KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**THỰC TẬP ĐỒ ÁN CHUYÊN NGÀNH**

**HỌC KỲ I, NĂM HỌC 2024**

**XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ BÁN ĐỒ CÔNG NGHỆ BẰNG REACTJS VÀ NODE.JS**

*Giáo viên hướng dẫn:*

ThS. Nguyễn Ngọc Đan Thanh

*Sinh viên thực hiện:*

Họ tên: Trần Trung Nghĩa

MSSV: 110121066

Lớp: DA21TTB

***Trà Vinh, tháng … năm 2024***

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**THỰC TẬP ĐỒ ÁN CHUYÊN NGÀNH**

**HỌC KỲ I, NĂM HỌC 2024**

**XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ BÁN ĐỒ CÔNG NGHỆ BẰNG REACTJS VÀ NODE.JS**

*Giáo viên hướng dẫn:*ThS. Nguyễn Ngọc Đan Thanh

*Sinh viên thực hiện:*

Họ tên: Trần Trung Nghĩa

MSSV: 110121066

Lớp: DA21TTB

***Trà Vinh, tháng … năm 2024***

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

1. **Quá trình thực hiện:**

* …
* …

1. **Về báo cáo**

* …
* …

1. **Về chương trình**

* …
* …

1. **Kết luận:** …

*Trà Vinh, ngày …... tháng …… năm 2024*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm 2024*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

Em xin gửi lời cảm ơn chân thành đến cô Nguyễn Ngọc Đan Thanh - giảng viên Bộ môn Công nghệ thông tin - Khoa Kỹ thuật và Công nghệ trường Đại học Trà Vinh đã giúp đỡ và hướng dẫn cho em rất nhiều trong quá trình tìm hiểu và thực hiện đồ án chuyên ngành.

Tuy nhiên trong quá trình thực hiện đồ án chuyên ngành, do chưa có nhiều kinh nghiệm và kiến thức còn hạn chế trong quá trình tìm hiểu cũng như làm đồ án nên em vẫn còn gặp một số hạn chế và sự thiếu sót. Rất mong nhận được sự quan tâm, nhận xét và những lời góp ý của cô Nguyễn Ngọc Đan Thanh cùng các thầy cô giảng viên bộ môn để cho đề tài của em được hoàn thiện chỉnh chu hơn.

Em xin chân thành cảm ơn.

**MỤC LỤC**

[CHƯƠNG 1 TỔNG QUAN 10](#_Toc182572403)

[1.1 Giới thiệu tổng quan về chủ đề 10](#_Toc182572404)

[1.2 Tổng quan về các hệ thống tương tự 10](#_Toc182572405)

[1.3 Phân tích các công nghệ được sử dụng 10](#_Toc182572406)

[1.4 Hạn chế của các hệ thống hiện có và đề xuất giải pháp 11](#_Toc182572407)

[CHƯƠNG 2 NGHIÊN CỨU LÝ THUYẾT 12](#_Toc182572408)

[2.1 ReactJS 12](#_Toc182572409)

[2.1.1 Giới thiệu [1] 12](#_Toc182572410)

[2.1.2 Lợi ích của việc sử dụng ReactJS [2] 12](#_Toc182572411)

[2.1.3 Bắt đầu cài đặt với ReactJS 13](#_Toc182572412)

[2.1.4 Giao diện trong ReactJS 14](#_Toc182572413)

[2.2 Node.js 26](#_Toc182572414)

[2.2.1 Giới thiệu [3] 26](#_Toc182572415)

[2.2.2 Lợi ích của việc sử dụng Node.js [6] 26](#_Toc182572416)

[2.2.3 Bắt đầu cài đặt với Node.js 27](#_Toc182572417)

[2.3 Kết luận 31](#_Toc182572418)

[CHƯƠNG 3 HIỆN THỰC HÓA NGHIÊN CỨU 32](#_Toc182572419)

[3.1 Mô tả bài toán 32](#_Toc182572420)

[3.2 Kiểm thử đầy đủ 32](#_Toc182572421)

[3.3 Thiết kế giao diện 32](#_Toc182572422)

[CHƯƠNG 4 KẾT QUẢ NGHIÊN CỨU 33](#_Toc182572423)

[4.1 Dữ liệu thử nghiệm 33](#_Toc182572424)

[4.2 Kết quả thử nghiệm 33](#_Toc182572425)

[4.3 Kết luận 33](#_Toc182572426)

[CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 34](#_Toc182572427)

[5.1 Kết luận 34](#_Toc182572428)

[5.2 Hướng phát triển 34](#_Toc182572429)

[DANH MỤC TÀI LIỆU THAM KHẢO 35](#_Toc182572430)

[PHỤ LỤC 36](#_Toc182572431)

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

No table of figures entries found.

[Hình 2.1 Kiểm tra phiên bản 13](#_Toc182570887)

[Hình 2.2 Ví dụ về component 18](#_Toc182570888)

[Hình 2.3 Tải Node.js 27](#_Toc182570889)

[Hình 2.4 Cài đặt Node.js 27](#_Toc182570890)

[Hình 2.5 Kiểm tra phiên bản Node.js 28](#_Toc182570891)

[Hình 2.6 Chạy server Node.js 29](#_Toc182570892)

**TÓM TẮT ĐỒ ÁN CHUYÊN NGÀNH**

**Vấn đề nghiên cứu**

Đồ án này tập trung vào việc nghiên cứu và xây dựng một website thương mại điện tử bán đồ công nghệ sử dụng ReactJS và Node.js. Vấn đề trọng tâm là làm thế nào để tích hợp các công nghệ hiện đại nhằm cung cấp trải nghiệm người dùng tốt, bảo đảm hiệu suất cao và khả năng mở rộng cho hệ thống.

**Hướng tiếp cận**

**Frontend**: Xây dựng giao diện người dùng bằng ReactJS với kiến trúc dựa trên component, sử dụng React Router để điều hướng và quản lý trạng thái hiệu quả bằng Redux. Giao diện được thiết kế thân thiện với người dùng, áp dụng Tailwind CSS để tăng tính thẩm mỹ và hỗ trợ responsive design cho mọi thiết bị.

**Backend**: Phát triển backend bằng Node.js và Express, xây dựng các RESTful API để giao tiếp với cơ sở dữ liệu MySQL, đồng thời cung cấp tính năng quản lý sản phẩm, người dùng và giỏ hàng thông qua hệ thống quản trị (Admin).

**Cách giải quyết vấn đề**

Để giải quyết vấn đề, tôi đã tiến hành phân tích yêu cầu nhằm xác định các tính năng cần thiết cho một website thương mại điện tử, bao gồm hiển thị sản phẩm, quản lý giỏ hàng và thanh toán. Dựa vào đó, tôi thiết kế hệ thống với các thành phần chính: frontend, backend, cơ sở dữ liệu và hệ thống quản trị. Sau cùng, tôi triển khai website với các chức năng như danh sách và chi tiết sản phẩm, thêm vào giỏ hàng và thanh toán.

**Kết quả đạt được**

Dự án đã hoàn thiện với các tính năng cần thiết cho một website thương mại điện tử bao gồm việc xây dựng giao diện người dùng và tối ưu trải nghiệm với ReactJS và Tailwind CSS. Backend đã được phát triển với RESTful API đảm bảo giao tiếp an toàn giữa frontend và backend. Ngoài ra, hệ thống quản trị cung cấp khả năng thêm, xóa, sửa thông tin sản phẩm trực tiếp từ backend. Sản phẩm có thể triển khai trên các nền tảng như Heroku hoặc Vercel, đảm bảo khả năng mở rộng và bảo mật.

**MỞ ĐẦU**

**Lý do chọn đề tài**

Trong bối cảnh hiện đại, khi công nghệ thông tin ngày càng phát triển và ảnh hưởng sâu rộng đến mọi lĩnh vực, mua sắm trực tuyến trở thành một xu hướng tất yếu. Ngành thương mại điện tử đã phát triển mạnh mẽ, đáp ứng nhu cầu ngày càng cao của người tiêu dùng trong việc tìm kiếm các sản phẩm nhanh chóng, tiện lợi và an toàn. Đặc biệt, đồ công nghệ là một trong những mặt hàng phổ biến và được ưa chuộng, với nhu cầu mua sắm trực tuyến ngày càng tăng.

Với mong muốn tìm hiểu sâu hơn về quy trình phát triển một website thương mại điện tử, cùng với tiềm năng ứng dụng các công nghệ hiện đại, đề tài "Xây dựng website thương mại điện tử bán đồ công nghệ sử dụng ReactJS và Node.js" được chọn nhằm cung cấp một giải pháp hiệu quả, nâng cao trải nghiệm người dùng và đảm bảo khả năng mở rộng cho hệ thống. Qua đề tài này, tôi có cơ hội thực hành và áp dụng các kiến thức về lập trình frontend, backend và quản lý cơ sở dữ liệu trong một dự án thực tiễn.

**Mục đích nghiên cứu**

Mục đích của đồ án là xây dựng một website thương mại điện tử hoàn chỉnh, cung cấp các tính năng cơ bản như quản lý sản phẩm, quản lý giỏ hàng, xử lý thanh toán và quản trị người dùng. Bằng cách tích hợp các công nghệ hiện đại như ReactJS cho frontend và Node.js cho backend, đề tài mong muốn mang đến một trải nghiệm người dùng mượt mà, tối ưu hóa hiệu suất và đảm bảo bảo mật thông tin.

**Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài là các công nghệ chính được sử dụng để xây dựng một hệ thống thương mại điện tử, bao gồm ReactJS, Node.js, Express, MySQL và Tailwind CSS. Đề tài tập trung vào cách các công nghệ này có thể kết hợp với nhau để tạo nên một hệ thống hoạt động hiệu quả, đáp ứng yêu cầu của người dùng và dễ dàng bảo trì.

**Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đồ án giới hạn trong việc xây dựng một website thương mại điện tử đơn giản dành cho các sản phẩm công nghệ. Đề tài bao gồm các nội dung như phân tích yêu cầu, thiết kế và triển khai hệ thống với các chức năng cơ bản như hiển thị sản phẩm, quản lý giỏ hàng, thanh toán và quản trị thông tin sản phẩm.

# TỔNG QUAN

## Giới thiệu tổng quan về chủ đề

Trong bối cảnh công nghệ thông tin không ngừng phát triển, mua sắm trực tuyến đã trở thành xu hướng phổ biến trên toàn thế giới. Các website thương mại điện tử hiện nay không chỉ đáp ứng yêu cầu về chức năng mà còn cần cung cấp trải nghiệm người dùng mượt mà, dễ dàng và an toàn. Đặc biệt, với sự phát triển nhanh chóng của ngành hàng công nghệ, việc xây dựng một trang web thương mại điện tử chuyên bán đồ công nghệ là điều cần thiết để đáp ứng nhu cầu của người tiêu dùng về tính tiện dụng, khả năng tương tác cao và trải nghiệm mua sắm an toàn.

Đồ án này tập trung vào nghiên cứu và phát triển một website thương mại điện tử bán đồ công nghệ sử dụng các công nghệ tiên tiến như ReactJS và Node.js, nhằm tạo ra một ứng dụng web có khả năng mở rộng, dễ bảo trì và cung cấp trải nghiệm người dùng vượt trội.

## Tổng quan về các hệ thống tương tự

Hiện nay, trên thị trường đã xuất hiện nhiều hệ thống thương mại điện tử nổi bật như Amazon, Lazada và Shopee. Các nền tảng này có điểm mạnh là khả năng quản lý kho hàng, xử lý đơn hàng và giao diện người dùng phù hợp với đại đa số người dùng. Tuy nhiên, nhiều hệ thống vẫn có những hạn chế về khả năng cá nhân hóa trải nghiệm người dùng và thiếu đi sự mượt mà trong quá trình sử dụng.

Từ những hạn chế trên, đề tài của em sẽ tập trung vào việc cải tiến các điểm sau:

Khả năng tương tác: Sử dụng ReactJS để xây dựng giao diện người dùng giúp cho trang web phản hồi nhanh hơn và dễ sử dụng.

Quản lý trạng thái tối ưu: Với Redux, các trạng thái của ứng dụng sẽ được quản lý tập trung, giúp giảm thiểu rủi ro sai lệch trong dữ liệu.

Bảo mật dữ liệu: Node.js kết hợp với các phương pháp mã hóa hiện đại sẽ giúp bảo vệ thông tin người dùng an toàn hơn.

## Phân tích các công nghệ được sử dụng

Để xây dựng một website thương mại điện tử toàn diện, các công nghệ dưới đây sẽ được áp dụng:

ReactJS: Đây là một thư viện JavaScript phổ biến với cấu trúc dựa trên component, cho phép tái sử dụng mã nguồn và tạo giao diện người dùng phản hồi nhanh chóng. ReactJS hỗ trợ các thư viện bổ trợ như Redux để quản lý trạng thái và React Router để điều hướng trang một cách linh hoạt. [1]

Node.js: Với đặc tính non-blocking I/O, Node.js là nền tảng lý tưởng để xây dựng RESTful API có khả năng xử lý lượng lớn yêu cầu từ phía người dùng một cách hiệu quả. Sự kết hợp với Express giúp tối ưu hóa các tác vụ backend và dễ dàng tích hợp với các cơ sở dữ liệu như MySQL. [2]

MySQL: Được sử dụng để lưu trữ và quản lý dữ liệu, MySQL cung cấp khả năng xử lý truy vấn nhanh chóng và hỗ trợ quản lý dữ liệu quan hệ, phù hợp cho nhu cầu lưu trữ thông tin sản phẩm, đơn hàng và người dùng. [3]

Các công nghệ này được chọn lựa không chỉ vì khả năng đáp ứng tốt yêu cầu về chức năng mà còn nhờ vào tính phổ biến và sự hỗ trợ cộng đồng mạnh mẽ, giúp dễ dàng bảo trì và mở rộng trong tương lai.

## Hạn chế của các hệ thống hiện có và đề xuất giải pháp

Các hệ thống thương mại điện tử hiện tại, dù có nhiều tính năng mạnh mẽ, vẫn tồn tại một số hạn chế đáng chú ý như:

Thiếu khả năng tùy chỉnh giao diện theo nhu cầu người dùng.

Khó khăn trong việc mở rộng hoặc điều chỉnh chức năng khi nhu cầu thay đổi.

Cơ chế bảo mật dữ liệu chưa đủ mạnh để chống lại các mối đe dọa ngày càng phức tạp của các hacker.

Đồ án này sẽ giải quyết những hạn chế trên bằng cách:

Tạo ra một hệ thống linh hoạt, dễ tùy chỉnh và mở rộng thông qua cấu trúc component của ReactJS.

Cải thiện tính bảo mật thông tin người dùng bằng các phương pháp mã hóa hiện đại trong Node.js.

Tối ưu hóa trải nghiệm người dùng với tốc độ phản hồi nhanh, giao diện đơn giản và dễ sử dụng.

# NGHIÊN CỨU LÝ THUYẾT

## ReactJS

### Giới thiệu [1]

ReactJS, được phát triển bởi Facebook vào năm 2011, ban đầu nhằm cải thiện tốc độ và hiệu suất giao diện người dùng trên trang web Facebook. Đến năm 2013, Facebook công bố mã nguồn mở ReactJS, giúp nó nhanh chóng trở thành một trong những thư viện phổ biến nhất cho phát triển ứng dụng web động. Năm 2015, Facebook tiếp tục giới thiệu React Native, một framework phát triển ứng dụng di động dựa trên ReactJS, cho phép xây dựng ứng dụng cho cả iOS và Android từ cùng một mã nguồn. Ngày nay, ReactJS đã trở thành thư viện phát triển web hàng đầu, được sử dụng rộng rãi bởi các công ty trên toàn thế giới, với sự đầu tư liên tục từ Facebook để đáp ứng nhu cầu của cộng đồng lập trình viên.

ReactJS là một thư viện JavaScript phát triển bởi Facebook, được sử dụng rộng rãi để xây dựng giao diện người dùng động và tương tác trong các ứng dụng web. Đây là một công cụ mạnh mẽ cho việc phát triển ứng dụng web động với hiệu suất cao và dễ bảo trì.

### Lợi ích của việc sử dụng ReactJS [2]

Hiệu suất cao: ReactJS sử dụng một cơ chế gọi là "Virtual DOM" để cải thiện hiệu suất. Thay vì cập nhật toàn bộ giao diện người dùng mỗi khi có thay đổi, React sẽ cập nhật chỉ những phần tử có thay đổi. Điều này giúp giảm tải cho trình duyệt và tăng tốc độ của ứng dụng.

Thư viện phong phú: ReactJS đi kèm với một cộng đồng lớn và nhiều thư viện mở rộng hỗ trợ. Bạn có thể dễ dàng tích hợp React với các thư viện và công cụ khác để phát triển các tính năng phức tạp.

Quản lý trạng thái tốt: React sử dụng mô hình quản lý trạng thái (state) rất tốt. Trạng thái của ứng dụng được quản lý một cách có hệ thống, dễ dàng theo dõi và bảo trì.

Cộng đồng lớn và hỗ trợ: ReactJS có một cộng đồng lớn và nhiều tài liệu học tập và hỗ trợ. Bạn có thể tìm kiếm giúp đỡ từ các nhà phát triển khác và chia sẻ kiến thức của mình trong cộng đồng này.

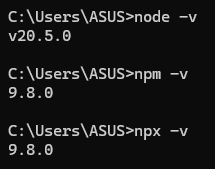
### Bắt đầu cài đặt với ReactJS

#### Cài đặt Node.js và npm [3]

Để có thể cài đặt được ứng dụng ReactJS trên máy tính, máy tính sẽ cần phải được cài đặt [Node.js](https://nodejs.org/en/), npm và npx sẽ được cài đặt cùng với Node.js.

Truy cập trang web chính thức của Node.js: Điều này giúp bạn tải xuống phiên bản mới nhất của Node.js. Trang web chính thức là <https://nodejs.org/>.

Để kiểm tra NodeJS và npx đã được cài đặt ở trên máy tính hay chưa, chúng ta mở Command Prompt và sử dụng câu lệnh node -v, npm -v và npx -v. Máy tính sẽ hiển thị phiên bản được cài đặt. Nếu nhìn thấy có lỗi xảy ra, không hiển thị phiên bản, hãy thử cài đặt lại.



Hình 2.1 Kiểm tra phiên bản

#### Tạo một ứng dụng React mới [1]

Các bước để khởi tạo một ứng dụng React có tên là my-app:

B1: Tạo một thư mục trên máy tính.

B2: Mở cửa sổ dòng lệnh bên trong thư mục đó.

B3: Chạy câu lệnh npx create-react-app my-app . Sau đó đợi để ứng dụng được khởi tạo.

B4: Sau khi đã khởi tạo thành công, di chuyển vào bên trong thư mục “hello-world” vừa được tạo bằng câu lệnh cd my-app.

B5: Khởi chạy ứng dụng bằng câu lệnh npm start.

Trình duyệt sẽ tự động mở ra và ứng dụng đã được khởi tạo thành công. Để tắt ứng dụng đó trên cửa sổ dòng lệnh sử dụng tổ hợp phím Ctrl + C.

Sử dụng lệnh create-react-app để tạo một dự án mới cho React.

npx create-react-app my-app

cd my-app

npm start

#### Sử dụng Tailwind CSS với React [4]

Sau khi đã tạo xong ứng dụng React chúng ta có thể sử dụng Tailwind CSS để làm cho ứng dụng đẹp mắt hơn.

Đầu tiên di chuyển đến thư mục my-app bằng lệnh cd my-app, sau khi đã ở trong thư mục my-app chúng ta cài đặt TailwindCSS thông qua npm, sau đó chạy lệnh init để tạo tệp tailwind.config.js

npm install -D tailwindcss

npx tailwindcss init

Thêm đường dẫn tới tất cả các tệp mẫu trong tệp tailwind.config.js

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

content: [

"./src/\*\*/\*.{js,jsx,ts,tsx}",

],

theme: {

extend: {},

},

plugins: [],

}

Thêm @tailwind chỉ thị cho từng lớp của Tailwind CSS vào tệp ./src/index.css

@tailwind base;

@tailwind components;

@tailwind utilities;

Giờ đây bạn đã có thể code ReactJS với Tailwind CSS và chạy chương trình bằng lệnh npm start.

### Giao diện trong ReactJS

### Javascript XML [5]

Javascript XML (JSX) là cú pháp do đội ngũ React phát triển, sử dụng chủ yếu trong React.

Mục đích của JSX được tạo ra để có thể tạo ra các element một cách tường minh và đơn giản. JSX cho phép developer có thể tạo ra các đoạn HTML nhanh chóng, kèm với khả năng có thể chèn các giá trị JS vào bên trong để tạo ra trang web có nội dung động.

import React from 'react';

function Greeting() {

const name = 'Nghia';

return (

<div>

<h1>Xin chào, {name}!</h1>

</div>

);

}

export default Greeting;

ReactJS có thể hoạt động mà không cần tới cú pháp JSX.

**JSX không phải là HTML**

JSX mặc dù có cú pháp trông rất giống HTML, tuy nhiên nó có một vài đặc điểm khác biệt so với HTML thông thường

**Single parent root**: Các component React cần phải return một thẻ bao ngoài duy nhất hoặc một array, không thể trả về nhiều hơn hai thẻ.

**className** thay vì class. Đây là một lý do kĩ thuật. class là một từ khoá trong JS. Vì vậy, đội ngũ React đã sử dụng className thay vì class để tránh các lỗi.

**style** nhận giá trị là một object, thay vì là cú pháp CSS thông thường

Các thuộc tính HTML sẽ được đổi tên theo kiểu camelCase

Đối với các đoạn JSX nằm trên nhiều dòng, JSX cần phải được bọc bên trong cặp ngoặc tròn ()

Component do chúng ta viết buộc phải được sử dụng ở dưới dạng tên viết hoa.

**Render giá trị JS với JSX**

JSX cho phép chúng ta có thể output các giá trị Javascript trực tiếp vào bên trong. Cú pháp để làm điều này là sử dụng dấu ngoặc nhọn {}. Xét ví dụ sau:

const App = () => {

const name = "Nghia"

const age = 20;

const img = "https://avatar.png";

return (

<div>

Hello, my name is {name}. I'm {age} years old.

<img src={img} />

</div>

);

};

**JSX với styling:**

Có nhiều cách để có thể style với JSX trong React, tương tự với HTML. Cơ bản sẽ có hai kiểu là External CSS và Inline CSS.

Với External CSS, cách viết CSS không có gì khác biệt. Điểm khác biệt ở đây là chúng ta cần dùng className thay cho class thông thường. Chúng ta có thể import file css vào bên trong component với cú pháp import "<file>.css". Xét ví dụ sau:

.App {

text-align: center;

font-weight: bold;

}

import "./App.css";

const App = () => {

return <div className="App">Hello, world!</div>;

};

Đối với Inline CSS, điểm khác biệt tương đối lớn so với HTML thông thường là:

style trong JSX nhận giá trị là một object (key-value)

Các key CSS phải được viết dưới dạng camelCase

Các value CSS cần phải được viết dưới dạng string hoặc number

Xét ví dụ sau:

const App = () => {

return (

<div style={{ background: "yellow", fontSize: 18 }}>Hello, World!</div>

);

};

Đối với Tailwind CSS sử dụng className: Thay vì khai báo kiểu CSS riêng, chỉ cần sử dụng className với các lớp tiện ích của Tailwind trong JSX. Đây là ví dụ:

const App = () => {

return (

<div className="bg-yellow-300 text-center font-bold text-xl p-4">

Hello, Tailwind CSS!

</div>

);

};

**Quy tắc của JSX**

JSX chỉ trả về một phần tử gốc duy nhất. Để trả về nhiều phần tử từ một thành phần, bạn cần bọc chúng trong một thẻ cha duy nhất. Ví dụ, bạn có thể sử dụng một cặp thẻ <div></div> hoặc <></>:

<div>

<h1>Hello Nghia</h1>

<img src="https://avatar.png">

…

</div>

Hoặc

<>

<h1>Hello Nghia</h1>

<img src="https://avatar.png">

…

</>

Thẻ rỗng này được gọi là Fragment. Fragment cho phép bạn nhóm các phần tử mà không để lại bất kỳ dấu vết nào trong cây HTML của trình duyệt.

**Đóng tất cả các thẻ**

JSX yêu cầu bạn phải đóng tất cả các thẻ rõ ràng: các thẻ đơn như là <img /> và các thẻ đôi như là <p>text</p>. Ví dụ:

<>

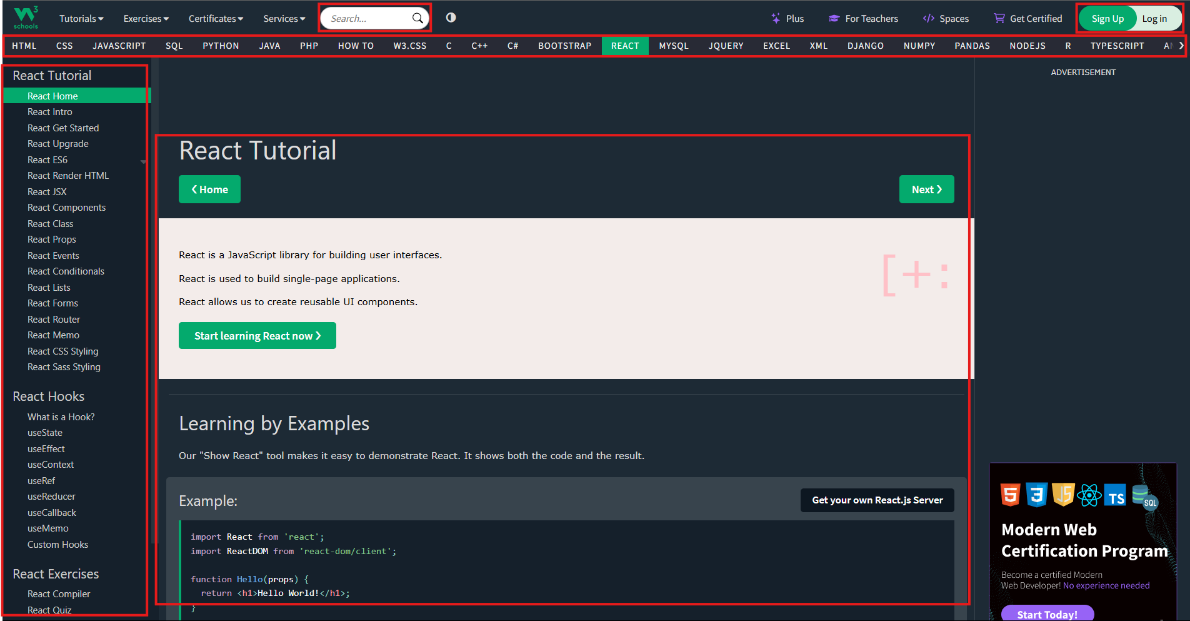
<img src="https://avatar.png" />

<h1>Name</h1>  
 <p>Infomation</p>

</>

#### Component trong ReactJS là gì? [5]

Ví dụ dưới đây, giao diện người dùng đã được chia nhỏ thành các phần riêng lẻ.



Hình 2.2 Ví dụ về component

Phần Search, thanh điều hướng, thanh bên và nội dung,... đều được coi là một thành phần riêng lẻ. Chúng ta có thể thực hiện hợp nhất tất cả các thành phần này để tạo thành một giao diện chính, là giao diện người dùng cuối cùng cho trang chủ.

Khi dự án phát triển, bạn sẽ nhận thấy rằng nhiều phần thiết kế có thể tái sử dụng component bạn đã viết, giúp tăng tốc quá trình phát triển.

**Component trong React là gì?**

Components là những thành phần giao diện (UI) được định nghĩa độc lập, có thể tái sử sụng và hoàn toàn tách biệt nhau.

Chúng ta có thể hiểu component là một hàm trong javascript. Chúng nhận bất kỳ đầu vào nào (hay còn gọi là “props“) và trả về các React elements thể hiện những gì được hiển thị trên trình duyệt. Vì vậy, việc sử dụng và chia nhỏ component hiệu quả sẽ giúp các lập trình viên trở nên chuyên nghiệp và giúp xây dựng một ứng dụng tốt hơn.

**Các bước tạo component trong React**

Dưới đây là các bước giúp tạo tạo một component trong React.

**Bước 1:** Xuất component

Tiền tố export default là một cú pháp JavaScript tiêu chuẩn (không riêng cho React). Nó cho phép bạn đánh dấu hàm chính trong một tệp để sau đó bạn có thể nhập nó từ các tệp khác.

**Bước 2:** Định nghĩa hàm

Với function Profile() { }, bạn định nghĩa một hàm JavaScript với tên Profile.

Lưu ý:Các component React là các hàm JavaScript thông thường, nhưng tên của chúng phải bắt đầu bằng một chữ cái viết hoa hoặc chúng sẽ không hoạt động!

**Bước 3:** Thêm mã Markup

Component trả về một thẻ <img /> với các thuộc tính src và alt. <img /> được viết giống HTML, nhưng bản chất là JavaScript bên trong! Cú pháp này được gọi là JSX, và nó cho phép bạn nhúng mã markup vào bên trong JavaScript.

Câu lệnh return có thể được viết trên cùng một dòng như trong component này:

return <img src="https://avatar.png" alt="my picture" />;

Nhưng nếu mã markup của bạn không nằm trên cùng một dòng như từ khóa return, bạn phải bọc nó trong một cặp dấu ngoặc đơn:

return (

<div>

<h1>Name</h1>

<img src="https://avatar.png" alt="my picture" />;

</div>

);

**Cách sử dụng một component trong React**

Sau khi định nghĩa component Profile của mình, bạn có thể lồng nó vào bên trong các component khác. Ví dụ, bạn có thể xuất một component StudentList sử dụng nhiều component Profile:

function Profile() {

return (

<div>

<h1>Name</h1>

<img src="https://avatar.png" alt="my picture" />;

</div>

);

}

export default function Gallery() {

return (

<section>

<h1>Danh sach hoc sinh Truong Dai hoc Tra Vinh</h1>

<Profile />

<Profile />

<Profile />

</section>

);

}

#### Props trong ReactJS là gì? [5]

Props là tham số đầu vào của các component trong React. Props là một trong những khái niệm cực kỳ quan trọng của React. Nó được sử dụng để truyền thông tin giữa các component. Mọi component cha có thể truyền thông tin đến các component con của nó bằng cách cung cấp chúng props.

const App = () => {

const x = 1;

const y = 2;

return (

<div>

<Sum a={x} b={y} />

</div>

)

}

const Sum = (props) => {

return <div>The value is: {props.a + props.b}</div>

}

**Cách truyền props vào một component trong React**

Bước 1: Truyền props từ component cha vào component con

Đầu tiên, hãy truyền một số props vào Avatar. Ví dụ, hãy truyền hai props: person (một đối tượng) và size (một con số):

export default function Profile() {

return (

<Avatar person={{ name: "Nghia", imageId: "001" }} size={100} />

);

}

Bước 2: Đọc props bên trong component con

Bạn có thể đọc các props này bằng cách liệt kê tên của chúng. Ví dụ: person và size được ngăn cách bằng dấu phẩy trong ({ và }) ngay sau hàm Avatar. Điều này cho phép bạn sử dụng chúng bên trong mã của component Avatar, giống như bạn thao tác với biến.

function Avatar({ person, size }) {

// person và size có sẵn ở đây

}

Thêm một số logic vào Avatar sử dụng các props **url,** **person** và **size** để render. Bây giờ bạn có thể cấu hình Avatar để render theo nhiều cách khác nhau với các props khác nhau. Hãy thử điều chỉnh các giá trị!

function Avatar({ url, person, size }) {

return (

<img className="avatar"

src={url}

alt={person.name}

width={size}

height={size}

/>

);

}

export default function Profile() {

return (

<div>

<Avatar

size={100}

url={https://avatar-tran.png}

person={{

name: "Tran",

imageId: "001",

}}

/>

<Avatar

size={200}

url={https://avatar-trung.png}

person={{

name: "Trung",

imageId: "002",

}}

/>

<Avatar

size={300}

url={https://avatar-Nghia.png}

person={{

name: "Nghia",

imageId: "003",

}}

/>

</div>

);

}

**Giá trị mặc định của props**

Nếu bạn muốn đặt một giá trị mặc định cho props để sử dụng khi không có giá trị được chỉ định, bạn có thể làm điều này bằng cách đặt = và giá trị mặc định ngay sau tham số:

function Avatar({ person, size = 100 }) {

// ...

}

Bây giờ, nếu <Avatar person={...} /> được render mà không có props size, size sẽ được đặt thành 100.

Giá trị mặc định chỉ được sử dụng khi props size bị thiếu hoặc nếu bạn truyền size={undefined}. Tuy nhiên, nếu bạn truyền size={null} hoặc size={0}, giá trị mặc định sẽ không được sử dụng.

**Children props trong React**

Các thẻ HTML có thể chứa bên trong nó các thẻ HTML khác, ta có ví dụ như div, p, ... Tương tự như vậy, các thẻ “HTML” do chúng ta tự tạo cũng có thể làm được điều tương tự thông qua một giá trị props đặc biệt có tên là children. Xét ví dụ sau:

const Card = (props) => {

return <div className="card">{props.children}</div>

}

const App = () => {

return (

<Card>

<div>Inside a card</div>

</Card>

)

}

Cũng tương tự như các props thông thường khác, children có thể nhận giá trị là bất cứ kiểu dữ liệu nào. Với ví dụ ở trên, children nhận vào giá trị là một React Element.

children props giúp chúng ta có khả năng “compose” các component lại với nhau. Thay vì cố định giá trị bên trong Card, lúc này Card có thể cho bất cứ component nào nằm trong nó có thêm các thuộc tính ở trên.

#### Event trong ReactJS là gì? [5]

Trong React, bạn có thể thêm các xử lý sự kiện (event handlers) vào JSX của bạn. Các event handlers là các hàm sẽ được kích hoạt khi có sự tương tác như nhấp chuột, di chuột qua, ,... và nhiều tương tác khác.

**Thêm xử lý sự kiện trong Reactjs**

Bạn có thể làm cho một nút hiển thị một thông báo khi người dùng nhấp chuột bằng cách làm theo ba bước sau:

Định nghĩa một hàm gọi là handleClick trong thành phần Button.

Thực hiện logic trong hàm này (sử dụng alert để hiển thị thông báo).

Thêm onClick={handleClick} vào thẻ <button>.

Dưới đây là mã ví dụ:

export default function Button() {

function handleClick() {

alert("Bạn đã click chuột vào tôi!");

}

return <button onClick={handleClick}>Nhấp chuột vào tôi</button>;

}

Trong ví dụ trên, chúng ta đã định nghĩa hàm handleClick và sau đó truyền nó như một prop cho <button> bằng cách sử dụng onClick={handleClick} - handleClick là một xử lý sự kiện. Thông thường, các hàm xử lý sự kiện được định nghĩa bên trong [component](https://kungfutech.edu.vn/bai-viet/reactjs/component-trong-react-la-gi) của bạn và có tên bắt đầu bằng handle, tiếp theo là tên của sự kiện như onClick, onMouseEnter, onchange, onSubmit,…

#### State trong ReactJS là gì? [5]

State về cơ bản là một giá trị biến đặc biệt trong React. Nó là giá trị mà khi thay đổi, React sẽ tiến hành việc tính toán lại kết quả của component và từ đó cập nhật lại giao diện. Để sử dụng được state, chúng ta cần import một function từ trong thư viện React là useState. useState và một số function khác trong thư viện được gọi là các “hooks”.

Xét ví dụ sau:

import React from "react";

const App = () => {

let count = 0;

const handleClick = () => {

count = count + 1;

console.log("count: ", count)

}

return (

<div>

<span>{count}</span>

<button onClick={handleClick}>Increase</button>

</div>

)

}

Với ví dụ trên, khi ta click vào button, giá trị của biến count sẽ được thay đổi và in giá trị count ra màn hình. Chúng ta cũng sẽ mong đợi rằng component App sẽ thực hiện việc tính toán lại để thay đổi giá trị trong cặp thẻ <span>. Từ đó, giao diện sẽ được cập nhật. Tuy nhiên thì giao diện sẽ **không** được cập nhật!

Thực tế, các biến thông thường như count trong ví dụ trên sẽ không làm cho React thực hiện việc tính toán lại dữ liệu và cập nhật giao diện. React sẽ hoàn toàn bỏ qua sự thay đổi của các biến đó. Khi chúng ta muốn cho React biết rằng nó cần tính toán lại giao diện, chúng ta cần sử dụng một khái niệm đặc biệt từ React: **“State”**

**Sử dụng state với React hooks**

State về cơ bản là một giá trị biến đặc biệt trong React. Nó là giá trị mà khi thay đổi, React sẽ tiến hành việc tính toán lại kết quả của component và từ đó cập nhật lại giao diện. Để sử dụng được state, chúng ta cần import một function từ trong thư viện React là useState. useState và một số function khác trong thư viện được gọi là các “hooks”.

Trong thực tế, người ta thường sử dụng cú pháp destructuring để khai báo biến state và setState. Cú pháp như sau:

const [count, setCount] = useState(10)

Xem ví dụ dưới đây:

import { useState } from 'react';

const App = () => {

const [count, setCount] = useState(10);

const onIncreaseClick = () => {

setCount(count + 1);

};

return (

<div>

<span>{count}</span>

<button onClick={onIncreaseClick}>Increase</button>

</div>

);

};

export default App;

## Node.js

### Giới thiệu [3]

Được phát hành vào năm 2009, **Node.js** (hay NodeJS) là một môi trường runtime **JavaScript đa nền tảng và mã nguồn mở**, cho phép lập trình viên tạo cả ứng dụng **front-end** và **back-end** bằng JavaScript.

Node.js được xây dựng trên **V8 JavaScript Engine**, được viết bằng **C++ và JavaScript**. Được phát triển bởi **Ryan Dahl**, Node.js phù hợp cho các ứng dụng web như **trang video**, **forum** hoặc **mạng xã hội** phạm vi hẹp. Nhờ khả năng chạy trên nhiều hệ điều hành khác nhau (Windows, Linux, macOS,..) và cung cấp nhiều **thư viện JavaScript module**, Node.js đơn giản hóa việc lập trình và giúp giảm thời gian phát triển.

Mục tiêu của Node.js là tạo ra một nền tảng có khả năng xử lý tốt các ứng dụng thời gian thực với khả năng mở rộng cao và hiệu quả trên nhiều hệ điều hành.

### Lợi ích của việc sử dụng Node.js [6]

Hiệu suất cao và không chặn (non-blocking): Node.js sử dụng mô hình xử lý không đồng bộ (asynchronous) và non-blocking I/O, giúp xử lý nhiều yêu cầu cùng lúc mà không bị chặn, tăng hiệu suất cho các ứng dụng thời gian thực.

Sử dụng chung JavaScript ở cả frontend và backend: Giúp các lập trình viên JavaScript dễ dàng làm việc trên cả hai phía, đơn giản hóa việc chuyển đổi dữ liệu và logic giữa frontend và backend.

Thư viện phong phú: Node.js có một hệ sinh thái mạnh mẽ qua NPM (Node Package Manager), cung cấp nhiều thư viện hữu ích, giúp tăng tốc độ phát triển và giảm thiểu thời gian viết mã thủ công.

Dễ mở rộng và linh hoạt: Node.js được thiết kế để dễ dàng mở rộng, phù hợp với kiến trúc microservices, giúp dễ quản lý, bảo trì và phát triển tính năng mới khi cần.

Cộng đồng lớn: Node.js có một cộng đồng lớn và năng động, cung cấp nhiều tài nguyên, hướng dẫn và hỗ trợ nhanh chóng trong quá trình phát triển.

Ứng dụng thời gian thực: Node.js đặc biệt mạnh trong việc phát triển các ứng dụng thời gian thực như chat, video streaming và các hệ thống thông báo nhờ vào khả năng xử lý sự kiện liên tục.

### Bắt đầu cài đặt với Node.js

#### Cài đặt Node.js [3]

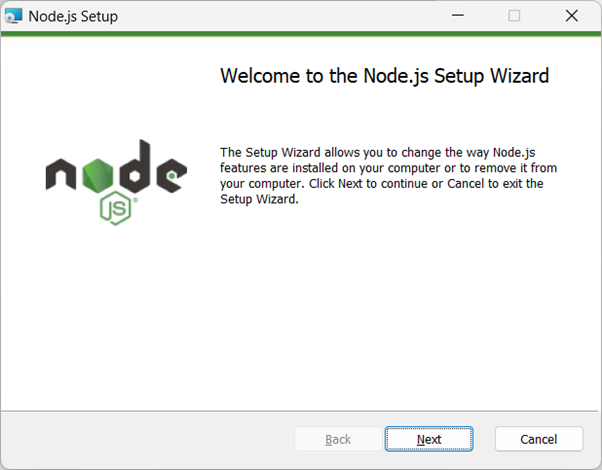
Truy cập trang web chính thức của Node.js: Điều này giúp bạn tải xuống phiên bản mới nhất của Node.js và chọn hệ điều hành phù hợp. Trang web chính thức là <https://nodejs.org/>.



Hình 2.3 Tải Node.js

**Cài đặt trên Windows**

Giả sử bạn đang làm việc với máy tính chạy Windows 10/Windows 11, hãy tải xuống trình cài đặt 64 bit cho Windows và bắt đầu cài đặt bằng cách nhấp đúp vào tệp đã tải xuống.



Hình 2.4 Cài đặt Node.js

Quá trình cài đặt sẽ đưa bạn qua một vài bước của trình hướng dẫn cài đặt. Nó cũng thêm thư mục cài đặt của tệp thực thi Node.js vào đường dẫn hệ thống.

Để kiểm tra NodeJS đã được cài đặt ở trên máy tính hay chưa, chúng ta mở Command Prompt và sử dụng câu lệnh node -v. Máy tính sẽ hiển thị phiên bản được cài đặt.



Hình 2.5 Kiểm tra phiên bản Node.js

#### Tạo cấu trúc thư mục cho Node.js [6]

Tạo thư mục cho dự án:

mkdir my-node-app

cd my-node-app

Khởi tạo dự án Node.js: Chạy lệnh sau để tạo file package.json:

npm init -y

Lệnh này sẽ tạo ra file package.json với các giá trị mặc định.

Tạo các thư mục cần thiết như: src, routes, controllers, models và config public.

**src/app.js**: File chính cho ứng dụng Node.js.

**routes/**: Chứa các định nghĩa route cho ứng dụng.

**controllers/**: Chứa các file xử lý logic cho từng route.

**models/**: Chứa các định nghĩa mô hình dữ liệu (nếu có cơ sở dữ liệu).

**config/**: Chứa các file cấu hình, ví dụ như cấu hình kết nối cơ sở dữ liệu.

**public/**: Chứa các tài nguyên tĩnh (CSS, JavaScript, ảnh).

File package.json chứa thông tin về dự án, các thư viện phụ thuộc và các script. Bạn có thể chỉnh sửa các thông tin trong file package.json theo nhu cầu của dự án.

#### Xây dựng một server đơn giản [6]

Node.js cho phép bạn xây dựng một server HTTP chỉ với một vài dòng mã. Dưới đây là hướng dẫn chi tiết.

Tạo file src/app.js:

// Import module HTTP

const http = require('http');

// Tạo server

const server = http.createServer((req, res) => {

res.statusCode = 200; // Trạng thái thành công

res.writeHead(200, {'Content-Type': 'text/html'});

res.end('Hello, world!\n');

});

// Khởi động server ở cổng 3000

server.listen(3000, () => {

console.log('Server đang chạy tại http://localhost:3000');

});

Giải thích mã nguồn:

**http.createServer()**: Tạo một server mới.

**req**: Đối tượng đại diện cho yêu cầu từ client.

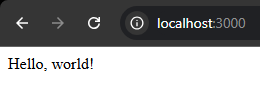
**res**: Đối tượng đại diện cho phản hồi mà server gửi đến client.

**res.end()**: Kết thúc phản hồi và gửi nội dung về client.

Chạy server bằng cách di chuyển đến thư mục my-node-app bằng lệnh

cd my-node-app. Sau đó chạy với lệnh node src/app.js.

Truy cập server: Mở trình duyệt và nhập địa chỉ http://localhost:3000. Bạn sẽ thấy thông điệp "Hello, world!" hiển thị trên giao diện.



Hình 2.6 Chạy server Node.js

#### Sử dụng các thư viện phổ biến [3]

Node.js có một hệ sinh thái phong phú với nhiều thư viện mạnh mẽ giúp tăng tốc quá trình phát triển. Dưới đây là một số thư viện phổ biến và cách sử dụng chúng.

**Cài đặt Express.js**

**Express.js** là một framework web phổ biến cho Node.js, giúp dễ dàng quản lý routing và middleware.

**Cài đặt Express**: Trong thư mục dự án, chạy lệnh:

npm install express

Tạo file src/app.js với Express:

const express = require('express');

const app = express();

const port = 3000;

// Định nghĩa route chính

app.get('/', (req, res) => {

res.send('Hello, world!');

});

// Khởi động server

app.listen(port, () => {

console.log(`Server đang chạy tại http://localhost:${port}`);

});

Chạy server như server node ở trên: node src/app.js và tuy cập vào đường dẫn http://localhost:3000 để xem kết quả.

**Các thư viện phổ biến khác**

Mongoose: Thư viện giúp làm việc với MongoDB.

npm install mongoose

Socket.io: Thư viện hỗ trợ ứng dụng thời gian thực.

npm install socket.io

Cors: Giúp quản lý CORS cho ứng dụng web.

npm install cors

## Kết luận

Khi phát triển trang web thương mai điện tử, việc áp dụng các công nghệ như ReactJS, Node.js, Express và MySQL mang lại nhiều lợi ích nổi bật. Sự kết hợp này không chỉ nâng cao trải nghiệm người dùng mà còn tối ưu hóa hiệu suất và khả năng tương thích trên nhiều nền tảng khác nhau.

**ReactJS** cho phép xây dựng giao diện người dùng có tính linh hoạt và tương tác cao thông qua các thành phần động mà không làm giảm chất lượng trải nghiệm. Cùng với đó, **Node.js** và **Express** có khả năng xử lý đồng thời và phản hồi nhanh chóng, tạo ra một máy chủ mạnh mẽ để hỗ trợ cho ứng dụng thương mại điện tử.

Việc sử dụng **MySQL** giúp quản lý cơ sở dữ liệu một cách hiệu quả và an toàn, bảo vệ thông tin cá nhân của người dùng. Thêm vào đó, việc sử dụng HTTPS trong giao tiếp giữa máy khách và máy chủ thông qua Node.js và Express đảm bảo tính bảo mật cao.

Các công nghệ này còn tăng cường tính mở rộng và khả năng bảo trì, giúp ứng dụng dễ dàng điều chỉnh theo sự phát triển của nhu cầu. Quản lý trạng thái hiệu quả của cả máy chủ và ứng dụng được thực hiện thông qua ReactJS và Node.js, giảm thiểu hiện tượng treo trang và cải thiện khả năng duyệt web.

Cuối cùng, sự kết hợp này giúp xây dựng một trang web thương mại điện tử có hiệu suất cao, khả năng mở rộng linh hoạt và bảo mật đáng tin cậy.

# HIỆN THỰC HÓA NGHIÊN CỨU

## Mô tả bài toán

Để đáp ứng nhu cầu ngày càng tăng về mua sắm trực tuyến, đặc biệt trong lĩnh vực công nghệ, việc xây dựng một website thương mại điện tử chuyên nghiệp là cần thiết. Dự án này tập trung vào việc phát triển một nền tảng trực tuyến, nơi khách hàng có thể dễ dàng tìm kiếm, so sánh và mua sắm các sản phẩm công nghệ như điện thoại di động, laptop và các phụ kiện công nghệ.

Các yêu cầu cụ thể của bài toán bao gồm:

**Chức năng quản lý sản phẩm:**

Quản trị viên có thể thêm, sửa, xóa và cập nhật thông tin sản phẩm, bao gồm hình ảnh, giá cả, mô tả và trạng thái tồn kho.

Hỗ trợ phân loại sản phẩm theo danh mục (ví dụ: điện thoại, laptop, phụ kiện,…).

**Chức năng giỏ hàng và thanh toán:**

Khách hàng có thể thêm sản phẩm vào giỏ hàng, điều chỉnh số lượng hoặc xóa sản phẩm khỏi giỏ hàng.

Cung cấp quy trình thanh toán an toàn và tích hợp với các phương thức thanh toán phổ biến (thẻ tín dụng, ví điện tử).

**Quản lý người dùng:**

Hỗ trợ đăng ký và đăng nhập tài khoản qua email hoặc tích hợp đăng nhập với Google hoặc Facebook.

Người dùng có thể quản lý thông tin cá nhân, lịch sử mua sắm và trạng thái đơn hàng.

**Tìm kiếm và bộ lọc:**

Hỗ trợ chức năng tìm kiếm sản phẩm theo tên hoặc từ khóa.

Bộ lọc theo giá, thương hiệu hoặc các thuộc tính sản phẩm khác để giúp người dùng dễ dàng tìm được sản phẩm phù hợp.

**Giao diện người dùng:**

Cung cấp trải nghiệm trực quan và thân thiện với người dùng trên cả máy tính và thiết bị di động.

Giao diện phản hồi nhanh, giảm thiểu thời gian tải trang.

**Quản lý đơn hàng:**

Quản trị viên có thể xem, xác nhận và cập nhật trạng thái đơn hàng (đang xử lý, đã giao, hủy).

**Khả năng mở rộng:**

Thiết kế hệ thống linh hoạt để dễ dàng thêm tính năng mới hoặc mở rộng quy mô khi lượng người dùng tăng.

**Mục tiêu chính**: Xây dựng một ứng dụng web thương mại điện tử vừa đáp ứng nhu cầu mua sắm của khách hàng, vừa hỗ trợ quản lý hiệu quả cho doanh nghiệp, đồng thời đảm bảo tính ổn định, bảo mật và khả năng mở rộng trong tương lai.

## Kiến trúc hệ thống

**1. Phạm vi kiến trúc hệ thống**

Hệ thống thương mại điện tử được xây dựng dựa trên kiến trúc 3 lớp gồm:

Frontend: Giao diện người dùng được phát triển bằng ReactJS và Tailwind CSS, đảm bảo tính thẩm mỹ và hiệu suất cao.

Backend: Hệ thống xử lý logic và dữ liệu sử dụng Express.js với cấu trúc RESTful API, cung cấp các dịch vụ chính như quản lý người dùng, sản phẩm, giỏ hàng, đơn hàng, thanh toán và tích hợp dịch vụ bên thứ ba.

Cơ sở dữ liệu: Sử dụng MySQL để lưu trữ dữ liệu quan trọng như người dùng, sản phẩm, giỏ hàng, đơn hàng.

Tích hợp dịch vụ bên thứ ba:

Google Login: Cung cấp khả năng đăng nhập nhanh bằng tài khoản Google.

Thanh toán trực tuyến: Tích hợp với các cổng thanh toán như PayPal, VNPay.

API vận chuyển: Tự động tính phí vận chuyển và theo dõi đơn hàng.

**2. Frontend**

Công nghệ: Giao diện người dùng được xây dựng với:

ReactJS: Framework mạnh mẽ giúp phát triển ứng dụng dạng SPA (Single Page Application).

Tailwind CSS: Thư viện CSS tối ưu hóa việc xây dựng giao diện responsive.

Các thành phần chính:

Header và Footer: Điều hướng chung cho người dùng.

Trang Home: Hiển thị danh sách sản phẩm nổi bật.

Danh sách sản phẩm: Liệt kê sản phẩm theo danh mục.

Giỏ hàng: Hiển thị các sản phẩm được chọn.

Trang chi tiết sản phẩm: Thông tin chi tiết về từng sản phẩm.

Thanh toán: Hỗ trợ các bước thanh toán và nhập địa chỉ.

Kiến trúc frontend:

Ứng dụng dạng SPA (Single Page Application).

Tách biệt giao diện thành các component độc lập: Header, Footer, ProductList, Cart, ProductDetails, giúp tái sử dụng và quản lý dễ dàng.

Sử dụng React Router để điều hướng giữa các trang.

**3. Backend**

Công nghệ: Sử dụng Express.js, một framework cho Node.js, để xây dựng backend với cấu trúc RESTful API.

Các module chính:

Quản lý người dùng: Xử lý đăng ký, đăng nhập, và phân quyền (khách hàng, quản trị viên).

Quản lý sản phẩm: CRUD sản phẩm, danh mục, và thương hiệu.

Xử lý giỏ hàng: Thêm, xóa, cập nhật sản phẩm trong giỏ hàng.

Đơn hàng: Lưu trữ thông tin đơn hàng, trạng thái giao hàng.

Thanh toán: Tích hợp các cổng thanh toán trực tuyến.

**4. Cơ sở dữ liệu**

Loại cơ sở dữ liệu: Sử dụng MySQL, hệ quản trị cơ sở dữ liệu quan hệ mạnh mẽ.

Các bảng dữ liệu và mối quan hệ: Dựa trên tệp ecommanagement.sql, hệ thống sử dụng các bảng chính:

Users: Quản lý thông tin người dùng.

Categories: Quản lý danh mục sản phẩm.

Brands: Quản lý thương hiệu.

Products: Quản lý thông tin sản phẩm.

ProductVariants: Biến thể sản phẩm (dung lượng, màu sắc).

Carts: Lưu trữ thông tin giỏ hàng.

Orders: Quản lý đơn hàng.

OrderDetails: Chi tiết sản phẩm trong từng đơn hàng.

Payments: Quản lý giao dịch thanh toán.

Reviews: Đánh giá sản phẩm.

Các mối quan hệ giữa bảng được thiết lập với foreign key, đảm bảo tính toàn vẹn dữ liệu. Ví dụ:

Users ↔ Orders: Một người dùng có thể có nhiều đơn hàng.

Orders ↔ OrderDetails: Mỗi đơn hàng chứa nhiều sản phẩm.

Products ↔ Categories: Sản phẩm thuộc về một danh mục.

Products ↔ Brands: Sản phẩm thuộc về một thương hiệu.

5. Yêu cầu phi chức năng

Khả năng chịu tải: Hệ thống có thể phục vụ đồng thời 100 người dùng mà không làm giảm hiệu suất.

Độ bảo mật: Sử dụng JWT (JSON Web Token) để xác thực và mã hóa dữ liệu.

Khả năng mở rộng: Dễ dàng mở rộng để phục vụ nhiều người dùng và tích hợp thêm dịch vụ.

6. Cách trình bày kiến trúc

Sơ đồ tổng quan hệ thống: Minh họa các thành phần chính và cách chúng giao tiếp.

Kiến trúc phân tầng: Phân chia thành 3 lớp:

Frontend: Giao tiếp với backend qua API.

Backend: Xử lý logic và dữ liệu.

Database: Lưu trữ dữ liệu quan trọng.

Cách phân chia logic:

Frontend gửi yêu cầu qua RESTful API đến backend.

Backend xử lý dữ liệu và giao tiếp với MySQL, sau đó trả kết quả về frontend.

7. Sơ đồ minh họa

Sơ đồ tổng quan hệ thống:

Người dùng tương tác với frontend.

Frontend gửi yêu cầu qua API đến backend.

Backend xử lý dữ liệu, giao tiếp với MySQL và dịch vụ bên thứ ba (Google Login, cổng thanh toán, API vận chuyển).

## Phân tích đặc tả yêu cầu hệ thống

### Thiết kế hệ thống

#### Sơ đồ usecase

#### Sơ đồ lớp

#### Sơ đồ tuần tự

#### Sơ đồ triển khai

## Thiết kế giao diện

## Sơ đồ website

## Màn hình trang chủ

## Màn hình trang quản trị

## Màn hình chức năng

## Màn hình chức năng…

## Kết chương

# KẾT QUẢ NGHIÊN CỨU

## Dữ liệu thử nghiệm

…

## Kết quả thử nghiệm

…

## Kết luận

…

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

…

## Hướng phát triển

…

# DANH MỤC TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | "ReactJS," [Online]. Available: https://react.dev/. [Accessed 2024]. |
| [2] | "KungfuTech," [Online]. Available: https://kungfutech.edu.vn/khoa-hoc/reactjs. [Accessed 2024]. |
| [3] | "Node.js," [Online]. Available: https://nodejs.org/en. [Accessed 2024]. |
| [4] | "Tailwind CSS," [Online]. Available: https://tailwindcss.com/. [Accessed 2024]. |
| [5] | "W3schools - React Tutorial," [Online]. Available: https://www.w3schools.com/react/default.asp. [Accessed 2024]. |
| [6] | "W3schools - Node.js Tutorial," [Online]. Available: https://www.w3schools.com/nodejs/default.asp. [Accessed 2024]. |
| [7] | "W3school - MySQL Tutorial," [Online]. Available: https://www.w3schools.com/mysql/default.asp. [Accessed 2024]. |

# PHỤ LỤC