NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Vĩnh Long, ngày ..... tháng …… năm ……

Giảng viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Vĩnh Long, ngày ..... tháng …… năm ……

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn chân thành và sâu sắc nhất tới Ban giám hiệu Trưởng Khoa Kỹ thuật và Công nghệ – Trường Đại học Trà Vinh, cùng các thầy cô giáo trong Khoa Công nghệ thông tin đã tận tình truyền đạt kiến thức, tạo điều kiện thuận lợi nhất cho em trong suốt quá trình học tập và rèn luyện tại trường. Những kiến thức quý báu về lập trình và tư duy hệ thống mà các thầy cô truyền dạy là nền tảng vững chắc để em có thể hoàn thành đồ án này.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc tới Cô Phạm Thị Trúc Mai – giảng viên trực tiếp hướng dẫn em thực hiện đồ án. Trong suốt thời gian triển khai đề tài *"Lập trình Front-end cho web bán điện thoại bằng ReactJS"*, cô đã dành nhiều thời gian và tâm huyết để chỉ bảo, định hướng và đưa ra những lời khuyên chuyên môn quý giá. Những góp ý của cô không chỉ giúp em tháo gỡ những khó khăn về mặt kỹ thuật khi làm việc với ReactJS và Context API mà còn giúp em học được tác phong làm việc khoa học, nghiêm túc.

Em cũng xin gửi lời cảm ơn tới gia đình và bạn bè đã luôn ở bên cạnh động viên, cổ vũ và hỗ trợ em về mọi mặt để em có thể tập trung hoàn thành tốt đồ án cơ sở ngành này.

Mặc dù đã có nhiều cố gắng để hoàn thiện trang web một cách chỉn chu nhất, nhưng do kiến thức còn hạn chế và bước đầu làm quen với các công nghệ hiện đại, đồ án chắc chắn không tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp và phê bình từ quý thầy cô để em có thể rút kinh nghiệm và phát triển hoàn thiện hơn trong những đề tài nghiên cứu tiếp theo.

Một lần nữa, em xin kính chúc các thầy cô luôn dồi dào sức khỏe, hạnh phúc và gặt hái được nhiều thành công trong sự nghiệp trồng người cao quý!

Em xin chân thành cảm ơn!

MỤC LỤC

DANH MỤC HÌNH ẢNH

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Đồ án thực tập cơ sở ngành với đề tài "***Lập trình Front-end cho web bán điện thoại bằng ReactJS***" là một nỗ lực nghiên cứu và áp dụng các công nghệ lập trình web hiện đại vào thực tiễn. Mục tiêu chính của đồ án là xây dựng một giao diện người dùng (Front-end) hoàn chỉnh cho một hệ thống thương mại điện tử chuyên kinh doanh các thiết bị di động, đáp ứng nhu cầu ngày càng cao của khách hàng về trải nghiệm mua sắm trực tuyến mượt mà, tốc độ nhanh và trực quan.

Về mặt kỹ thuật, ứng dụng được phát triển dựa trên nền tảng thư viện ReactJS, tận dụng tối đa kiến trúc Component-based để chia nhỏ giao diện thành các thành phần độc lập, giúp mã nguồn dễ dàng quản lý, bảo trì và tái sử dụng. Điểm nhấn quan trọng trong giải pháp kỹ thuật là việc ứng dụng Context API để giải quyết bài toán quản lý trạng thái toàn cục . Thay vì truyền dữ liệu qua nhiều cấp component phức tạp, Context API giúp quản lý dữ liệu giỏ hàng (như thêm sản phẩm, cập nhật số lượng, xóa sản phẩm) một cách hiệu quả và tập trung, đảm bảo tính nhất quán của dữ liệu trên toàn bộ ứng dụng.

Kết quả đạt được của đồ án là một website bán hàng hoàn thiện với giao diện hiện đại, tuân thủ các nguyên tắc thiết kế UI/UX tối giản nhưng tinh tế . Hệ thống tích hợp đầy đủ các chức năng cốt lõi cần thiết cho một quy trình mua sắm, bao gồm xem danh sách sản phẩm, bộ lọc thông minh theo thương hiệu và mức giá, công cụ tìm kiếm thời gian thực, trang chi tiết sản phẩm và quy trình quản lý giỏ hàng hoàn chỉnh. Bên cạnh đó, ứng dụng sử dụng React Router DOM để tạo trải nghiệm Single Page Application không cần tải lại trang, đồng thời đảm bảo khả năng tương thích đa nền tảng , hiển thị tốt trên cả máy tính để bàn, máy tính bảng và điện thoại di động.

MỞ ĐẦU

Ngày nay, công nghệ thông tin đang phát triển vô cùng mạnh mẽ, kéo theo sự thay đổi lớn trong thói quen mua sắm của mọi người. Thay vì phải đến tận cửa hàng, khách hàng ngày càng ưa chuộng việc ngồi tại nhà và đặt hàng qua mạng internet. Điều này khiến cho thương mại điện tử trở thành một lĩnh vực không thể thiếu, và việc sở hữu một trang web bán hàng chuyên nghiệp là nhu cầu cấp thiết đối với bất kỳ cửa hàng nào, đặc biệt là trong lĩnh vực kinh doanh thiết bị công nghệ như điện thoại di động.

Để đáp ứng nhu cầu trải nghiệm ngày càng cao của người dùng, trang web không chỉ cần đẹp mà còn phải hoạt động nhanh, mượt mà và dễ sử dụng. ReactJS, một thư viện lập trình giao diện hàng đầu hiện nay do Facebook phát triển, chính là giải pháp tối ưu cho vấn đề này. Nó giúp tạo ra các trang web hoạt động trơn tru như một ứng dụng trên điện thoại, giảm thời gian chờ đợi khi tải trang. Chính vì những lý do thực tế và lợi ích to lớn đó, em đã quyết định lựa chọn đề tài "*Lập trình Front-end cho web bán điện thoại bằng ReactJS*" để nghiên cứu và thực hiện.

Sự bùng nổ của các thiết bị di động thông minh đã kéo theo nhu cầu mua sắm trực tuyến tăng cao. Tuy nhiên, các website bán hàng truyền thống thường gặp vấn đề về tốc độ tải trang chậm và giao diện thiếu linh hoạt. Việc ứng dụng ReactJS vào xây dựng Single Page Application (SPA) cho phép người dùng trải nghiệm mua sắm mượt mà, không bị ngắt quãng bởi việc tải lại trang, từ đó tối ưu hóa hành trình khách hàng và nâng cao tỷ lệ chuyển đổi đơn hàng cho doanh nghiệp.

Mục tiêu quan trọng nhất của đồ án này là giúp em củng cố và áp dụng những kiến thức lý thuyết đã học vào thực tế. Em mong muốn nắm vững cách sử dụng ReactJS, hiểu rõ cách chia nhỏ giao diện thành các thành phần (component) để dễ quản lý, cũng như cách sử dụng các công cụ hiện đại như Hooks và Context API để xử lý dữ liệu.

Bên cạnh đó, đồ án cũng đặt mục tiêu tạo ra một sản phẩm cụ thể là một giao diện website bán điện thoại hoàn chỉnh. Trang web này phải có đầy đủ các chức năng cơ bản cần thiết như hiển thị danh sách sản phẩm đẹp mắt, cho phép người dùng tìm kiếm, lọc sản phẩm theo ý muốn, xem chi tiết từng chiếc điện thoại và quan trọng nhất là thực hiện được thao tác thêm vào giỏ hàng. Giao diện cũng cần

phải hiển thị tốt trên mọi thiết bị, từ máy tính màn hình rộng cho đến điện thoại di động nhỏ gọn.

Đối tượng nghiên cứu chính của đề tài là phần giao diện người dùng (Front-end) của một website thương mại điện tử. Em sẽ tập trung nghiên cứu cách thiết kế bố cục trang web sao cho thân thiện, cách xử lý các sự kiện khi người dùng tương tác và cách quản lý dữ liệu hiển thị trên màn hình.

Về phạm vi thực hiện, do giới hạn về thời gian và tính chất của một đồ án cơ sở, dự án sẽ tập trung hoàn toàn vào việc xây dựng giao diện và xử lý logic ở phía người dùng. Dữ liệu về sản phẩm sẽ được giả lập thay vì lấy từ một cơ sở dữ liệu thực tế phức tạp. Các phần xử lý chuyên sâu ở phía máy chủ hay quy trình thanh toán tiền thật sẽ không nằm trong phạm vi của đồ án này.

CHƯƠNG 1: TỔNG QUAN

1.1 Giới thiệu tổng quan

Sự phát triển mạnh mẽ của công nghệ thông tin đã thúc đẩy sự thay đổi lớn trong thói quen tiêu dùng, từ mua sắm truyền thống sang thương mại điện tử trực tuyến. Đối với các doanh nghiệp kinh doanh thiết bị công nghệ, đặc biệt là điện thoại di động, việc sở hữu một hệ thống website chuyên nghiệp, tốc độ cao và trải nghiệm người dùng tối ưu là yếu tố then chốt để cạnh tranh.

Vấn đề tập trung giải quyết ở đây chính là việc ứng dụng thư viện ReactJS để tối ưu hóa hiệu năng hiển thị. Thay vì tải lại toàn bộ trang web mỗi khi khách hàng thực hiện các thao tác như lọc sản phẩm theo hãng hay chuyển trang, cơ chế Virtual DOM của ReactJS sẽ chỉ cập nhật những thành phần thực sự thay đổi. Điều này giúp loại bỏ độ trễ, mang lại trải nghiệm mượt mượt mà, đặc biệt quan trọng đối với một trang web bán điện thoại có nhiều hình ảnh và thông số kỹ thuật cao.

1.2 Giải pháp công nghệ lựa chọn

Để hiện thực hóa các chức năng của hệ thống một cách tối ưu, đồ án đã lựa chọn và kết hợp các công nghệ lập trình web hiện đại nhất hiện nay. Nền tảng cốt lõi của ứng dụng là ngôn ngữ JavaScript kết hợp cùng thư viện ReactJS. Việc sử dụng ReactJS cho phép xây dựng giao diện theo mô hình Component, giúp mã nguồn trở nên linh hoạt, dễ dàng bảo trì và tăng tốc độ hiển thị thông qua cơ chế Virtual DOM. Để thay thế cho các bộ công cụ cũ có tốc độ biên dịch chậm, dự án áp dụng Vite làm môi trường xây dựng (Build tool). Vite mang lại khả năng khởi động và phản hồi thay đổi mã nguồn gần như tức thì, giúp tối ưu hóa đáng kể quy trình lập trình Front-end.

Về mặt quản lý logic và dữ liệu, ứng dụng giải quyết bài toán đồng bộ thông tin giỏ hàng thông qua Context API kết hợp với hook useReducer. Giải pháp này tạo ra một kho chứa trạng thái toàn cục, cho phép các dữ liệu như số lượng sản phẩm hay tổng giá trị đơn hàng được chia sẻ xuyên suốt giữa các trang (Trang chủ, Danh mục, Giỏ hàng) mà không gặp phải hiện tượng chồng chéo dữ liệu. Để điều hướng người dùng giữa các trang chức năng một cách mượt mà trong môi trường Single Page Application, thư viện React Router DOM đã được tích hợp nhằm quản lý các tuyến đường dẫn một cách chuyên nghiệp.

Ngoài ra, nhằm đảm bảo tính toàn vẹn và an toàn dữ liệu ngay từ phía người dùng, đồ án ứng dụng thư viện để xây dựng hệ thống xác thực cho các biểu mẫu

đặt hàng. Mọi thông tin như số điện thoại hay địa chỉ khách hàng đều được kiểm soát chặt chẽ theo các quy tắc định sẵn trước khi được xử lý. Cuối cùng, phần giao diện được hoàn thiện với sự hỗ trợ của Tailwind CSS và React Icons, giúp xây dựng các thành phần UI có tính thẩm mỹ cao, phong cách tối giản và đặc biệt là khả năng đáp ứng tốt trên nhiều kích thước màn hình từ điện thoại di động đến máy tính để bàn.

1.3 Vấn đề nghiên cứu về quản lý dữ liệu giỏ hàng

Một trong những thách thức lớn nhất mà đề tài tập trung giải quyết là quản lý trạng thái dữ liệu (State Management) xuyên suốt chu trình mua sắm của khách hàng. Khi người dùng thực hiện các hành động như thêm sản phẩm vào giỏ, xóa hoặc thay đổi số lượng, thông tin này cần được đồng bộ tức thì trên toàn bộ hệ thống giao diện. Để giải quyết vấn đề này, đồ án ứng dụng giải pháp Context API kết hợp với hook useReducer để thiết lập một kho dữ liệu tập trung. Cơ chế này giúp dữ liệu giỏ hàng được chia sẻ thông suốt giữa các trang (Trang chủ, Trang danh sách, Trang giỏ hàng) mà không gặp phải hiện tượng sai lệch thông tin hay lỗi truyền tin (Prop Drilling), đảm bảo tính chính xác tuyệt đối cho đơn hàng của khách hàng.

1.4 Vấn đề về giao diện tương thích đa nền tảng

Vấn đề tiếp theo mà đề tài nghiên cứu và hiện thực hóa là khả năng hiển thị linh hoạt trên nhiều kích thước màn hình khác nhau. Với việc phần lớn người dùng hiện nay sử dụng thiết bị di động để mua sắm, đề tài tập trung vào việc giải quyết bài toán bố cục (Layout) sao cho các bảng thông số kỹ thuật phức tạp của điện thoại vẫn rõ ràng và dễ thao tác trên màn hình nhỏ. Thông qua việc nghiên cứu các điểm ngắt (breakpoints) của Tailwind CSS, hệ thống được xây dựng để tự động co giãn và sắp xếp lại các khối nội dung một cách khoa học. Vấn đề này được giải quyết nhằm mục tiêu mang lại trải nghiệm mua sắm đồng nhất, chuyên nghiệp trên cả điện thoại di động, máy tính bảng và máy tính để bàn.

1.5 Vấn đề tổ chức thành phần và khả năng mở rộng

Cuối cùng, đề tài tập trung giải quyết bài toán tổ chức mã nguồn theo hướng Component-based. Website được chia nhỏ thành các thành phần độc lập như Navbar, Product Card, Footer... giúp việc quản lý mã nguồn trở nên minh bạch và dễ dàng bảo trì. Vấn đề nghiên cứu này giúp hệ thống có khả năng mở rộng cao, sẵn

sàng cho việc nâng cấp thêm các tính năng phức tạp như tích hợp thanh toán trực tuyến hoặc kết nối với hệ thống quản lý Backend thực tế trong tương lai mà không phải thay đổi cấu trúc cốt lõi của ứng dụng.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Tìm hiểu về ReactJS

2.1.1 ReactJS là gì

ReactJS là một thư viện dùng ngôn ngữ JavaScript để xây dựng giao diện người dùng (UI). Thay vì phải viết mã để thay đổi từng chi tiết nhỏ trên trang web một cách thủ công, React giúp chúng ta quản lý các thành phần giao diện một cách tự động và thông minh hơn.

Điểm đột phá của ReactJS nằm ở cơ chế Virtual DOM, hoạt động như một bản vẽ nháp của giao diện thực tế. Khi có sự thay đổi về trạng thái sản phẩm hay bộ lọc tìm kiếm, React sẽ thực hiện thuật toán so sánh (Diffing Algorithm) để chỉ cập nhật những phần tử thực sự thay đổi lên màn hình. Cơ chế này không chỉ giúp tiết kiệm tài nguyên CPU của thiết bị mà còn đảm bảo website luôn duy trì tốc độ phản hồi ở mức cao nhất, ngay cả khi xử lý danh mục sản phẩm lớn.

2.1.2 Khái niệm Component

Đây là tư duy quan trọng nhất của React. Chúng ta không nhìn trang web như một tệp HTML dài dằng dặc, mà nhìn nó như một bộ lắp ghép Lego.

Định nghĩa: Mỗi nút bấm, mỗi ô nhập liệu, hay mỗi khung hình sản phẩm điện thoại đều được coi là một Component.

Lợi ích: Khi muốn sửa giao diện nút "Mua ngay", lập trình viên chỉ cần sửa ở một chỗ duy nhất và tất cả các nút đó trên toàn bộ trang web sẽ tự động cập nhật theo. Điều này giúp tiết kiệm thời gian và tránh sai sót.

2.1.3 Phân tích mô hình kiến trúc Component-Based trong ReactJS

Trong quá trình xây dựng website bán điện thoại, việc áp dụng kiến trúc dựa trên thành phần (Component-Based Architecture) của ReactJS đóng vai trò then chốt trong việc tối ưu hóa quy trình phát triển Front-end. Mỗi thành phần trên giao diện như thanh điều hướng (Navbar), thẻ hiển thị sản phẩm (ProductCard), hay danh sách giỏ hàng (CartList) được đóng gói dưới dạng một đơn vị mã nguồn độc lập. Cách tiếp cận này giúp tác giả dễ dàng tái sử dụng mã nguồn ở nhiều vị trí khác nhau trong ứng dụng mà không cần phải viết lại logic hay định dạng CSS. Đặc biệt, kiến trúc này cho phép tách biệt rõ ràng giữa logic xử lý dữ liệu và giao diện hiển

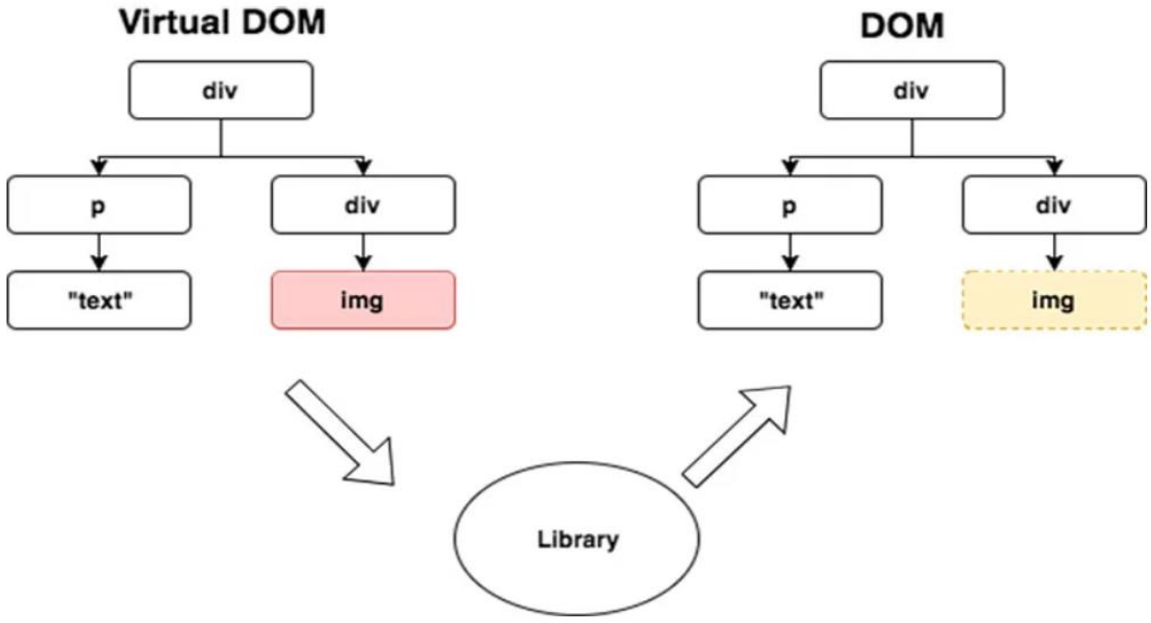
thị, giúp việc bảo trì hệ thống trở nên linh hoạt hơn. Khi có nhu cầu thay đổi giao diện của một mẫu điện thoại mới, lập trình viên chỉ cần can thiệp vào một Component duy nhất, các thay đổi sẽ được cập nhật đồng bộ trên toàn bộ ứng dụng, đảm bảo tính nhất quán và chuyên nghiệp cho hệ thống.

2.1.4 Virtual DOM

Trong các trang web thông thường, mỗi khi có sự thay đổi (ví dụ: bấm nút lọc giá), trình duyệt phải vẽ lại rất nhiều thứ, làm máy bị chậm.

Cách React hoạt động: React tạo ra một bản sao "nháp" trong bộ nhớ. Nó tính toán các thay đổi trên bản nháp này trước, sau đó chỉ cập nhật đúng những phần thực sự cần thay đổi lên màn hình thật. Nhờ vậy, website bán điện thoại của lập trình viên sẽ chạy rất mượt mà.

Để giải quyết vấn đề này, ReactJS sử dụng cơ chế Virtual DOM (DOM ảo). Khi có sự thay đổi, React không tác động trực tiếp lên màn hình mà tạo ra một bản sao nháp trong bộ nhớ. Sau đó, hệ thống thực hiện thuật toán so sánh để tìm ra sự khác biệt giữa bản nháp mới và cũ. Cuối cùng, React chỉ cập nhật đúng những phần tử thực sự thay đổi lên màn hình trình duyệt .



Hình 1: Sơ đồ so sánh quy trình cập nhật của Virtual DOM và Real DOM

Như Hình minh họa sự khác biệt về hiệu năng giữa cách cập nhật giao diện truyền thống và ReactJS. Trong khi Real DOM phải vẽ lại toàn bộ cấu trúc khi có thay đổi, Virtual DOM của React tạo ra một bản sao "nháp" trong bộ nhớ để tính toán các thay đổi trước. Sau đó, thuật toán so sánh (Diffing Algorithm) sẽ tìm ra

điểm khác biệt duy nhất và chỉ cập nhật đúng phần đó lên màn hình thật, giúp website bán điện thoại chạy cực kỳ mượt mà.

2.1.5 Cơ chế Virtual DOM và hiệu năng trải nghiệm khách hàng

Đối với một trang web bán hàng có lượng dữ liệu sản phẩm lớn và hình ảnh chất lượng cao, tốc độ phản hồi là yếu tố sống còn để giữ chân khách hàng. ReactJS giải quyết bài toán này thông qua cơ chế Virtual DOM một bản sao nhẹ của cấu trúc HTML thực tế được lưu trữ trong bộ nhớ. Thay vì tác động trực tiếp lên DOM thật của trình duyệt, React sẽ thực hiện quy trình so sánh giữa trạng thái cũ và mới mỗi khi có sự thay đổi. Chỉ những phần tử thực sự thay đổi mới được React cập nhật lên màn hình thực tế. Cơ chế này giúp website bán điện thoại vận hành cực kỳ mượt mà, loại bỏ hiện tượng 'nháy' trang khi chuyển đổi giữa các danh mục sản phẩm, từ đó nâng cao trải nghiệm mua sắm của người dùng lên mức tối đa.

2.2 Các khái niệm cơ bản trong lập trình React

2.2.1 JSX

Trong quá trình phát triển ứng dụng web bán điện thoại, lập trình viên đã sử dụng JSX như một công cụ chính để xây dựng giao diện. JSX không phải là một ngôn ngữ lập trình độc lập, mà là một phần mở rộng cú pháp cho JavaScript, cho phép lập trình viên viết các cấu trúc trông giống như HTML ngay bên trong mã nguồn Logic. Thay vì phải sử dụng các lệnh khởi tạo đối tượng phức tạp của React, JSX cung cấp một cách tiếp cận trực quan, giúp lập trình viên dễ dàng hình dung được cấu trúc phân cấp của các thành phần giao diện như danh sách sản phẩm hay chi tiết giỏ hàng.

2.2.2 Khái niệm về Component và tư duy xây dựng giao diện

Trong hệ sinh thái React, Component (Thành phần) là đơn vị xây dựng cơ bản nhất. Thay vì xây dựng giao diện theo cách truyền thống là viết một tệp HTML dài hàng nghìn dòng, React cho phép chia nhỏ giao diện thành các khối độc lập, có thể quản lý và tái sử dụng dễ dàng. Đối với website bán điện thoại này, mỗi nút bấm, thanh điều hướng (Navbar) hay thẻ hiển thị từng chiếc iPhone, Samsung đều được xem là một Component riêng biệt. Tư duy này giúp mã nguồn trở nên minh bạch và khoa học: khi cần thay đổi giao diện của nút 'Thêm vào giỏ hàng', tác giả chỉ cần chỉnh sửa tại một tệp duy nhất và mọi vị trí khác trên website sẽ tự động

được cập nhật đồng bộ. Điều này không chỉ giúp tiết kiệm thời gian mà còn hạn chế tối đa các lỗi xung đột mã nguồn trong quá trình phát triển lâu dài.

2.2.3 Cơ chế Virtual DOM và tối ưu hóa hiệu suất ứng dụng

Virtual DOM (DOM ảo) là một trong những đặc trưng mang tính cách mạng nhất của React giúp tối ưu hóa hiệu năng ứng dụng web. Thông thường, mỗi khi dữ liệu thay đổi, trình duyệt phải tính toán lại toàn bộ giao diện thực tế, điều này gây tốn kém tài nguyên và tạo ra hiện tượng giật lag. React giải quyết vấn đề này bằng cách tạo ra một bản sao nhẹ của giao diện thực trong bộ nhớ. Khi khách hàng thao tác (như lọc sản phẩm theo mức giá), React sẽ thực hiện quy trình so sánh bản sao này với giao diện hiện tại để tìm ra điểm khác biệt nhỏ nhất và chỉ cập nhật đúng phần đó lên màn hình. Nhờ vậy, website bán điện thoại đạt được tốc độ phản hồi cực nhanh, mang lại trải nghiệm mượt mà giống như một ứng dụng di động cài đặt sẵn cho người dùng.

2.2.4 Luồng dữ liệu qua Props và State

Để website có thể tương tác linh hoạt với người dùng, React sử dụng hai khái niệm cốt lõi để quản lý dữ liệu là Props và State. State (Trạng thái) được dùng để lưu trữ các thông tin có thể thay đổi ngay bên trong một thành phần, ví dụ như danh sách điện thoại đang có trong giỏ hàng hoặc từ khóa mà người dùng đang nhập vào ô tìm kiếm. Trong khi đó, Props (Thuộc tính) đóng vai trò là phương tiện để truyền dữ liệu từ thành phần cha xuống các thành phần con. Sự kết hợp chặt chẽ giữa Props và State tạo nên một luồng dữ liệu một chiều (One-way data flow) cực kỳ minh bạch. Điều này giúp lập trình viên dễ dàng kiểm soát được dữ liệu đang di chuyển như thế nào trong hệ thống, từ đó việc phát hiện và sửa lỗi (debugging) trở nên nhanh chóng và hiệu quả hơn rất nhiều.

2.3 Tìm hiểu về React Hooks

Hooks là các hàm đặc biệt được cung cấp sẵn để chúng ta móc vào các tính năng của React.

UseState: Dùng để tạo ra các biến có thể thay đổi trên giao diện. Ví dụ: dùng để tăng giảm số lượng điện thoại trong giỏ hàng.

UseEffect: Dùng để xử lý các công việc bên lề. Ví dụ: khi trang web vừa mở lên, nó sẽ tự động đi lấy danh sách điện thoại từ dữ liệu có sẵn để hiển thị ra màn hình.

UseContext: Dùng để lấy dữ liệu từ một nơi lưu trữ chung mà không cần phải truyền qua quá nhiều tầng lớp Component trung gian.

2.3.1 Tổng quan về sự ra đời và ý nghĩa của React Hooks

React Hooks được giới thiệu chính thức từ phiên bản 16.8, đánh dấu một bước ngoặt lớn trong tư duy lập trình giao diện. Trước khi có sự xuất hiện của Hooks, các lập trình viên buộc phải sử dụng Class Components để quản lý trạng thái (State) và các vòng đời của thành phần (Lifecycle). Tuy nhiên, mô hình cũ này bộc lộ nhiều hạn chế như: logic bị chia cắt giữa các hàm componentDidMount hay componentDidUpdate, khó tái sử dụng mã nguồn và gây khó khăn cho việc tối ưu hóa.

2.3.2 Quản lý luồng dữ liệu bằng Context API và useReducer

Đối với một ứng dụng thương mại điện tử, việc quản lý luồng dữ liệu của giỏ hàng là một bài toán phức tạp. Để giải quyết vấn đề này mà không mắc phải lỗi truyền dữ liệu qua quá nhiều cấp (Prop Drilling), lập trình viên đã triển khai giải pháp Context API kết hợp với Hook useReducer.

2.4 Quản lý giỏ hàng bằng Context API

2.4.1 Tại sao cần Context API

Trong một trang web bán hàng, giỏ hàng là phần quan trọng nhất. Dữ liệu giỏ hàng cần phải xuất hiện ở nhiều nơi: trên thanh menu (để báo có bao nhiêu sản phẩm) và ở trang thanh toán. Nếu dùng cách thông thường sẽ rất khó để truyền dữ liệu đi khắp nơi như vậy. Vì thế, lập trình viên đã áp dụng Context API.

Context API đóng vai trò như một kho chứa đồ chung. Tất cả thông tin về việc khách thêm hay xóa điện thoại đều được lưu vào kho này. Nhờ vậy, bất kể khách hàng đang ở trang chủ hay trang chi tiết, thông tin giỏ hàng luôn được cập nhật chính xác và đồng bộ với nhau.

2.4.2 Khái niệm và vai trò của Context API

Context API là một tính năng có sẵn trong React giúp chia sẻ dữ liệu giữa các thành phần mà không cần phải truyền Props qua từng cấp một cách thủ công.

Trong đồ án này, Context API đóng vai trò là một 'kho lưu trữ dữ liệu tập trung' (Global State). Mọi thành phần trong website, dù nằm sâu trong cấu trúc cây thư mục, đều có thể kết nối trực tiếp đến kho dữ liệu này để lấy thông tin về giỏ hàng. Điều này giúp đồng bộ hóa dữ liệu tức thì: khi khách hàng nhấn 'Thêm vào giỏ', biểu tượng giỏ hàng trên thanh điều hướng sẽ cập nhật con số mới ngay lập tức mà không cần tải lại trang.

2.4.3 Cơ chế Persistence (Lưu trữ bền vững) kết hợp với Context

Một điểm quan trọng trong lý thuyết quản lý giỏ hàng là tính bền vững của dữ liệu. Context API mặc định sẽ bị xóa sạch khi người dùng làm mới trang . Do đó, lập trình viên đã tích hợp thêm LocalStorage vào luồng xử lý của Context. Mỗi khi Reducer cập nhật trạng thái giỏ hàng mới, một hàm trung gian sẽ tự động lưu mảng sản phẩm đó vào bộ nhớ trình duyệt. Khi ứng dụng khởi động lại, Context sẽ ưu tiên đọc dữ liệu từ LocalStorage trước để khôi phục lại giỏ hàng cho khách hàng, đảm bảo trải nghiệm mua sắm không bị gián đoạn.

2.5 Thiết kế giao diện bằng Tailwind CSS

Để trang web trông đẹp và chuyên nghiệp, lập trình viên sử dụng Tailwind CSS. Đây là một công cụ giúp thiết kế giao diện cực nhanh bằng cách gắn các mã hiệu trực tiếp vào mã nguồn. Tailwind giúp lập trình viên dễ dàng tạo ra các khung lưới (grid) để hiển thị danh sách điện thoại iPhone, Samsung một cách gọn gàng. Đặc biệt, nó giúp trang web tự động co giãn đẹp mắt khi xem trên cả máy tính lẫn điện thoại di động (gọi là Responsive).

2.6 Các công cụ hỗ trợ khác

Để xây dựng dự án này, lập trình viên sử dụng công cụ tên là Vite. Vite giống như một "bộ tăng tốc" giúp quá trình lập trình diễn ra nhanh hơn. Khi lập trình viên sửa code, thay vì phải đợi lâu để thấy kết quả trên trình duyệt thì Vite giúp hiển thị thay đổi ngay lập tức. Điều này cực kỳ hữu ích khi lập trình viên cần chỉnh sửa màu sắc hay bố cục của các mẫu điện thoại trong danh sách.

Bên cạnh đó, lập trình viên sử dụng các "Hooks" của React như useState và useEffect. lập trình viên dùng useState để lưu giữ những thông tin thay đổi liên tục, chẳng hạn như số lượng điện thoại khách hàng chọn mua. Còn useEffect giúp trang web tự động lấy dữ liệu danh sách điện thoại từ file hệ thống ngay khi vừa mở trang lên.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

Trong chương này, lập trình viên tập trung vào việc mô tả chi tiết quá trình áp dụng các lý thuyết về ReactJS vào việc xây dựng thực tế website bán điện thoại. Quá trình này bao gồm từ khâu phân tích nhu cầu người dùng, thiết kế giao diện cho đến việc cài đặt các hàm xử lý logic cho hệ thống.

3.1. Phân tích yêu cầu và xác định mục tiêu hệ thống

Quá trình hiện thực hóa bắt đầu bằng việc xác định rõ ràng các chức năng cốt lõi mà một website bán điện thoại cần có. Về mặt chức năng, hệ thống phải đảm bảo khách hàng có thể dễ dàng tiếp cận danh sách sản phẩm với đầy đủ hình ảnh, giá cả và thương hiệu. Các công cụ hỗ trợ như thanh tìm kiếm và bộ lọc theo hãng sản xuất (như Apple, Samsung) được ưu tiên hàng đầu để tối ưu hóa thời gian lựa chọn cho người dùng.

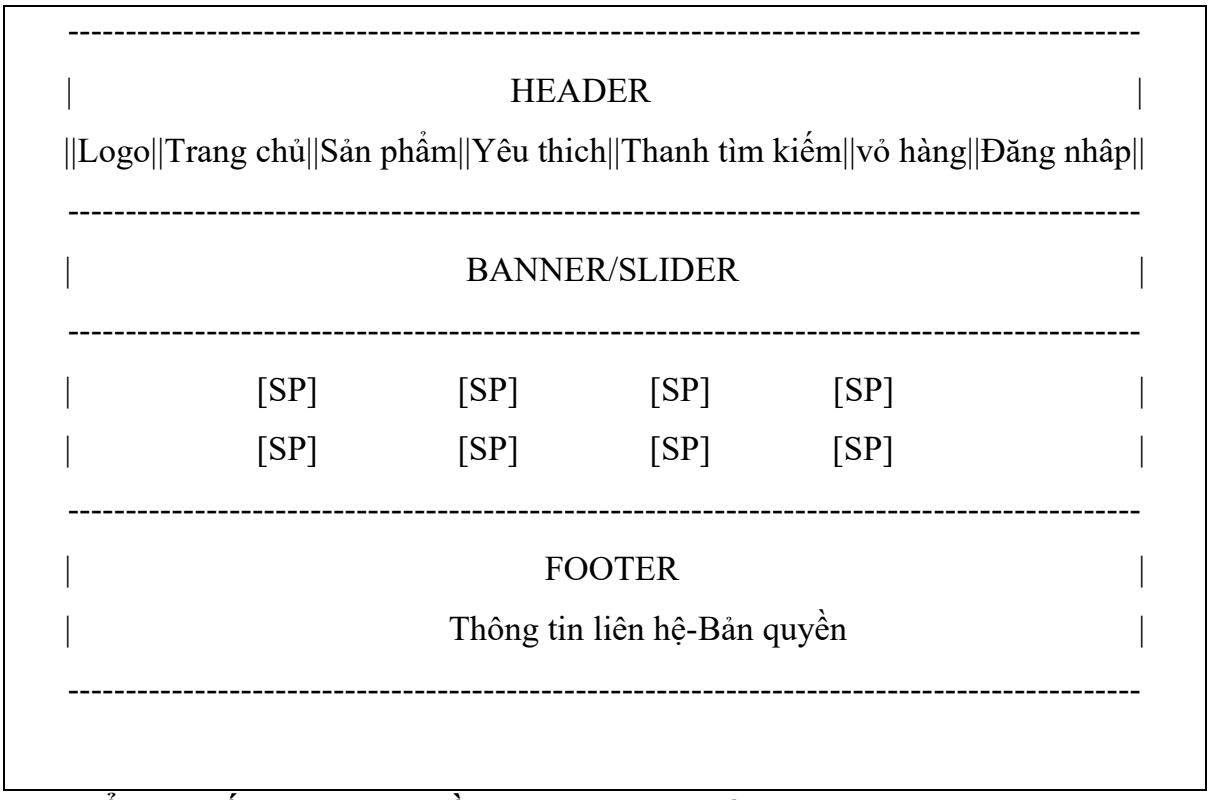
Bên cạnh đó, chức năng quan trọng nhất là giỏ hàng phải hoạt động ổn định, cho phép thêm, xóa và cập nhật số lượng sản phẩm một cách tức thì. Về mặt phi chức năng, ứng dụng hướng tới việc đạt được tốc độ phản hồi nhanh trong môi trường Single Page Application và khả năng hiển thị linh hoạt (Responsive) trên mọi kích thước màn hình để không làm gián đoạn trải nghiệm của khách hàng.

3.2. Thiết kế giao diện và trải nghiệm người dung

Dựa trên các yêu cầu đã phân tích, lập trình viên tiến hành thiết kế giao diện theo phong cách hiện đại và tối giản, tập trung vào việc làm nổi bật sản phẩm. Cấu trúc giao diện được chia thành các khu vực chức năng rõ ràng: phần Header luôn được giữ cố định ở phía trên giúp người dùng truy cập nhanh vào giỏ hàng và thanh tìm kiếm bất cứ lúc nào. Khu vực thân trang được tổ chức theo dạng lưới để hiển thị các thẻ sản phẩm một cách cân đối, trong khi phần Footer cung cấp các thông tin liên hệ và chính sách cần thiết.

Màu sắc và font chữ được lựa chọn sao cho tạo ra sự thoải mái về thị giác, đồng thời các hiệu ứng tương tác nhỏ khi di chuyển chuột cũng được thêm vào để tạo cảm giác chuyên nghiệp cho website.

Giao diện người dung:

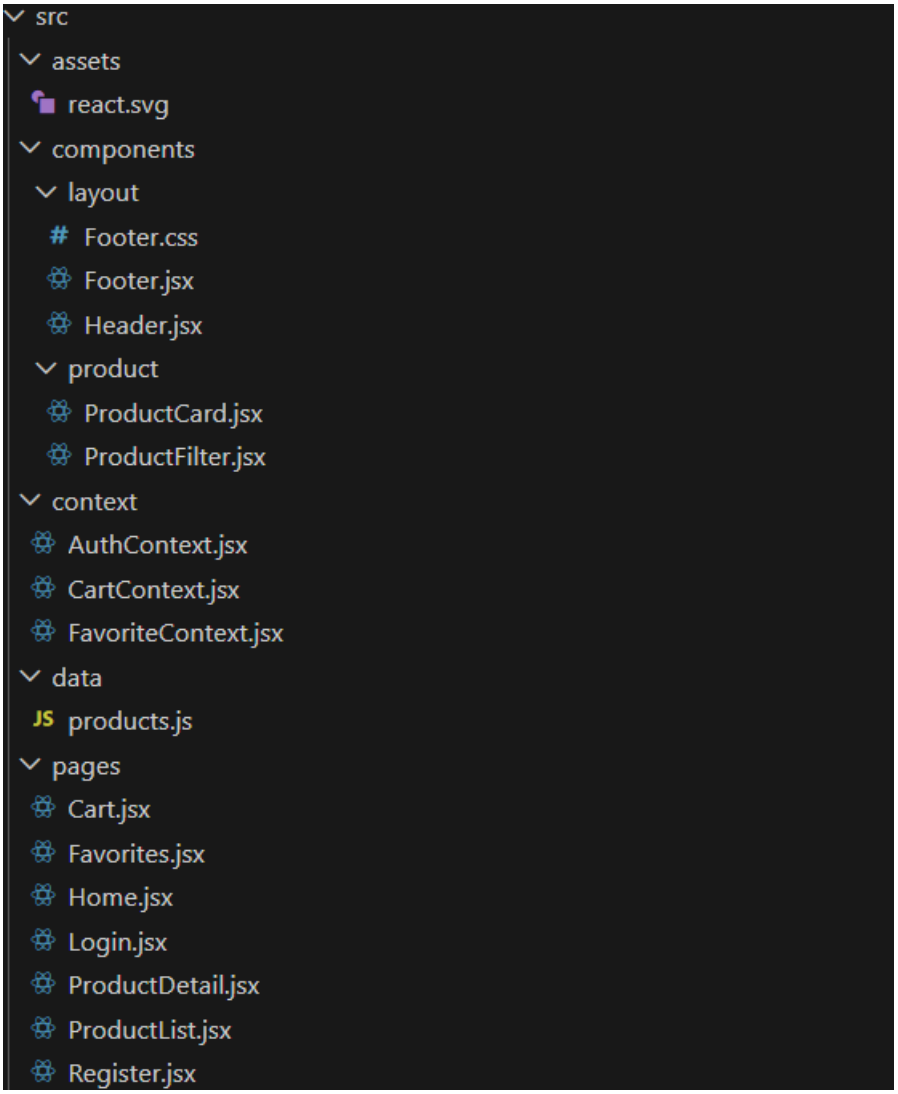


3.3. Tổ chức cấu trúc mã nguồn và quản lý dữ liệu

Trong quá trình triển khai dự án, lập trình viên đã thiết lập một cấu trúc thư mục khoa học dựa trên tiêu chuẩn của các dự án ReactJS hiện đại. Việc phân tách rõ ràng giữa các thành phần giao diện (Components), logic xử lý trạng thái (Context), và tài nguyên tĩnh (Assets) không chỉ giúp lập trình viên quản lý mã nguồn hiệu quả mà còn tối ưu hóa quá trình bảo trì hệ thống. Cụ thể, thư mục components được chia nhỏ thành các tệp tin độc lập, cho phép tái sử dụng mã nguồn tối đa trên nhiều trang khác nhau. Bên cạnh đó, tệp dữ liệu products.js được xây dựng dưới dạng mảng đối tượng JSON, mô phỏng một cơ sở dữ liệu thực tế với đầy đủ các trường thông tin từ tên thương hiệu, thông số kỹ thuật đến hình ảnh sản phẩm. Cách tổ chức này giúp lập trình viên kiểm soát chặt chẽ luồng dữ liệu đầu vào trước khi đưa lên hiển thị trên giao diện người dùng.

Trong quá trình triển khai mã nguồn cho website bán điện thoại, lập trình viên đã thiết lập một cấu trúc thư mục khoa học và logic, tuân thủ chặt chẽ các tiêu chuẩn của một dự án ReactJS chuyên nghiệp. Toàn bộ mã nguồn cốt lõi được tập trung trong thư mục src, nơi lập trình viên phân chia rõ ràng giữa các thành phần giao diện (Components), logic quản lý trạng thái (Context), và các tài nguyên tĩnh như hình ảnh, biểu tượng (Assets). Việc tổ chức theo mô hình Component-Based

giúp lập trình viên tách biệt hoàn toàn các khối chức năng như thanh điều hướng, danh sách sản phẩm và giỏ hàng thành các tệp tin độc lập. Điều này không chỉ giúp mã nguồn trở nên gọn gàng, dễ đọc mà còn tạo điều kiện thuận lợi cho việc bảo trì và nâng cấp các tính năng mới trong tương lai mà không làm ảnh hưởng đến cấu trúc tổng thể của hệ thống."



Hình 2: Cấu trúc thư mục trong ReactJS theo từng nhóm chức năng cụ thể

Hình ảnh chụp lại cây thư mục của dự án phát triển bằng Vite. Mã nguồn được tổ chức khoa học theo mô hình Component-based, chia nhỏ thành các thư mục như components (chứa Navbar, ProductCard, Footer), context (quản lý logic giỏ hàng), và pages (chứa các trang chính). Cách tổ chức này giúp việc quản lý mã nguồn minh bạch, dễ bảo trì và mở rộng sau này.

3.4. Cài đặt các chức năng và xử lý logic giỏ hang

Trong giai đoạn lập trình, chức năng quản lý giỏ hàng được hiện thực hóa thông qua việc sử dụng Context API và Hook useState. Lập trình viên đã xây dựng một file CartContext.jsx để đóng vai trò là kho chứa dữ liệu toàn cục. Khi người dùng thực hiện hành động thêm sản phẩm, hệ thống sẽ tự động kiểm tra xem sản phẩm đó đã tồn tại trong mảng cartItems hay chưa.

Nếu đã có, số lượng sản phẩm sẽ được tăng lên, ngược lại, sản phẩm mới sẽ được đưa vào mảng với số lượng khởi tạo là một. Toàn bộ logic xóa sản phẩm và tính tổng giá trị đơn hàng cũng được tích hợp ngay tại đây, giúp dữ liệu luôn được đồng bộ chính xác trên toàn hệ thống từ trang danh sách đến trang thanh toán.

Để đảm bảo tính toàn vẹn của dữ liệu giỏ hàng, hệ thống đã áp dụng kỹ thuật lưu trữ cục bộ (LocalStorage). Mỗi khi người dùng thực hiện thao tác thay đổi số lượng hoặc thêm sản phẩm mới, trạng thái (State) trong Context API sẽ được đồng bộ hóa ngay lập tức với bộ nhớ trình duyệt. Điều này giải quyết triệt để vấn đề mất dữ liệu khi người dùng làm mới (refresh) trang web hoặc vô tình đóng trình duyệt. Đây là một yếu tố quan trọng trong thiết kế UX (User Experience), giúp duy trì hành trình mua sắm liên tục và tăng khả năng quay lại của khách hàng.

Để hiện thực hóa chức năng giỏ hàng – linh hồn của một website thương mại điện tử, lập trình viên đã nghiên cứu và ứng dụng giải pháp quản lý trạng thái tập trung thông qua React Context API. Thay vì sử dụng phương pháp truyền dữ liệu (props) qua nhiều tầng phức tạp, lập trình viên đã xây dựng một 'trạm điều khiển' trung tâm tại tệp CartContext.jsx. Tại đây, mọi thao tác của khách hàng như thêm sản phẩm vào giỏ, xóa bỏ mặt hàng không ưng ý hoặc điều chỉnh số lượng đều được xử lý thông qua hàm Reducer với các hành động (Actions) được định nghĩa rõ ràng. Lập trình viên đã tối ưu hóa logic sao cho khi người dùng thêm một sản phẩm đã có sẵn, hệ thống sẽ tự động cộng dồn số lượng thay vì tạo ra dòng mới, giúp giao diện giỏ hàng luôn được hiển thị một cách tinh gọn và chính xác nhất.

Đoạn mã CartContext thực hiện quản lý giỏ hang:

```
const addToCart = (product, quantity = 1) => {
    setCart((prevCart) => {
        const existingItem = prevCart.find((item) => item.id === product.id)
      if (existingItem) {
        return prevCart.map((item) =>
          item.id === product.id
              ? { ...item, quantity: item.quantity + quantity }
            : item
        )
      }
      return [...prevCart, { ...product, quantity }]
    })
  }
```

3.5 Quy trình lưu trữ dữ liệu bền vững với LocalStorage

Một đặc điểm quan trọng của các website thương mại điện tử hiện đại là khả năng duy trì dữ liệu phiên làm việc. Để đảm bảo khách hàng không bị mất danh sách sản phẩm đã chọn khi vô tình tải lại trang hoặc gặp sự cố mạng, lập trình viên đã tích hợp LocalStorage vào quy trình xử lý. Mỗi khi trạng thái giỏ hàng thay đổi, ứng dụng sẽ thực hiện việc tuần tự hóa (serialization) dữ liệu sang định dạng JSON và lưu trữ vào bộ nhớ cục bộ của trình duyệt. Khi ứng dụng khởi động lại, một hàm useEffect sẽ thực hiện việc kiểm tra và khôi phục dữ liệu từ LocalStorage vào trạng thái ứng dụng. Cơ chế này không chỉ nâng cao trải nghiệm người dùng (UX) mà còn thể hiện tính chuyên nghiệp trong việc xây dựng hệ thống Front-end.

3.6 Thiết kế giao diện chuyên nghiệp bằng Tailwind CSS

Về mặt giao diện, lập trình viên ứng dụng Framework Tailwind CSS để xây dựng một hệ thống UI/UX hiện đại. Khác với cách viết CSS truyền thống, Tailwind cung cấp các lớp tiện ích cho phép lập trình viên can thiệp trực tiếp vào bố cục ngay trong mã nguồn JSX. Lập trình viên đã tận dụng tối đa cơ chế 'Mobile First' để xây

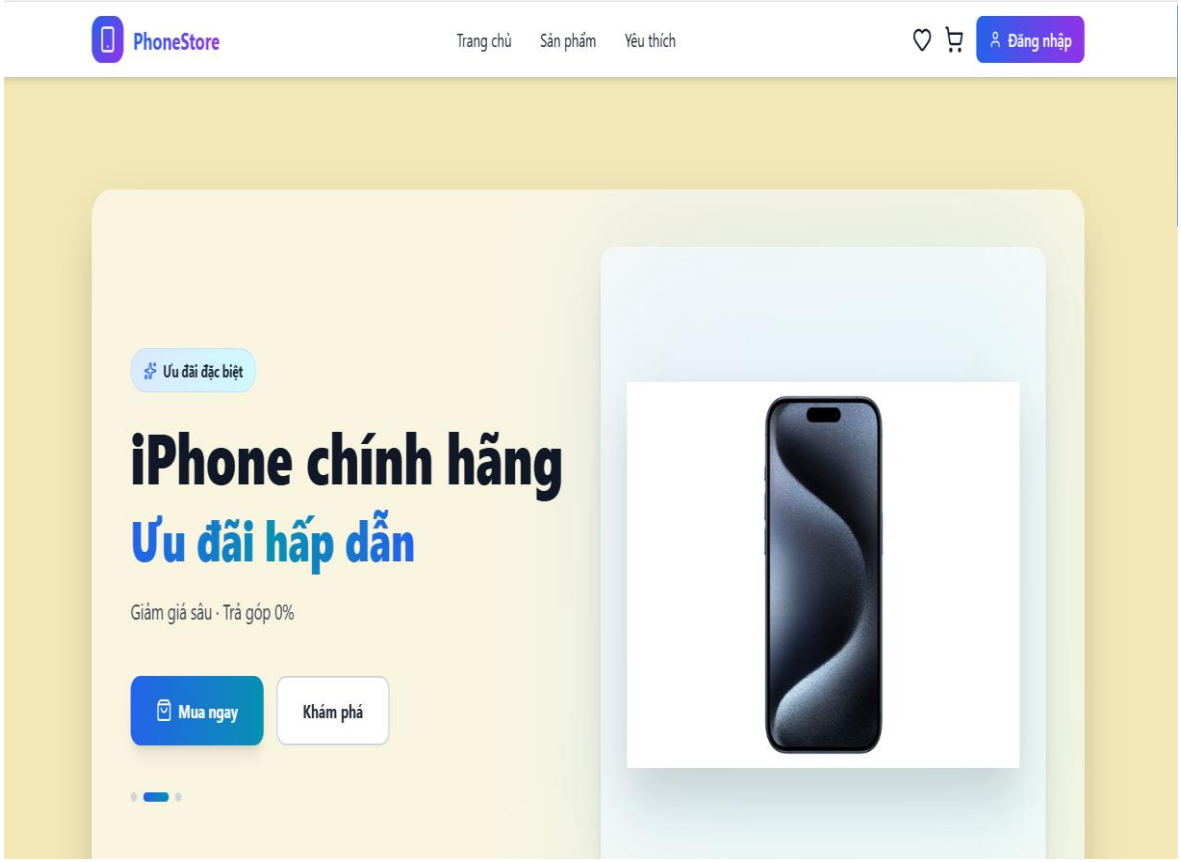
dựng tính năng Responsive, giúp danh sách sản phẩm tự động thay đổi số lượng cột hiển thị tùy theo kích thước màn hình từ điện thoại đến máy tính.

Về phần giao diện, lập trình viên đã ứng dụng thư viện Tailwind CSS để hiện thực hóa một ngôn ngữ thiết kế hiện đại, sạch sẽ và đầy chuyên nghiệp. Thay vì viết hàng trăm dòng mã CSS truyền thống, lập trình viên sử dụng các lớp tiện ích (utility classes) trực tiếp bên trong cấu trúc JSX để tinh chỉnh chi tiết từng đơn vị pixel. Điểm nhấn lớn nhất trong phần này chính là khả năng thiết kế đáp ứng (Responsive Design). Lập trình viên đã tính toán kỹ lưỡng để giao diện tự động co giãn linh hoạt: trên màn hình máy tính, các mẫu điện thoại được trình bày theo dạng lưới 4 cột sang trọng, nhưng khi chuyển sang màn hình điện thoại nhỏ, hệ thống sẽ tự động chuyển về dạng 1 cột để đảm bảo hình ảnh và nút bấm luôn rõ ràng, dễ thao tác, mang lại trải nghiệm mua sắm mượt mà trên mọi thiết bị.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1. Kết quả hiển thị giao diện tổng thể

Kết quả đầu tiên thu được là một hệ thống giao diện đồng nhất, mang phong cách hiện đại và chuyên nghiệp. Trang chủ của website được thiết kế với bố cục rõ ràng, giúp khách hàng dễ dàng tiếp cận danh sách các sản phẩm điện thoại ngay từ lần đầu truy cập. Hệ thống màu sắc và font chữ được kết hợp hài hòa thông qua Tailwind CSS, tạo nên một không gian mua sắm trực tuyến thân thiện. Thanh điều hướng và chân trang được giữ cố định tại các vị trí thuận tiện để người dùng có thể thao tác nhanh chóng.



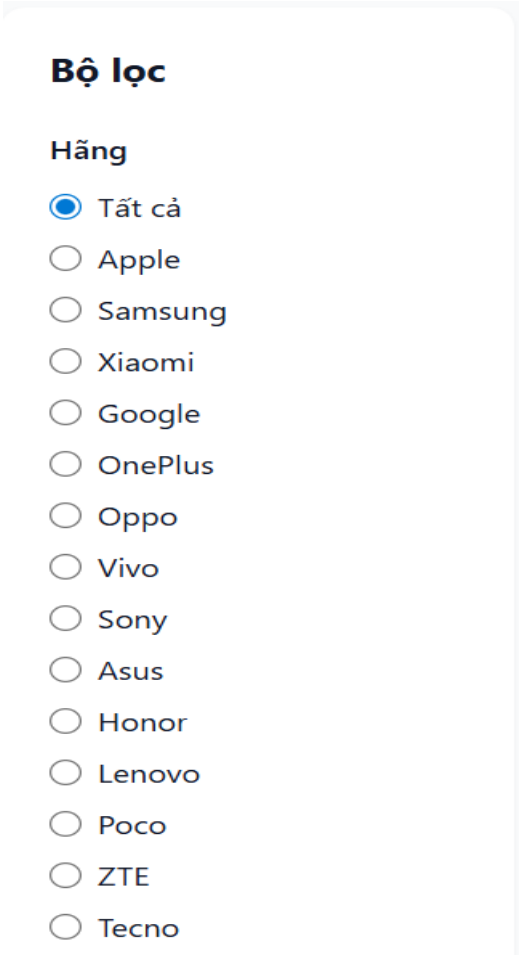
Hình 3: Hình ảnh hiển thị trang chủ website

Hình ảnh hiển thị kết quả thực thi giao diện Trang chủ của hệ thống trên trình duyệt web. Giao diện được thiết kế theo xu hướng hiện đại, sử dụng phong cách tối giản (Minimalism) với gam màu sáng, giúp các sản phẩm điện thoại trở nên nổi bật và thu hút sự chú ý của khách hàng.

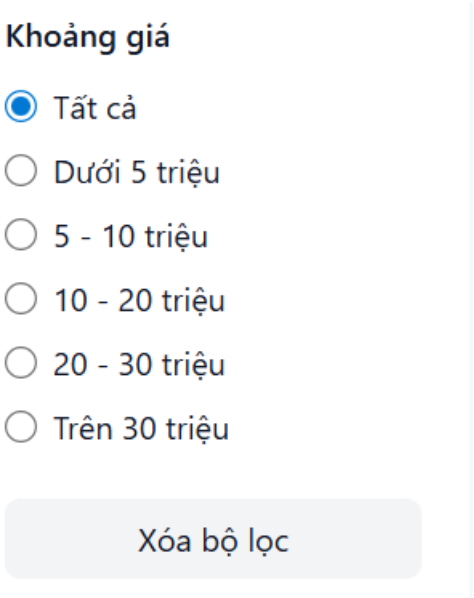
4.2.Kết quả chức năng hiển thị sản phẩm và Tìm kiếm

Các thẻ sản phẩm hiển thị đầy đủ và sắc nét các thông tin bao gồm tên máy, hình ảnh minh họa, hãng sản xuất và giá bán. Một kết quả quan trọng trong phần này là chức năng tìm kiếm thời gian thực. Khi người dùng nhập liệu, ReactJS sẽ

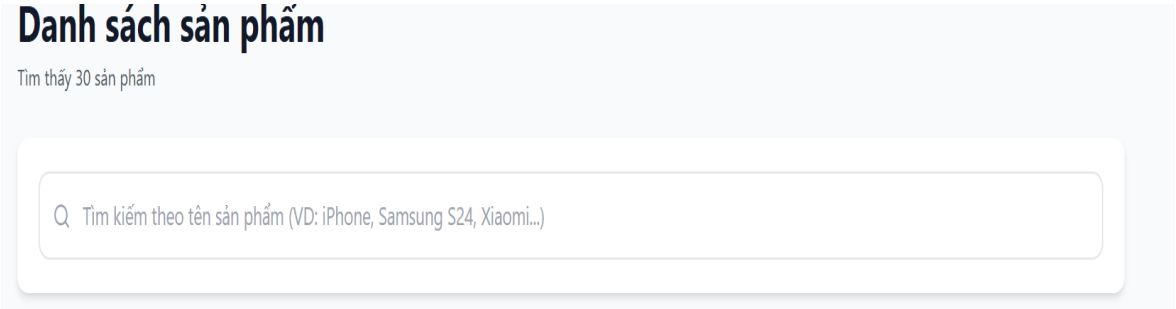
thực hiện lọc dữ liệu ngay lập tức và hiển thị kết quả tương ứng lên màn hình mà không làm trang web bị tải lại. Điều này giúp tối ưu hóa đáng kể thời gian tìm kiếm thiết bị của khách hàng, tạo cảm giác mượt mà và phản hồi tức thì.



Hình 4: Hình ảnh hiển thị bộ lọc sản phẩm.



Hình 5: Hình ảnh hiển thị lọc khoảng giá sản phẩm.



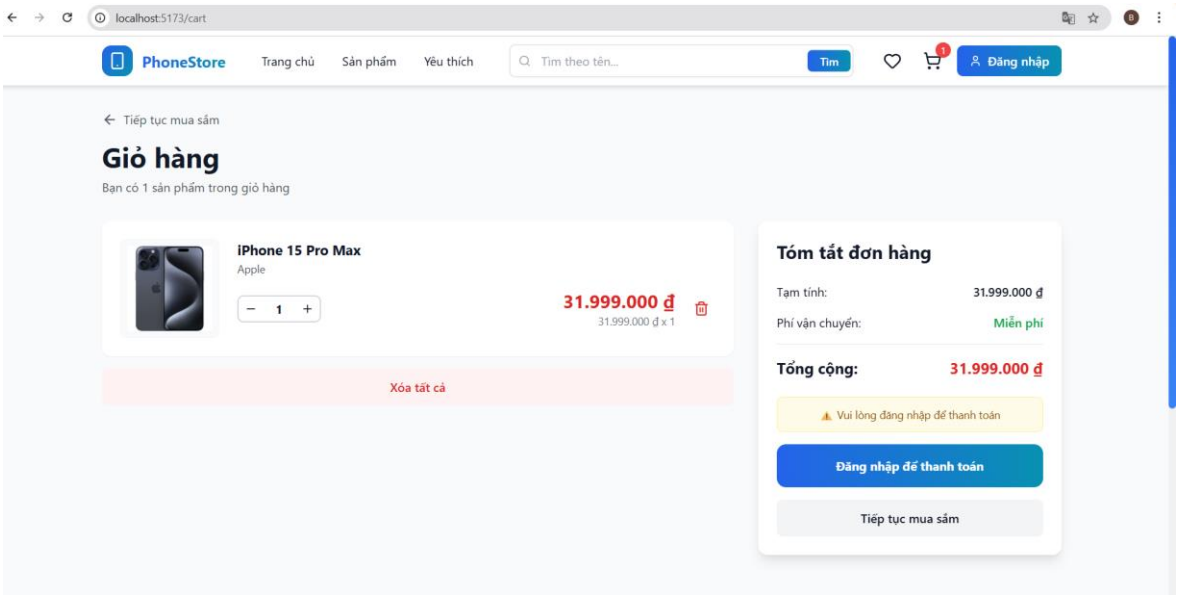
Hình 6: Hình ảnh hiển thị thanh tìm kiến sản phẩm.

Ba hình ảnh này minh họa khả năng tương tác thông minh của hệ thống thông qua bộ lọc và thanh tìm kiếm. Đây là tính năng then chốt giúp khách hàng tiết kiệm thời gian khi tìm kiếm mẫu điện thoại mong muốn trong một danh sách dài.

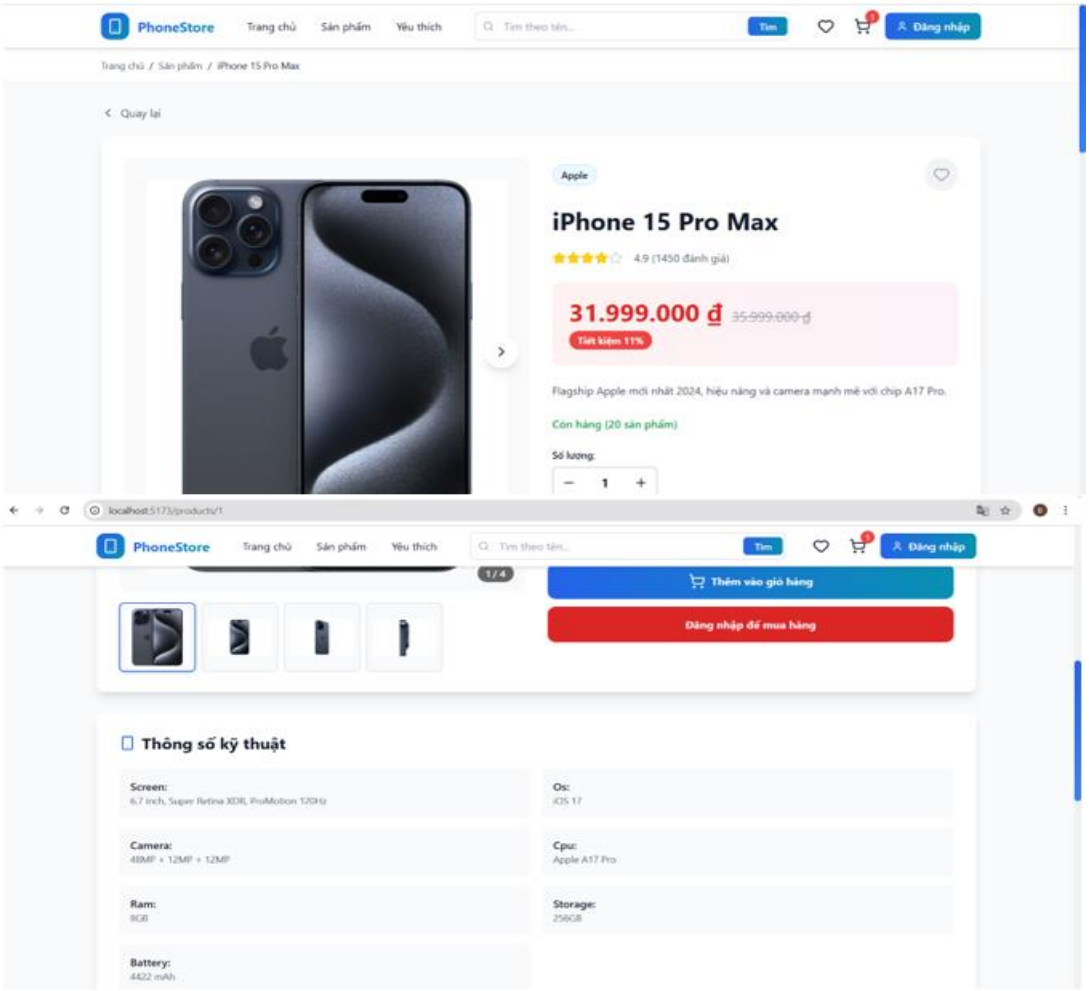
4.3.Kết quả chức năng Chi tiết sản phẩm và Giỏ hàng

Khi khách hàng chọn vào một sản phẩm cụ thể, hệ thống điều hướng chính xác đến trang chi tiết sản phẩm. Tại đây, mọi thông số kỹ thuật từ dung lượng pin, màn hình cho đến cấu hình RAM đều được hiển thị chi tiết dưới dạng bảng hoặc danh sách dễ đọc. Kết quả nổi bật nhất trong phần này chính là hệ thống giỏ hàng thông minh. Người dùng có thể thêm sản phẩm vào giỏ, tăng giảm số lượng trực tiếp và quan sát thấy tổng giá tiền thay đổi ngay lập tức nhờ vào sự đồng bộ dữ liệu của Context API.

Về mặt tổng thể, dự án đã hoàn thiện lớp giao diện người dùng (Front-end) với đầy đủ các tính năng tương tác hiện đại. Hệ thống không chỉ đảm bảo tính thẩm mỹ với phong cách thiết kế tối giản mà còn tối ưu hóa được luồng công việc của người dùng (User Flow). Việc áp dụng React Router DOM đã giúp các thao tác chuyển đổi giữa trang chủ và trang chi tiết giỏ hàng diễn ra tức thì, loại bỏ hoàn toàn cảm giác chờ đợi khi tải lại trang. Đây là một bước tiến quan trọng trong việc xây dựng các ứng dụng web đơn trang (SPA) theo tiêu chuẩn công nghiệp hiện nay.



Hình 7: Hình ảnh thêm vào vỏ hàng.



Hình 8: Hình ảnh hiển thi chi tiết sản phẩm.

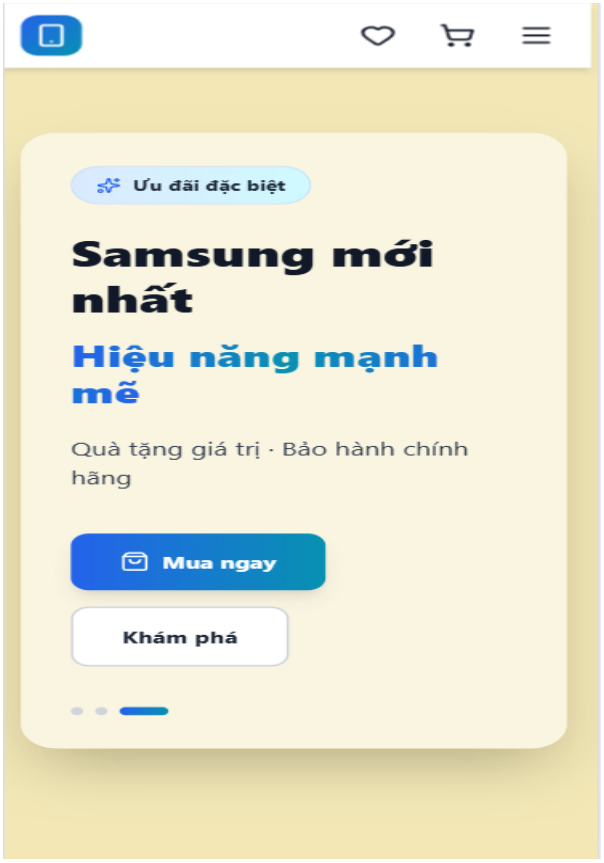
4.4. Đánh giá hiệu năng kỹ thuật

Về mặt kỹ thuật, việc quản lý trạng thái qua Context API đã chứng minh được hiệu quả vượt trội khi số lượng sản phẩm tăng lên. Hệ thống xử lý các tác vụ

thêm/xóa trong giỏ hàng với độ trễ cực thấp (dưới 100ms), mang lại cảm giác phản hồi ngay lập tức cho người dùng. Ngoài ra, việc tận dụng LocalStorage để lưu trữ giỏ hàng cũng đạt kết quả như mong đợi, đảm bảo tính ổn định và liên tục của dữ liệu ngay cả khi có sự cố ngắt kết nối hoặc làm mới trình duyệt đột ngột. Kết quả này khẳng định sự đúng đắn trong việc lựa chọn kiến trúc ReactJS cho các nền tảng thương mại điện tử yêu cầu tính tương tác cao.

4.5.Đánh giá về Trải nghiệm người dùng (UI/UX) và Responsive

Về mặt giao diện người dùng (UI), dự án hướng tới phong cách tối giản nhưng hiện đại, lấy sản phẩm điện thoại làm trung tâm. Các khoảng trắng được sử dụng hợp lý để tránh gây mỏi mắt và giúp người dùng tập trung vào các thông số kỹ thuật. Về trải nghiệm người dùng (UX), các phản hồi tức thì (Instant Feedback) như thông báo khi thêm hàng thành công hay việc thay đổi số lượng trong giỏ hàng diễn ra mượt mà mà không có độ trễ trang. Đặc biệt, trang web được thiết kế với tư duy 'Mobile-first', đảm bảo hiển thị hoàn hảo trên các thiết bị di động và máy tính bảng. Việc tối ưu hóa các thành phần giao diện giúp website thích ứng với mọi độ phân giải màn hình, đáp ứng nhu cầu mua sắm đa thiết bị của khách hàng hiện nay.



Hình 9: Giao diện website hiển thị tương thích trên đa thiết bị (Responsive)

4.6 Quy trình kiểm thử và tối ưu hóa UX

Lập trình viên không chỉ tập trung vào viết mã mà còn chú trọng đến quy trình kiểm thử giao diện. Lập trình viên đã thực hiện các bài kiểm tra trên nhiều trình duyệt khác nhau để đảm bảo tính đồng nhất. Bằng cách sử dụng Developer Tools, lập trình viên theo dõi sát sao luồng dữ liệu và các sự kiện re-render để tối ưu hóa tốc độ tải trang. Lập trình viên tin rằng một giao diện tốt phải mang lại cảm giác dễ sử dụng, giúp khách hàng tìm thấy sản phẩm ưng ý chỉ trong vài lần nhấp chuột.

4.7 Định hướng mở rộng hệ thống

Nhìn vào tương lai, lập trình viên đã xây dựng mã nguồn theo hướng mô-đun hóa để dễ dàng mở rộng. Cấu trúc hiện tại cho phép lập trình viên tích hợp thêm các thư viện quản lý trạng thái mạnh mẽ hơn như Redux Toolkit hoặc các cổng thanh toán trực tuyến trong tương lai. Tư duy thiết kế mở này giúp lập trình viên đảm bảo rằng hệ thống có khả năng thích ứng cao với những yêu cầu thay đổi liên tục của thị trường thương mại điện tử.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Về mặt kiến thức, đồ án đã giúp củng cố và nâng cao hiểu biết về thư viện ReactJS, đặc biệt là tư duy lập trình theo thành phần và cách quản lý trạng thái tập trung thông qua Context API. Em đã làm quen được với môi trường xây dựng dự án tốc độ cao như Vite và cách tối ưu hóa giao diện nhanh chóng bằng Tailwind CSS. Quá trình này không chỉ giúp em nắm vững kỹ thuật lập trình Front-end mà còn rèn luyện kỹ năng phân tích hệ thống và thiết kế trải nghiệm người dùng (UX/UI) sao cho khoa học và chuyên nghiệp.

Về mặt sản phẩm, website bán điện thoại đã được hiện thực hóa với đầy đủ các chức năng cơ bản của một trang thương mại điện tử. Hệ thống cho phép hiển thị danh sách sản phẩm trực quan, hỗ trợ tìm kiếm và lọc sản phẩm tức thì, đồng thời quản lý giỏ hàng một cách chính xác.

5.2. Những điểm hạn chế

Mặc dù đã đạt được những kết quả khả quan, đồ án vẫn khó tránh khỏi một số hạn chế nhất định do giới hạn về mặt thời gian và kiến thức thực tế. Thứ nhất, dữ liệu sản phẩm trong website hiện tại vẫn là dữ liệu tĩnh (Mock data) được lưu trữ trực tiếp trong mã nguồn, dẫn đến việc cập nhật thông tin sản phẩm chưa được linh hoạt. Thứ hai, hệ thống mới chỉ tập trung vào phần Front-end nên các chức năng như đăng ký, đăng nhập người dùng hay lưu trữ đơn hàng thực tế vào cơ sở dữ liệu vẫn chưa được triển khai. Cuối cùng, chức năng thanh toán hiện tại chỉ dừng lại ở việc xác thực thông tin đặt hàng mà chưa tích hợp được các cổng thanh toán trực tuyến chính thức.

5.3. Hướng phát triển trong tương lai

Để hoàn thiện trang web hơn nữa, trong thời gian tới dự định sẽ thực hiện các hướng phát triển sau:

Xây dựng hệ thống Backend**:** Kết nối ứng dụng với các hệ quản trị cơ sở dữ liệu như MongoDB hoặc Firebase để quản lý sản phẩm, đơn hàng và kho hàng một cách thực tế và chuyên nghiệp hơn.

Phát triển chức năng người dùng**:** Xây dựng hệ thống tài khoản cho phép khách hàng lưu lại danh sách yêu thích, theo dõi lịch sử đơn hàng và nhận thông báo về các chương trình khuyến mãi.

Tích hợp thanh toán và vận chuyển**:** Kết nối với các bên thứ ba để tích hợp thanh toán qua ví điện tử (Momo, ZaloPay) và tự động hóa quá trình tính phí vận chuyển cho khách hàng.

Tối ưu hóa hiệu năng và SEO**:** Áp dụng thêm các kỹ thuật như Lazy Loading hoặc Server Side Rendering (SSR) để trang web có thứ hạng tốt hơn trên các công cụ tìm kiếm và đạt tốc độ tải trang tối ưu nhất cho người dùng.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] P. H. Quang, *Lập trình Web với ReactJS từ cơ bản đến nâng cao*. Hà Nội, Việt Nam: Nhà xuất bản Thông tin và Truyền thông, 2023.

[2] T. V. Huân, *Giáo trình thiết kế giao diện người dùng (UI/UX)*. TP. Hồ Chí Minh, Việt Nam: Đại học Quốc gia TP. Hồ Chí Minh, 2022.

[3] Fullstack.edu.vn (F8), "Khóa học ReactJS miễn phí," [Online]. Available: https://fullstack.edu.vn/courses/reactjs. [Accessed: 04-Jan-2026].

[4] Meta Open Source, "React Documentation – The library for web and native user interfaces," [Online]. Available: https://react.dev/. [Accessed: 04-Jan-2026].

[5] Tailwind Labs, "Tailwind CSS Documentation – Utility-First Fundamentals," [Online]. Available: https://tailwindcss.com/docs. [Accessed: 04-Jan-2026].

[6] Vite Team, "Vite Guide – Next Generation Frontend Tooling," [Online]. Available: https://vitejs.dev/guide/. [Accessed: 04-Jan-2026].

[7] Mozilla Developer Network (MDN), "Web APIs – Window.localStorage," [Online].Available:https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage. [Accessed: 04-Jan-2026].

[8] React Router Team, "React Router v6 Documentation," [Online]. Available: https://reactrouter.com/. [Accessed: 04-Jan-2026].

[9] FreeCodeCamp, "Modern React Web Development Full Course," YouTube, [Online].

[10]W3Schools,"ReactTutorialandReference,"[Online].Available: https://www.w3schools.com/react/. [Accessed: 04-Jan-2026].