các test case chưa rõ ràng, bị trùng lặp hoặc phân nhóm chưa phù hợp, đánh nhãn `[AI-Edit]` kèm lý do và cách sửa cụ thể.
- Đề xuất bổ sung các test case còn thiếu nhằm đảm bảo độ phủ toàn diện, đánh nhãn `[AI-New]`.

Phản hồi từ AI có cấu trúc rõ ràng, dễ theo dõi và bám sát yêu cầu đánh giá. Ngoài ra, hệ thống còn đưa ra một số góp ý về việc cải thiện phần chú thích (remarks), cách nhóm test case hợp lý hơn, và nhấn mạnh vào các vấn đề thường bị bỏ sót như accessibility hoặc tính phản hồi (responsiveness).

Nhìn chung, việc sử dụng AI để hỗ trợ đánh giá checklist giúp tiết kiệm thời gian, phát hiện các điểm mù trong kiểm thử thủ công, và cải thiện chất lượng checklist một cách đáng kể. Phản hồi từ AI sau khi sử dụng prompt được nhóm tiếp thu, rà soát và tích hợp vào checklist chính thức nhằm đảm bảo tính đầy đủ, rõ ràng và hiệu quả trong kiểm thử giao diện người dùng.

6.2 Kiểm chứng lại kết quả & Tinh chỉnh

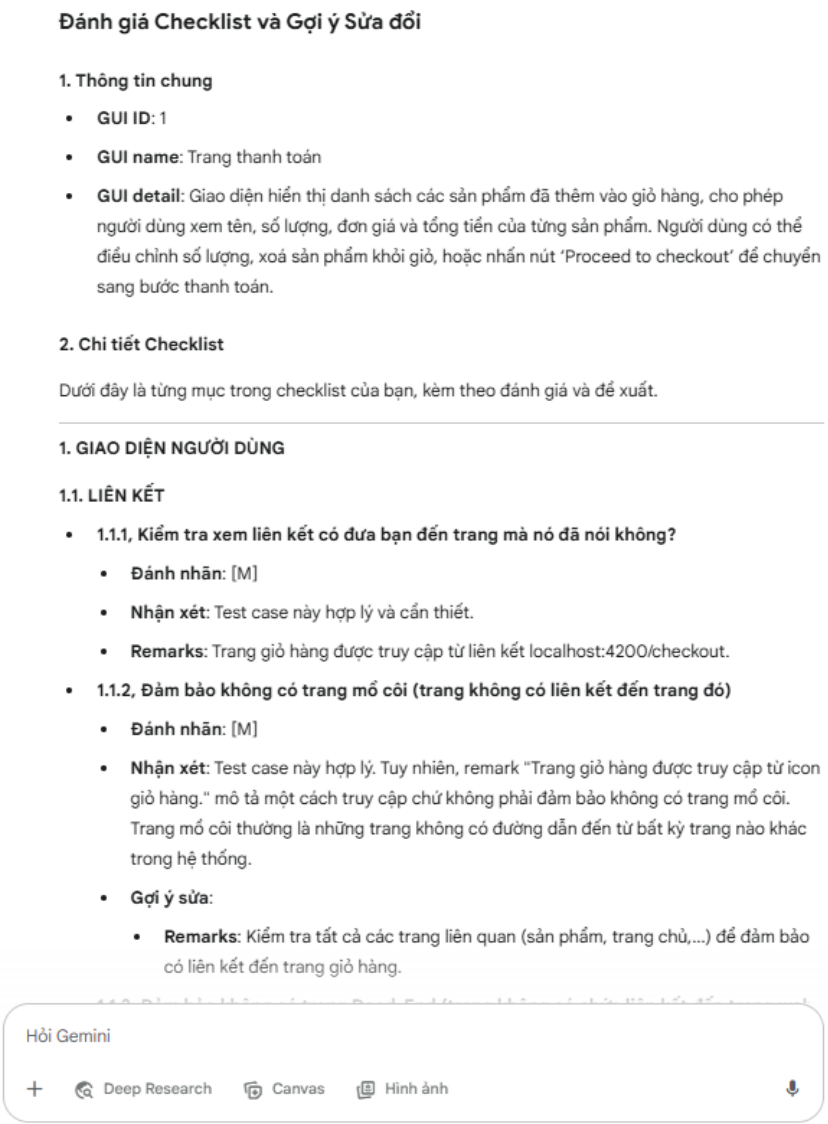
Sau khi nhận được các phản hồi và đề xuất kiểm thử từ công cụ AI (ChatGPT), tiến hành các bước kiểm chứng và tinh chỉnh để đảm bảo rằng các kết quả phù hợp với bối cảnh cụ thể của phần mềm và không bị trùng lặp hoặc lệch hướng. Các bước thực hiện bao gồm:

- **Đối chiếu thủ công:** Mỗi test case hoặc lỗi được AI đề xuất đều được nhóm so sánh thủ công với các mục có trong checklist gốc. Nếu một test case đã tồn tại hoặc gần trùng lặp về nội dung, bỏ qua hoặc gộp lại để tránh dư thừa. Đồng thời, cũng kiểm tra xem AI có đề xuất những mục không liên quan đến màn hình đang test hay không (ví dụ: lỗi liên quan đến tìm kiếm sản phẩm trong khi chỉ test màn hình thanh toán).
- **Phân tích ảnh chụp màn hình (screenshot):** Dựa trên ảnh giao diện thực tế, xác minh các lỗi được AI chỉ ra như:
  - Sai chính tả ("Contakt" thay vì "Contact").
  - Lỗi font chữ không đồng bộ hoặc khó đọc.
  - Các thành phần bị lệch căn lề, tràn viền, hoặc không canh giữa.
  - Màu sắc hover chưa rõ ràng, thiếu độ tương phản.

  Các lỗi này được đối chiếu với giao diện thật để đảm bảo phản hồi AI là chính xác, không bị "hallucinate" (AI tự suy diễn hoặc tạo ra thông tin không đúng với ngữ cảnh thực tế).
- **Kiểm tra theo nguyên tắc thiết kế giao diện:** Các phản hồi từ AI được nhóm đánh giá lại theo các nguyên tắc thiết kế GUI chuẩn như:
  - **Usability:** Giao diện dễ hiểu, dễ thao tác, phản hồi nhanh chóng.
  - **Accessibility:** Có thể thao tác bằng bàn phím (tab tới nút, focus), dễ đọc, độ tương phản cao.
  - **Consistency:** Thiết kế thống nhất về màu sắc, font chữ, bố cục giữa các thành phần.

Nếu đề xuất nào từ AI không đáp ứng các tiêu chí này, nhóm sẽ loại bỏ hoặc điều chỉnh cho hợp lý hơn với sản phẩm thật.

- **Mở rộng checklist:** Những test case mới và hợp lý được AI gợi ý (ví dụ như kiểm tra tính đúng đắn của đơn giá, kiểm tra phản hồi hover, kiểm tra định dạng tiền tệ) sẽ được xem xét kĩ lưỡng lại một lần nữa trước khi thêm vào checklist chính. Các test case này được đánh dấu là "AI-generated" để phân biệt với các mục tạo thủ công. Đồng thời, cũng ghi chú thêm ở cột "Remark" để giải thích rõ lý do bổ sung, tăng tính minh bạch cho quá trình kiểm thử.



Đánh giá Checklist và Gợi ý Sửa đổi

1. Thông tin chung

- **GUI ID**: 1
- **GUI name**: Trang thanh toán
- **GUI detail**: Giao diện hiển thị danh sách các sản phẩm đã thêm vào giỏ hàng, cho phép người dùng xem tên, số lượng, đơn giá và tổng tiền của từng sản phẩm. Người dùng có thể điều chỉnh số lượng, xoá sản phẩm khỏi giỏ, hoặc nhấn nút 'Proceed to checkout' để chuyển sang bước thanh toán.

2. Chi tiết Checklist

Dưới đây là từng mục trong checklist của bạn, kèm theo đánh giá và đề xuất.

---

1. GIAO DIỆN NGƯỜI DÙNG

1.1. LIÊN KẾT

- **1.1.1, Kiểm tra xem liên kết có đưa bạn đến trang mà nó đã nói không?**
  - **Đánh nhãn**: [M]
  - **Nhận xét**: Test case này hợp lý và cần thiết.
  - **Remarks**: Trang giỏ hàng được truy cập từ liên kết localhost:4200/checkout.
- **1.1.2, Đảm bảo không có trang mồ côi (trang không có liên kết đến trang đó)**
  - **Đánh nhãn**: [M]
  - **Nhận xét**: Test case này hợp lý. Tuy nhiên, remark "Trang giỏ hàng được truy cập từ icon giỏ hàng." mô tả một cách truy cập chứ không phải đảm bảo không có trang mồ côi. Trang mồ côi thường là những trang không có đường dẫn đến từ bất kỳ trang nào khác trong hệ thống.
  - **Gợi ý sửa**:
    - **Remarks**: Kiểm tra tất cả các trang liên quan (sản phẩm, trang chủ,...) để đảm bảo có liên kết đến trang giỏ hàng.

Hình 4: Prompt được sử dụng

7 Test case được tạo bởi AI và Kết quả Review

Trong quá trình phát triển phần mềm, việc kiểm thử giao diện người dùng (GUI Testing) đóng vai trò quan trọng nhằm đảm bảo rằng sản phẩm cuối cùng thân thiện, dễ sử dụng và không xảy ra lỗi

hiển thị hay chức năng. Tuy nhiên, việc kiểm thử giao diện thủ công thường tốn nhiều thời gian, dễ bỏ sót lỗi và phụ thuộc nhiều vào kinh nghiệm cá nhân. Nhờ sự phát triển của trí tuệ nhân tạo (AI), đặc biệt là các mô hình sinh ngôn ngữ như Gemini, quá trình kiểm thử GUI có thể được hỗ trợ hiệu quả hơn. Các công cụ AI có thể nhanh chóng phân tích ảnh chụp giao diện, checklist và gợi ý các lỗi giao diện, các điểm không nhất quán hoặc những chỗ chưa tuân thủ tiêu chuẩn thiết kế.

Trong phần này, công cụ Gemini được dùng để review checklist thủ công do nhóm tự tạo, đưa ra nhận xét, cảnh báo các mục không rõ ràng, gợi ý bổ sung test case còn thiếu và xác nhận logic hợp lý của checklist. Quy trình kiểm thử kết hợp AI và kiểm thử thủ công giúp đảm bảo chất lượng giao diện, phát hiện lỗi chính xác hơn, đồng thời tiết kiệm thời gian và công sức so với kiểm thử hoàn toàn thủ công.

Dựa trên checklist ban đầu, AI đã hỗ trợ theo ba hướng chính:

- Xác nhận các test case do nhóm tạo là hợp lý và đầy đủ – gán nhãn `[M]` .
- Gợi ý chỉnh sửa mô tả cho rõ ràng, cụ thể hơn – gán nhãn `[AI-Edit]`.
- Đề xuất mới các test case còn thiếu để checklist trở nên đầy đủ hơn – gán nhãn `[AI-New]`.

7.1 Các Test Case được giữ nguyên `[M]`

AI đã rà soát toàn bộ checklist và xác nhận nhiều test case do nhóm tạo là hợp lý, phản ánh đúng các tiêu chuẩn cơ bản của giao diện người dùng, như: điều hướng đúng trang, bố cục rõ ràng, độ tương phản màu sắc, kích thước font hợp lý, và tiêu đề trang chính xác. Những test case này không cần chỉnh sửa thêm và được giữ nguyên như ban đầu.

7.2 Các Test Case cần chỉnh sửa `[AI-Edit]`

Một số test case trong checklist ban đầu được AI đề xuất chỉnh sửa nhằm nâng cao tính rõ ràng và tránh trùng lặp. Các đề xuất tập trung vào những điểm sau:

- **Gộp test case kiểm thử đầu vào không hợp lệ:** Các test case liên quan đến việc nhập số âm, ký tự chữ, ký tự đặc biệt hoặc để trống trong ô số lượng có thể được gộp lại thành một test case tổng quát về *"Xác thực dữ liệu đầu vào"*.
  Ví dụ: Thay vì 3 test case riêng biệt như "Không cho nhập số âm", "Không cho nhập ký tự đặc biệt", có thể viết lại thành một dòng: "Kiểm tra xác thực ô nhập số lượng không cho phép giá trị không hợp lệ".

- **Chỉnh sửa mô tả chưa rõ ràng:** Một số test case mô tả không đủ bối cảnh hoặc không nêu rõ hành vi mong đợi. AI đề xuất viết lại cụ thể hơn.
  Ví dụ: Dòng "Hệ thống phản hồi khi cập nhật số lượng" nên bổ sung thành "Khi người dùng thay đổi số lượng sản phẩm, hệ thống hiển thị phản hồi như loading spinner hoặc cập nhật giá ngay lập tức".

- **Loại bỏ hoặc hợp nhất test case trùng lặp:** Một số test case kiểm thử cùng một nội dung với test case khác (VD: vị trí icon giỏ hàng được highlight) có thể gây lặp thông tin. AI đề xuất loại bỏ hoặc gộp chúng lại.
  Ví dụ: Hai dòng kiểm thử "Kiểm tra icon giỏ hàng có hiển thị" và "Icon giỏ hàng được làm nổi bật khi đang ở trang giỏ" có thể gộp thành một: "Kiểm tra icon giỏ hàng hiển thị rõ và được làm nổi bật khi đang ở trang giỏ".

- **Bổ sung ví dụ minh họa để làm rõ:** Một vài test case nói chung chung như "Kiểm tra phản hồi hệ thống" được AI đề xuất bổ sung ví dụ cụ thể.
  Ví dụ: Nên viết rõ "Phản hồi hệ thống" là gì – như loading spinner, thông báo thành công/thất bại, hoặc cập nhật tổng tiền.

Các chỉnh sửa này giúp checklist trở nên mạch lạc, nhất quán, và dễ sử dụng hơn trong quá trình kiểm thử thực tế.

7.3 Các Test Case mới được đề xuất `[AI-New]`

Dựa trên phân tích từ công cụ AI Gemini, các test case mới được đề xuất nhằm bổ sung cho các khía cạnh chưa được kiểm tra đầy đủ trong checklist ban đầu. Các đề xuất này được phân thành các nhóm như sau:

- **Responsiveness và Layout:**
  - Bố cục trang có phù hợp và hiển thị đúng trên các kích thước màn hình khác nhau (desktop, tablet, mobile) không?
  - Các phần tử trên trang (ví dụ: bảng sản phẩm, nút bấm, tổng tiền) có được căn chỉnh và bố trí hợp lý trên mọi thiết bị không?
  - Khoảng cách giữa các phần tử có đủ để tránh chồng chéo hoặc quá sát nhau không?
  - Có sự nhất quán về khoảng cách (padding/margin) giữa các phần tử tương tự không?
- **Cửa sổ / Pop-up:**
  - Nếu có pop-up xác nhận (ví dụ: khi xóa sản phẩm), pop-up đó có hiển thị rõ ràng thông tin, nút xác nhận/hủy và có thể đóng bằng các cách thông thường (Escape, click bên ngoài) không?
- **Trải nghiệm người dùng (UX) – Điều hướng và phản hồi:**
  - Tên sản phẩm trong giỏ hàng có phải là liên kết đưa người dùng về trang chi tiết sản phẩm không?
  - Khi số lượng sản phẩm thay đổi, tổng tiền của sản phẩm đó và tổng tiền toàn bộ giỏ hàng có được cập nhật ngay lập tức và chính xác không?
  - Các hành động quan trọng (ví dụ: "Proceed to checkout", "Update Cart") có nổi bật và dễ nhận biết không?
  - Nút "Tiếp tục mua sắm"(hoặc tương tự) có được cung cấp để người dùng dễ dàng quay lại trang sản phẩm không?
  - Nếu giỏ hàng trống, trang có hiển thị thông báo rõ ràng về việc giỏ hàng trống và gợi ý người dùng mua sắm không?
- **Tương tác:**
  - Trường nhập liệu số lượng có hỗ trợ các nút tăng/giảm số lượng (spinner buttons) không? Nếu có, chúng có hoạt động đúng không?

- Khi di chuột qua các phần tử tương tác (ví dụ: nút bấm, liên kết), có hiệu ứng hover rõ ràng không?
- Người dùng có thể dễ dàng xóa từng sản phẩm khỏi giỏ hàng không?
- Có chức năng "Chọn tất cả"hoặc "Xóa tất cả"sản phẩm khỏi giỏ hàng không (nếu phù hợp với logic nghiệp vụ)?
- Nếu có trường nhập mã giảm giá/coupon, trường đó có dễ tìm, dễ nhập và có nút áp dụng rõ ràng không?

- **Xử lý lỗi và thông báo:**
  - Khi nhập số lượng không hợp lệ (ví dụ: số 0, số âm, ký tự chữ, số lượng vượt quá tồn kho), hệ thống có hiển thị thông báo lỗi cụ thể và dễ hiểu không?
  - Các thông báo (lỗi, thành công, cảnh báo) có hiển thị ở vị trí dễ thấy, rõ ràng và biến mất (hoặc có thể đóng) một cách hợp lý không?
  - Khi xóa sản phẩm, có thông báo xác nhận việc xóa thành công (ví dụ: "Sản phẩm đã được xóa khỏi giỏ hàng") không?
  - Nếu giỏ hàng trống, trang có hiển thị thông báo rõ ràng về việc giỏ hàng trống không?
  - Kiểm tra khả năng thêm cùng một sản phẩm nhiều lần và hệ thống có gộp chúng lại hoặc tạo dòng riêng biệt (tùy theo thiết kế) không?
  - Kiểm tra khi số lượng được cập nhật, giá trị "Total"của sản phẩm đó và "Total"của toàn bộ giỏ hàng có được cập nhật ngay lập tức và chính xác không?

- **Khả năng tiếp cận (Accessibility):**
  - Các phần tử tương tác (nút, liên kết, trường nhập liệu) có đủ kích thước để dễ dàng thao tác bằng ngón tay trên thiết bị di động không?
  - Các hình ảnh có thuộc tính 'alt text' phù hợp để hỗ trợ người dùng khiếm thị sử dụng trình đọc màn hình không?
  - Thứ tự focus khi sử dụng phím Tab có theo trình tự logic và dễ đoán không?

7.4 Kết luận

Việc tích hợp AI vào quy trình kiểm thử giao diện người dùng đã mang lại nhiều giá trị thực tiễn cho nhóm. Cụ thể:

- **Xác nhận và củng cố:** AI đã giúp xác nhận các test case thủ công mà nhóm xây dựng là hợp lý, phản ánh đúng các yêu cầu cơ bản về giao diện và trải nghiệm người dùng.
- **Phát hiện lỗi mô tả và phân nhóm:** Một số test case bị mô tả chưa rõ ràng, bị trùng lặp hoặc phân nhóm chưa hợp lý đã được AI chỉ ra, từ đó nhóm có cơ sở để tinh chỉnh lại checklist theo hướng rõ ràng, có hệ thống hơn.
- **Bổ sung test case còn thiếu:** Các test case mới được đề xuất bởi AI bao phủ những khía cạnh quan trọng thường dễ bị bỏ sót như *khả năng truy cập (accessibility)*, *xử lý lỗi (error handling)*, *tương tác người dùng*, và *xác minh logic xử lý backend*.

- **Đánh giá tính phù hợp của đề xuất:** Mặc dù AI đã hỗ trợ hiệu quả trong việc đề xuất và chỉnh sửa test case, nhóm vẫn cần rà soát lại tất cả các đề xuất để đảm bảo phù hợp với ngữ cảnh cụ thể của trang web đang được kiểm thử. Một số test case do AI sinh ra có thể mang tính tổng quát hoặc được đề xuất dựa trên giả định mặc định, dẫn đến không phù hợp với thực tế triển khai.

  - **Với các test case `[AI-New]`:**
    * Một số đề xuất kiểm tra hình ảnh sản phẩm (VD: thuộc tính `alt text`) không áp dụng được vì trang giỏ hàng chỉ hiển thị tên sản phẩm, không có ảnh minh họa.
    * Các test case liên quan đến mã giảm giá (discount code) không phù hợp nếu chức năng này chưa được triển khai trong giao diện hiện tại.
  - **Với các test case `[AI-Edit]`:**
    * Một vài gợi ý chỉnh sửa yêu cầu bổ sung ví dụ cụ thể hoặc làm rõ logic, tuy nhiên nếu test case ban đầu đã đủ rõ trong ngữ cảnh nội bộ nhóm thì việc chỉnh sửa là không bắt buộc.
    * Một số đề xuất gộp test case có thể làm mất đi tính chi tiết phục vụ truy vết lỗi, do đó cần cân nhắc giữ nguyên để thuận tiện cho việc phân tích sau kiểm thử.

  Do đó, mỗi test case do AI đề xuất hoặc chỉnh sửa đều cần được đánh giá lại dựa trên: (i) chức năng thực tế của trang web, (ii) khả năng hiện có của hệ thống, và (iii) mục tiêu kiểm thử cụ thể.

  Vì vậy, nhóm vẫn cần đối chiếu từng đề xuất của AI với giao diện thật để đánh giá tính khả thi và mức độ liên quan trước khi đưa vào checklist chính thức.

8 Test case được tạo thủ công

Các test case này được xây dựng dựa trên nguyên tắc thiết kế giao diện người dùng và đánh giá trực quan:

- **Tính nhất quán bố cục:** Các cột (Sản phẩm, Số lượng, Giá, Tổng) căn chỉnh đều nhau.
- **Khoảng cách và padding:** Các hàng sản phẩm có khoảng cách hợp lý, dễ nhìn.
- **Hiển thị thành phần:** Tất cả nút và text đều hiển thị đầy đủ, không bị khuất.
- **Kiểu chữ và nhãn:** Font chữ rõ ràng, thống nhất, dễ phân biệt tiêu đề và nội dung.
- **Khả năng đáp ứng (Responsive):** Giao diện không vỡ khi thu nhỏ màn hình.

9 Kết luận

Việc sử dụng AI trong kiểm thử giao diện giúp:

- Phát hiện lỗi nhanh và bao quát hơn (ví dụ: lỗi căn lề, chính tả).
- Gợi ý mở rộng checklist hợp lý theo tiêu chuẩn giao diện.

- Nâng cao hiệu quả kiểm thử GUI, hỗ trợ người kiểm thử tiết kiệm thời gian và tránh sót lỗi.

Quy trình này kết hợp giữa AI và con người, giúp tăng độ chính xác và mở rộng phạm vi kiểm thử GUI theo cách linh hoạt và có hệ thống.