

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**KHOA KỸ THUẬT VÀ CÔNG NGHỆ**



**ISO 9001:2015**

**HÀ MINH CHIẾN**

**XÂY DỰNG WEBSITE  
BÁN LINH KIỆN ĐIỆN TỬ  
VÀ TỐI ƯU HOÁ  
TRẢI NGHIỆM NGƯỜI DÙNG**

**ĐỒ ÁN TỐT NGHIỆP  
NGÀNH CÔNG NGHỆ THÔNG TIN**

**TRÀ VINH, NĂM 2024**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**KHOA KỸ THUẬT VÀ CÔNG NGHỆ**

**XÂY DỰNG WEBSITE**  
**BÁN LINH KIỆN ĐIỆN TỬ**  
**VÀ TỐI ƯU HOÁ**  
**TRẢI NGHIỆM NGƯỜI DÙNG**

**ĐỒ ÁN TỐT NGHIỆP**  
**NGÀNH CÔNG NGHỆ THÔNG TIN**

Sinh viên: **Hà Minh Chiến**  
Lớp: **DA20TTA**  
MSSV: **110120010**  
GVHD: **Ths. Đoàn Phước Miền**

**TRÀ VINH, NĂM 2024**

## LỜI MỞ ĐẦU

Trong thời đại số hóa hiện nay, mạng máy tính đã trở thành một phần không thể thiếu của cuộc sống hàng ngày. Chúng không chỉ giúp kết nối con người với nhau mà còn là nền tảng cho các hệ thống thông tin, quản lý dữ liệu, giao dịch tài chính, và nhiều lĩnh vực quan trọng khác. Sự phát triển mạnh mẽ của công nghệ thông tin đã tạo điều kiện thuận lợi cho sự ra đời và bùng nổ của các website thương mại điện tử, đặc biệt là trong lĩnh vực bán lẻ linh kiện điện tử.

Website bán linh kiện điện tử không chỉ mang lại tiện ích cho người tiêu dùng bằng cách cung cấp một kênh mua sắm nhanh chóng và tiện lợi, mà còn mở ra cơ hội kinh doanh rộng lớn cho các doanh nghiệp trong ngành. Tuy nhiên, để thành công trong môi trường cạnh tranh khốc liệt này, việc xây dựng một website không chỉ dừng lại ở việc trưng bày sản phẩm. Đó còn là việc tối ưu hóa trải nghiệm người dùng (UX) và đảm bảo an toàn bảo mật thông tin.

Nhằm đáp ứng nhu cầu ngày càng cao của người dùng, website bán linh kiện điện tử cần phải được thiết kế với giao diện thân thiện, dễ sử dụng, và có tốc độ tải trang nhanh. Đồng thời, các biện pháp bảo mật cũng phải được triển khai một cách nghiêm ngặt để bảo vệ thông tin cá nhân và giao dịch của khách hàng.

Trong đề tài này, chúng ta sẽ đi sâu vào việc xây dựng và tối ưu hóa một website bán linh kiện điện tử từ khía cạnh thiết kế giao diện, chức năng, trải nghiệm người dùng, đến các biện pháp bảo mật cần thiết. Qua đó, tôi hy vọng sẽ mang đến những thông tin hữu ích và thiết thực, giúp các nhà phát triển và quản lý website có thể tạo ra một nền tảng thương mại điện tử hiệu quả, an toàn và hấp dẫn, đáp ứng được yêu cầu của thị trường và mang lại sự hài lòng cho khách hàng.

## **LỜI CẢM ƠN**

Trước hết, tôi xin gửi tới toàn thể các thầy, cô giáo đang giảng dạy tại trường, đặc biệt là các thầy cô thuộc Bộ môn Công nghệ Thông tin đã tận tình, giảng dạy, truyền đạt kiến thức cho tôi.

Tôi cảm ơn thầy Đoàn Phước Miên trong thời gian qua đã cung cấp cho tôi những kiến thức liên quan đến đề tài “ Xây dựng website bán linh kiện điện tử và tối ưu hoá trải nghiệm người dùng ”, tận tâm hỗ trợ, tư vấn và hướng dẫn nhiệt tình để tôi có thể hoàn thành đồ án chuyên ngành này một cách tốt nhất và kịp với thời gian quy định.

Vì sự hiểu biết của tôi về đề tài còn hạn chế rất mong hội đồng thông cảm và bỏ qua những thiếu sót trong quá trình làm đồ án và góp ý thêm cho tôi để lần sau tôi có thêm nhiều ý tưởng và hoàn thiện vốn kiến thức cũng như đồ án tiếp theo mà tôi sẽ làm.

Tôi xin chân thành cảm ơn thầy Đoàn Phước Miên !

*Trà Vinh, ngày ..... tháng 6 năm 2024*

Sinh viên thực hiện

**Hà Minh Chiến**

[illegible]

**Giảng viên hướng dẫn**  
(ký và ghi rõ họ tên)

.....

.....

.....

.....

.....

4. Điểm mới đề tài:

.....

.....

.....

.....

.....

5. Giá trị thực trên đề tài:

.....

.....

.....

.....

.....

.....

.....

7. Đề nghị sửa chữa bổ sung:

.....

.....

.....

.....

.....

.....

.....

8. Đánh giá:

.....

.....

.....

.....

Trà Vinh, ngày ..... tháng ..... năm 2024  
Giảng viên hướng dẫn  
(Ký & ghi rõ họ tên)

[illegible]

**Giảng viên chấm**  
(ký và ghi rõ họ tên)



**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**  
(*Của cán bộ chấm đồ án, khóa luận*)

Họ và tên người nhận xét: .....

Chức danh: ..... Học vị: .....

Chuyên ngành: .....

Cơ quan công tác: .....

Tên sinh viên: .....

Tên đề tài đồ án, khóa luận tốt nghiệp: .....

.....

.....

**I. Ý KIẾN NHẬN XÉT**

**1. Nội dung:**

.....

.....

.....

.....

.....

.....

.....

.....

.....

**2. Điểm mới các kết quả của đồ án, khóa luận:**

.....

.....

.....

**3. Ứng dụng thực tế:**

.....

.....

.....

.....

.....

.....

(Các câu hỏi của giáo viên phản biện)

### III. KẾT LUẬN

(Ghi rõ đồng ý hay không đồng ý cho bảo vệ đồ án khóa luận tốt nghiệp)

....., ngày ..... tháng ..... năm 2024

Người nhận xét  
(Ký & ghi rõ họ tên)

## MỤC LỤC

<b>CHƯƠNG 1. ĐẶT VẤN ĐỀ.....</b>	<b>1</b>
1.1. Lý do chọn đề tài.....	1
1.2. Mục tiêu .....	1
1.3. Nội dung.....	1
1.4. Đối tượng và phạm vi nghiên cứu .....	3
1.5. Phương pháp nghiên cứu .....	4
<b>CHƯƠNG 2. CƠ SỞ LÝ THUYẾT.....</b>	<b>5</b>
2.1. Mô tả đề tài .....	5
2.2. ReactJS.....	6
2.2.1. Giới thiệu về ReactJS.....	6
2.3. NodeJS .....	8
2.3.1 Giới thiệu về NodeJS .....	8
2.4 Node package manager – NPM .....	10
2.4.1 Giới thiệu về NPM .....	10
2.5 Javascript.....	11
2.5.1 Giới thiệu về Javascript.....	11
2.6 MongoDB .....	13
2.6.1 Giới thiệu về MongoDB .....	13
2.7 Nội dung nghiên cứu.....	18
2.7.1 ReactJS.....	18
2.7.2 NodeJS .....	21
2.7.3 Javascript .....	21
2.8 Một số thư viện được sử dụng .....	23
2.8.1 Thư viện styled-components.....	23
2.8.2 Thư viện react-router-dom.....	24
2.8.3 Thư viện Antd .....	25
2.8.4 Thư viện jwt-decode .....	26
2.8.5 Thư viện Redux .....	26
2.8.6 Thư viện tanstack/react-query .....	27
2.8.7 Thư viện dotenv.....	28
2.8.8 Thư viện Axios.....	29
<b>CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU .....</b>	<b>31</b>
3.1. Cài đặt NodeJS.....	31
3.2. Tiến hành tạo Project ReactJS .....	34
3.3 Tiến hành xây dựng .....	36

3.4 Thiết kế dữ liệu .....	43
3.4.1 <i>Lược đồ use case tổng quan của hệ thống</i> .....	43
3.4.2 <i>Mô tả các use case</i> .....	44
<b>CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU.....</b>	<b>49</b>
4.1. Giao diện trang chủ.....	49
4.2. Giao diện chi tiết sản phẩm.....	50
4.3 Giao diện danh mục sản phẩm của từng sản phẩm.....	51
4.3.1 Giao diện danh mục bàn phím .....	51
4.3.2 Giao diện danh mục CPU .....	51
4.3.3 Giao diện danh mục Case .....	52
4.3.4 Giao diện danh mục Chuột .....	52
4.3.5 Giao diện danh mục GPU .....	53
4.3.6 Giao diện danh mục Laptop.....	53
4.3.7 Giao diện danh mục Main.....	54
4.3.8 Giao diện danh mục Màn hình.....	54
4.3.9 Giao diện danh mục PSU.....	55
4.3.10 Giao diện danh mục Ram.....	55
4.3.11 Giao diện danh mục SSD.....	56
4.3.12 Giao diện danh mục Thiết bị mạng.....	56
4.4 Giao diện trang thông tin người dùng.....	57
4.5 Giao diện trang đơn hàng.....	57
4.6 Giao diện trang chi tiết đơn hàng.....	58
4.7 Giao diện trang giỏ hàng.....	59
4.8 Giao diện trang thanh toán.....	59
4.9 Giao diện trang đặt hàng thành công .....	60
4.10 Giao diện trang đăng nhập .....	60
4.11 Giao diện trang đăng ký.....	61
4.12 Giao diện trang quản lý người dùng .....	61
4.13 Giao diện trang quản lý sản phẩm .....	62
<b>CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....</b>	<b>63</b>
5.1. Kết luận.....	63
5.2. Hướng phát triển .....	63
<b>DANH MỤC TÀI LIỆU THAM KHẢO.....</b>	<b>64</b>

## DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH

Hình 2.1 Mô hình dữ liệu trong MongoDB <sup>[4]</sup> .....	16
Hình 3.1 Chọn phiên bản NodeJS.....	31
Hình 3.2 Cài đặt NodeJS.....	31
Hình 3.3 Cài đặt NodeJS.....	32
Hình 3.4 Cài đặt NodeJS.....	32
Hình 3.5 Cài đặt NodeJS.....	33
Hình 3.6 Cài đặt NodeJS.....	33
Hình 3.7 Cài đặt NodeJS.....	34
Hình 3.8 Tạo ReactJS Project .....	35
Hình 3.9 Trở vào Folder.....	35
Hình 3.10 Chạy lệnh Npm Start.....	36
Hình 3.11 Giao diện ReactJS .....	36
Hình 3.12 Chạy lệnh Npm Install .....	37
Hình 3.13 Giao diện ReactJS .....	37
Hình 3.14 Các thành phần ở Backend .....	38
Hình 3.15 Các gói thư viện Frontend .....	38
Hình 3.16 Các gói thư viện Backend .....	39
Hình 3.17 Các thành phần Frontend .....	39
Hình 3.18 Các thành phần ở Backend .....	40
Hình 3.19 Mô hình use case tổng quan của hệ thống .....	43
Hình 3.20 Use case xem sản phẩm .....	44
Hình 3.21 Use case tìm kiếm sản phẩm.....	44
Hình 3.22 Use case đặt hàng.....	45
Hình 3.23 Use case xem đơn hàng.....	45
Hình 3.24 Use case đăng ký.....	45
Hình 3.25 Use case đăng nhập.....	46
Hình 3.26 Use case quản lý giỏ hàng.....	46
Hình 3.27 Use case đăng xuất.....	47
Hình 3.28 Use case Quản lý sản phẩm .....	47
Hình 3.29 Use case Quản lý người dùng .....	48
Hình 3.30 Use case Quản lý đơn hàng.....	48
Hình 4.1 Giao diện trang chủ.....	49
Hình 4.2 Giao diện chi tiết sản phẩm.....	50
Hình 4.3 Giao diện danh mục bàn phím .....	51
Hình 4.4 Giao diện danh mục CPU .....	51
Hình 4.5 Giao diện danh mục Case .....	52
Hình 4.6 Giao diện danh mục Chuột .....	52
Hình 4.7 Giao diện danh mục GPU .....	53
Hình 4.8 Giao diện danh mục Laptop.....	53
Hình 4.9 Giao diện danh mục Main.....	54
Hình 4.10 Giao diện danh mục Màn hình.....	54
Hình 4.11 Giao diện danh mục PSU.....	55
Hình 4.12 Giao diện danh mục Ram.....	55
Hình 4.13 Giao diện danh mục SSD.....	56
Hình 4.14 Giao diện danh mục Thiết bị mạng.....	56
Hình 4.15 Giao diện Thông tin người dùng.....	57

Hình 4.16 Giao diện Đơn hàng .....	57
Hình 4.17 Giao diện Chi tiết đơn hàng .....	58
Hình 4.18 Giao diện Giỏ hàng .....	59
Hình 4.19 Giao diện trang Thanh toán .....	59
Hình 4.20 Giao diện trang Đặt hàng thành công .....	60
Hình 4.21 Giao diện trang Đăng nhập .....	60
Hình 4.22 Giao diện trang Đăng ký .....	61
Hình 4.23 Giao diện trang Quản lý người dùng .....	61
Hình 4.24 Giao diện trang Quản lý sản phẩm.....	62
Hình 4.25 Giao diện trang Quản lý đơn hàng.....	62

# DANH MỤC TỪ VIẾT TẮT

(Sắp xếp danh mục từ viết tắt theo thứ tự alphabet của từ viết tắt)

Từ viết tắt	Ý nghĩa
API	Application programming interface
COD	Cash On Delivery
CSS	Cascading Style Sheets
HTML	Hyper Text Markup Language
JS	JavaScript
NPM	Node Package Manager
UI	User Interface
UX	User Experience

## **CHƯƠNG 1. ĐẶT VẤN ĐỀ**

### **1.1. Lý do chọn đề tài**

Trong thời kỳ thương mại điện tử phát triển mạnh mẽ, việc bán linh kiện điện tử trực tuyến không chỉ là một chiến lược kinh doanh cần thiết mà còn là bước tiến quan trọng giúp doanh nghiệp khai thác được nhiều cơ hội. Việc xây dựng một trang web chuyên về bán linh kiện điện tử mang lại nhiều lợi ích to lớn. Với sự gia tăng của xu hướng thương mại điện tử, người tiêu dùng ngày càng có chiều hướng tìm kiếm và mua sắm sản phẩm trực tuyến. Điều này tạo ra cơ hội lớn cho doanh nghiệp để tận dụng và kết nối một cách thuận lợi với khách hàng.

Vì vậy, xây dựng một website bán linh kiện điện tử là cách hiệu quả để đáp ứng nhu cầu của người tiêu dùng hiện đại. Trước tình hình đó, tôi đã lựa chọn thực hiện đề tài

" Xây dựng website bán linh kiện điện tử và tối ưu hoá trải nghiệm người dùng " nhằm học hỏi, nâng cao kiến thức, và tìm hiểu về các website để có thể tạo ra những trang web hoàn thiện, đáp ứng xu hướng mua sắm trực tuyến ngày càng phổ biến.

### **1.2. Mục tiêu**

- Tìm hiểu và nghiên cứu React, NodeJS và MongoDB.
- Xây dựng một nền tảng thương mại điện tử thân thiện, dễ sử dụng cho người dùng.
- Tối ưu hóa trải nghiệm người dùng.

### **1.3. Nội dung**

#### **1. Tìm hiểu công nghệ:**

- Tìm hiểu và nắm vững thư viện React để xây dựng giao diện người dùng (Frontend).
- Tìm hiểu và sử dụng NodeJS để xây dựng máy chủ và xử lý logic (Backend).
- Tìm hiểu và sử dụng MongoDB để lưu trữ cơ sở dữ liệu.

#### **2. Thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX):**

- Xác định giao diện người dùng (UI) và trải nghiệm người dùng (UX) phù hợp với mục đích của trang web.
- Tạo ra giao diện thân thiện, dễ sử dụng.



- Sắp xếp các linh kiện vào các danh mục và phân loại rõ ràng để người dùng dễ dàng tìm kiếm.

### 3. Chức năng cơ bản:

- Đăng nhập và đăng ký tài khoản cho người dùng.
- Hiện thị danh sách sản phẩm linh kiện, kèm theo thông tin chi tiết và hình ảnh.
- Cho phép người dùng thêm sản phẩm vào giỏ hàng và thực hiện thanh toán.
- Cung cấp ô tìm kiếm để người dùng nhập từ khóa tìm kiếm.

### 4. Giỏ hàng và thanh toán

- Cho phép người dùng thêm và xóa sản phẩm từ giỏ hàng.
- Hiện thị tổng giá trị của giỏ hàng và chi phí vận chuyển (nếu có).
- Cung cấp nhiều phương thức thanh toán như thanh toán trực tuyến qua cổng thanh toán an toàn, thanh toán khi nhận hàng (COD), và các phương thức thanh toán điện tử khác.

### 5. Tối ưu hóa trải nghiệm người dùng:

- Tối ưu hoá trải nghiệm người dùng trên cơ sở hỗ trợ email.

### 6. Quản trị:

- Quản lý sản phẩm: người quản trị có thể thêm, xóa, sửa sản phẩm.
- Quản lý người dùng: người quản trị có thể thêm, xóa, sửa thông tin người dùng.
- Thống kê: thống kê các đơn hàng.

## 1.4. Đối tượng và phạm vi nghiên cứu

### Đối tượng nghiên cứu:

Các công nghệ liên quan đến phát triển ứng dụng web, cụ thể là React, NodeJS và MongoDB.

Người dùng của website bao gồm khách hàng mua sắm linh kiện điện tử và quản trị viên quản lý sản phẩm và người dùng

### Phạm vi nghiên cứu:

#### 1. Về công nghệ:

- Frontend: Tìm hiểu và áp dụng React để phát triển giao diện người dùng. Điều này bao gồm các khái niệm như component-based architecture, state management, routing, và tích hợp API.
- Backend: Nghiên cứu NodeJS để xây dựng server, xử lý yêu cầu từ client, và quản lý logic của ứng dụng. Điều này cũng bao gồm việc sử dụng ExpressJS để tạo các API và kết nối với cơ sở dữ liệu.
- Database: Sử dụng MongoDB để lưu trữ và quản lý dữ liệu. Nghiên cứu cách thiết kế mô hình dữ liệu phù hợp với yêu cầu của hệ thống và tích hợp MongoDB với NodeJS.

#### 2. Về tính năng và giao diện:

- Thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX) nhằm tạo ra một trang web thân thiện và dễ sử dụng.
- Phát triển các tính năng cơ bản cho người dùng như đăng ký, đăng nhập, xem danh sách sản phẩm, tìm kiếm sản phẩm, thêm vào giỏ hàng, và thanh toán.
- Cung cấp các tính năng quản trị như quản lý sản phẩm, quản lý người dùng, và thống kê đơn hàng.

#### 3. Về người dùng:

- Nghiên cứu nhu cầu và hành vi của người dùng (khách hàng mua linh kiện điện tử) để thiết kế giao diện và chức năng phù hợp.

#### 4. Phạm vi kỹ thuật

- Xây dựng và triển khai ứng dụng web trên môi trường phát triển Visual Studio Code.
- Đảm bảo tính bảo mật và hiệu suất của hệ thống trong quá trình phát triển và triển khai.
- Tích hợp các phương thức thanh toán an toàn và đáng tin cậy.

### 1.5. Phương pháp nghiên cứu

*Phương pháp lý thuyết:* Đọc hiểu, tìm hiểu các tài liệu về React và NodeJS để thiết kế phần Frontend và Backend cho website.

*Phương pháp thực nghiệm:* Xây dựng website bán lẻ điện tử trên môi trường Visual Studio Code sử dụng thư viện React để thiết kế giao diện website và cơ sở dữ liệu là MongoDB để lưu trữ thông tin.

## CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

### 2.1. Mô tả đề tài

Đề tài này nhằm xây dựng một website thương mại điện tử chuyên bán các linh kiện điện tử, tập trung vào việc cung cấp một nền tảng trực tuyến tiện lợi và tối ưu cho người dùng. Mục tiêu chính là phát triển một hệ thống website thân thiện, dễ sử dụng và cung cấp đầy đủ các chức năng cần thiết cho việc mua sắm và quản lý sản phẩm. Các công nghệ được sử dụng để xây dựng hệ thống này bao gồm React cho phần giao diện người dùng (Frontend), NodeJS cho phần máy chủ và xử lý logic (Backend), và MongoDB để lưu trữ và quản lý cơ sở dữ liệu.

Website sẽ bao gồm các chức năng cơ bản như đăng ký và đăng nhập người dùng, hiển thị danh sách sản phẩm với thông tin chi tiết và hình ảnh, cho phép người dùng thêm sản phẩm vào giỏ hàng và thực hiện thanh toán với nhiều phương thức khác nhau. Ngoài ra, website cũng sẽ cung cấp tính năng tìm kiếm nâng cao và bộ lọc để người dùng dễ dàng tìm kiếm và lựa chọn sản phẩm theo các tiêu chí như giá, thương hiệu, loại sản phẩm, và các tính năng kỹ thuật.

Để tối ưu hóa trải nghiệm người dùng, website sẽ có giao diện thân thiện, dễ sử dụng, hỗ trợ qua các kênh chat trực tuyến và email, và cung cấp nhiều phương thức thanh toán an toàn và tiện lợi. Hệ thống cũng bao gồm các chức năng quản trị như quản lý sản phẩm, quản lý người dùng và thống kê đơn hàng để hỗ trợ người quản trị trong việc điều hành và giám sát hoạt động của website.

## 2.2. ReactJS

### 2.2.1. Giới thiệu về ReactJS

ReactJS là một opensource được phát triển bởi Facebook, ra mắt vào năm 2013, bản thân nó là một thư viện Javascript được dùng để xây dựng các tương tác với các thành phần trên website. Một trong những điểm nổi bật nhất của ReactJS đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client nữa.

ReactJS là một thư viện JavaScript chuyên giúp các nhà phát triển xây dựng giao diện người dùng hay UI. Trong lập trình ứng dụng front-end, lập trình viên thường sẽ phải làm việc chính trên 2 thành phần sau: UI và xử lý tương tác của người dùng. UI là tập hợp những thành phần mà bạn nhìn thấy được trên bất kỳ một ứng dụng nào, ví dụ có thể kể đến bao gồm: menu, thanh tìm kiếm, những nút nhấn, card... Giả sử bạn đang lập trình một website thương mại điện tử, sau khi người dùng chọn được sản phẩm ưng ý rồi và nhấn vào nút “Thêm vào giỏ hàng”, thì việc tiếp theo mà bạn phải làm đó là thêm sản phẩm được chọn vào giỏ hàng và hiển thị lại sản phẩm đó khi user vào xem => xử lý tương tác.

Trước khi có ReactJS, lập trình viên thường gặp rất nhiều khó khăn trong việc sử dụng “vanilla JavaScript” (JavaScript thuần) và JQuery để xây dựng UI. Điều đó đồng nghĩa với việc quá trình phát triển ứng dụng sẽ lâu hơn và xuất hiện nhiều bug, rủi ro hơn. Vì vậy vào năm 2011, Jordan Walke – một nhân viên của Facebook đã khởi tạo ReactJS với mục đích chính là cải thiện quá trình phát triển UI.

#### **Ưu điểm:**

Ngoài việc hỗ trợ xây dựng giao diện nhanh, hạn chế lỗi trong quá trình code, cải thiện performance website thì những tính năng đặc biệt dưới đây có thể là lý do khiến bạn “chốt sale” với ReactJS và bắt đầu tìm hiểu nó từ bây giờ:

Phù hợp với đa dạng thể loại website: ReactJS khiến cho việc khởi tạo website dễ dàng hơn bởi vì bạn không cần phải code nhiều như khi tạo trang web thuần chỉ dùng JavaScript, HTML và nó đã cung cấp cho bạn đủ loại “đồ chơi” để bạn có thể dùng cho nhiều trường hợp.

Tái sử dụng các Component: Nếu bạn xây dựng các Component đủ tốt, đủ flexible để có thể thỏa các “yêu cầu” của nhiều dự án khác nhau, bạn chỉ tốn thời gian xây dựng ban đầu và sử dụng lại hầu như toàn bộ ở các dự án sau. Không chỉ riêng mỗi ReactJS mà

các framework hiện nay cũng đều cho phép chúng ta thực hiện điều đó, ví dụ Flutter chẳng hạn.

Có thể sử dụng cho cả Mobile application: Hầu hết chúng ta đều biết rằng ReactJS được sử dụng cho việc lập trình website, nhưng thực chất nó được sinh ra không chỉ làm mỗi điều đó. Nếu bạn cần phát triển thêm ứng dụng Mobile, thì hãy sử dụng thêm React Native – một framework khác được phát triển cũng chính Facebook, bạn có thể dễ dàng “chia sẻ” các Component hoặc sử dụng lại các Business Logic trong ứng dụng.

Thân thiện với SEO: SEO là một phần không thể thiếu để đưa thông tin website của bạn lên top đầu tìm kiếm của Google. Bản chất ReactJS là một thư viện JavaScript, Google Search Engine hiện nay đã crawl và index được code JavaScript, tuy nhiên bạn cũng cần thêm một vài thư viện khác để hỗ trợ điều này nhé!

Debug dễ dàng: Facebook đã phát hành 1 Chrome extension dùng trong việc debug trong quá trình phát triển ứng dụng. Điều đó giúp tăng tốc quá trình release sản phẩm cung như quá trình coding của bạn.

Công cụ phát triển web hot nhất hiện nay: Nếu bạn nhìn vào số liệu thống kê từ Google Trend ở Việt Nam ở hình bên dưới, dạo lướt qua các trang tuyển dụng hàng đầu ở Việt Nam như Topdev, Itviec, v.v bạn sẽ thấy số lượng tuyển dụng cho vị trí React Developer là cực kỳ lớn cùng với mức lương vô cùng hấp dẫn và độ phổ biến hiện tại của ReactJS trên thị trường Việt Nam là như thế nào.

### **Nhược điểm:**

Khả năng học khá cao ban đầu: Đối với những người mới bắt đầu, việc học React có thể đôi khi khá phức tạp. Nó yêu cầu hiểu biết về JavaScript cơ bản và một số khái niệm về lập trình hướng thành phần.

Những quyết định kiến trúc cần được đưa ra: React không cung cấp một cách tiếp cận kiến trúc cụ thể, điều này có thể làm cho việc quản lý trạng thái, routing và xử lý dữ liệu trở nên phức tạp nếu không được quản lý một cách cẩn thận.

Tích hợp API và thư viện bên thứ ba: Mặc dù ReactJS cung cấp một hệ sinh thái mạnh mẽ với các thư viện và công cụ hỗ trợ, nhưng việc tích hợp với các thư viện hoặc API bên ngoài có thể gặp phải một số thách thức.

Hiệu suất có thể bị ảnh hưởng nếu không được quản lý tốt: Mặc dù React có cơ chế cập nhật hiệu suất cao như Virtual DOM, nhưng việc quản lý trạng thái, sử dụng không hiệu quả của lifecycle methods và render optimization có thể dẫn đến hiệu suất kém.

## 2.3. NodeJS

### 2.3.1 Giới thiệu về NodeJS

NodeJS được phát hành vào năm 2009, NodeJS, hay còn được biết với tên gọi chính thức là Node.js, là môi trường thời gian chạy (runtime environment) JavaScript đa nền tảng và mã nguồn mở. NodeJS cho phép các lập trình viên tạo cả ứng dụng front-end và back-end bằng JavaScript.

NodeJS là mã nguồn mở: Điều này có nghĩa là mã nguồn của NodeJS được cung cấp công khai. Và được duy trì bởi những người đóng góp từ khắp nơi trên thế giới.

NodeJS hỗ trợ đa nền tảng: NodeJS không phụ thuộc vào bất kỳ phần mềm hệ điều hành nào mà đều có thể hoạt động trên Linux, macOS hoặc Windows.

NodeJS là môi trường thời gian chạy mã JavaScript: Khi bạn viết code JavaScript trong trình soạn thảo văn bản, code đó không thể thực hiện bất kỳ tác vụ nào trừ khi bạn thực thi (hoặc chạy) nó. Và để chạy code, bạn cần có môi trường thời gian chạy.

Các trình duyệt như Chrome và Firefox có môi trường thời gian chạy. Đó là lý do tại sao họ có thể chạy code JavaScript. Trước khi NodeJS được tạo, JavaScript chỉ có thể chạy trên trình duyệt và chỉ được sử dụng để xây dựng các ứng dụng front-end.

NodeJS cung cấp môi trường thời gian chạy bên ngoài trình duyệt. Nó cũng được xây dựng trên công cụ JavaScript của Chrome (V8 Engine). Điều này giúp bạn có thể xây dựng các ứng dụng back-end bằng cách sử dụng cùng ngôn ngữ lập trình JavaScript mà bạn quen thuộc.

### Các thuật ngữ liên quan đến NodeJS

*I/O (input/output)*: Là viết tắt của input/output, thuật ngữ I/O chủ yếu đề cập đến sự tương tác của chương trình với hệ thống.

Ví dụ: Các hoạt động I/O có thể bao gồm việc đọc/ ghi dữ liệu từ/ vào disk, tạo các yêu cầu HTTP và trao đổi với cơ sở dữ liệu. Hoạt động này rất chậm so với việc truy cập bộ nhớ (RAM) hoặc thực hiện công việc trực tiếp trên CPU.

Không đồng bộ (Asynchronous): Thực thi không đồng bộ đề cập đến cách thực thi không theo trình tự xuất hiện trong code. Trong lập trình không đồng bộ, chương trình sẽ không đợi tác vụ hoàn thành mà đã có thể chuyển sang tác vụ tiếp theo.

Không chặn (Non-blocking): Chặn (blocking) đề cập đến hành động chặn việc thực thi tiếp theo cho đến khi tác vụ đó kết thúc trong khi không chặn (non-blocking) đề cập đến hành động không chặn việc thực thi. Kết hợp với thuật ngữ “không đồng bộ” ở trên, bạn có thể hiểu rằng các phương thức non-blocking diễn ra một cách không đồng bộ.

*Sự kiện (Event) và Lập trình Hướng sự kiện (Event-driven programming):* Sự kiện là các hành động do người dùng hoặc hệ thống tạo ra, như nhấp chuột, tải xuống tệp hoàn tất hoặc lỗi phần cứng hoặc phần mềm. Lập trình hướng sự kiện là một mô hình lập trình trong đó luồng chương trình được xác định bởi các sự kiện. Một chương trình hướng sự kiện sẽ thực hiện các hành động để đáp lại các sự kiện. Một sự kiện xảy ra sẽ kích hoạt hàm callback.

### **Ưu điểm:**

*NodeJS có thể mở rộng:* Các ứng dụng NodeJS có khả năng mở rộng cao vì chúng hoạt động không đồng bộ vì các yêu cầu đồng thời có thể được xử lý rất hiệu quả bằng NodeJS.

NodeJS hoạt động trên một luồng đơn nên khi có một yêu cầu đến, NodeJS sẽ bắt đầu xử lý yêu cầu đó và sẵn sàng xử lý yêu cầu tiếp theo.

Tính năng hấp dẫn nhất của NodeJS là khả năng phân vùng các ứng dụng theo chiều ngang và quy trình phân vùng này chủ yếu đạt được nhờ sử dụng các tiến trình con. Bằng cách sử dụng tính năng này, các phiên bản ứng dụng riêng biệt được cung cấp cho các đối tượng mục tiêu khác nhau và cho phép chúng đáp ứng sở thích của khách hàng.

*Thời gian thực thi code nhanh:* Công cụ thời gian chạy (runtime motor) JavaScript V8 được NodeJS sử dụng và cũng được Google Chrome sử dụng. Một trình bao bọc được trung tâm cung cấp cho JavaScript và vì lý do đó, công cụ thời gian chạy trở nên nhanh hơn.

Công cụ JavaScript V8 của Google Chrome là nền tảng của NodeJS, cho phép thực thi mã nhanh hơn. Công cụ này biên dịch code JavaScript thành code máy, giúp code được triển khai hiệu quả và dễ dàng hơn và nhanh hơn.



Đồng thời, việc sử dụng các khái niệm như lập trình không đồng bộ và cách vận hành non-blocking trên các hoạt động I/O cũng giúp nâng cao hiệu suất của NodeJS.

*Khả năng tương thích trên nhiều nền tảng:* Các loại hệ điều hành khác nhau như Windows, UNIX, LINUX, MacOS và các thiết bị di động khác đều có thể sử dụng NodeJS.

*Sử dụng JavaScript:* NodeJS sử dụng JavaScript. Hầu hết các lập trình viên đều quen thuộc với JavaScript, vì vậy đối với họ, việc hiểu NodeJS trở nên rất dễ dàng hơn.

*Truyền dữ liệu nhanh:* Thời gian xử lý những dữ liệu đã được truyền đến các luồng khác nhau thường sẽ mất nhiều thời gian. Trong khi đó, để xử lý dữ liệu, NodeJS chỉ mất một khoảng thời gian rất ngắn và thực hiện với tốc độ nhanh.

NodeJS tiết kiệm rất nhiều thời gian vì các tệp được NodeJS xử lý và tải lên đồng thời. Do đó, tốc độ tổng thể của truyền dữ liệu và video được cải thiện nhờ NodeJS.

*Không có bộ đệm:* Dữ liệu không bao giờ được lưu vào bộ đệm trong ứng dụng NodeJS.

*Tiết kiệm thời gian, công sức và chi phí:* NodeJS được trang bị một kho thư viện khổng lồ – NPM (Node Package Manager). Các developer có thể sử dụng lại các module trong code và đưa các chức năng đa dạng vào bất kỳ ứng dụng nào.

Kho lưu trữ nguồn mở này giúp giảm đáng kể chi phí và nỗ lực phát triển, đồng thời cũng rút ngắn thời gian triển khai và thúc đẩy các giải pháp đổi mới.

Một trong những lợi ích chính của NodeJS là khả năng viết toàn bộ cơ sở hạ tầng của bất kỳ ứng dụng web nào chỉ bằng một ngôn ngữ – JavaScript. Do đó, lập trình viên không cần tốn nhiều thời gian học các ngôn ngữ khác nhau để đáp ứng.

*Hỗ trợ cộng đồng mạnh mẽ:* NodeJS có sự hỗ trợ cộng đồng và hỗ trợ mạnh mẽ vì là mã nguồn mở. Vì vậy, các nhà phát triển có thể tìm kiếm sự trợ giúp từ các chuyên gia từ khắp nơi trên thế giới. Điều này thúc đẩy các dự án phát triển.

## **2.4 Node package manager – NPM**

### ***2.4.1 Giới thiệu về NPM***

NPM là viết tắt của Node package manager là một công cụ tạo và quản lý các thư viện lập trình Javascript cho Node.js. Trong cộng đồng Javascript, các lập trình viên chia sẻ hàng trăm nghìn các thư viện với các đoạn code đã thực hiện sẵn một chức năng nào đó.

Nó giúp cho các dự án mới tránh phải viết lại các thành phần cơ bản, các thư viện lập trình hay thậm chí cả các framework.

## **Công dụng của NPM**

Với NPM, công việc sẽ đơn giản đi rất nhiều, chúng giúp bạn thực hiện việc quản lý đơn giản hơn rất nhiều. Các thư viện đều có sẵn trên npm, bạn chạy một dòng lệnh để tải về và dễ dàng include chúng hơn.

Mỗi đoạn code này có thể phụ thuộc vào rất nhiều các mã nguồn mở khác, thật may mắn khi các công cụ quản lý thư viện ra đời, nếu không sẽ mất rất nhiều công sức trong việc quản lý các thư viện này.

Cộng đồng sử dụng npm rất lớn, hàng nghìn các thư viện được phát hành, hỗ trợ Javascript ES6, React, Express, Grunt, Duo... Hiện nay cũng đã xuất hiện thêm Yarn một công cụ tương tự npm, được Facebook phát triển với nhiều tính năng vượt trội có khả năng sẽ thay thế npm.

Nếu như bạn từng code Php thì sẽ biết Composer là công cụ quản lý thư viện của nó, tương tự như NPM là công cụ quản lý thư viện Javascript.

## **2.5 Javascript**

### ***2.5.1 Giới thiệu về Javascript***

JavaScript là ngôn ngữ lập trình website phổ biến hiện nay, nó được tích hợp và nhúng vào HTML giúp website trở nên sống động hơn. JavaScript đóng vai trò như là một phần của trang web, thực thi cho phép Client-side script từ phía người dùng cũng như phía máy chủ (Nodejs) tạo ra các trang web động.

JavaScript là một ngôn ngữ lập trình thông dịch với khả năng hướng đến đối tượng. Là một trong 3 ngôn ngữ chính trong lập trình web và có mối liên hệ lẫn nhau để xây dựng một website sống động, chuyên nghiệp:

HTML: Hỗ trợ trong việc xây dựng layout, thêm nội dung dễ dàng trên website.

CSS: Hỗ trợ việc định dạng thiết kế, bố cục, style, màu sắc,...

JavaScript: Tạo nên những nội dung “động” trên website. Cùng tìm hiểu rõ hơn ở phần dưới đây.

JS là viết tắt của JavaScript, khi có JS bạn sẽ hiểu đó đang nói đến JavaScript.

Nhiệm vụ của Javascript là xử lý những đối tượng HTML trên trình duyệt. Nó có thể can thiệp với các hành động như thêm / xóa / sửa các thuộc tính CSS và các thẻ HTML một cách dễ dàng. Hay nói cách khác, Javascript là một ngôn ngữ lập trình trên trình duyệt ở phía client. Tuy nhiên, hiện nay với sự xuất hiện của NodeJS đã giúp cho Javascript có thể làm việc ở backend. Bạn thử truy cập vào một số website trên internet thì sẽ thấy có những hiệu ứng slide, menu xổ xuống, các hình ảnh chạy qua chạy lại rất đẹp tất cả các chức năng này đều được xử lý bằng Javascript.

Trong những năm gần đây, sự xuất hiện của các framework như NodeJS (chuyên code backend), ExpressJS (NodeJS framework), và nhiều thư viện frontend khác như Angular, jQuery, ReactJS ra đời, giúp tạo ra một cơn sốt với từ khóa Javascript Fullstack.

### **Lịch sử phát triển của Javascript**

Brendan Eich chính là người đã phát triển JS tại Netscape với tiền thân là Mocha. Sau đó, Mocha được đổi thành LiveScript và cuối cùng mới đổi thành JavaScript.

Năm 1998, JavaScript với phiên bản mới nhất là ECMAScript 2 phát hành và đến năm 1999 thì ECMAScript 3 được ra mắt.

Năm 2016, ứng dụng JavaScript đã đạt kỷ lục lên tới 92% website sử dụng, đồng thời cũng được đánh giá là một công cụ cực kỳ quan trọng đối với lập trình viên.

### **Ưu điểm**

Một số ưu điểm nổi bật của ngôn ngữ lập trình JS như sau:

Chương trình rất dễ học.

Những lỗi Javascript rất dễ để phát hiện, từ đó giúp bạn sửa lỗi một cách nhanh chóng hơn.

Những trình duyệt web có thể dịch thông qua HTML mà không cần sử dụng đến một compiler.

JS có thể hoạt động ở trên nhiều nền tảng và các trình duyệt web khác nhau.

Được các chuyên gia đánh giá là một loại ngôn ngữ lập trình nhẹ và nhanh hơn nhiều so với các ngôn ngữ lập trình khác.

JS còn có thể được gắn trên một số các element hoặc những events của các trang web.

Những website có sử dụng JS thì chúng sẽ giúp cho trang web đó có sự tương tác cũng như tăng thêm nhiều trải nghiệm mới cho người dùng.

Người dùng cũng có thể tận dụng JS với mục đích là để kiểm tra những input thay vì cách kiểm tra thủ công thông qua hoạt động truy xuất database.

Giao diện của ứng dụng phong phú với nhiều thành phần như Drag and Drop, Slider để cung cấp đến cho người dùng một Rich Interface (giao diện giàu tính năng).

Giúp thao tác với người dùng phía Client và tách biệt giữa các Client với nhau.

## **Nhược điểm**

Bên cạnh những ưu điểm kể trên thì JS vẫn có những nhược điểm riêng tương tự như các ngôn ngữ lập trình khác hiện nay. Cụ thể:

JS Code Snippet khá lớn.

JS dễ bị các hacker và scammer khai thác hơn.

JS cũng không có khả năng đa luồng hoặc đa dạng xử lý.

Có thể được dùng để thực thi những mã độc ở trên máy tính của người sử dụng.

Những thiết bị khác nhau có thể sẽ thực hiện JS khác nhau, từ đó dẫn đến sự không đồng nhất.

Vì tính bảo mật và an toàn nên các Client-Side Javascript sẽ không cho phép đọc hoặc ghi các file.

JS không được hỗ trợ khi bạn sử dụng ở trong tình trạng thiết bị được kết nối mạng.

## **2.6 MongoDB**

### ***2.6.1 Giới thiệu về MongoDB***

MongoDB là một phần mềm mã nguồn mở dùng để quản trị cơ sở dữ liệu NoSQL<sup>[4]</sup>.

NoSQL (Not only SQL) được sử dụng thay thế cho cơ sở dữ liệu quan hệ (Relational Database – RDB) truyền thống. Cơ sở dữ liệu NoSQL khá hữu ích trong khi làm việc với các tập dữ liệu phân tán lớn. MongoDB là một công cụ có thể quản lý thông tin hướng document cũng như lưu trữ hoặc truy xuất thông tin.

Trong khi đó, ngôn ngữ truy vấn có cấu trúc (SQL) là ngôn ngữ lập trình được tiêu chuẩn hóa, dùng để quản lý cơ sở dữ liệu quan hệ. Dữ liệu được chuẩn hóa SQL dưới dạng schema và table và mọi table đều có cấu trúc cố định.

Hiện nay, có nhiều công ty toàn cầu sử dụng MongoDB để lưu trữ lượng dữ liệu “không lồ” của họ như Facebook, Nokia, eBay, Adobe, Google,...

## **Công dụng của MongoDB**

MongoDB giúp các tổ chức lưu trữ lượng lớn dữ liệu trong khi vẫn hoạt động nhanh chóng. Ngoài lưu trữ dữ liệu, MongoDB còn được sử dụng trong các trường hợp sau:

- Tích hợp một lượng lớn dữ liệu đa dạng
- Mô tả các cấu trúc dữ liệu phức tạp, biến hoá
- Cung cấp dữ liệu cho các ứng dụng hiệu suất cao
- Hỗ trợ các ứng dụng đám mây lai và đa đám mây
- Hỗ trợ phương pháp phát triển Agile

Thay vì sử dụng các table và row như trong cơ sở dữ liệu quan hệ, vì là cơ sở dữ liệu NoSQL, MongoDB được tạo thành từ collection và document. Document được tạo thành từ các cặp khóa-giá trị (là đơn vị dữ liệu cơ bản của MongoDB). Còn collection, tương đương với table trong SQL, là nơi chứa các bộ document.

## **Các thuật ngữ MongoDB thường dùng**

### **\_id**

\_id là một trường bắt buộc trong mọi document của MongoDB. \_id được sử dụng để đại diện cho tính duy nhất của một document trong một collection. Trường \_id hoạt động giống như khóa chính (primary key) của document.

\_id là một số thập lục phân 12 byte đảm bảo tính duy nhất của mọi document. Bạn có thể cung cấp \_id trong khi chèn document. Trong 12 byte này:

- 4 byte đầu tiên đại diện cho thời điểm hiện tại (dựa trên hệ giây của Unix Epoch);
- 3 byte tiếp theo cho id máy;

- 2 byte tiếp theo cho process id của máy chủ MongoDB;
- 3 byte cuối cùng là giá trị gia tăng đơn giản.

Nếu bạn không cung cấp được số id thì MongoDB sẽ tự động cung cấp một id duy nhất cho document của bạn.

## **Document**

Document là đơn vị lưu trữ dữ liệu cơ bản trong cơ sở dữ liệu MongoDB. Document mang vai trò tương tự như row trong các hệ thống cơ sở dữ liệu quan hệ truyền thống.

Document là một cách để sắp xếp và lưu trữ dữ liệu dưới dạng một tập hợp các cặp field-value. Document trong MongoDB không cần phải có cùng một bộ field hoặc cấu trúc với các document khác trong cùng một collection.

Đồng thời, các field chung trong document của một collection có thể chứa các loại dữ liệu khác nhau.

## **Collection**

Collection là một tập hợp các document MongoDB. Collection tương tự như table trong hệ thống cơ sở dữ liệu quan hệ. Các collection có tính chất schema less, do đó các document trong cùng một collection có thể có các trường khác nhau.

Thông thường, một collection chứa các document có mục đích tương tự hoặc liên quan với nhau.

## **Database**

Trong MongoDB, database là một container vật lý chứa tập hợp các collection. Một database có thể chứa 0 collection hoặc nhiều collection.

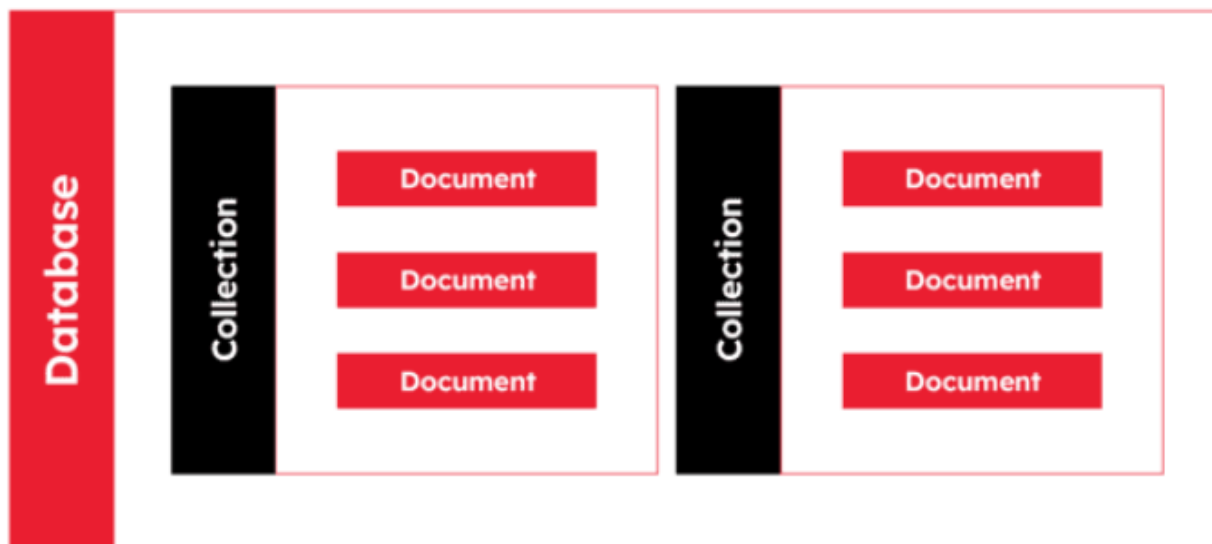
Một phiên bản máy chủ MongoDB có thể lưu trữ nhiều database và không có giới hạn về số lượng database có thể được lưu trữ trên một phiên bản, nhưng giới hạn ở không gian bộ nhớ ảo có thể được phân bổ bởi hệ điều hành.

## **MongoDB lưu trữ dữ liệu**

Như chúng ta biết rằng MongoDB là một máy chủ cơ sở dữ liệu và dữ liệu được lưu trữ trong các cơ sở dữ liệu này. Hay nói cách khác, môi trường MongoDB cung

cấp cho bạn một máy chủ mà bạn có thể khởi động và sau đó tạo nhiều cơ sở dữ liệu trên đó bằng MongoDB.

Nhờ vào cơ sở dữ liệu NoSQL, dữ liệu được lưu trữ dưới dạng collection và document. Do đó, cơ sở dữ liệu, collection và document có mối liên hệ với nhau như hình dưới đây:



Hình 2.1 Mô hình dữ liệu trong MongoDB<sup>[4]</sup>

Trong máy chủ MongoDB, bạn có thể tạo nhiều cơ sở dữ liệu và nhiều collection.

Cách cơ sở dữ liệu MongoDB chứa các collection cũng giống như cách cơ sở dữ liệu MySQL chứa các table.

Bên trong collection, chúng ta có document. Các document này chứa dữ liệu mà bạn muốn lưu trữ trong cơ sở dữ liệu MongoDB và một collection có thể chứa nhiều document. Đồng thời, với tính chất schema-less (không cần một cấu trúc lưu trữ dữ liệu), document này không nhất thiết phải giống với document khác.

Các document được tạo bằng cách sử dụng các field (trường). Các field là các cặp khóa-giá trị trong document, giống như các column trong cơ sở dữ liệu quan hệ. Giá trị của các field có thể là bất kỳ loại dữ liệu BSON nào như double, string, boolean, v.v.

MongoDB lưu trữ dữ liệu ở định dạng BSON document. Ở đây, BSON là đại diện cho định dạng mã hoá nhị phân của các tài liệu JSON (chữ B trong BSON là viết

tất của Binary). Hay nói cách khác, trong phần backend, máy chủ MongoDB chuyển đổi dữ liệu JSON thành dạng nhị phân, được gọi là BSON, và BSON này có thể được lưu trữ và truy vấn hiệu quả hơn.

Kích thước tối đa của BSON document là 16 MB.

Trong MongoDB document, bạn được phép lưu trữ dữ liệu lồng nhau. Việc lồng dữ liệu này cho phép bạn tạo các mối quan hệ phức tạp giữa dữ liệu và lưu trữ chúng trong cùng một document, giúp cho quá trình làm việc và tìm nạp dữ liệu hiệu quả hơn so với SQL.

## **Ưu điểm MongoDB**

MongoDB mang đến cho người sử dụng một số ưu điểm:

- Không schema: Giống như các cơ sở dữ liệu NoSQL khác, MongoDB không yêu cầu các schema được xác định trước.
- MongoDB lưu trữ bất kỳ loại dữ liệu nào: Điều này cho phép người dùng linh hoạt tạo số lượng trường trong document theo nhu cầu, và giúp việc mở rộng cơ sở dữ liệu MongoDB trở nên dễ dàng hơn so với cơ sở dữ liệu quan hệ truyền thống.
- Hướng document: Một trong những ưu điểm của việc sử dụng document là các đối tượng này ánh xạ tới các kiểu dữ liệu gốc trong một số ngôn ngữ lập trình. Việc có các document được nhúng cũng làm giảm nhu cầu kết nối cơ sở dữ liệu, điều này có thể làm giảm chi phí.
- Khả năng mở rộng: Kiến trúc mở rộng theo chiều ngang của MongoDB giúp bạn tạo ra một ứng dụng có thể xử lý được lưu lượng truy cập tăng đột biến khi doanh nghiệp của bạn phát triển. Ngoài ra, việc phân chia dữ liệu (sharding) cho phép cơ sở dữ liệu phân phối dữ liệu trên một cụm máy. MongoDB cũng hỗ trợ tạo vùng dữ liệu dựa trên shard key.
- Hỗ trợ bên thứ ba: MongoDB hỗ trợ một số công cụ lưu trữ và cung cấp API công cụ lưu trữ có thể cắm được (pluggable storage engine API) cho phép các bên thứ ba phát triển công cụ lưu trữ dữ liệu riêng.
- Linh hoạt lưu trữ tệp dung lượng lớn: MongoDB phát triển hệ thống tệp



riêng GridFS, gần giống với hệ thống tệp phân tán Hadoop. Việc sử dụng hệ thống tệp nhằm để lưu trữ các tệp vượt qua kích thước giới hạn của BSON (16 MB cho mỗi document).

## **Khuyết điểm MongoDB**

Mặc dù MongoDB mang lại nhiều giá trị lớn, công cụ này vẫn có một số nhược điểm:

- Tính liên tục: Với chiến lược chuyển đổi dự phòng tự động, người dùng chỉ có thể thiết lập một node master trong cụm MongoDB. Nếu node master bị lỗi, một node khác sẽ tự động chuyển đổi thành master mới. Quá trình chuyển đổi này đảm bảo tính liên tục, nhưng không diễn ra tức thời mà có thể mất tới một phút.
- Giới hạn ghi: Node master duy nhất của MongoDB cũng làm giới hạn lại tốc độ ghi dữ liệu vào cơ sở dữ liệu. Việc ghi dữ liệu phải được ghi trên node master và việc ghi thông tin mới vào cơ sở dữ liệu bị giới hạn bởi khả năng của node master đó.
- Tính nhất quán của dữ liệu: MongoDB không cung cấp tính toàn vẹn tham chiếu đầy đủ thông qua việc sử dụng các ràng buộc khóa ngoại (foreign-key), điều này có thể ảnh hưởng đến tính nhất quán của dữ liệu.
- Bảo mật: Tính năng xác thực người dùng không được mặc định bật trong cơ sở dữ liệu MongoDB. Để bảo mật hệ thống trước các cuộc tấn công của tin tặc, bạn có thể thủ công thiết lập các cài đặt chặn những kết nối lạ và không an toàn.

## **2.7 Nội dung nghiên cứu**

### **2.7.1 ReactJS**

Cần nghiên cứu các thành phần sau:

*Components:*

- Functional Components: Hiểu cách tạo và sử dụng Functional Components, là cách đơn giản nhất để tạo ra một component trong ReactJS.
- Class Components: Nắm vững cách tạo và quản lý Class Components, cho phép bạn sử dụng các tính năng như state và lifecycle methods.

- Lifecycle Methods: Tìm hiểu về các lifecycle methods như `componentDidMount`, `componentDidUpdate`, và `componentWillUnmount` để thực hiện các hành động trong quá trình vòng đời của component.

Ví dụ về tạo 1 lớp Component:

```
class Car extends React.Component {  
  render() {  
    return <h2>Hi, I am a Car!</h2>;  
  }  
}
```

*Props:*

- Introduction to Props: Hiểu về khái niệm Props (properties), là cách truyền dữ liệu từ parent component đến child component trong ReactJS.
- Passing Props: Học cách truyền và nhận props trong functional components và class components.
- Default Props: Biết cách sử dụng default props để thiết lập giá trị mặc định cho props.

```
function Car(props)  
{  
  return <h2>I am a {props.color} Car!</h2>;  
}  
const root =  
ReactDOM.createRoot(document.getElementById('root'));  
root.render(<Car color="red"/>);
```

*State:*

- Introduction to State: Nắm vững khái niệm về State, là nơi lưu trữ và quản lý dữ liệu thay đổi của component.
- Using State in Functional Components: Sử dụng React Hooks như `useState` để quản lý state trong functional components.
- Using State in Class Components: Hiểu cách sử dụng `this.state` và `this.setState()` để quản lý state trong class components.

```
class Car extends React.Component {
  constructor(props) {
    super(props); this.state = {brand: "Ford"};
  } render()
  {
    return ( <div> <h1>My Car</h1> </div> );
  }
}
```

### *State vs. Props:*

- Difference between State and Props: Hiểu rõ sự khác biệt giữa State và Props, bao gồm phạm vi sử dụng và cách quản lý.
- When to Use State vs. Props: Biết cách lựa chọn giữa sử dụng state và props dựa trên yêu cầu của ứng dụng.

### *Reusable Components:*

- Component Reusability: Tìm hiểu cách tạo các component có thể tái sử dụng để giảm lặp lại mã và tăng khả năng bảo trì.
- Higher-Order Components (HOCs): Biết cách sử dụng HOCs để mở rộng chức năng của các component.

### *Component Lifecycle:*

- Understanding the Component Lifecycle: Hiểu rõ về vòng đời của một component và các hành động xảy ra ở mỗi giai đoạn của vòng đời.
- Lifecycle Methods: Nắm vững các lifecycle methods và cách sử dụng chúng để thực hiện các hành động như khởi tạo, cập nhật và hủy bỏ.

### *Handling Events:*

- Event Handling: Học cách xử lý sự kiện trong ReactJS và cách gán các hàm xử lý sự kiện với các elements.

```
function Football() {
  const shoot = (a, b) => {
    alert(b.type);
  } return (
    <button onClick={ (event) => shoot("Goal!", event)}>Take the
    shot!</button> );
  } const root =
  ReactDOM.createRoot(document.getElementById('root'));
  root.render(<Football />);
}
```

### 2.7.2 NodeJS

#### *Nền tảng Node.js*

- Cơ bản về Node.js: Hiểu về cách hoạt động của Node.js, bao gồm cách nó xử lý các yêu cầu HTTP, xử lý không đồng bộ và sử dụng callback.
- Học về npm: npm là trình quản lý gói cho Node.js, bạn cần biết cách sử dụng nó để quản lý các gói phụ thuộc của dự án.

#### *Express.js*

- Express.js: Express là một framework web cho Node.js, giúp bạn xây dựng các ứng dụng web dễ dàng và nhanh chóng.
- Routing và Middleware: Hiểu cách định tuyến yêu cầu HTTP và sử dụng middleware trong Express.js để xử lý các yêu cầu.

#### *Tích hợp với ReactJS*

- Tích hợp với frontend: Xây dựng các endpoint API trong ứng dụng Node.js để cung cấp dữ liệu từ file CSV cho frontend ReactJS.
- Fetch API hoặc Axios: Sử dụng Fetch API hoặc thư viện Axios trong ReactJS để gửi yêu cầu HTTP đến backend Node.js và nhận dữ liệu trả về.

### 2.7.3 Javascript

Nghiên cứu về JavaScript để sử dụng trong ReactJS và Node.js gồm các nội dung sau:

#### *Cú pháp và Cấu trúc cơ bản của JavaScript*

- Tìm hiểu về các cú pháp cơ bản của JavaScript như biến, hàm, điều kiện, vòng lặp, và các cấu trúc dữ liệu như mảng và đối tượng.
- Hiểu về phạm vi và hoạt động của từ khóa this trong JavaScript.

### *ES6+ và Các tính năng mới*

- Nắm vững các tính năng mới của ES6+ như let/const, arrow functions, template literals, destructuring, spread/rest operators, promises, async/await, và modules.
- Áp dụng các tính năng này trong mã JavaScript cho cả ReactJS và Node.js để viết mã sáng sủa và dễ hiểu.

### *Lập trình Hàm trong JavaScript*

- Hiểu về cách làm việc với hàm trong JavaScript, bao gồm cách khai báo hàm, gọi hàm, truyền đối số, và sử dụng hàm callback.
- Học về cách sử dụng hàm bất đồng bộ và xử lý các tác vụ không đồng bộ bằng promises và async/await.

### *Phát triển ứng dụng web với ReactJS*

- Tìm hiểu về các khái niệm cơ bản của ReactJS như components, state, props, lifecycle methods, và hooks.
- Áp dụng các nguyên tắc của ReactJS để phát triển giao diện người dùng linh hoạt và dễ bảo trì.

### *Xử lý Dữ liệu trong ReactJS*

- Hiểu về cách quản lý trạng thái của ứng dụng trong ReactJS bằng cách sử dụng state và props.
- Tìm hiểu cách sử dụng các thư viện quản lý trạng thái như Redux hoặc Context API để quản lý trạng thái của ứng dụng lớn hơn.

### *Phát triển Backend với Node.js:*

- Nắm vững cách sử dụng Node.js để phát triển các ứng dụng web backend.
- Tìm hiểu về Express.js và các khái niệm như routing, middleware, và xử lý yêu cầu và phản hồi.

### *Tích hợp Backend và Frontend*

- Học cách tạo và sử dụng API trong Node.js để cung cấp dữ liệu cho ứng dụng ReactJS.
- Tìm hiểu cách giao tiếp với API từ phía frontend bằng cách sử dụng các phương thức HTTP như GET, POST, PUT, và DELETE.

## 2.8 Một số thư viện được sử dụng

### 2.8.1 Thư viện *styled-components*

#### *Mô tả*

Styled-components là một thư viện mạnh mẽ cho phép bạn viết CSS trong các ứng dụng React dưới dạng component. Điều này giúp bạn gắn chặt kiểu dáng với logic của component, dễ dàng tái sử dụng và bảo trì code. Bằng cách sử dụng tagged template literals trong JavaScript, styled-components cho phép bạn viết các styles trực tiếp trong các file JavaScript/TypeScript của mình, giúp tránh xung đột tên lớp và tăng tính module hóa cho các style của bạn.

#### *Tính năng nổi bật*

- **Scoped Styles:** Mỗi styled component tạo ra một class name độc đáo, tránh xung đột tên lớp giữa các component khác nhau.
- **Dynamic Styling:** Hỗ trợ styling động dựa trên props, giúp dễ dàng thay đổi style dựa trên trạng thái hoặc dữ liệu.
- **Theming:** Hỗ trợ theming với ThemeProvider, cho phép dễ dàng quản lý và thay đổi theme cho toàn bộ ứng dụng.
- **CSS Features:** Hỗ trợ đầy đủ các tính năng của CSS, bao gồm pseudo-classes, pseudo-elements, media queries, và keyframes cho animations.
- **Server-Side Rendering (SSR):** Tương thích với SSR, giúp giảm thời gian tải trang và cải thiện SEO.
- **Developer Experience:** Tích hợp tốt với các công cụ phát triển như VSCode, cung cấp autocompletion và highlight cho CSS.

#### *Sử dụng*

- Bạn có thể cài đặt styled-components thông qua npm bằng cách chạy lệnh: `npm install styled-components`.
- Sau đó, bạn có thể import styled-components vào trong ứng dụng của mình và sử dụng chúng để tạo và kiểu dáng cho các component với lệnh: `import styled from 'styled-components'`.

## 2.8.2 Thư viện react-router-dom

### Mô tả

React-router-dom là một thư viện mạnh mẽ dành cho React, cung cấp các công cụ và thành phần để quản lý điều hướng và routing trong ứng dụng web. Thư viện này giúp bạn dễ dàng xây dựng các ứng dụng đơn trang (SPA) với các route khác nhau, quản lý lịch sử trình duyệt, và điều hướng động. Bằng cách sử dụng react-router-dom, bạn có thể tạo các ứng dụng có cấu trúc rõ ràng, dễ bảo trì và mở rộng.

### Tính năng nổi bật

- **Declarative Routing:** Xây dựng các route bằng cách sử dụng JSX, cho phép bạn định nghĩa các route một cách rõ ràng và dễ hiểu trong component của mình.
- **Nested Routes:** Hỗ trợ nested routing, cho phép bạn tạo các route lồng nhau và quản lý điều hướng phức tạp một cách dễ dàng.
- **Dynamic Routing:** Hỗ trợ dynamic routing, giúp bạn điều hướng dựa trên các điều kiện hoặc dữ liệu động.
- **Route Parameters:** Cho phép bạn sử dụng các tham số trong route, giúp xây dựng các URL động và truyền dữ liệu thông qua URL.
- **Programmatic Navigation:** Cung cấp các công cụ để điều hướng một cách lập trình thông qua JavaScript, giúp bạn điều khiển điều hướng một cách linh hoạt.
- **Redirects:** Hỗ trợ redirect, cho phép bạn chuyển hướng người dùng từ một route này sang một route khác.
- **Route Guards:** Tích hợp các công cụ để bảo vệ route, đảm bảo rằng chỉ những người dùng được xác thực hoặc có quyền truy cập mới có thể truy cập vào các route nhất định.

### Sử dụng

- Bạn có thể cài đặt React-router-dom thông qua npm bằng cách chạy lệnh: `npm install react-router-dom`.
- Sau đó, bạn có thể import React-router-dom vào trong ứng dụng của mình với lệnh: `import { BrowserRouter as Router, Route, Switch } from 'react-router-dom'`.

### 2.8.3 Thư viện Antd

#### Mô tả

Ant Design là một thư viện giao diện người dùng (UI) mạnh mẽ và toàn diện dành cho React, cung cấp một bộ sưu tập phong phú các thành phần UI được thiết kế sẵn. Được phát triển bởi Alibaba, Ant Design không chỉ tập trung vào việc cung cấp các thành phần UI chất lượng cao mà còn hướng đến việc tạo ra trải nghiệm người dùng tốt nhất với thiết kế nhất quán và phong cách chuyên nghiệp. Thư viện này rất phổ biến trong cộng đồng phát triển web nhờ vào tính dễ sử dụng, khả năng tùy chỉnh cao và hỗ trợ mạnh mẽ.

#### Tính năng nổi bật

- **Thành phần phong phú:** Cung cấp hơn 50 thành phần UI, bao gồm các button, form, table, modal, menu, và nhiều thành phần khác.
- **Thiết kế nhất quán:** Tuân thủ các nguyên tắc thiết kế nhất quán, giúp tạo ra các ứng dụng web có giao diện đẹp và đồng nhất.
- **Tùy chỉnh cao:** Cho phép tùy chỉnh giao diện và hành vi của các thành phần thông qua các props và các công cụ tùy chỉnh chủ đề.
- **Hỗ trợ quốc tế hóa (i18n):** Cung cấp các công cụ và thành phần hỗ trợ dịch và đa ngôn ngữ, giúp xây dựng các ứng dụng quốc tế hóa dễ dàng.
- **Documentation đầy đủ:** Cung cấp tài liệu chi tiết và ví dụ thực tế, giúp các nhà phát triển dễ dàng bắt đầu và sử dụng thư viện.
- **Hỗ trợ TypeScript:** Cung cấp các định nghĩa TypeScript đầy đủ, giúp phát triển an toàn và hiệu quả hơn trong các dự án TypeScript.
- **Hệ sinh thái phong phú:** Tích hợp tốt với các công cụ và thư viện khác, cung cấp các giải pháp toàn diện từ thiết kế đến phát triển.

#### Sử dụng

- Bạn có thể cài đặt Ant Design thông qua npm bằng cách chạy lệnh: `npm install antd`.
- Sau đó, bạn có thể import Ant Design vào trong ứng dụng của mình với lệnh: `import { Button, DatePicker } from 'antd', import 'antd/dist/antd.css'.`



### 2.8.4 Thư viện *jwt-decode*

#### *Mô tả*

Jwt-decode là một thư viện nhỏ gọn dành cho JavaScript, cho phép bạn giải mã JSON Web Tokens (JWT) một cách dễ dàng. Thư viện này được thiết kế để hoạt động cả trên trình duyệt lẫn trên Node.js, giúp bạn trích xuất và đọc các thông tin được lưu trữ trong JWT mà không cần phải gửi chúng tới server để xác thực. Điều này rất hữu ích trong các ứng dụng web, nơi mà bạn có thể cần phải truy cập thông tin người dùng hoặc trạng thái phiên từ các token JWT.

#### *Tính năng nổi bật*

- **No Dependencies:** Thư viện không có phụ thuộc bên ngoài, giúp giảm bớt kích thước và độ phức tạp của dự án.
- **Supports All JWTs:** Hỗ trợ giải mã mọi JWT được mã hóa bằng base64url.
- **Lightweight:** Thư viện rất nhẹ, giúp cải thiện hiệu năng của ứng dụng.
- **Browser and Node.js Compatibility:** Hoạt động tốt trên cả trình duyệt và môi trường Node.js.

#### *Sử dụng*

- Bạn có thể cài đặt jwt-decode thông qua npm bằng cách chạy lệnh: `npm install jwt-decode`.
- Sau đó, bạn có thể import jwt-decode vào trong ứng dụng của mình với lệnh: `import jwtDecode from 'jwt-decode';`.

### 2.8.5 Thư viện *Redux*

#### *Mô tả*

Redux là một thư viện quản lý trạng thái ứng dụng JavaScript phổ biến, thường được sử dụng với các thư viện như React để xây dựng các ứng dụng UI phức tạp. Redux giúp bạn quản lý trạng thái của ứng dụng một cách dễ dàng, nhất quán và có thể đoán trước được. Bằng cách lưu trữ trạng thái của toàn bộ ứng dụng trong một đối tượng duy nhất được gọi là "store", Redux cho phép bạn dễ dàng theo dõi, kiểm soát và cập nhật trạng thái ứng dụng.

#### *Tính năng nổi bật*

- **Single Source of Truth:** Tất cả trạng thái của ứng dụng được lưu trữ trong một store duy nhất, giúp dễ dàng theo dõi và quản lý.
- **Predictable State Updates:** Các trạng thái chỉ có thể được cập nhật bằng cách phát hành (dispatch) các hành động (actions), giúp dễ dàng kiểm soát và dự đoán các thay đổi trạng thái.
- **Centralized State Management:** Giúp đơn giản hóa việc quản lý trạng thái trong các ứng dụng lớn với nhiều thành phần phức tạp.
- **Middleware:** Hỗ trợ middleware để xử lý các tác vụ bất đồng bộ và các hoạt động phụ trợ khác.
- **DevTools:** Tích hợp với Redux DevTools giúp theo dõi các thay đổi trạng thái, kiểm tra và gỡ lỗi dễ dàng.
- **Immutability:** Tính bất biến của state giúp ngăn ngừa các thay đổi không mong muốn và đảm bảo tính nhất quán của dữ liệu.
- **Ease of Testing:** Redux store và các reducers dễ dàng kiểm tra vì chúng là các hàm thuần túy.

#### *Sử dụng*

- Bạn có thể cài đặt Redux thông qua npm bằng cách chạy lệnh: `npm install redux react-redux`.

#### **2.8.6 Thư viện *tanstack/react-query***

##### *Mô tả*

Tanstack/react-query là một thư viện quản lý trạng thái dữ liệu phía client cho ứng dụng React, được thiết kế để giải quyết các vấn đề liên quan đến fetching, caching và updating dữ liệu từ server. Thư viện này cung cấp các hooks để sử dụng để tương tác với dữ liệu, giảm bớt sự phức tạp của việc quản lý state trong ứng dụng và cải thiện hiệu suất bằng cách tự động quản lý cache dữ liệu.

##### *Tính năng nổi bật*

- **Hooks-Based:** Sử dụng hooks như `useQuery`, `useMutation`, `useQueryClient` để tương tác với dữ liệu một cách đơn giản và hiệu quả.
- **Fetching Data:** Hỗ trợ việc fetching dữ liệu từ server với các options linh hoạt như caching, stale-while-revalidate, paginated queries.

- **Caching:** Tự động quản lý cache dữ liệu để giảm số lần fetching dữ liệu và cải thiện hiệu suất ứng dụng.
- **Optimistic Updates:** Hỗ trợ cập nhật dữ liệu optimistic để cải thiện trải nghiệm người dùng trong khi đang chờ xử lý từ server.
- **Server-Side Rendering (SSR):** Hỗ trợ tốt cho Server-Side Rendering để đảm bảo dữ liệu được load trước khi render ra client.
- **Typescript Support:** Cung cấp các định nghĩa typescript hoàn chỉnh, giúp việc phát triển ứng dụng an toàn hơn.
- **DevTools:** Có các công cụ hỗ trợ giám sát và debug dễ dàng như DevTools và React Query DevTools.

### *Sử dụng*

- Bạn có thể cài đặt tanstack/react-query thông qua npm bằng cách chạy lệnh: `npm install @tanstack/react-query`.
- Sau đó, bạn có thể import tanstack/react-query vào trong ứng dụng của mình với lệnh: `import { useQuery, useMutation, } from '@tanstack/react-query'`.

### **2.8.7 Thư viện dotenv**

#### *Mô tả*

Dotenv là một thư viện Node.js nhằm giúp quản lý biến môi trường (environment variables) trong ứng dụng. Biến môi trường là các giá trị cấu hình như các khóa API, thông tin kết nối cơ sở dữ liệu, cấu hình mật khẩu, và các giá trị khác mà không nên được lưu trực tiếp trong mã nguồn, nhưng thay vào đó được quản lý bằng cách sử dụng biến môi trường. Điều này giúp bảo mật thông tin nhạy cảm và linh hoạt khi triển khai ứng dụng trên nhiều môi trường khác nhau.

#### *Tính năng nổi bật*

- **Quản lý biến môi trường:** Cho phép định nghĩa và sử dụng các biến môi trường trong ứng dụng.
- **Giữ thông tin nhạy cảm riêng biệt:** Ngăn chặn việc lưu trực tiếp các giá trị nhạy cảm như khóa API hay mật khẩu trong mã nguồn.

- **Dễ dàng triển khai:** Giúp bạn dễ dàng chuyển đổi giữa các môi trường phát triển (development), thử nghiệm (testing), và sản phẩm (production) mà không cần sửa đổi mã nguồn.
- **Cấu hình linh hoạt:** Cho phép đọc các biến môi trường từ một tập tin `.env` hoặc các tập tin cấu hình khác một cách dễ dàng.

### *Sử dụng*

- Bạn có thể cài đặt dotenv thông qua npm bằng cách chạy lệnh: `npm install dotenv`.
- Sau đó, bạn có thể tạo 1 file `.env` trong dự án của bạn để quản lý biến môi trường.

### **2.8.8 Thư viện Axios**

#### *Mô tả*

Axios là một thư viện HTTP client phổ biến trong JavaScript, được sử dụng chủ yếu để thực hiện các yêu cầu HTTP từ phía client (trình duyệt hoặc Node.js) đến server và xử lý các phản hồi từ server. Axios hỗ trợ các tính năng như tạo và gửi yêu cầu HTTP, xử lý các yêu cầu bất đồng bộ, quản lý và hủy các yêu cầu, cũng như biến đổi dữ liệu phản hồi trả về từ server.

#### *Tính năng nổi bật*

- **Đơn giản và dễ sử dụng:** Cung cấp một API dễ hiểu và linh hoạt để thực hiện các yêu cầu HTTP.
- **Hỗ trợ Promise:** Hỗ trợ việc xử lý yêu cầu bất đồng bộ bằng cách sử dụng Promise, giúp quản lý mã nguồn dễ dàng hơn.
- **Chuyển đổi dữ liệu tự động:** Axios tự động chuyển đổi các dữ liệu JSON từ/phải JavaScript object khi gửi hoặc nhận phản hồi từ server.
- **Hủy yêu cầu (Cancel requests):** Cung cấp khả năng hủy yêu cầu HTTP một cách dễ dàng và hiệu quả, giúp tối ưu hóa hiệu suất ứng dụng.
- **Interceptors:** Cho phép bạn xử lý và thay đổi các yêu cầu và phản hồi HTTP trước khi chúng được gửi hoặc nhận.
- **Hỗ trợ trên cả trình duyệt và Node.js:** Axios có thể được sử dụng cả trong môi trường trình duyệt (client-side) và môi trường Node.js (server-side).

- **Được sử dụng rộng rãi và hỗ trợ mạnh mẽ:** Axios là một trong những thư viện HTTP client phổ biến và có sự hỗ trợ rộng rãi từ cộng đồng và nhà phát triển.

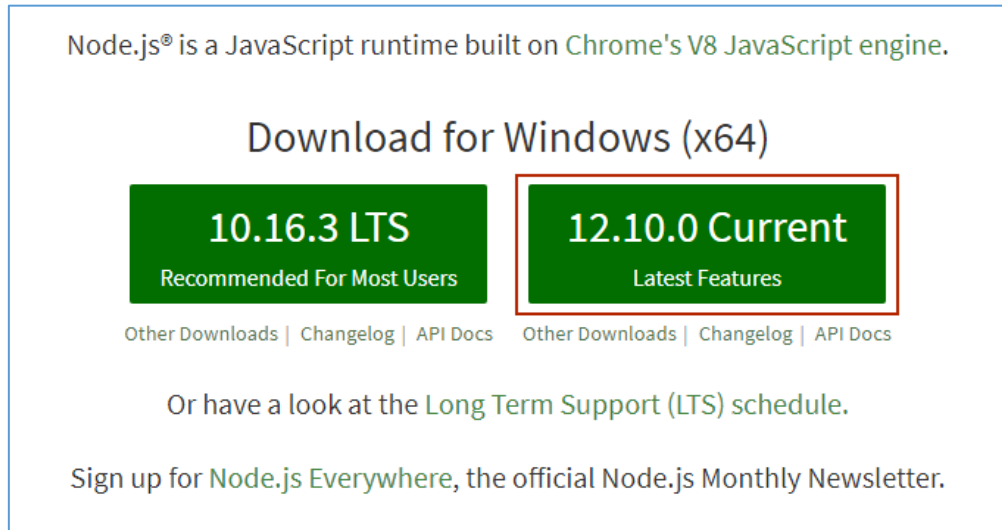
### *Sử dụng*

- Bạn có thể cài đặt axios thông qua npm bằng cách chạy lệnh: `npm install axios`.
- Sau đó, bạn có thể import axios vào trong ứng dụng của mình với lệnh: `import axios from 'axios'`.

## CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU

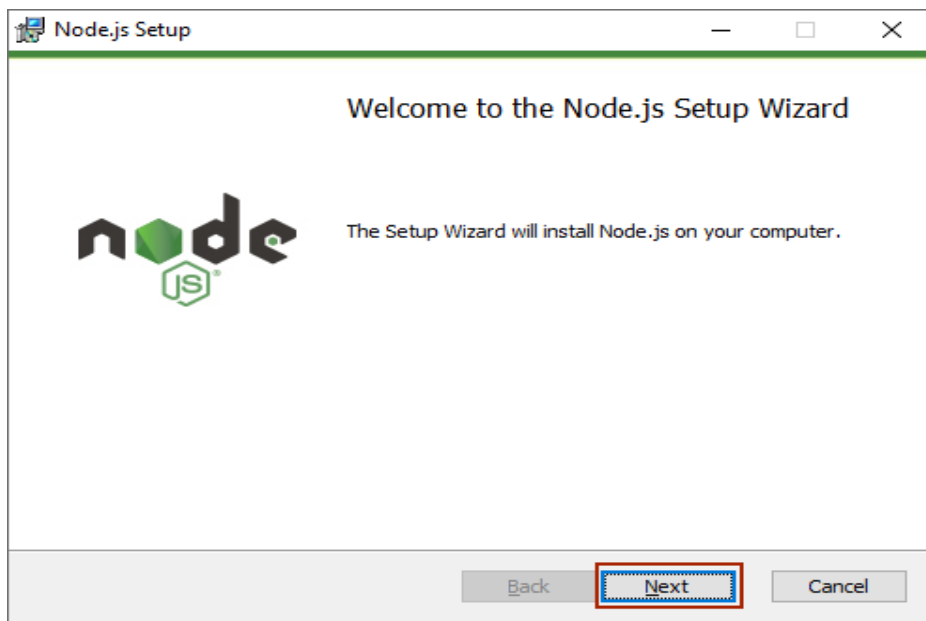
### 3.1. Cài đặt NodeJS

Bước 1: Vào trang <https://nodejs.org/en/download/current> sau đó tải phiên bản phù hợp



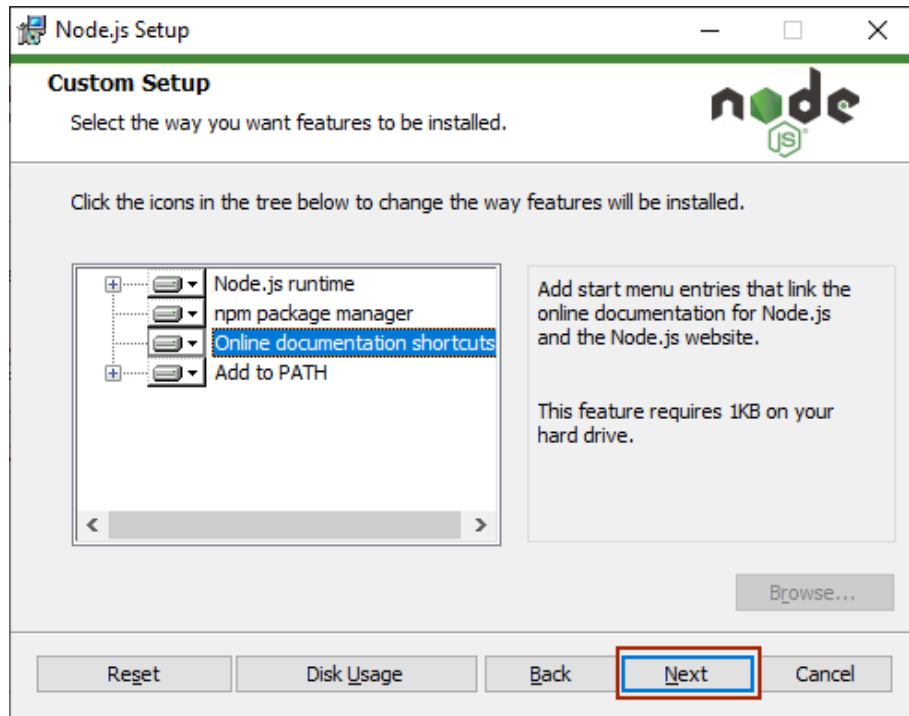
Hình 3.1 Chọn phiên bản NodeJS

Bước 2: Tiến hành cài đặt



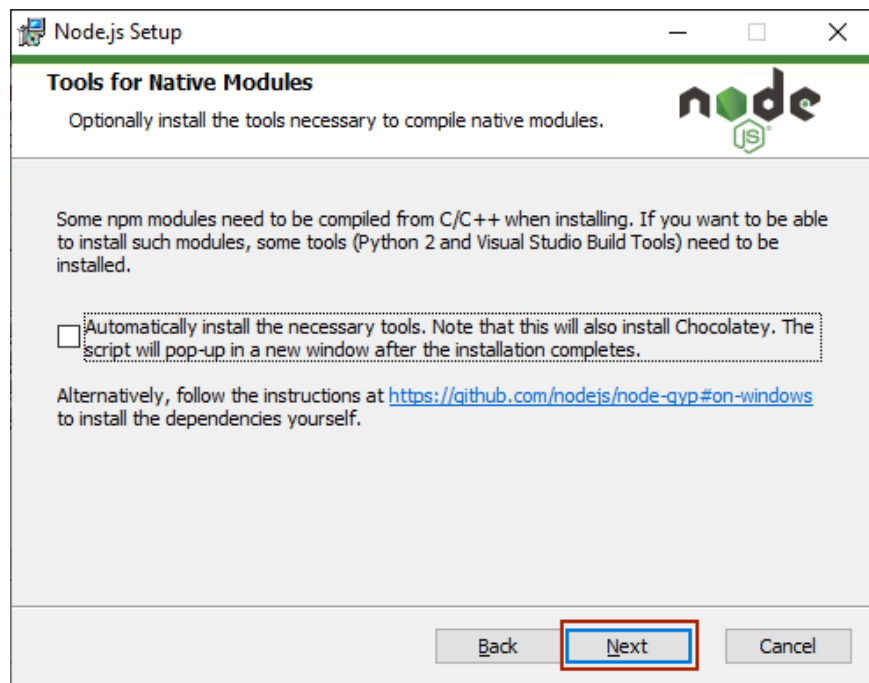
Hình 3.2 Cài đặt NodeJS

Bước 3: Bấm next như hình dưới



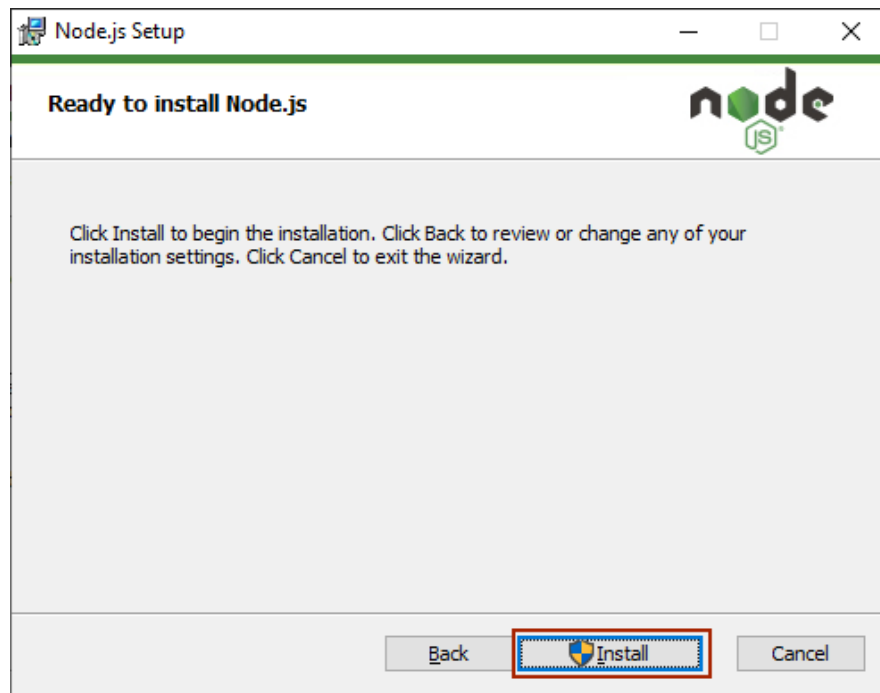
Hình 3.3 Cài đặt NodeJS

Bước 4: Tiếp tục next



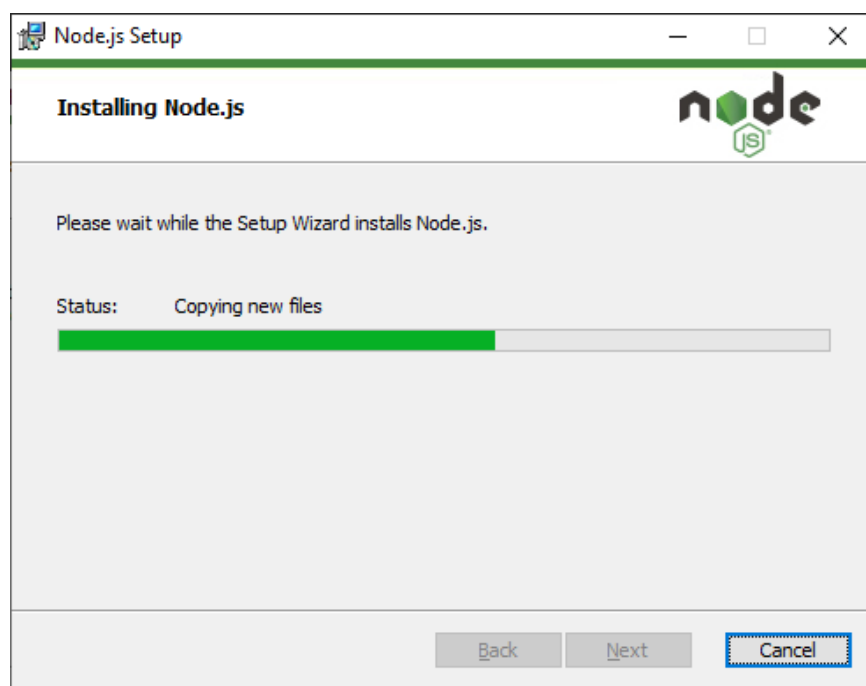
Hình 3.4 Cài đặt NodeJS

## Bước 5: Nhấn install



Hình 3.5 Cài đặt NodeJS

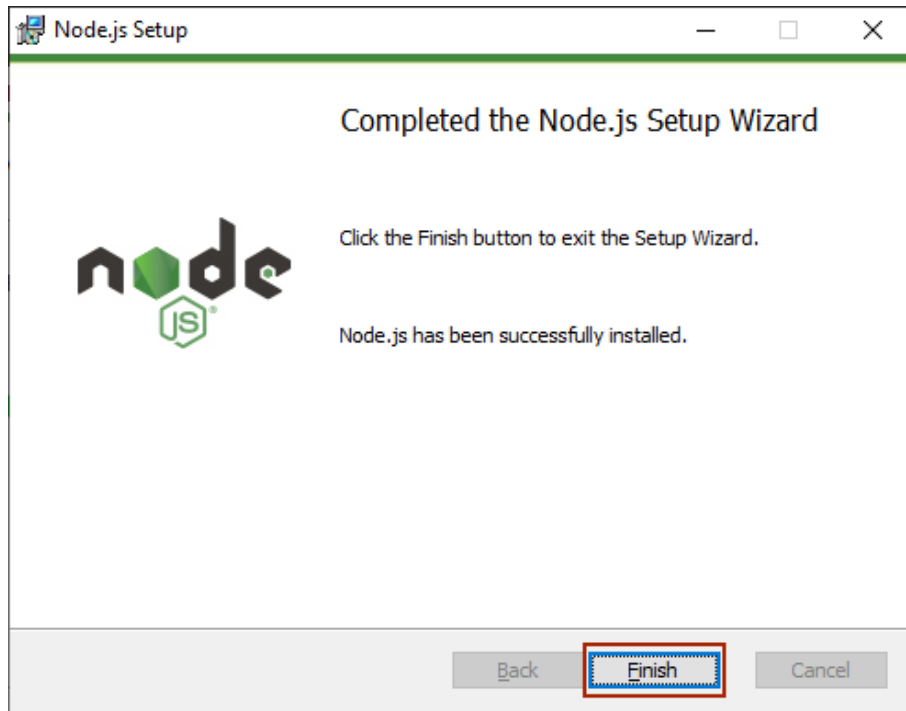
## Bước 6: Đợi cài đặt



Hình 3.6 Cài đặt NodeJS



## Bước 7: Bấm finish



Hình 3.7 Cài đặt NodeJS

Sau khi thực hiện tất cả các bước đã cài đặt thành công NodeJS.

### 3.2. Tiến hành tạo Project ReactJS

Bước 1: Cài đặt môi trường NodeJS như mục 2.6.1 trên.

Bước 2: Cài đặt bộ cài app React.

Có thể cài ở bất kỳ ổ đĩa hay thư mục nào bằng lệnh cmd. Bằng câu lệnh:

**- npm install -g create-react-app**

Bước 3: Tạo project React.js trong ổ đĩa D với tên bất kỳ.

Ta lần lượt tiến hành: trở vào ổ đĩa D và thực thi lệnh tạo project với tên react-project (tên project tùy ý) như bên dưới.

**- create-react-app react-project**

```
C:\WINDOWS\system32\cmd.exe
D:\>create-react-app react-project

Creating a new React app in D:\react-project.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...

> core-js@2.6.9 postinstall D:\react-project\node_modules\babel-runtime\n
ode_modules\core-js
> node scripts/postinstall || echo "ignore"

> core-js@3.2.1 postinstall D:\react-project\node_modules\core-js
> node scripts/postinstall || echo "ignore"

+ react-dom@16.10.1
+ react-scripts@3.1.2
+ react@16.10.1
added 1441 packages from 686 contributors and audited 904866 packages in
103.675s
found 0 vulnerabilities

Initialized a git repository.

Success! Created react-project at D:\react-project
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back
    !

We suggest that you begin by typing:

  cd react-project
  npm start

Happy hacking!
D:\>
```

Hình 3.8 Tạo ReactJS Project

Bước 4: Thực thi lệnh chuyển đổi đến thư mục dự án và khởi động dự án

Ta lần lượt tiến hành: trở vào thư mục react-project với lệnh cmd sau:

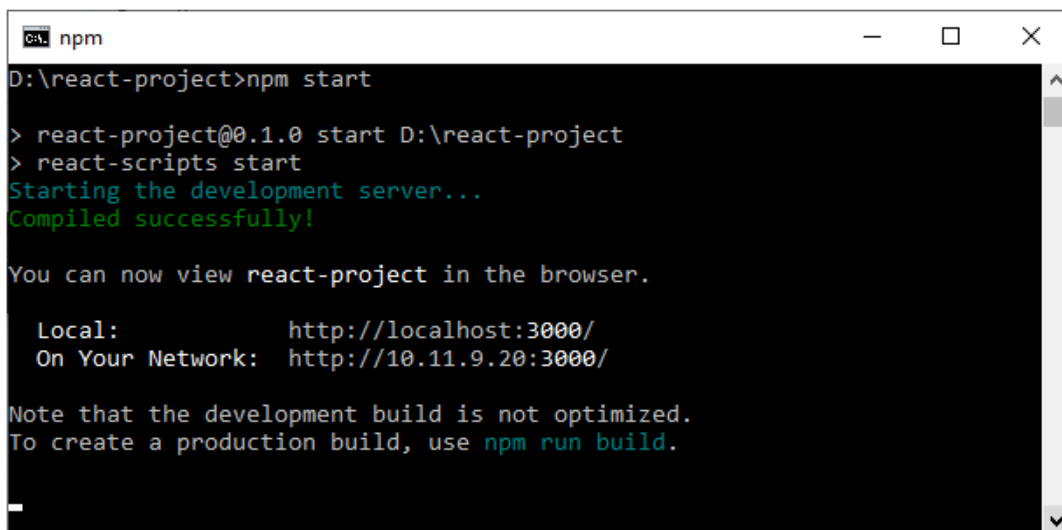
**cd react-project**

```
C:\WINDOWS\system32\cmd.exe
D:\>cd react-project

D:\react-project>
```

Hình 3.9 Trở vào Folder

Thực thi chạy project React bằng lệnh **npm start**



```
npm
D:\react-project>npm start

> react-project@0.1.0 start D:\react-project
> react-scripts start
Starting the development server...
Compiled successfully!

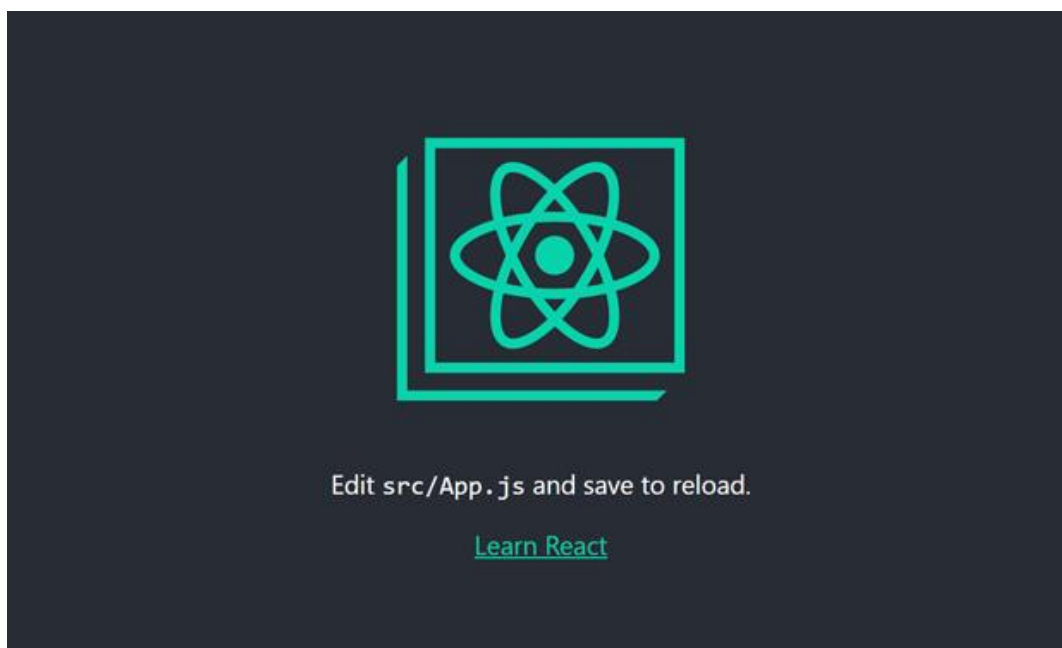
You can now view react-project in the browser.

  Local:            http://localhost:3000/
  On Your Network:  http://10.11.9.20:3000/

Note that the development build is not optimized.
To create a production build, use npm run build.
```

Hình 3.10 Chạy lệnh Npm Start

Sau bước khởi động, ứng dụng React sẽ tự động mở một trang web mặc định thông báo kết quả cài đặt trên trình duyệt với đường dẫn: <http://localhost:3000/>. Như vậy, quá trình cài đặt đã thành công.

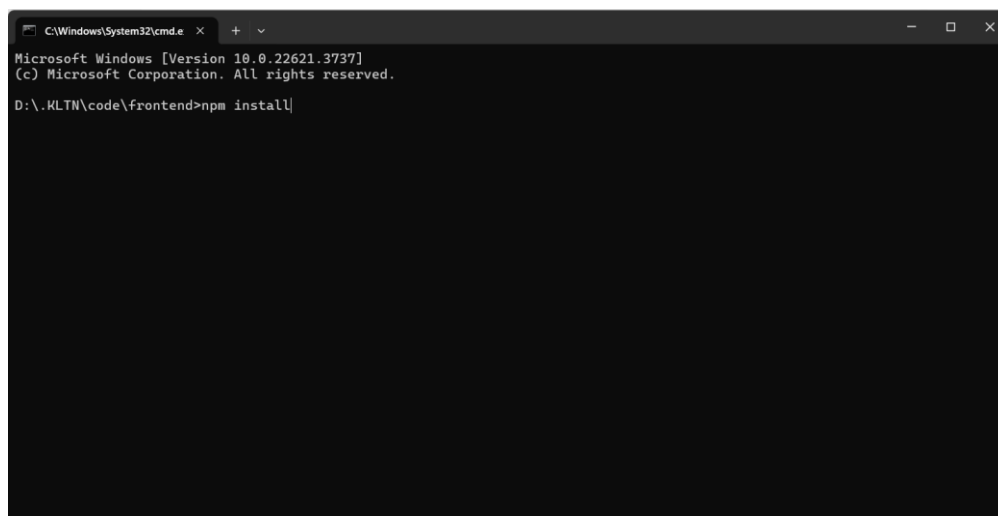


Hình 3.11 Giao diện ReactJS

### 3.3 Tiến hành xây dựng

Ở đây đề tài này chúng ta cần tạo 2 Folder một Folder dùng để chứa Frontend (cách tạo như ở phần 3.2 tạo bằng React) và Folder còn lại sẽ dùng để chứa Backend (tự tạo và cài đặt theo ý muốn)

Bước 1: Sau khi tạo Frontend bằng thư viện React thành công tại Folder chứa Frontend tiến hành chạy lệnh `npm install` để tạo `node_modules`.



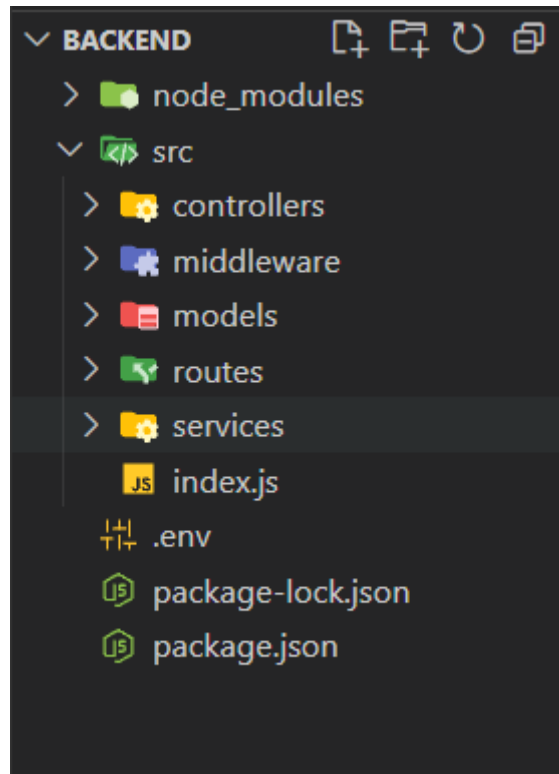
Hình 3.12 Chạy lệnh Npm Install

Bước 2: Chạy câu lệnh `npm start` để chạy Web trên Localhost và sau đó tiến hành thiết kế giao diện.



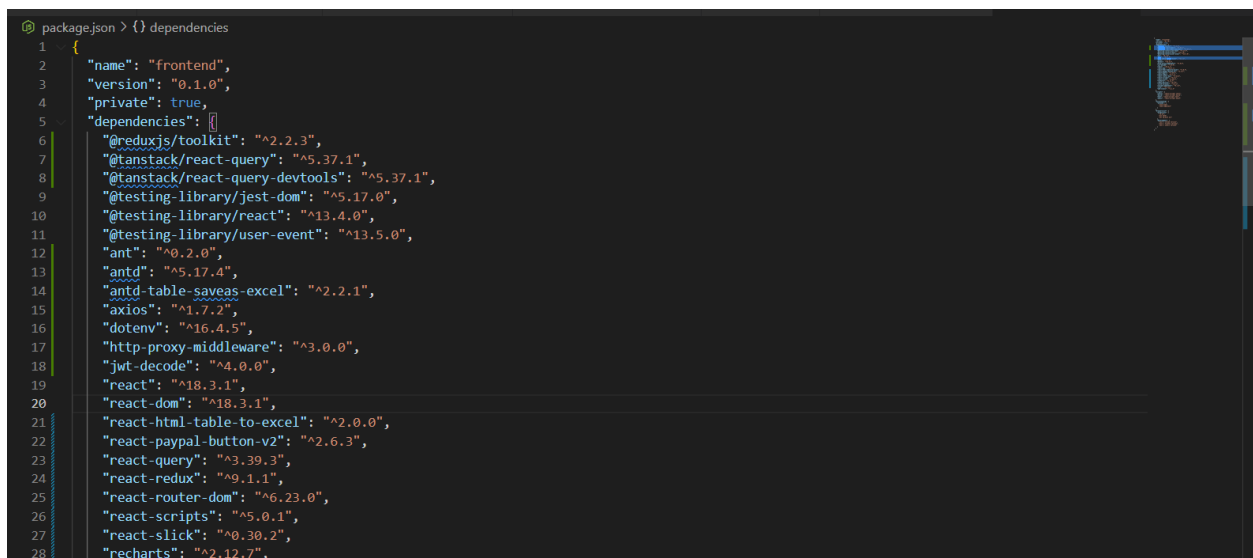
Hình 3.13 Giao diện ReactJS

Về phần Backend thì sẽ được cài đặt như hình bên dưới :



Hình 3.14 Các thành phần ở Backend

Bước 3: Tiến hành cài đặt các thư viện cho phần Frontend như styled-components, Antd, redux, tanstack/react-query, axios, dotenv, jwt-decode, ... để thuận tiện trong việc xây dựng phần giao diện cho đề tài.



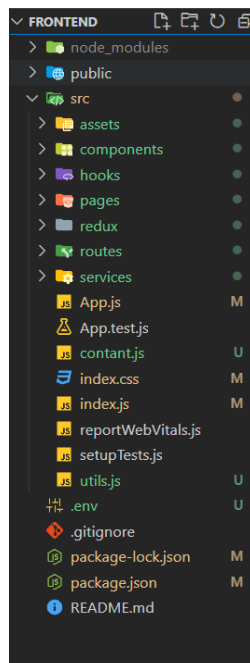
Hình 3.15 Các gói thư viện Frontend

Tiến hành cài đặt các thư viện cho phần Backend như bcrypt , body-parser, cookie-parser , cors, dotenv, express, jsonwebtoken, mongoose, nodemailer, ... để thuận tiện cho việc xây dựng phần Backend để truyền tải các thông tin dữ liệu.

```
package.json > {} dependencies
1  {
2    "name": "backend",
3    "version": "1.0.0",
4    "main": "index.js",
5    > Debug
6    "scripts": {
7      "test": "echo \\\"Error: no test specified\\\" && exit 1",
8      "start": "nodemon src/index.js"
9    },
10   "author": "",
11   "license": "ISC",
12   "description": "",
13   "dependencies": {
14     "bcrypt": "^5.1.1",
15     "body-parser": "^1.20.2",
16     "cookie-parser": "^1.4.6",
17     "cors": "^2.8.5",
18     "dotenv": "^16.4.5",
19     "express": "^4.19.2",
20     "jsonwebtoken": "^9.0.2",
21     "mongoose": "^8.3.4",
22     "nodemailer": "^6.9.13",
23     "nodemon": "^3.1.0",
24     "redux-persist": "^6.0.0"
25   }
26 }
```

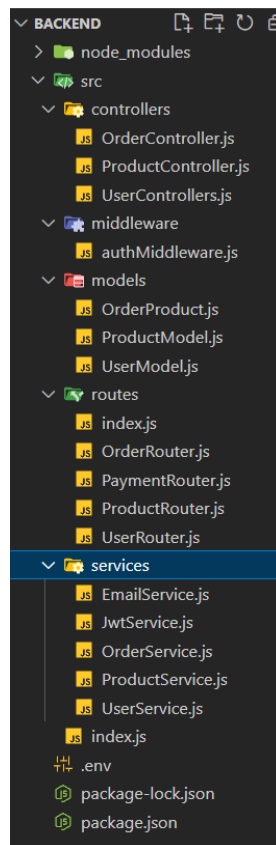
Hình 3.16 Các gói thư viện Backend

Bước 4: Ở Frontend tiến hành tạo các component để có thể tái sử dụng, tạo các trang giao diện và các thành phần cần thiết cho đề tài.



Hình 3.17 Các thành phần Frontend

Về phần Backend tạo các file đọc, truyền dữ liệu cho các trang ở phần Frontend và tạo các mô hình dữ liệu để lưu trữ dữ liệu cho phần Frontend.



Hình 3.18 Các thành phần ở Backend

Mô hình dữ liệu:

OrderProduct:

```
const orderSchema = new mongoose.Schema({
  orderItems: [
    {
      name: { type: String, required: true },
      amount: { type: Number, required: true },
      image: { type: String, required: true },
      price: { type: Number, required: true },
      discount: { type: Number },
      product: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'Product',
        required: true,
      },
    },
  ],
  shippingAddress: {
    fullName: { type: String, required: true },
    address: { type: String, required: true },
    city: { type: String, required: true },
    phone: { type: Number, required: true },
  },
  paymentMethod: { type: String, required: true },
  itemsPrice: { type: Number, required: true },
  shippingPrice: { type: Number, required: true },
  totalPrice: { type: Number, required: true },
  user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  isPaid: { type: Boolean, default: false },
  paidAt: { type: Date },
  isDelivered: { type: Boolean, default: false },
  deliveredAt: { type: Date },
},
{
  timestamps: true,
});
```



### UserModel:

```
const userSchema = new mongoose.Schema(  
  {  
    name: { type: String },  
    email: { type: String, required: true, unique: true },  
    password: { type: String, required: true },  
    isAdmin: { type: Boolean, default: false, required: true },  
    phone: { type: Number },  
    address: { type: String },  
    avatar: { type: String },  
    city: { type: String },  
  },  
  {  
    timestamps: true,  
  }  
);
```

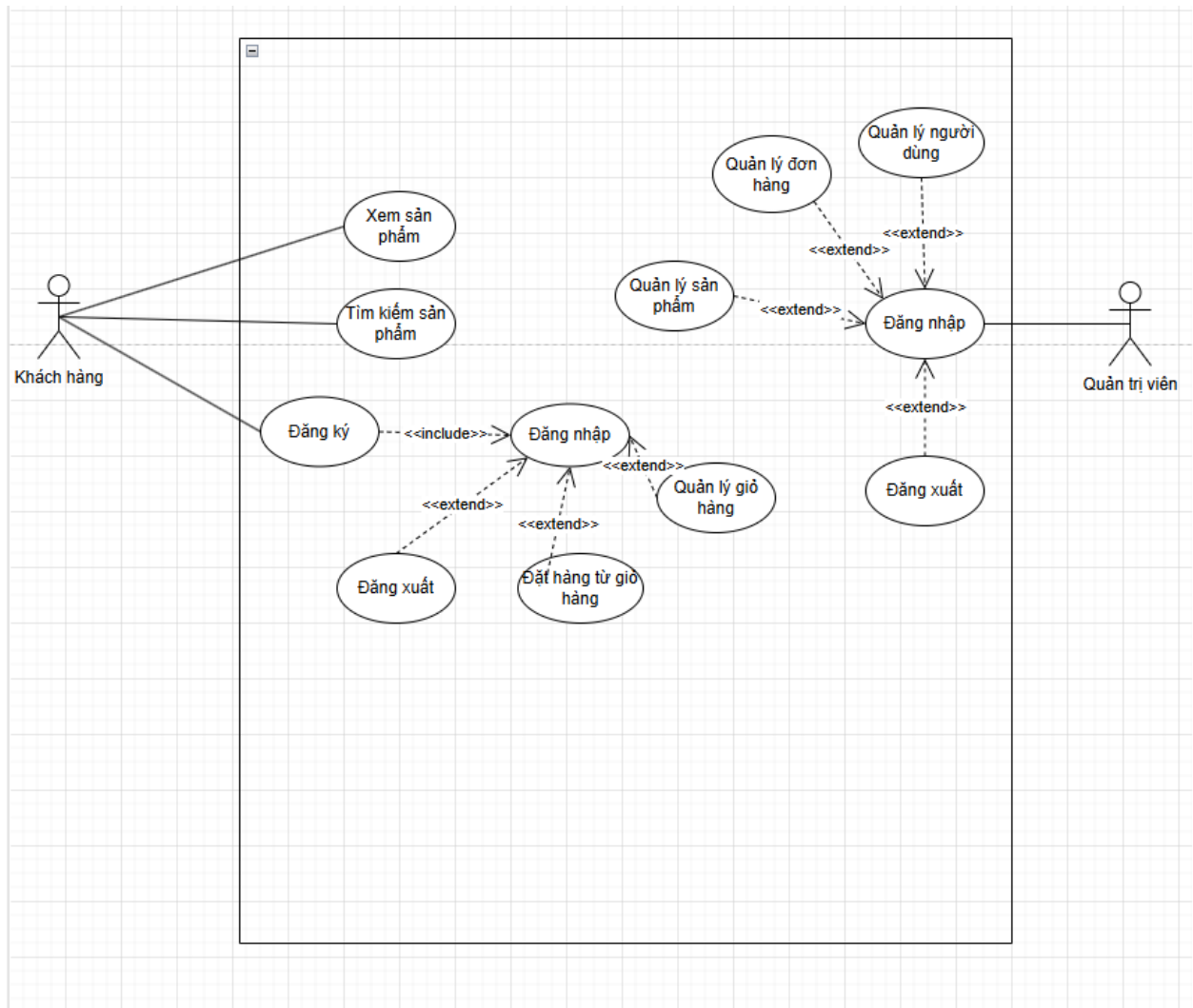
### ProductModel:

```
const productSchema = new mongoose.Schema(  
  {  
    name: { type: String, required: true, unique: true },  
    image: { type: String, required: true },  
    type: { type: String, required: true },  
    price: { type: Number, required: true },  
    countInStock: { type: Number, required: true },  
    rating: { type: Number, required: true },  
    description: { type: String },  
    discount: { type: Number },  
    sold: { type: Number },  
  },  
  {  
    timestamps: true,  
  }  
);
```

Bước 5: Sau khi cài đặt và thiết kế mô hình dữ liệu ở phía Backend hoàn thành thì bắt đầu xây dựng các giao diện tương ứng ở phía Frontend để gọi tới các API ở Backend để sử dụng.

### 3.4 Thiết kế dữ liệu

#### 3.4.1 Lược đồ use case tổng quan của hệ thống

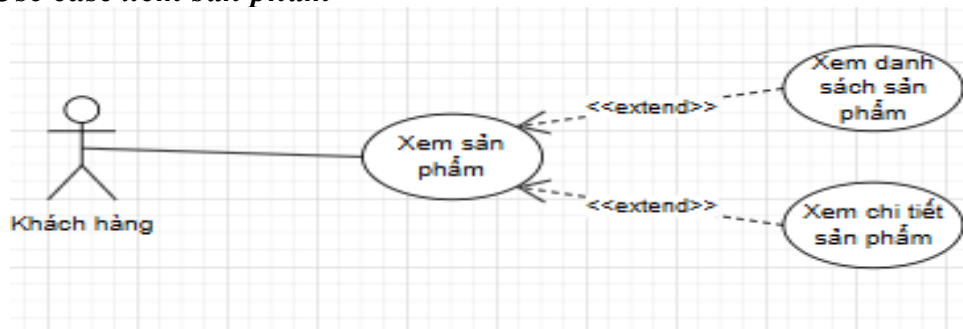


Hình 3.19 Mô hình use case tổng quan của hệ thống

Mô tả: Khách hàng có quyền đăng ký tài khoản, xem sản phẩm, tìm kiếm sản phẩm, ngoài ra sau khi đăng nhập, khách hàng có thể quản lý giỏ hàng của bản thân và đặt hàng từ giỏ hàng. Quản trị viên có quyền quản lý sản phẩm, quản lý đơn đặt hàng, quản lý tài khoản.

### 3.4.2 Mô tả các use case

#### 3.4.2.1 Use case xem sản phẩm



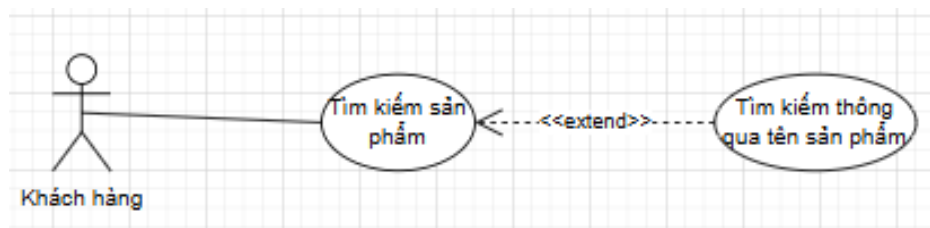
Hình 3.20 Use case xem sản phẩm

Tên use case: xem sản phẩm

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể xem danh sách sản phẩm và chi tiết sản phẩm

#### 3.4.2.2 Use case tìm kiếm sản phẩm



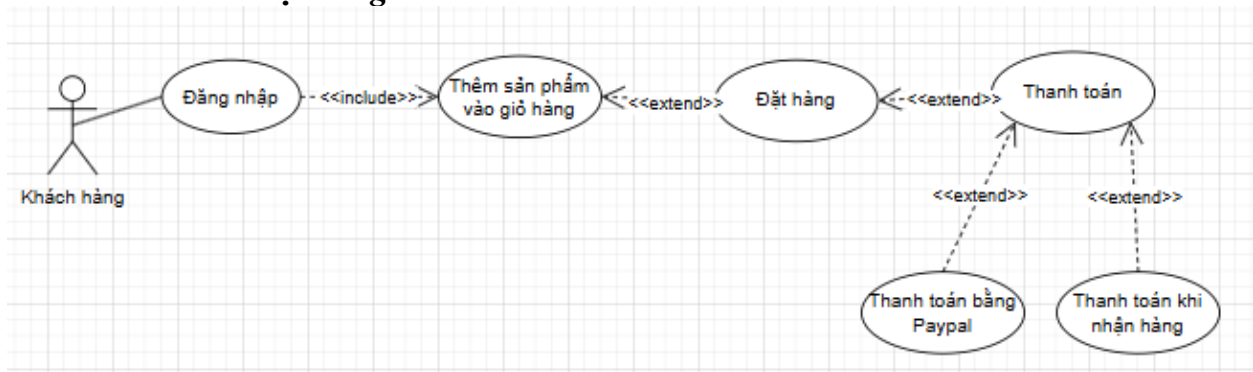
Hình 3.21 Use case tìm kiếm sản phẩm

Tên use case: tìm kiếm sản phẩm

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể tìm sản phẩm thông qua tên sản phẩm

### 3.4.2.3 Use case đặt hàng



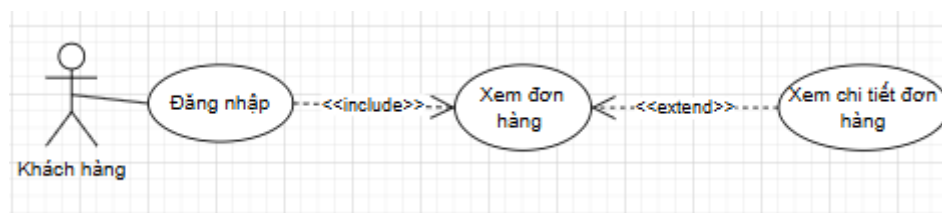
Hình 3.22 Use case đặt hàng

Tên use case: đặt hàng

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể đặt hàng (có sản phẩm trong giỏ hàng) và thanh toán theo 2 phương thức thanh toán khi nhận hàng hoặc thanh toán bằng Paypal

### 3.4.2.4 Use case xem đơn hàng đã đặt



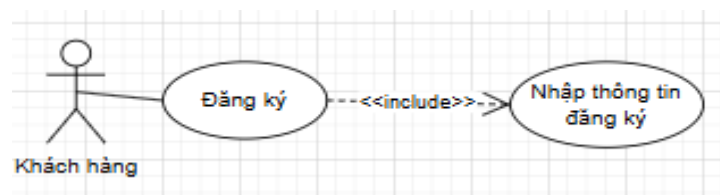
Hình 3.23 Use case xem đơn hàng

Tên use case: xem đơn hàng đã đặt

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể xem các đơn hàng đã đặt và có thể xem các chi tiết đơn hàng đó

### 3.4.2.5 Use case đăng ký



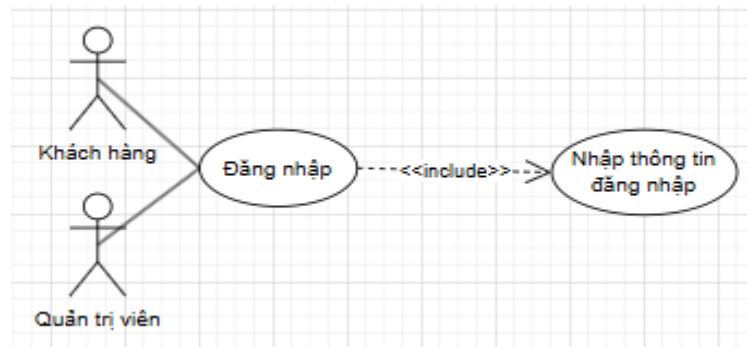
Hình 3.24 Use case đăng ký

Tên use case: đăng ký

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể đăng ký tài khoản

#### **3.4.2.6 Use case đăng nhập**



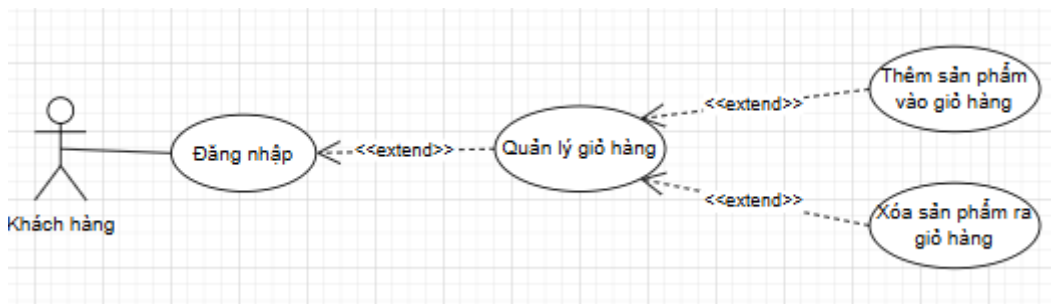
Hình 3.25 Use case đăng nhập

Tên use case: đăng nhập

Actor: khách hàng, quản trị viên

Mô tả use case: use case này cho phép khách hàng và quản trị viên có thể đăng nhập vào hệ thống

#### **3.4.2.7 Use case quản lý giỏ hàng**



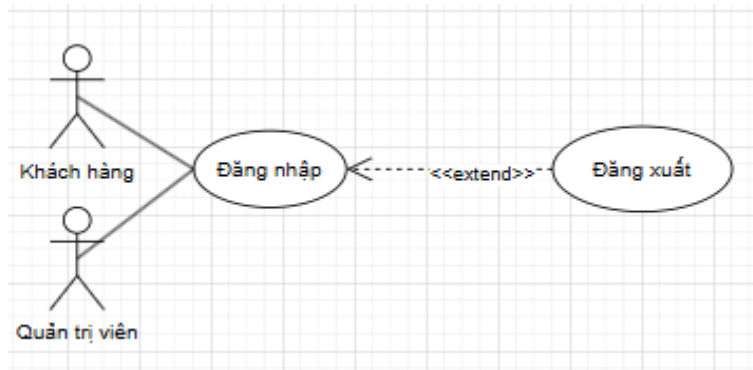
Hình 3.26 Use case quản lý giỏ hàng

Tên use case: quản lý giỏ hàng

Actor: khách hàng

Mô tả use case: use case này cho phép khách hàng có thể thêm sản phẩm vào giỏ hàng và xóa sản phẩm ra khỏi giỏ hàng

#### **3.4.2.8 Use case đăng xuất**



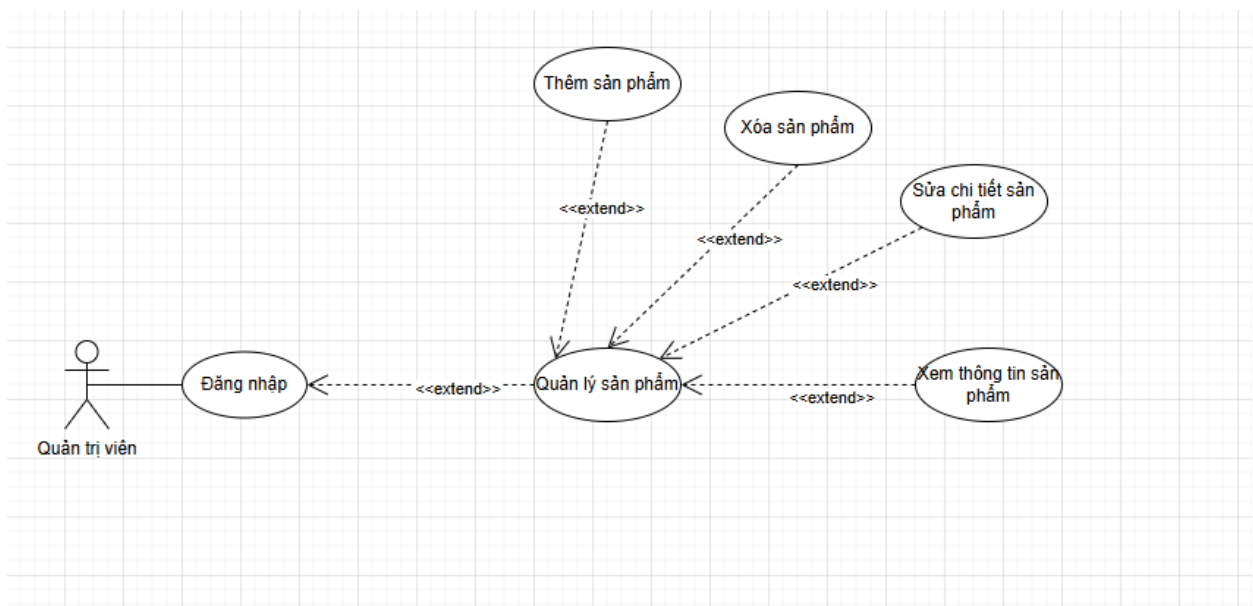
Hình 3.27 Use case đăng xuất

Tên use case: đăng xuất

Actor: khách hàng, quản trị viên

Mô tả use case: use case này cho phép khách hàng và quản trị viên có thể thoát khỏi hệ thống. Use case này chỉ thực hiện được sau khi khách hàng, quản trị viên đăng nhập.

#### 3.4.2.9 Use case quản lý sản phẩm



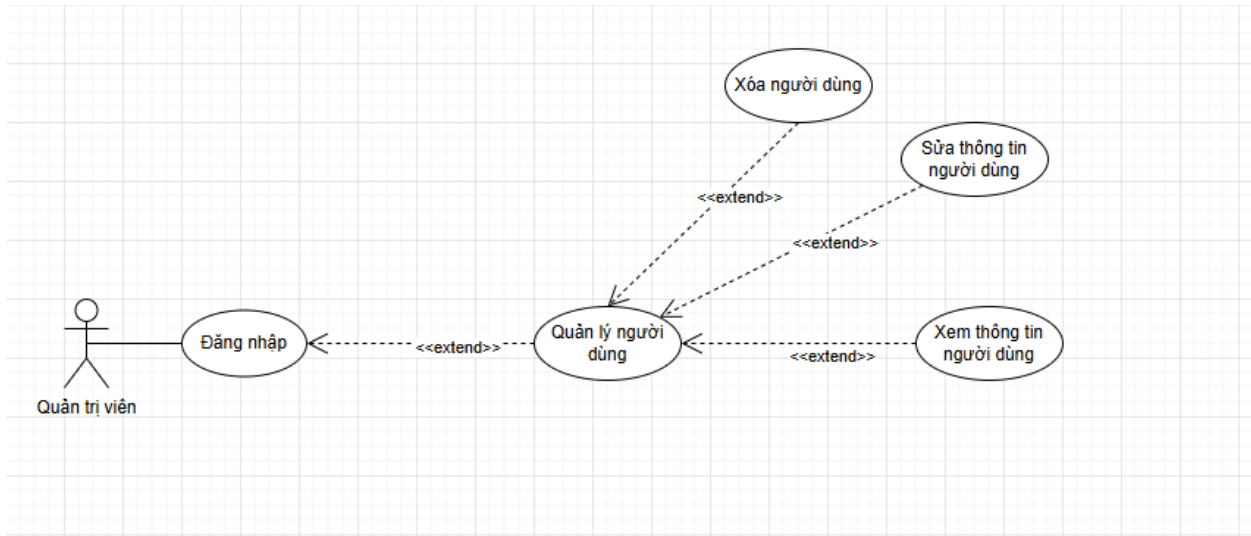
Hình 3.28 Use case Quản lý sản phẩm

Tên use case: quản lý sản phẩm

Actor: quản trị viên

Mô tả use case: use case này cho phép quản trị viên có thể thêm, sửa, xóa quản lý sản phẩm và chi tiết sản phẩm. Use case này chỉ thực hiện được sau khi quản trị viên đăng nhập

#### 3.4.2.10 Use case quản lý người dùng



Hình 3.29 Use case Quản lý người dùng

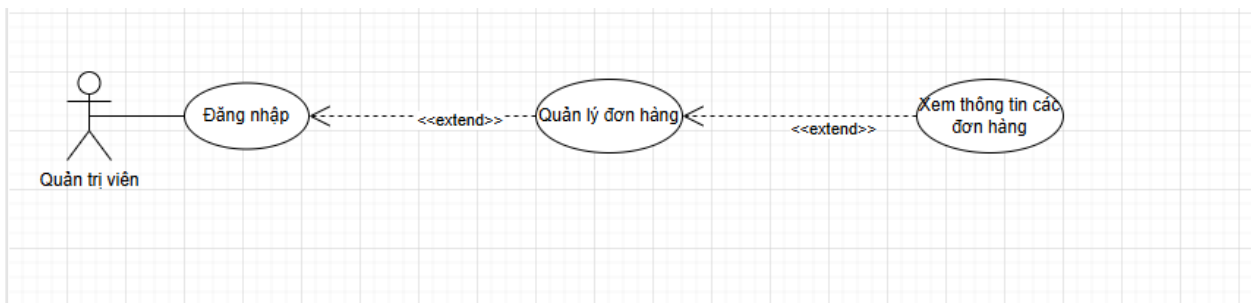
Tên use case: quản lý người dùng

Actor: quản trị viên

Mô tả use case: use case này cho phép quản trị viên có thể xem, sửa và xóa người dùng.

Use case này chỉ thực hiện được sau khi quản trị viên đăng nhập

#### 3.4.2.11 Use case quản lý đơn hàng



Hình 3.30 Use case Quản lý đơn hàng

Tên use case: quản lý đơn hàng

Actor: quản trị viên

Mô tả use case: use case này cho phép quản trị viên có thể xem các đơn hàng. Use case này chỉ thực hiện được sau khi quản trị viên đăng nhập

## CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU

### 4.1. Giao diện trang chủ

Đây là giao diện trang chủ ở đây người dùng có thể xem các sản phẩm, tìm kiếm sản phẩm và thêm sản phẩm vào giỏ hàng, ngoài ra còn có thể cập nhật thông tin cá nhân và xem các đơn hàng đã đặt

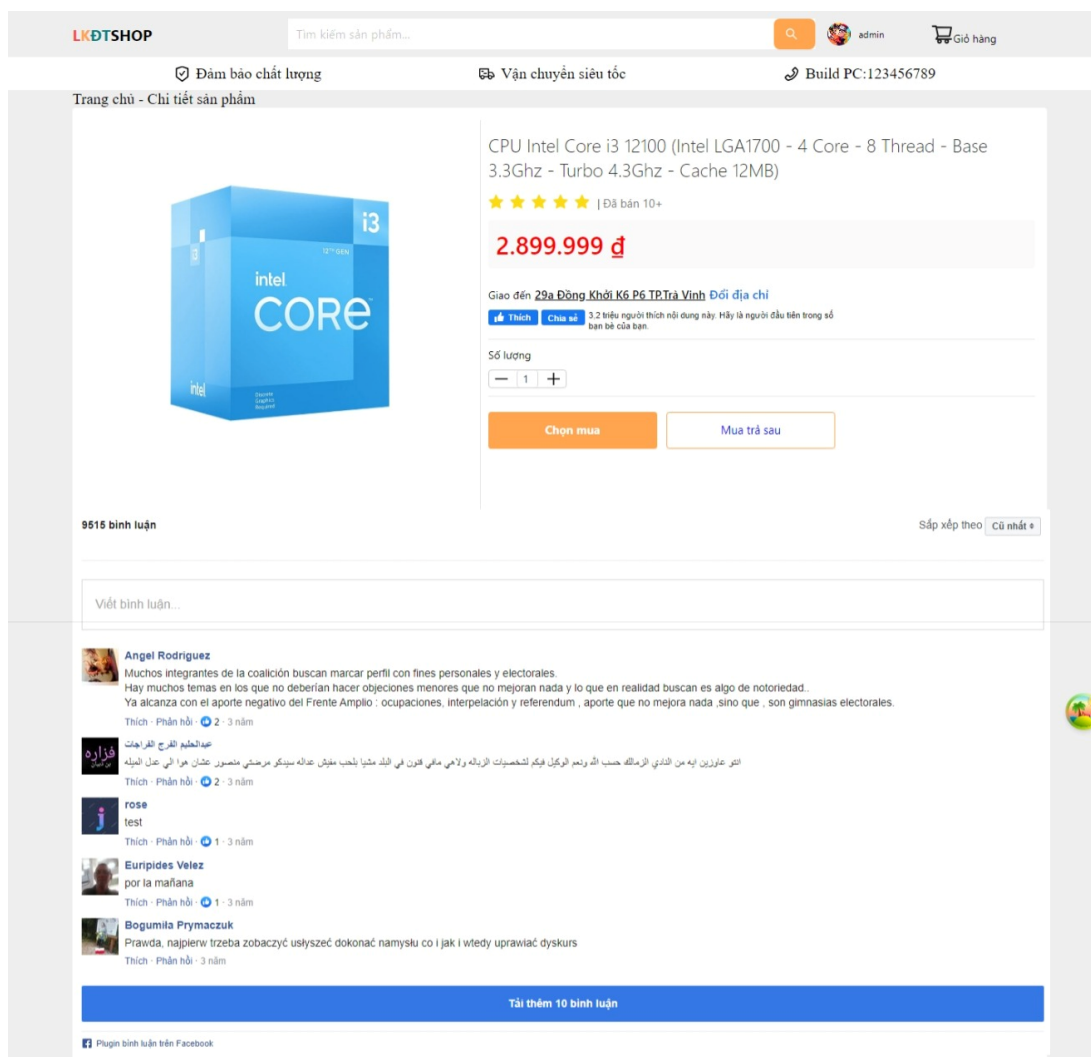


Hình 4.1 Giao diện trang chủ



## 4.2. Giao diện chi tiết sản phẩm

Đây là giao diện của trang chi tiết sản phẩm ở đây người dùng có thể xem chi tiết thông tin sản phẩm, tăng và giảm số lượng sản phẩm để thêm vào giỏ hàng

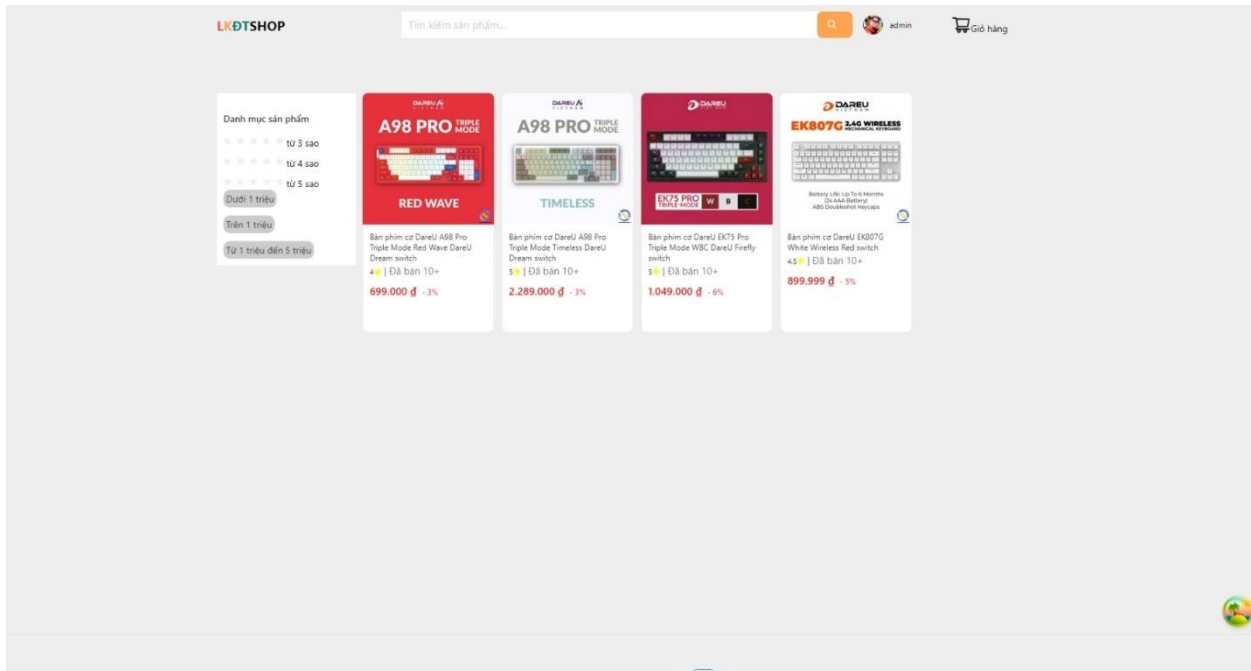


Hình 4.2 Giao diện chi tiết sản phẩm

## 4.3 Giao diện danh mục sản phẩm của từng sản phẩm

### 4.3.1 Giao diện danh mục bàn phím

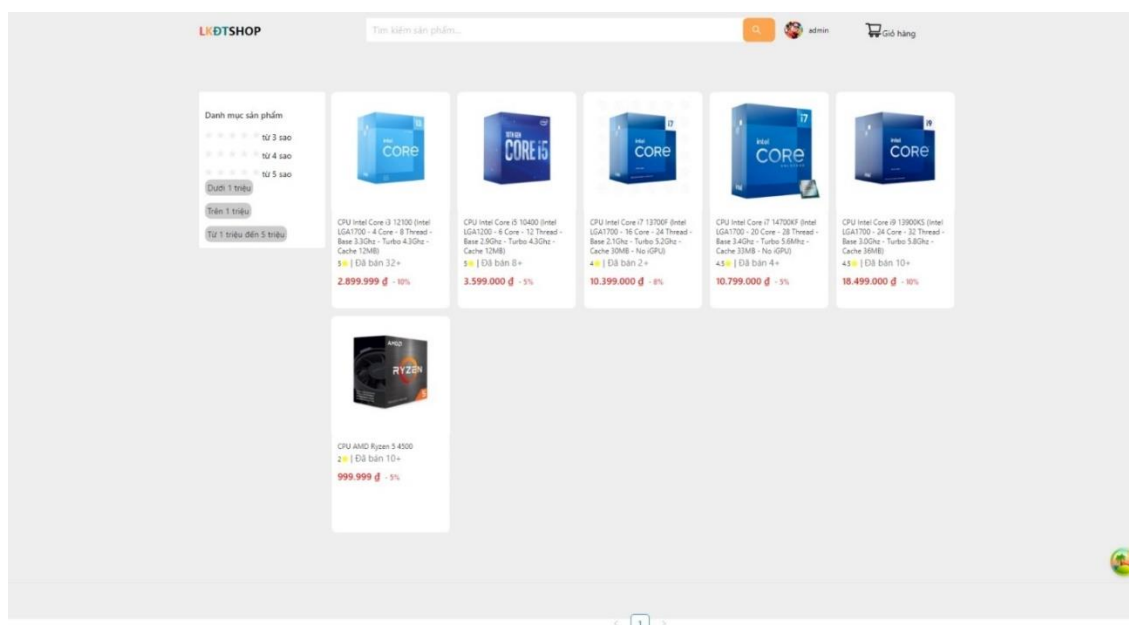
Đây là giao diện của trang danh mục bàn phím ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.3 Giao diện danh mục bàn phím

### 4.3.2 Giao diện danh mục CPU

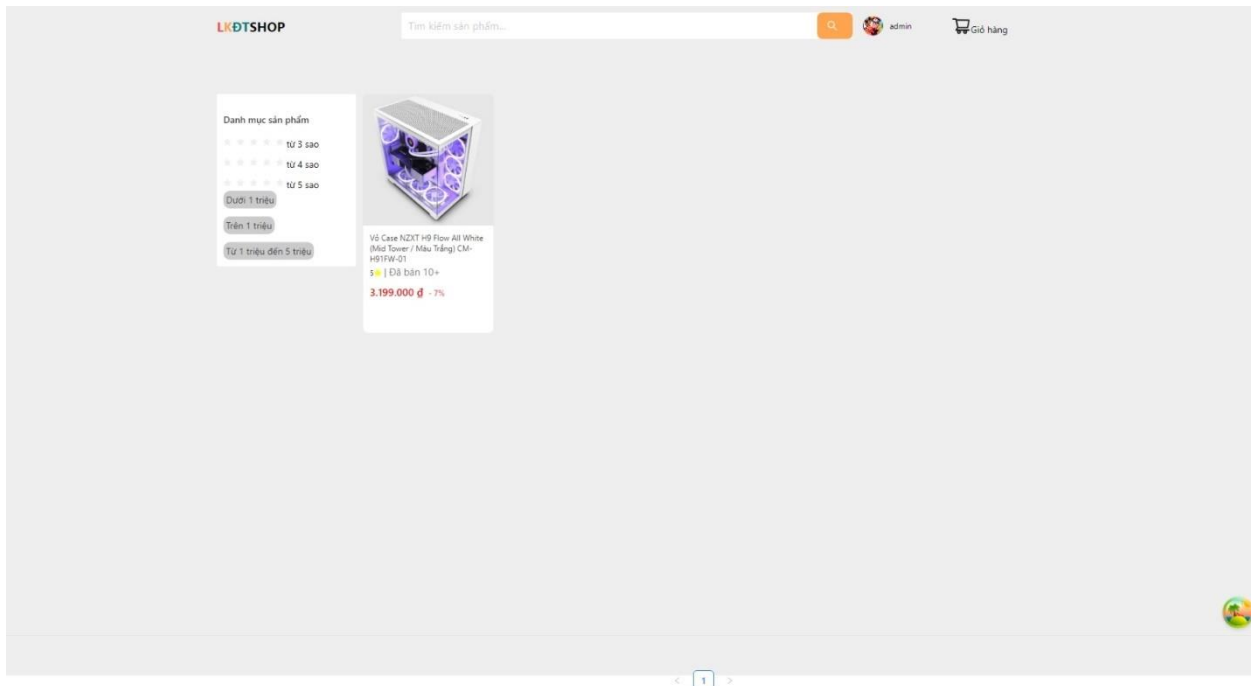
Đây là giao diện của trang danh mục CPU ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.4 Giao diện danh mục CPU

### 4.3.3 Giao diện danh mục Case

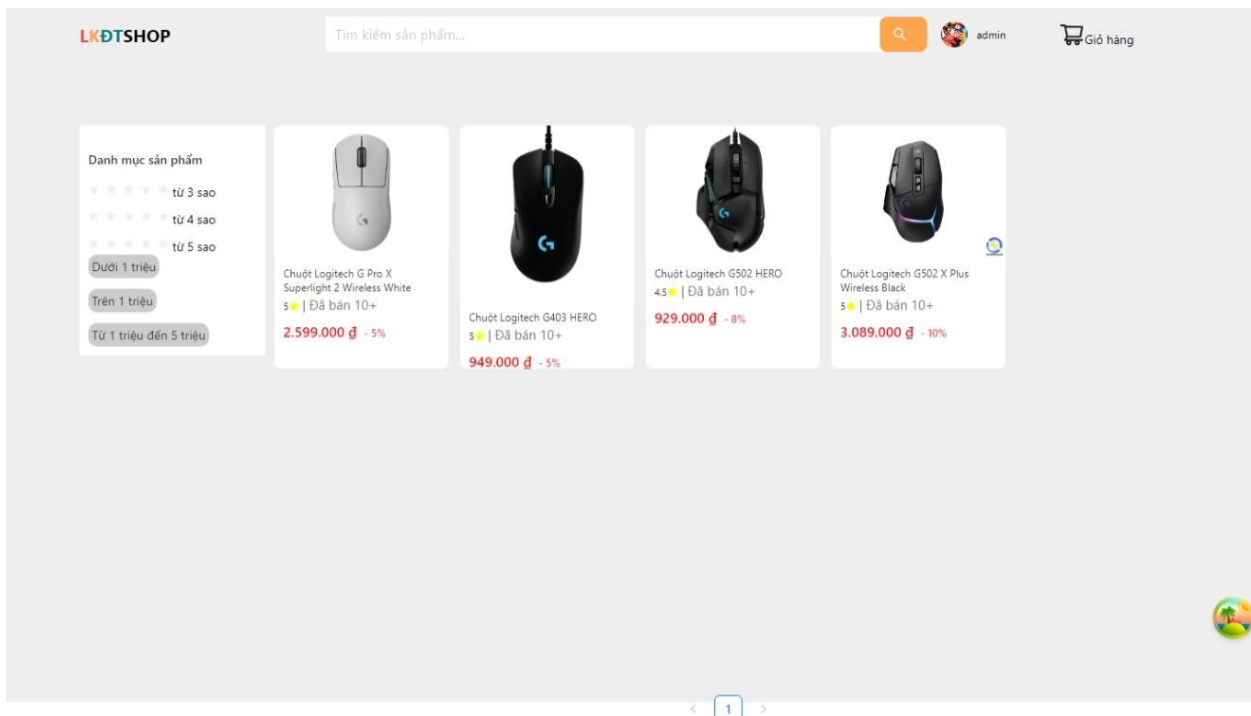
Đây là giao diện của trang danh mục Case ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.5 Giao diện danh mục Case

### 4.3.4 Giao diện danh mục Chuột

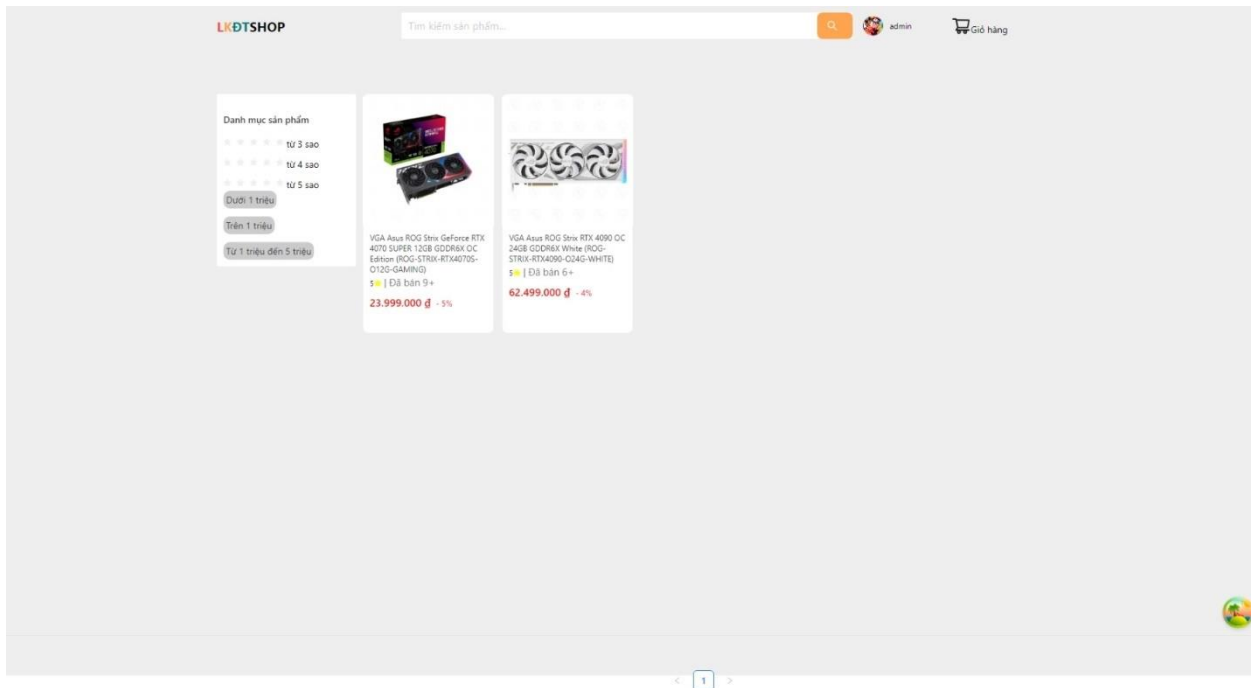
Đây là giao diện của trang danh mục Chuột ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.6 Giao diện danh mục Chuột

### 4.3.5 Giao diện danh mục GPU

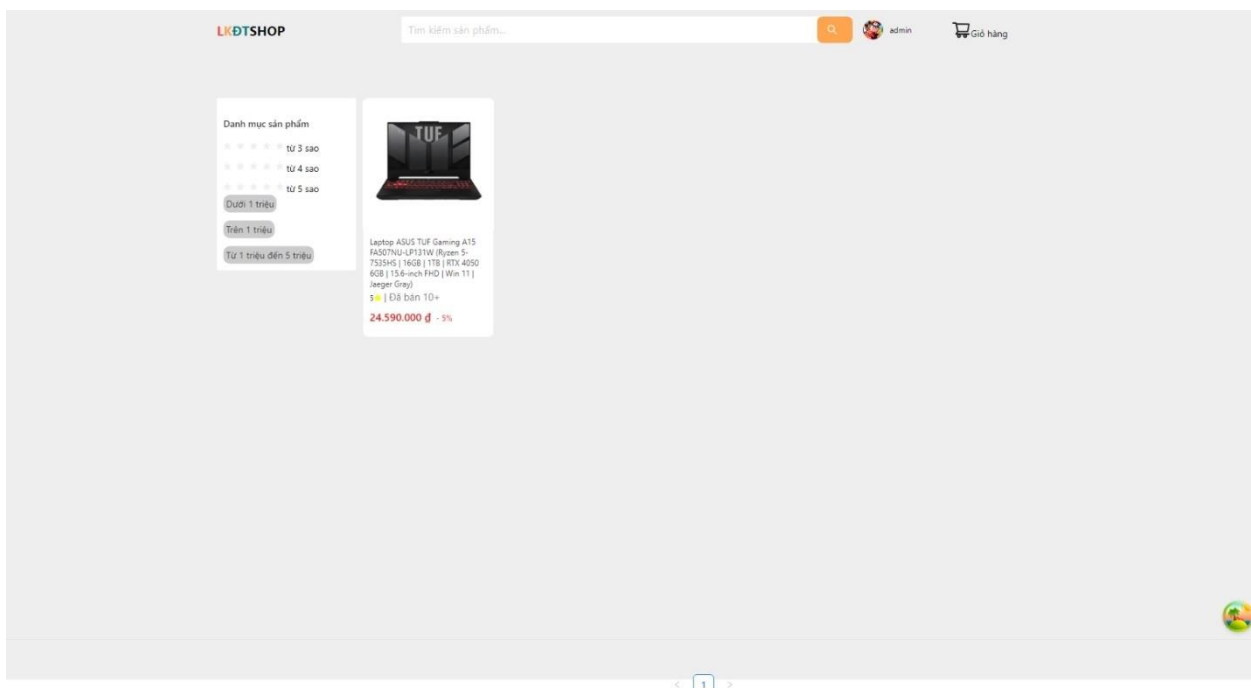
Đây là giao diện danh mục GPU ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.7 Giao diện danh mục GPU

### 4.3.6 Giao diện danh mục Laptop

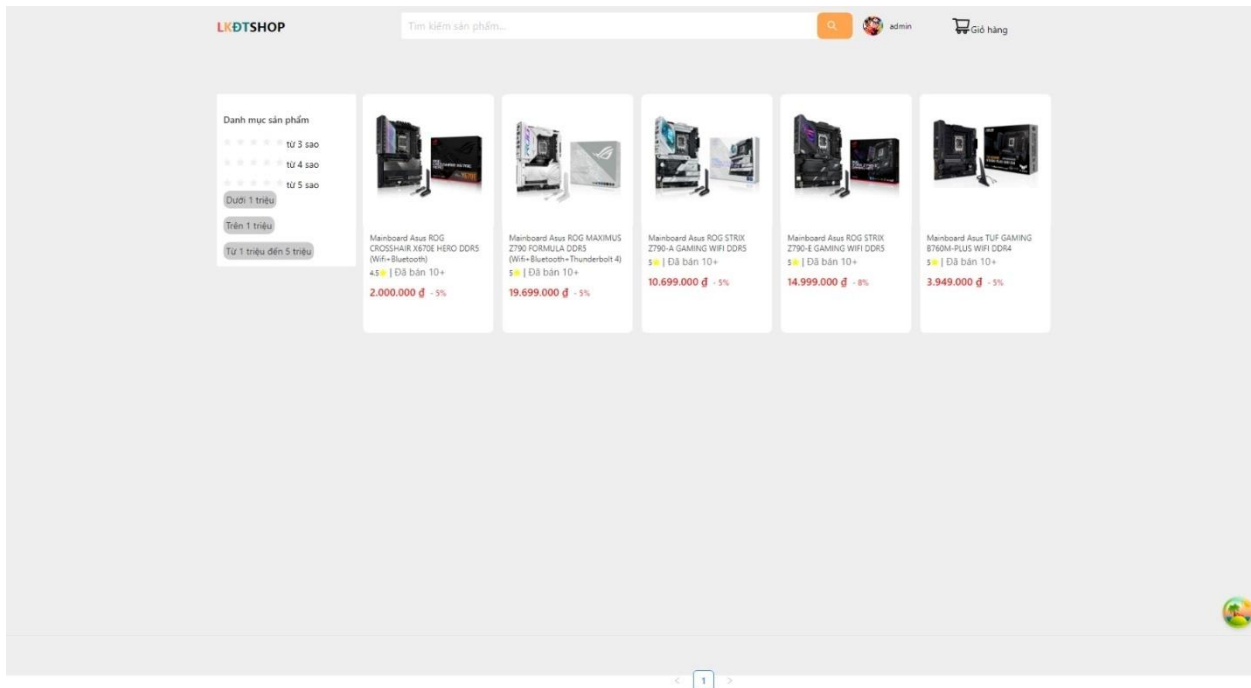
Đây là giao diện danh mục Laptop ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.8 Giao diện danh mục Laptop

### 4.3.7 Giao diện danh mục Main

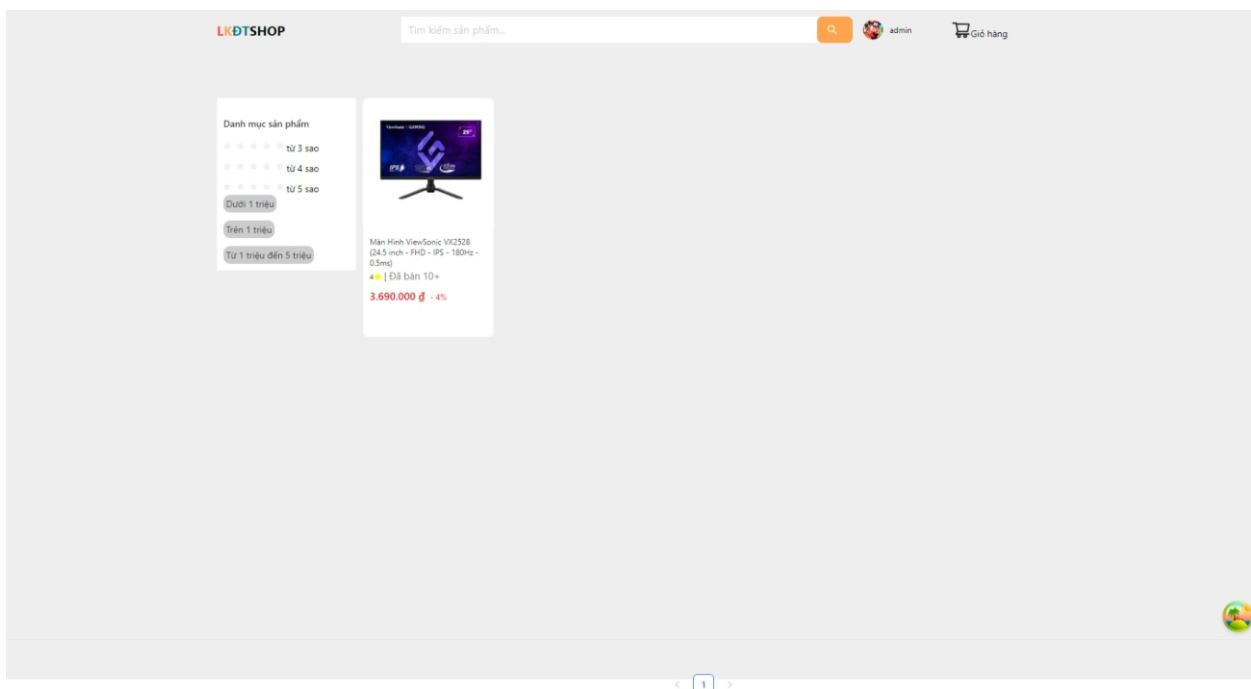
Đây là giao diện danh mục Main ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.9 Giao diện danh mục Main

### 4.3.8 Giao diện danh mục Màn hình

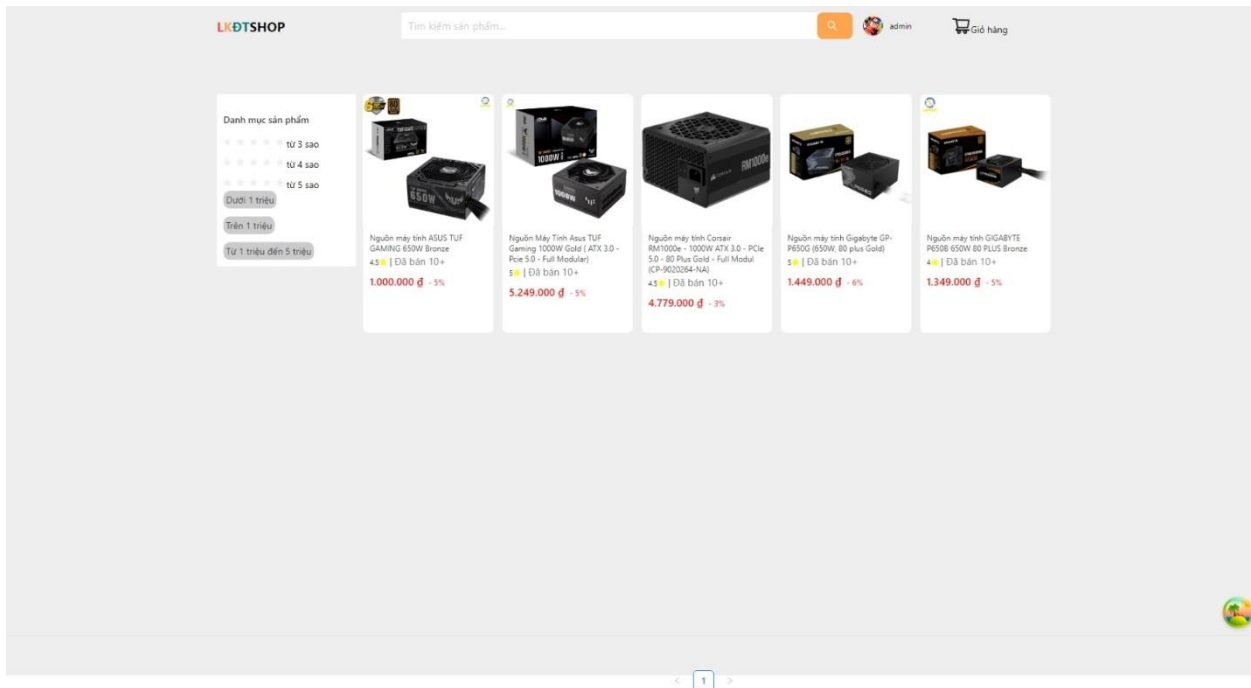
Đây là giao diện danh mục Màn hình ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.10 Giao diện danh mục Màn hình

### 4.3.9 Giao diện danh mục PSU

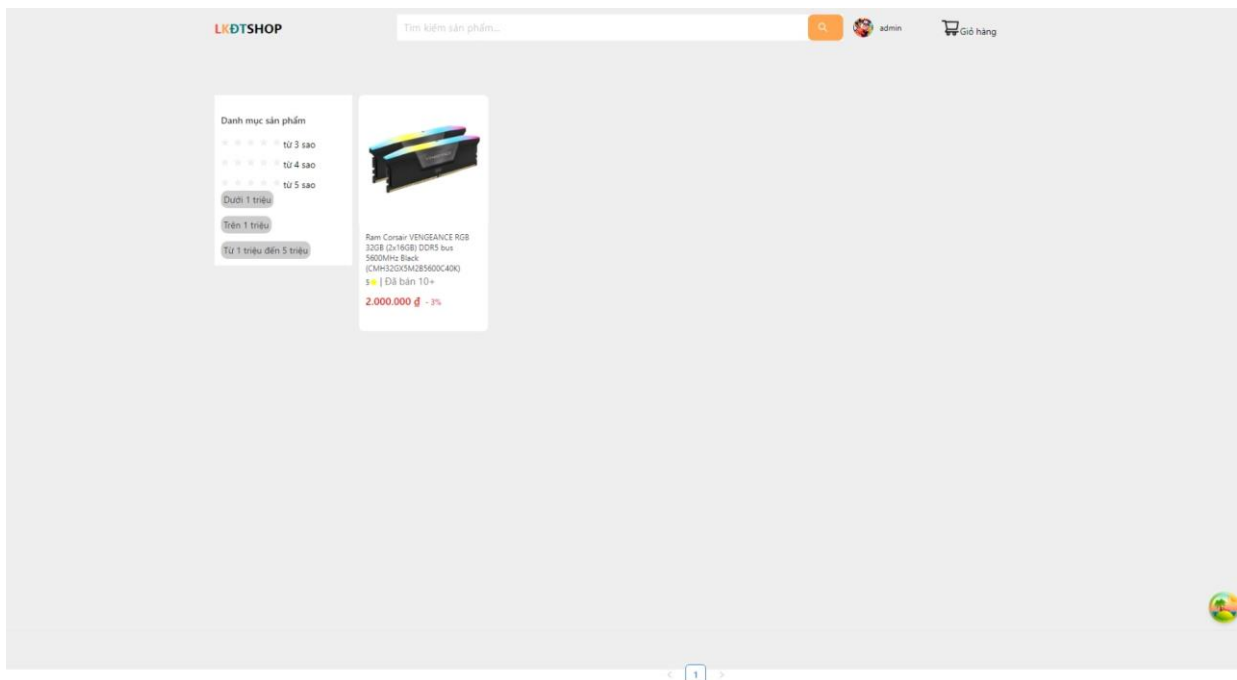
Đây là giao diện danh mục PSU ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.11 Giao diện danh mục PSU

### 4.3.10 Giao diện danh mục Ram

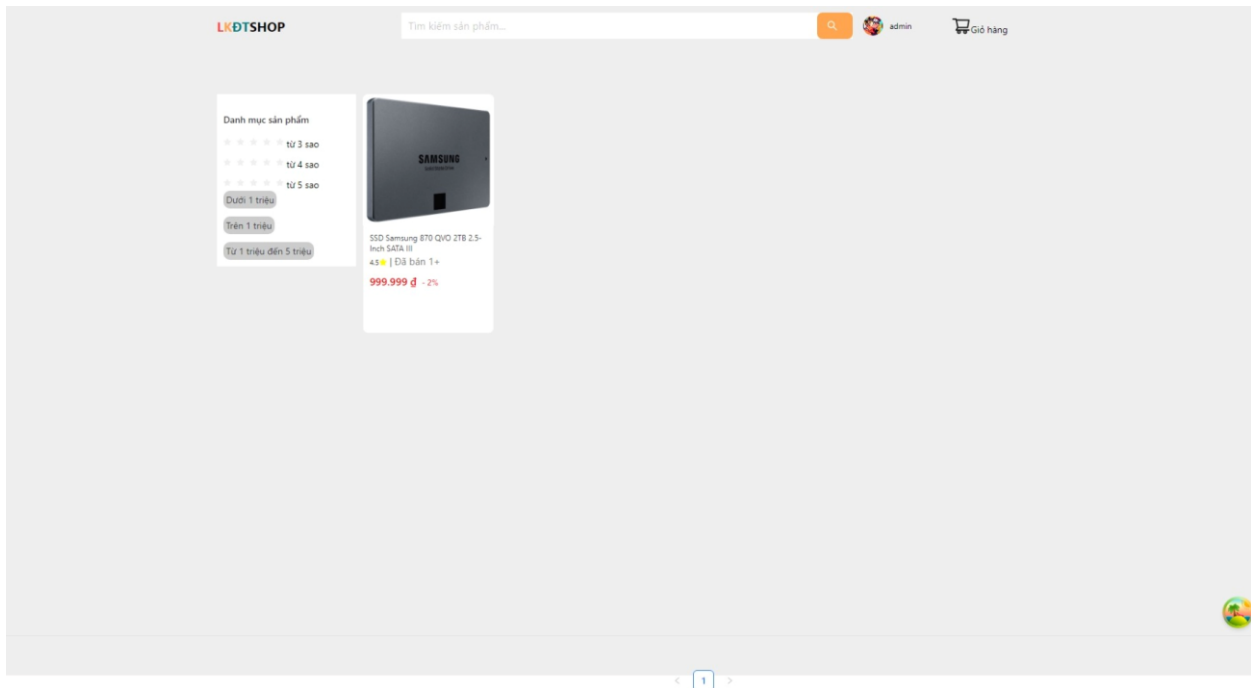
Đây là giao diện danh mục Ram ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.12 Giao diện danh mục Ram

#### 4.3.11 Giao diện danh mục SSD

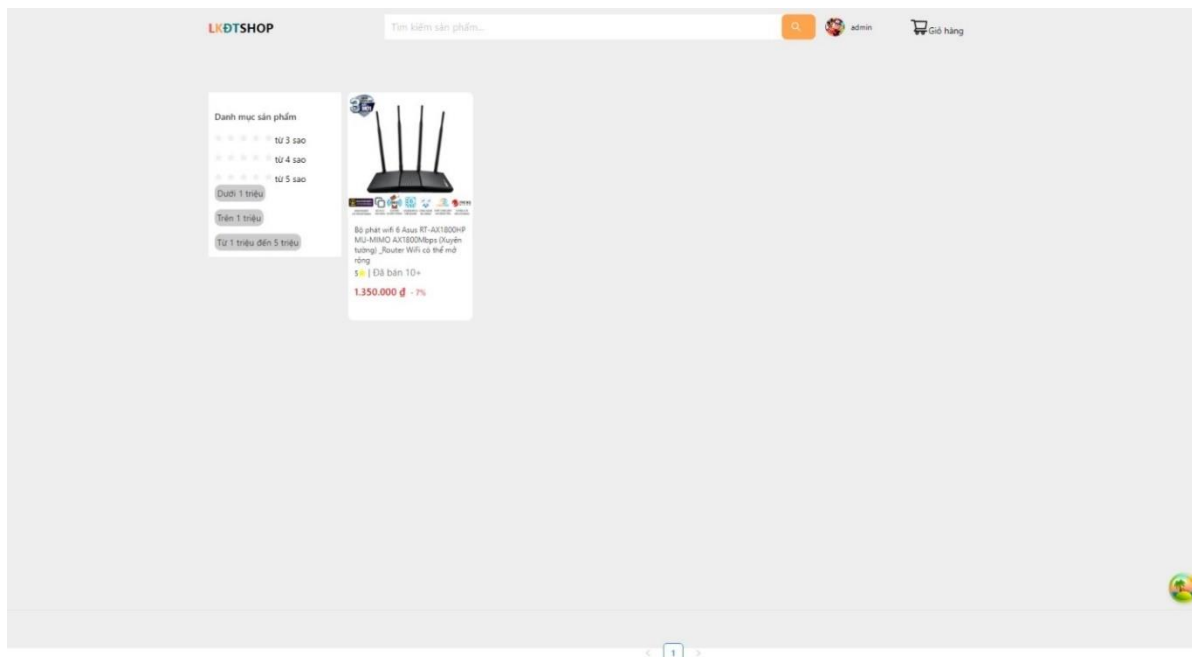
Đây là giao diện danh mục SSD ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm



Hình 4.13 Giao diện danh mục SSD

#### 4.3.12 Giao diện danh mục Thiết bị mạng

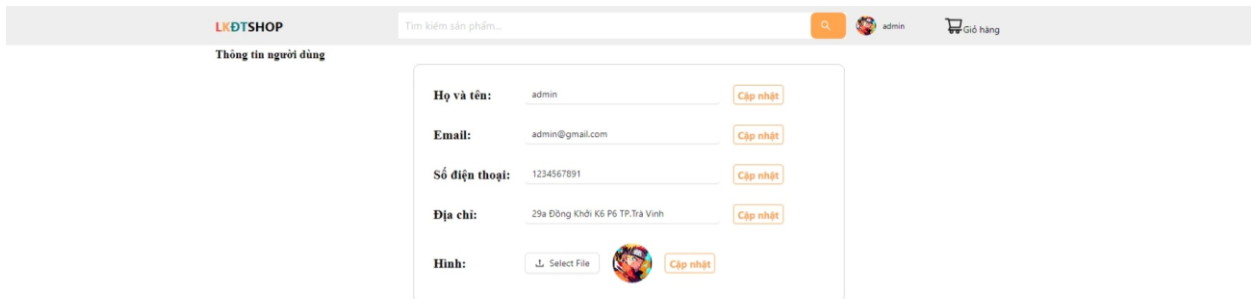
Đây là giao diện danh mục Thiết bị mạng ở đây người dùng có thể lọc sản phẩm theo số sao của sản phẩm và giá của sản phẩm




Hình 4.14 Giao diện danh mục Thiết bị mạng

#### 4.4 Giao diện trang thông tin người dùng

Đây là trang thông tin người dùng ở đây người dùng có thể cập nhật thông tin cá nhân của mình



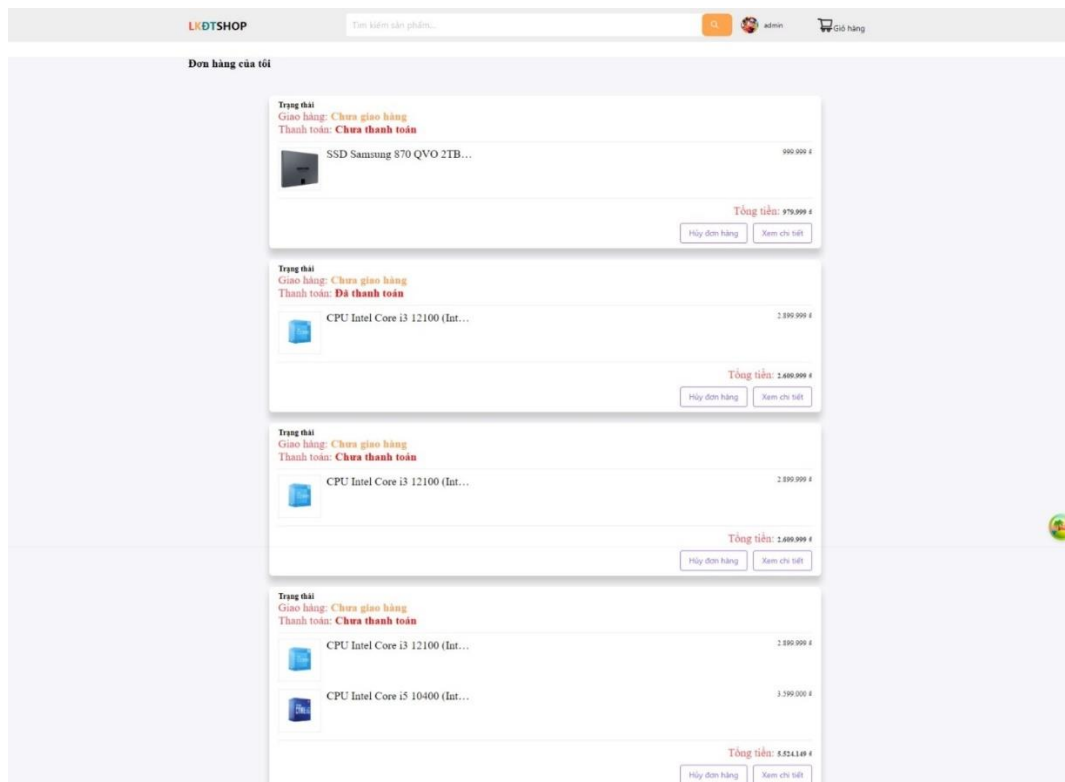
Thông tin người dùng

Họ và tên:	admin	Cập nhật
Email:	admin@gmail.com	Cập nhật
Số điện thoại:	1234567891	Cập nhật
Địa chỉ:	29a Đồng Khởi K6 P6 TP.Trà Vinh	Cập nhật
Hình:	<input type="button" value="Select File"/> 	Cập nhật

Hình 4.15 Giao diện Thông tin người dùng

#### 4.5 Giao diện trang đơn hàng

Đây là giao diện trang đơn hàng ở đây người dùng có thể hủy đơn hàng mình đặt và xem chi tiết đơn hàng



Đơn hàng của tôi

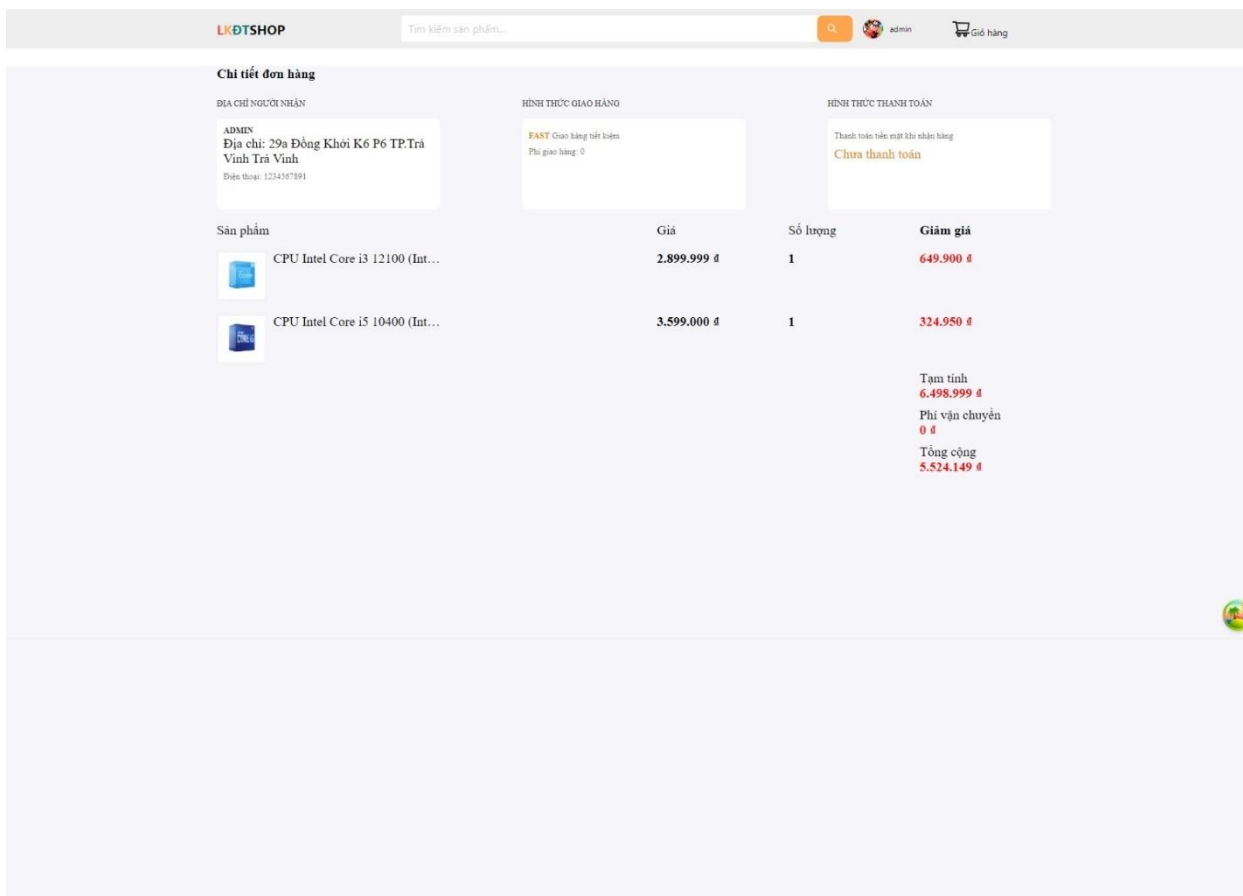
Trạng thái: Chưa giao hàng	Giao hàng: Chưa giao hàng	Thanh toán: Chưa thanh toán	SSD Samsung 870 QVO 2TB...	990.000 đ	Tổng tiền: 970.000 đ	Hủy đơn hàng	Xem chi tiết
Trạng thái: Chưa giao hàng	Giao hàng: Chưa giao hàng	Thanh toán: Đã thanh toán	CPU Intel Core i3 12100 (Int...	2.890.000 đ	Tổng tiền: 2.400.000 đ	Hủy đơn hàng	Xem chi tiết
Trạng thái: Chưa giao hàng	Giao hàng: Chưa giao hàng	Thanh toán: Chưa thanh toán	CPU Intel Core i3 12100 (Int...	2.890.000 đ	Tổng tiền: 2.400.000 đ	Hủy đơn hàng	Xem chi tiết
Trạng thái: Chưa giao hàng	Giao hàng: Chưa giao hàng	Thanh toán: Chưa thanh toán	CPU Intel Core i3 12100 (Int...	2.890.000 đ	Tổng tiền: 8.524.140 đ	Hủy đơn hàng	Xem chi tiết
			CPU Intel Core i5 10400 (Int...	3.590.000 đ			

Hình 4.16 Giao diện Đơn hàng





## 4.6 Giao diện trang chi tiết đơn hàng

Đây là trang chi tiết đơn hàng ở đây người dùng có thể xem được các chi tiết của đơn hàng mình đặt gồm tên sản phẩm, giá, số lượng, ...



The screenshot displays the 'Chi tiết đơn hàng' (Order Details) page. At the top, there is a navigation bar with the 'LKĐTSHOP' logo, a search bar, and user account links. The main content area is divided into several sections:

- Chi tiết đơn hàng**: A section containing order details such as the user's name (ADMIN), address (29a Đồng Khởi K6 P6 TP.Trà Vinh), and phone number (1234567891).
- HÌNH THỨC GIAO HÀNG**: A section showing the delivery method as 'FAST Giao hàng tiết kiệm' with a delivery fee of 0.
- HÌNH THỨC THANH TOÁN**: A section showing the payment method as 'Thanh toán tiền mặt khi nhận hàng' (Cash on delivery) with a status of 'Chưa thanh toán' (Not paid).
- Sản phẩm**: A table listing the products in the order.

Sản phẩm	Giá	Số lượng	Giảm giá
 CPU Intel Core i3 12100 (Int...	2.899.999 đ	1	649.900 đ
 CPU Intel Core i5 10400 (Int...	3.599.000 đ	1	324.950 đ

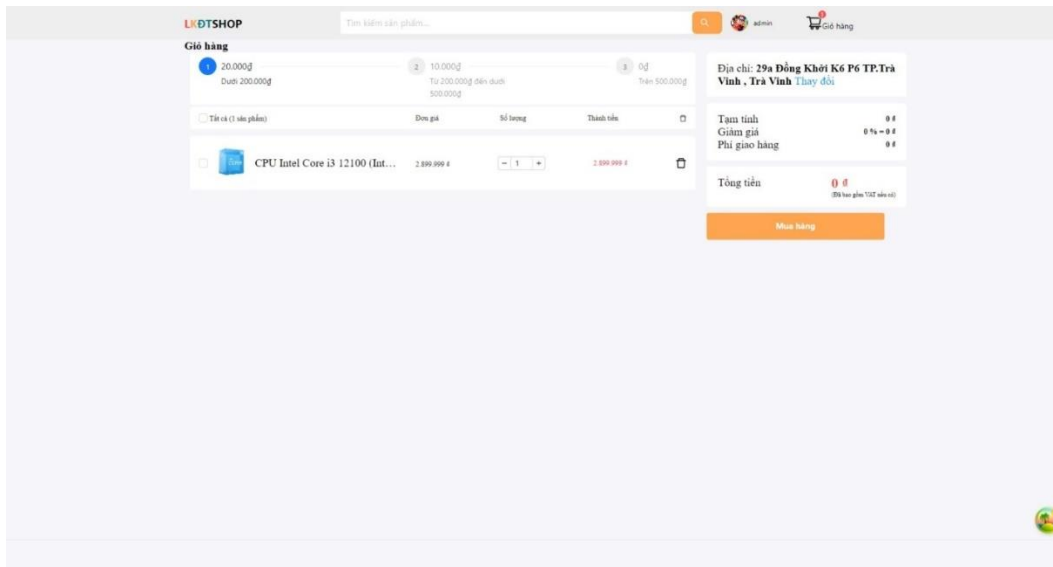
At the bottom right, a summary of the order is provided:

- Tạm tính: 6.498.999 đ
- Phí vận chuyển: 0 đ
- Tổng cộng: 5.524.149 đ

Hình 4.17 Giao diện Chi tiết đơn hàng

#### 4.7 Giao diện trang giỏ hàng

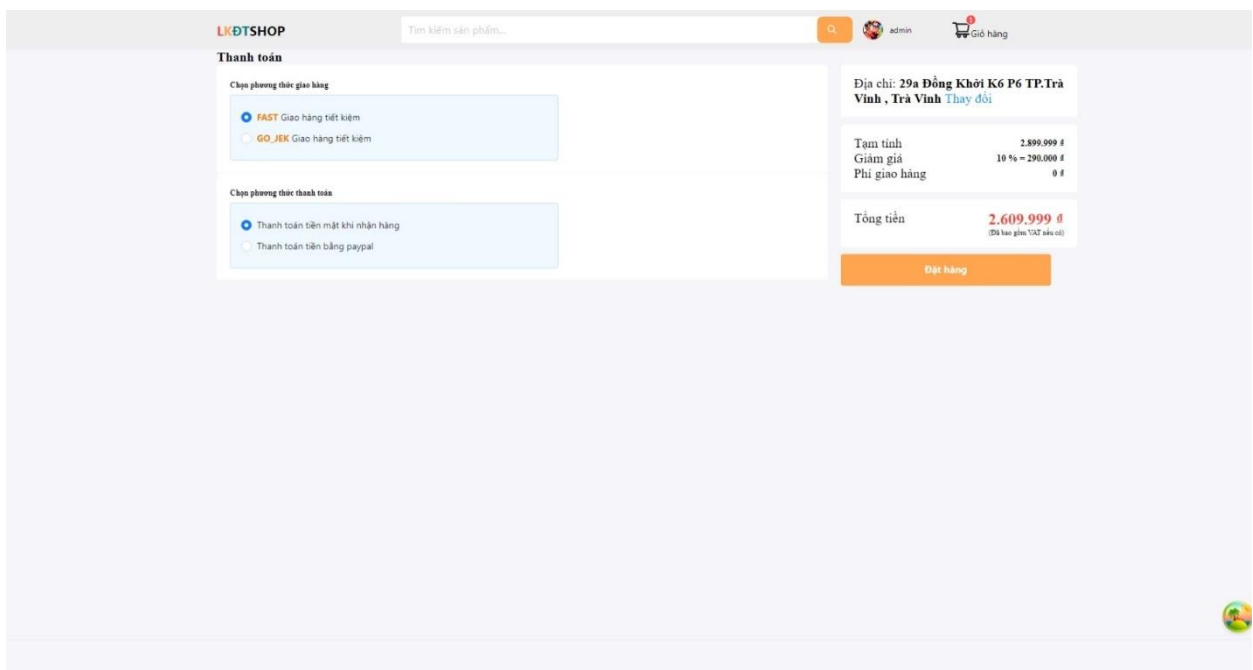
Đây là giao diện trang giỏ hàng ở đây người dùng có thể chọn các sản phẩm đã được thêm vào giỏ hàng để mua hàng, có thể tăng giảm số lượng sản phẩm và xóa sản phẩm



Hình 4.18 Giao diện Giỏ hàng

#### 4.8 Giao diện trang thanh toán

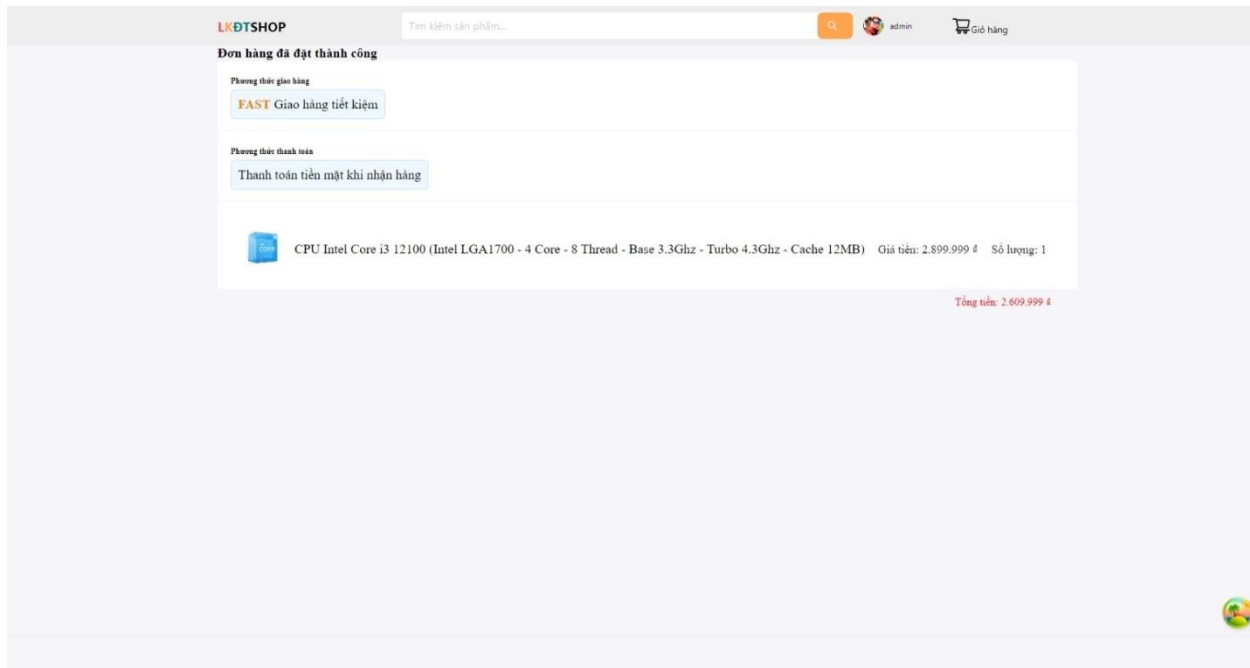
Đây là giao diện trang thanh toán ở đây người dùng có thể chọn phương thức giao hàng và chọn phương thức thanh toán



Hình 4.19 Giao diện trang Thanh toán

#### 4.9 Giao diện trang đặt hàng thành công

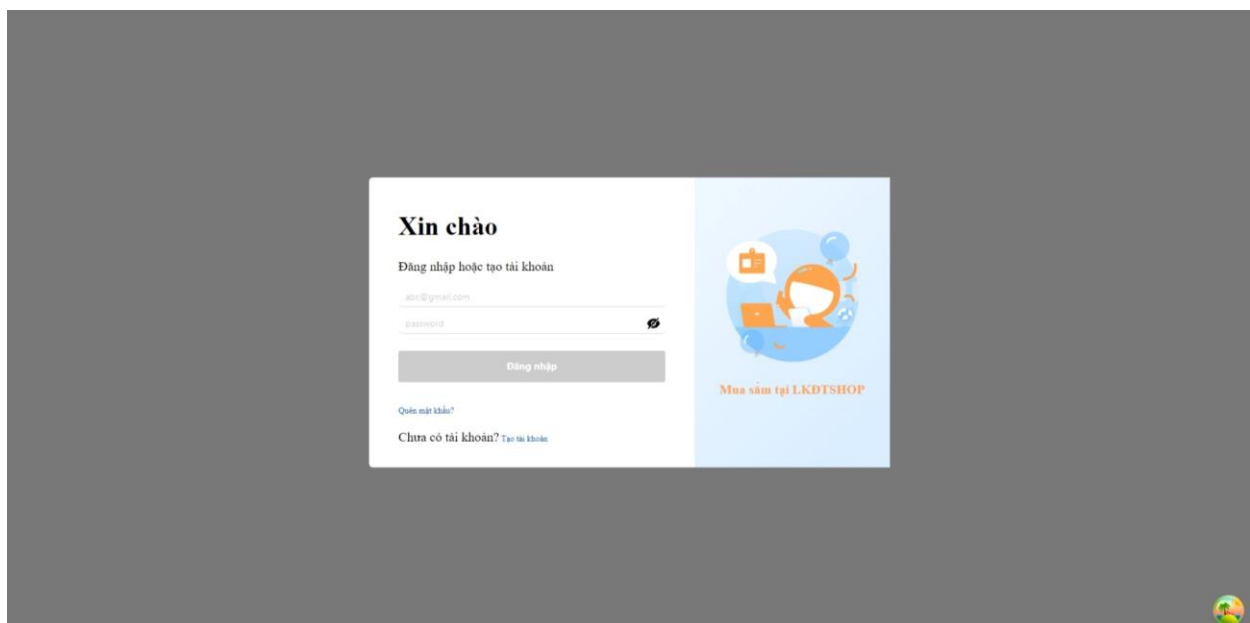
Đây là giao diện trang đặt hàng thành công ở đây người dùng có thể thấy được thông tin đơn hàng của mình đặt thành công



Hình 4.20 Giao diện trang Đặt hàng thành công

#### 4.10 Giao diện trang đăng nhập

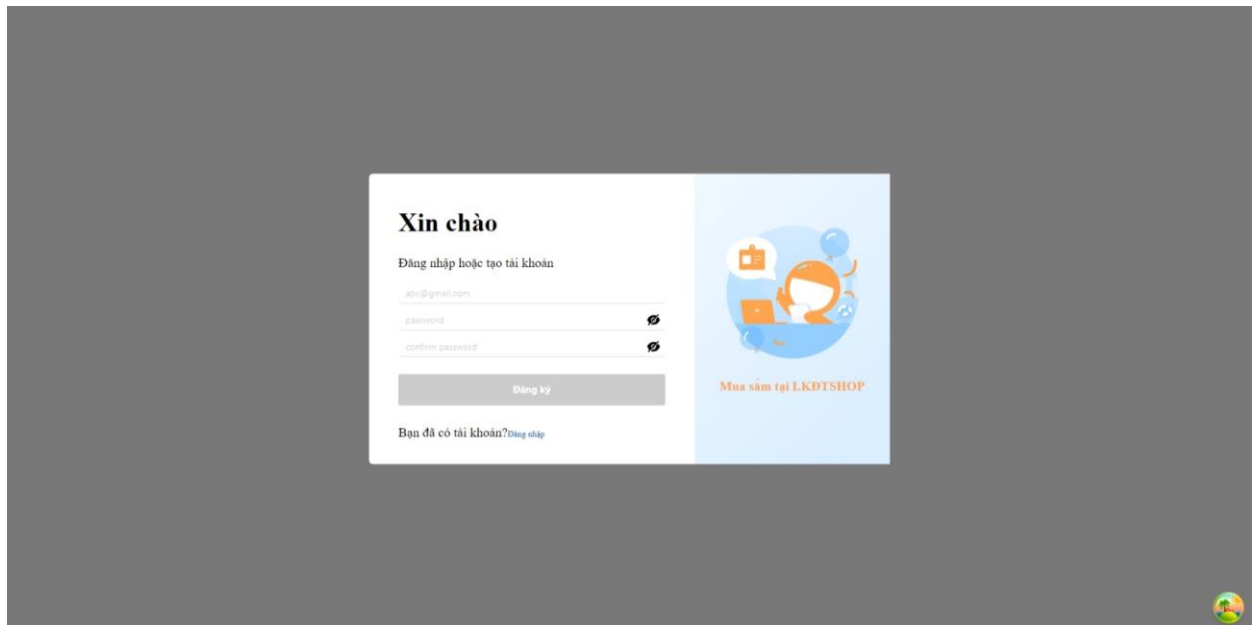
Đây là giao diện trang đăng nhập ở đây người dùng có thể nhập thông tin tài khoản để đăng nhập



Hình 4.21 Giao diện trang Đăng nhập

#### 4.11 Giao diện trang đăng ký

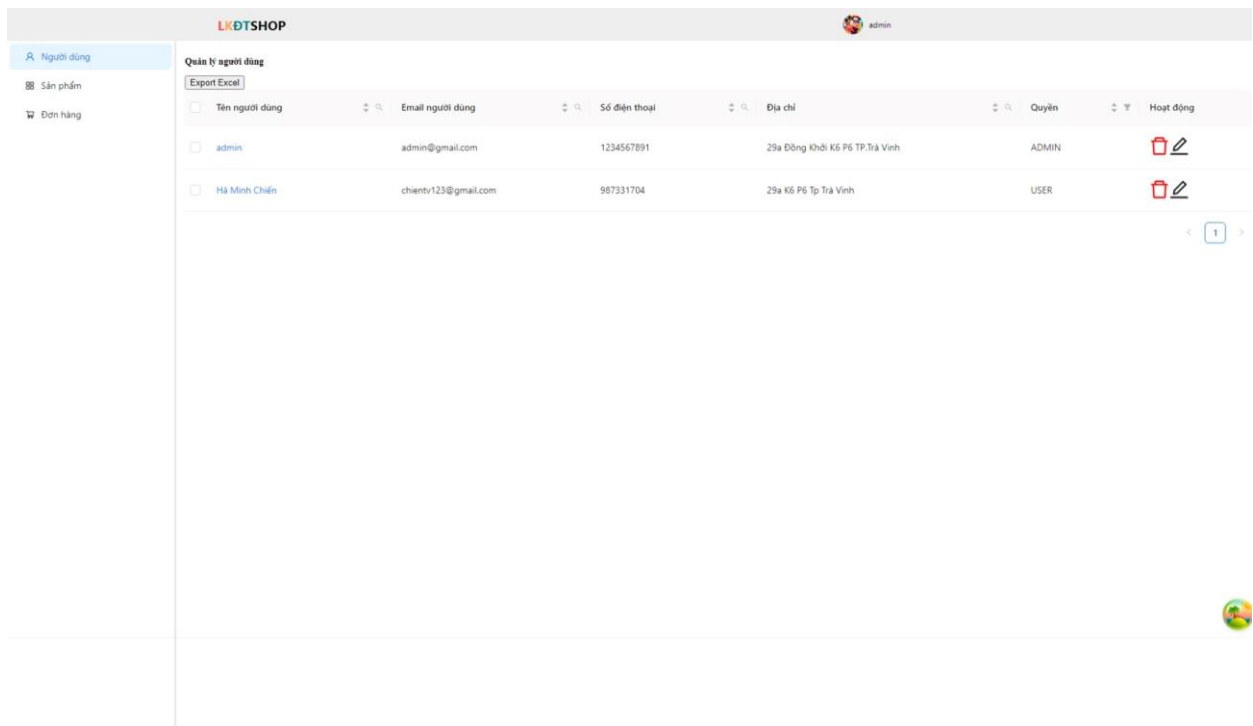
Đây là giao diện trang đăng ký ở đây người dùng có thể nhập thông tin để đăng ký tài khoản



Hình 4.22 Giao diện trang Đăng ký

#### 4.12 Giao diện trang quản lý người dùng

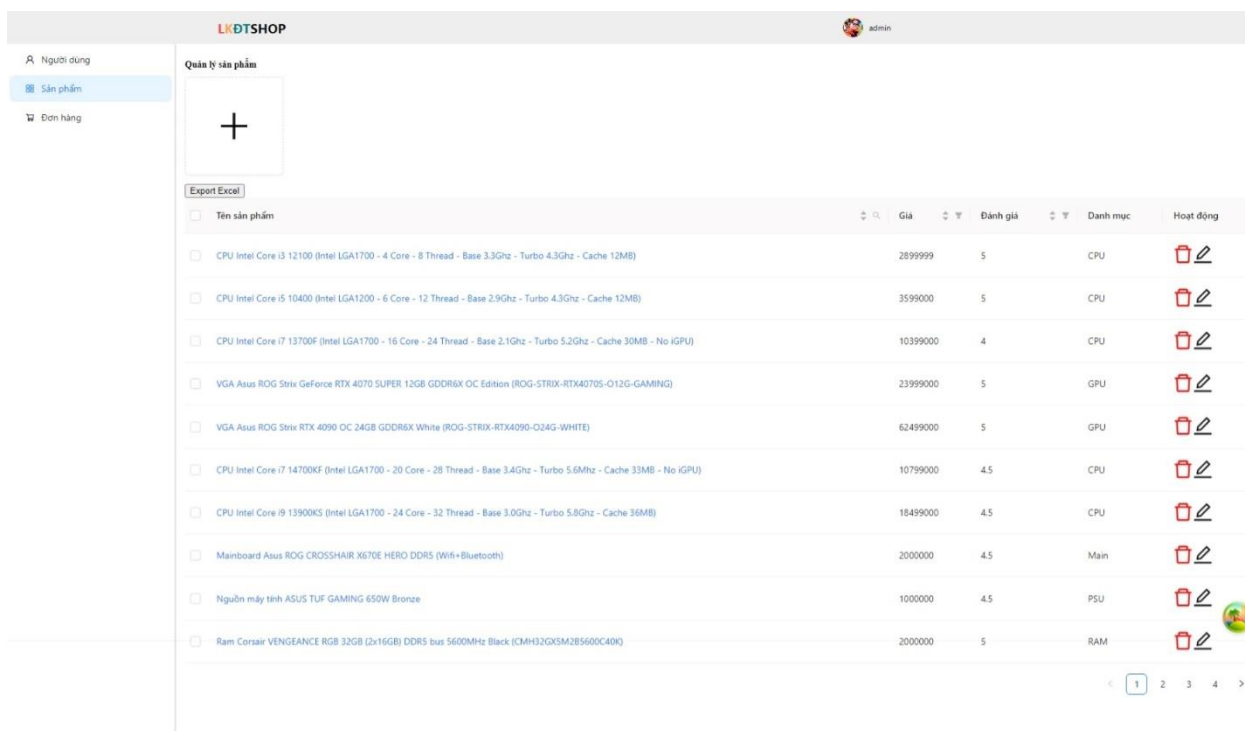
Đây là giao diện trang quản lý người dùng ở đây quản trị viên có thể xóa và sửa thông tin của người dùng



Hình 4.23 Giao diện trang Quản lý người dùng

### 4.13 Giao diện trang quản lý sản phẩm

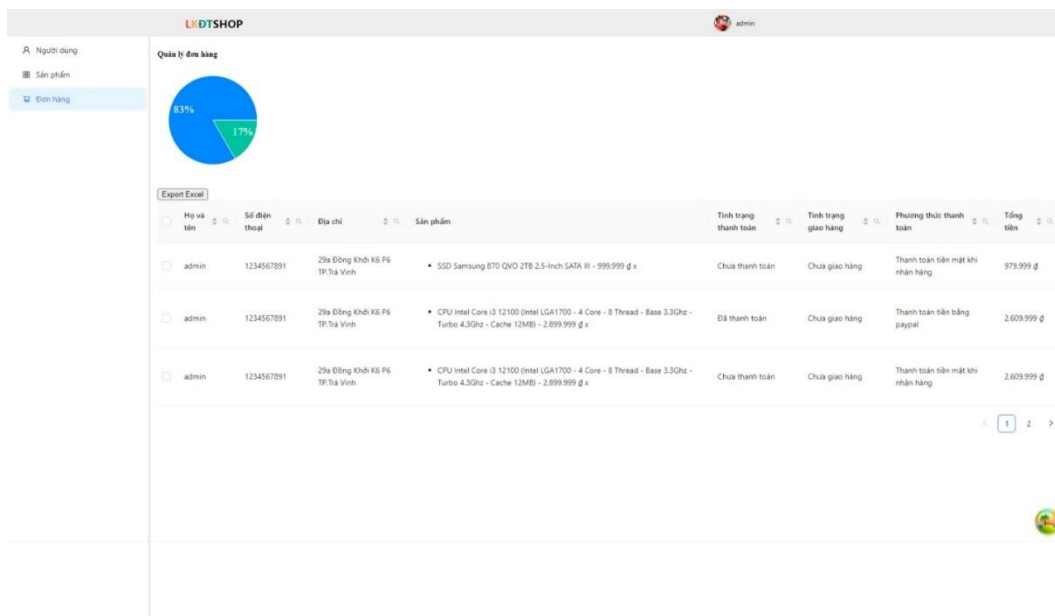
Đây là giao diện trang quản lý sản phẩm ở đây quản trị viên có thể xem, thêm, xóa, sửa và in ra danh sách các sản phẩm



Hình 4.24 Giao diện trang Quản lý sản phẩm

### 4.14 Giao diện trang quản lý đơn hàng

Đây là giao diện trang quản lý đơn hàng ở đây quản trị viên có thể thấy được các đơn hàng được khách hàng đặt



Hình 4.25 Giao diện trang Quản lý đơn hàng

## CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1. Kết luận

Trong đồ án "Xây dựng website bán linh kiện điện tử và tối ưu hoá trải nghiệm người dùng", tôi đã hoàn thành các mục tiêu đề ra và đạt được những kết quả quan trọng như sau:

- Tôi đã nghiên cứu và ứng dụng thành công các công nghệ React, NodeJS và MongoDB để xây dựng một website bán linh kiện điện tử.
- Giao diện người dùng (UI) và trải nghiệm người dùng (UX) được thiết kế thân thiện và dễ sử dụng, giúp người dùng dễ dàng tìm kiếm và mua sắm sản phẩm.
- Các chức năng cơ bản như đăng nhập/đăng ký, hiển thị danh sách sản phẩm, thêm vào giỏ hàng và thanh toán đã được triển khai hoàn chỉnh.
- Tính năng tìm kiếm và bộ lọc giúp người dùng tìm kiếm sản phẩm theo nhiều tiêu chí như giá, số sao và loại sản phẩm.
- Hệ thống giỏ hàng và thanh toán đa dạng, hỗ trợ nhiều phương thức thanh toán như thanh toán trực tuyến, thanh toán khi nhận hàng (COD).

### 5.2. Hướng phát triển

Đưa sản phẩm vào thực tế tiếp cận người dùng để có những đánh giá thực tế về trải nghiệm và đưa ra những ý kiến để sản phẩm có thể được cải thiện toàn diện hơn về giao diện và chức năng trong tương lai.

Nghiên cứu và phát triển thêm các tính năng nâng cao như hệ thống đánh giá và nhận xét sản phẩm từ người dùng, tích hợp AI để gợi ý sản phẩm, và tối ưu hóa hiệu suất website.

Khám phá và ứng dụng các công nghệ mới như GraphQL, TypeScript để cải thiện hiệu suất và bảo mật cho website.

Tăng cường khả năng tương tác người dùng qua các kênh truyền thông xã hội và ứng dụng di động.

Thêm chức năng xác nhận email của người dùng.

## DANH MỤC TÀI LIỆU THAM KHẢO

### Sách/Giáo trình:

[1] Phạm Minh Dương, Tài liệu giảng dạy môn Phân tích và thiết kế hệ thống thông tin (lưu hành nội bộ), trường ĐH Trà Vinh, 2014.

[2] Nguyễn Văn Ba, Phân tích và thiết kế hệ thống thông tin, NXB Đại học Quốc gia Hà Nội, 2009.

[3] Phan Thị Phương Nam, Tài liệu giảng dạy môn Hệ quản trị cơ sở dữ liệu (lưu hành nội bộ), trường ĐH Trà Vinh, 2014.

[4] Đoàn Phước Miên, Phạm Thị Trúc Mai, Tài liệu giảng dạy môn Thiết kế và lập trình web (lưu hành nội bộ), trường ĐH Trà Vinh, 2014.

### Website:

[1] React, 2024, <https://vi.legacy.reactjs.org/>

[2] React, 2024, <https://react.dev/>

[3] Tuong Uyen, NodeJS là gì, 04/12/2023, <https://itviec.com/blog/nodejs-la-gi/>

[4] Tuong Uyen, MongoDB là gì, 29/06/2023, <https://itviec.com/blog/mongodb-la-gi/>