BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**NIÊN LUẬN**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**XÂY DỰNG WEBSITE BÁN GIÀY**

**Sinh viên: Đào Minh Khoa**

**Mã số: B1809248**

**Khóa: K44**

**Cần Thơ, 12/2021**

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**NIÊN LUẬN**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**XÂY DỰNG WEBSITE BÁN GIÀY**

**Người hướng dẫn Sinh viên thực hiện**

Ths. Lê Hoàng ThảoĐào Minh Khoa

MSSV: B1809248

Khóa: K44

***Cần Thơ, 12/2021***

**LỜI CẢM ƠN**

Em xin chân thành cảm ơn quý thầy cô đã giúp đỡ em thực hiên bài niên luận này. Đặc biệt thầy **Lê Hoàng Thảo** đã tận tình giúp đỡ, hướng dẫn em tìm hiểu và nắm bắt các kiến thức mới, cùng với đó là vận dụng nó để khai thác đề tài và giải quyết vấn đề để hoàn thành bài niên luận này. Em cũng cám ơn các Bạn bè trong khoa CNTT & TT đã nhiệt tình chia sẻ kiến thức của mình cho em.

Em xin chân thành cám ơn!

**MỤC LỤC**

[PHẦN 1: GIỚI THIỆU 1](#_Toc90155154)

[I. ĐẶT VẤN ĐỀ 1](#_Toc90155155)

[II. MỤC TIÊU ĐỀ TÀI 1](#_Toc90155156)

[1. Mục tiêu về kiến thức 1](#_Toc90155157)

[2. Mục tiêu về sản phẩm 1](#_Toc90155158)

[III. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU 1](#_Toc90155159)

[IV. NỘI DUNG NGHIÊN CỨU 2](#_Toc90155160)

[PHẦN 2: NỘI DUNG 2](#_Toc90155161)

[1. Đặc tả yêu cầu 2](#_Toc90155162)

[2. Đặc tả usecase 2](#_Toc90155163)

[2.2 Mô tả chi tiết use case khách hàng. 2](#_Toc90155164)

[2.3 Sơ đồ use case quản trị 4](#_Toc90155165)

[3. Mô hình cơ sở dữ liệu và đặc tả cơ sở dữ liệu 4](#_Toc90155166)

[3.1 Mô hình cơ sở dữ liệu 4](#_Toc90155167)

[3.2 Đặc tả cơ sở dữ liệu 4](#_Toc90155168)

[PHẦN 3: CƠ SỞ LÝ THUYẾT 10](#_Toc90155169)

[I. NGÔN NGỮ PHÍA CLIENT 10](#_Toc90155170)

[1. HTML 10](#_Toc90155171)

[2. CSS 11](#_Toc90155172)

[3. Javascript 12](#_Toc90155173)

[4. ReactJS 12](#_Toc90155174)

[5. Thư viện axios 17](#_Toc90155175)

[II. NGÔN NGỮ PHÍA SERVER 17](#_Toc90155176)

[1. Cơ sở dữ liệu MySQL 17](#_Toc90155177)

[2. NodeJS 18](#_Toc90155178)

[3. ExpressJS 18](#_Toc90155179)

[PHẦN 4: XÂY DỰNG HỆ THỐNG 20](#_Toc90155180)

[I. XÂY DỰNG CHỨC NĂNG CHO NGƯỜI QUẢN LÝ 20](#_Toc90155181)

[1. Chức năng thêm sản phẩm 20](#_Toc90155182)

[2. Chức năng sửa sản phẩm 21](#_Toc90155183)

[3. Chức năng xem thông tin sản phẩm 22](#_Toc90155184)

[4. Chức năng xóa sản phẩm 22](#_Toc90155185)

[5. Chức năng xác nhận hóa đơn và hủy hóa đơn 22](#_Toc90155186)

[II. XÂY DỰNG CHỨC NĂNG CHO KHÁCH HÀNG 23](#_Toc90155187)

[1. Chức năng đăng ký 23](#_Toc90155188)

[2. Chức năng đăng nhập 23](#_Toc90155189)

[3. Chức năng xem thông tin sản phẩm 24](#_Toc90155190)

[4. Chức năng thêm thông tin khách hàng 24](#_Toc90155191)

[5. Chức năng tìm kiếm sản phẩm 25](#_Toc90155192)

[6. Chức năng lọc sản phẩm 25](#_Toc90155193)

[7. Chức năng thêm sản phẩm vào giỏ hàng 26](#_Toc90155194)

[8. Chức năng thanh toán khi nhận hàng 26](#_Toc90155195)

[9. Chức năng thanh toán qua vnpay 27](#_Toc90155196)

[10. Chức năng theo dõi đơn hàng 28](#_Toc90155197)

[PHẦN 5: KẾT LUẬN 29](#_Toc90155198)

[1. Kết quả đạt được 29](#_Toc90155199)

[2. Hướng phát triển 29](#_Toc90155200)

[3. Tài liệu tham khảo 29](#_Toc90155201)

**Danh Mục Bảng**

[Bảng 1: Nội dung nghiên cứu 2](#_Toc90150516)

[Bảng 2: Use case xem thông tin sản phẩm. 2](#_Toc90150517)

[Bảng 3: Use case nhập thông tin khách hàng. 2](#_Toc90150518)

[Bảng 4: Use case đăng nhập. 3](#_Toc90150519)

[Bảng 5: Use case mua sản phẩm 3](#_Toc90150520)

[Bảng 6: Use case thêm sản phẩm vào hàng 3](#_Toc90150521)

[Bảng 7: Bảng khách hàng 5](#_Toc90150522)

[Bảng 8: Bảng tài khoản 5](#_Toc90150523)

[Bảng 9: Bảng giỏ hàng 5](#_Toc90150524)

[Bảng 10: Bảng kho hàng 5](#_Toc90150525)

[Bảng 11: Bảng sản phẩm 6](#_Toc90150526)

[Bảng 12: Bảng loại sản phẩm 6](#_Toc90150527)

[Bảng 13: Bảng thương hiệu 6](#_Toc90150528)

[Bảng 14: Bảng kích thước sản phẩm 6](#_Toc90150529)

[Bảng 15: Bảng hình ảnh sản phẩm 7](#_Toc90150530)

[Bảng 16: Bảng hóa đơn nhập 7](#_Toc90150531)

[Bảng 17: Bảng hóa đơn xuất 7](#_Toc90150532)

[Bảng 18: Bảng chi tiết hóa đơn xuất 7](#_Toc90150533)

[Bảng 19: Danh sách một số status code thông dụng được trả về từ response 8](#_Toc90150534)

[Bảng 20: Một số phương thức (response methods) hỗ trợ hay dùng nhất 20](#_Toc90150535)

**Danh Mục Hình**

[Hình 1: Sơ đồ use case khách hàng. 2](#_Toc90151072)

[Hình 2: Sơ đồ use case quản lý 4](#_Toc90151073)

[Hình 3: Mô hình cơ sở dữ liệu 4](#_Toc90151074)

[Hình 4: Kiến trúc các thành phần của REST API 9](#_Toc90151075)

[Hình 5: Lưu đồ xử lý yêu cầu của REST API server 10](#_Toc90151076)

[Hình 6: Logo html 10](#_Toc90151077)

[Hình 7: Logo css 11](#_Toc90151078)

[Hình 8: Logo javascript 12](#_Toc90151079)

[Hình 9: Logo reactjs 12](#_Toc90151080)

[Hình 10: Function components hợp lệ 13](#_Toc90151081)

[Hình 11: Function components sử dụng arrow function 13](#_Toc90151082)

[Hình 12: Ví dụ class components 14](#_Toc90151083)

[Hình 13: Props 15](#_Toc90151084)

[Hình 14: State trong class components 15](#_Toc90151085)

[Hình 15: State trong function components 16](#_Toc90151086)

[Hình 16: LifeCycle trong ReactJS 16](#_Toc90151087)

[Hình 17: Logo mysql 17](#_Toc90151088)

[Hình 18: Logo nodejs 18](#_Toc90151089)

[Hình 19: Logo expressjs 18](#_Toc90151090)

[Hình 20: Ví dụ cấu trúc route 19](#_Toc90151091)

[Hình 21: Một số phương thức (response methods) hỗ trợ hay dùng nhất 20](#_Toc90151092)

[Hình 22: Giao diện thêm sản phẩm 21](#_Toc90151093)

[Hình 23: Giao diện sửa sản phẩm 21](#_Toc90151094)

[Hình 24: Giao diện xem thông tin sản phẩm 22](#_Toc90151095)

[Hình 25: Giao diện xóa sản phẩm 22](#_Toc90151096)

[Hình 26: Giao diện xác nhận đơn và hủy đơn hàng 22](#_Toc90151097)

[Hình 27: Giao diện đăng ký 23](#_Toc90151098)

[Hình 28: Giao diện đăng nhập 24](#_Toc90151099)

[Hình 29: Giao diện xem thông tin sản phẩm 24](#_Toc90151100)

[Hình 30: Giao diện thêm thông tin khách hàng 24](#_Toc90151101)

[Hình 31: Giao diện tìm kiếm sản phẩm 25](#_Toc90151102)

[Hình 32: Giao diện lọc sản phẩm 25](#_Toc90151103)

[Hình 33: Giao diện thêm sản phẩm vào giỏ hàng 26](#_Toc90151104)

[Hình 34: Giao diện thanh toán khi nhận hàng 26](#_Toc90151105)

[Hình 35: Giao diện thanh toán qua vnpay 27](#_Toc90151106)

[Hình 36: Giao diện chọn hình thức thanh toán 27](#_Toc90151107)

[Hình 37: Giao diện nhập thông tin thanh toán 28](#_Toc90151108)

[Hình 38: Giao diện đơn hàng 28](#_Toc90151109)

# PHẦN 1: GIỚI THIỆU

## I. ĐẶT VẤN ĐỀ

Hiện nay, việc kinh doanh kết hợp với công nghệ thông tin đang là xu hướng của thế giới. Đặc biệt, tình hình Covid diễn ra khá phức tạp ở nước ta đòi hỏi người dân phải hạn chế ra ngoài. Cho nên việc số lượng khách mua sắm tại các cửa hàng cũng giảm xuống. Vì vậy, việc kinh doanh bán hàng cần sử dụng internet để quảng bá đến mọi người ngay tại ngôi nhà của họ, mặc dù khách hàng không ra ngoài nhưng vẫn có thể đi shopping, mua những yếu phẩm cần thiết cho mình. Vì thế, việc tạo ra một website bán điện thoại để giúp khách hàng mua sắm và cập nhật tin tức tại nhà là điều rất cần thiết.

## II. MỤC TIÊU ĐỀ TÀI

### 1. Mục tiêu về kiến thức

* Nắm được kiến thức cơ bản của HTML, CSS, Javascript, MySQL, NodeJS.
* Tìm hiểu thư viên ReatJS, framework ExpressJS và xây dựng website.

### 2. Mục tiêu về sản phẩm

* Xây dựng và lập trình được website bán giày.
* Quản lý, thống kê được sản phẩm, số lượng sản phẩm.
* Tạo các ràng buộc cho website.
* Phân tích thiết kế được cơ sở dữ liệu.
* Truy xuất cơ sở dữ liệu.
* Tùy chỉnh thêm, sửa, xóa sản phẩm.
* Ứng dụng ReactJS và framework ExpressJS vào website.

## III. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU

- Đối tượng nghiên cứu chính của đề tài là sử dụng ReactJS và ExpressJS để xây dựng website bán giày. Ngoài ra, cũng sẽ nghiên cứu thêm về việc xây dựng một kiến trúc hệ thống ứng dụng sử dụng máy chủ dữ liệu (API server) và công cụ quản lý (Web client). Đối tượng nghiên cứu đầy đủ sẽ là một hệ thống gồm 2 thành phần:

- REST API server: xây dựng một server cung cấp các thao tác tạo, thêm, sửa, xóa (CRUD) dữ liệu của hệ thống thông qua HTTP request.

- Web client: xây dựng một ứng dụng web giao tiếp với server thông qua REST API, nhằm mục đích quản lý dữ liệu cốt lõi của hệ thống.

## IV. NỘI DUNG NGHIÊN CỨU

|  |  |
| --- | --- |
| **STT** | **Nội dung** |
| **1** | Tìm hiểu về API và RESTful API. |
| **2** | Tìm hiểu NodeJS với framework Express.  Xây dựng model dữ liệu ( các bảng của cơ sở dữ liệu) của đề tài. |
| **3** | Tìm hiểu hệ quản trị cơ sở dữ liệu MySQL.  Cấu hình MySQL trong NodeJS. |
| **4** | Tìm hiểu về ReactJS: components, state và props.  Sử dụng webhook trong react.  Routing bằng react-router-dom và chuyển hướng đăng nhập. |
| **5** | Tìm hiểu về axios.  Trao đổi với API server thông qua axios. |
| **6** | Tìm hiểu về AntDesign và các components mà thư viện cung cấp. |

Bảng 1: Nội dung nghiên cứu

# 

# PHẦN 2: NỘI DUNG

### 1. Đặc tả yêu cầu

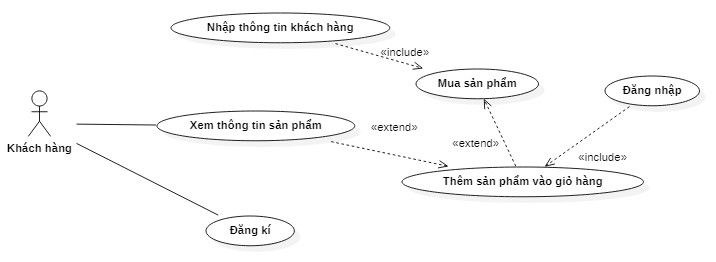
Website có 2 nhiệm vụ chính là giới thiệu sản phẩm và chức năng đặt hàng thông qua giao diện thân thiện, dễ sử dụng.

Hệ thống cho phép người dùng tương qua giao diện website. Sau khi truy cập website, khách hàng có thể chọn sản phẩm, xem thông tin chi tiết của sản phẩm như: tên sản phẩm, hình sản phẩm, giá sản phẩm, kích thước và số lượng sản phẩm. khách hàng có thể thêm sản phẩm vào giỏ hàng. Để thêm sản phẩm vào giỏ hàng, khách hàng cần đăng nhập tài khoản. Nếu chưa có tài khoản, khách hàng có thể đăng ký tài khoản gồm: tên đăng nhập và mật khẩu. Sau đó, khách hàng sẽ được cung cấp cho một tài khoản để sử dụng. khi mua sản phẩm, khách hàng cần cung cấp thông tin của khách hàng bao gồm: họ tên, số điện thoại, giới tính, email, địa chỉ. Để thuận lợi cho việc mua hàng, khách hàng có thể thêm thông tin trước trong phần hồ sơ. Ngoài ra, khách hàng có thể thanh toán trực tiếp qua cổng thanh toán thương mại điện tử VNpay.

Quản trị viên sẽ tương tác trên giao diện dành cho quản trị viên. Hệ thống cho phép quản trị viên quản lý được số lượng sản phẩm được nhập vào, thêm sản phẩm, sửa và xóa sản phẩm. Nhận thông tin khách hàng mua sản phẩm, quản lý danh sách khách hàng, thông tin hóa đơn của khách hàng, thống kê sản số hóa đơn, doanh thu và sản phẩm bán chạy nhất.

### 2. Đặc tả usecase

#### 2.1 Usecase khách hàng



Hình 1: Sơ đồ use case khách hàng.

### 2.2 Mô tả chi tiết use case khách hàng.

#### 2.2.1 Xem thông tin sản phẩm.

|  |  |
| --- | --- |
| **Tên use case** | Xem thông tin sản phẩm |
| **Mô tả** | Cho phép khách hàng xem thông tin chi tiết của một sản phẩm như: tên, giá, cấu hình, hiệu năng,… |
| **Tiền điều kiện** |  |
| **Tác nhân chính** | Khách hàng |
| **Cách kích hoạt** | Click vào sản phẩm trên giao diện website |

Bảng 2: Use case xem thông tin sản phẩm.

#### 2.2.2 Nhập thông tin khách hàng.

|  |  |
| --- | --- |
| **Tên use case** | Nhập thông tin khách hàng |
| **Mô tả** | Cho phép khách hàng nhập thông tin cá nhân để mua hàng |
| **Tiền điều kiện** | Đã đăng ký tài khoản |
| **Tác nhân chính** | Khách hàng |
| **Cách kích hoạt** | Nhấn vào biểu tượng người dùng và chọn hồ sơ |

Bảng 3: Use case nhập thông tin khách hàng.

#### 2.2.3 Đăng nhập.

|  |  |
| --- | --- |
| **Tên use case** | Đăng nhập |
| **Mô tả** | Cho phép người dùng đăng nhập tài khoản của mình vào website |
| **Tiền điều kiện** | Đã đăng kí tài khoản |
| **Tác nhân chính** | Khách hàng |
| **Cách kích hoạt** | Cần có tài khoản và mật khẩu để tiến hành đăng nhập |

Bảng 4: Use case đăng nhập.

#### 2.2.4 Mua sản phẩm.

|  |  |
| --- | --- |
| **Tên use case** | Mua sản phẩm |
| **Mô tả** | Cho phép khách hàng mua sản phẩm từ website |
| **Tiền điều kiện** | Đã đăng nhập tài khoản |
| **Tác nhân chính** | Khách hàng |
| **Cách kích hoạt** | Click vào nút mua và điền thông tin khách hàng |

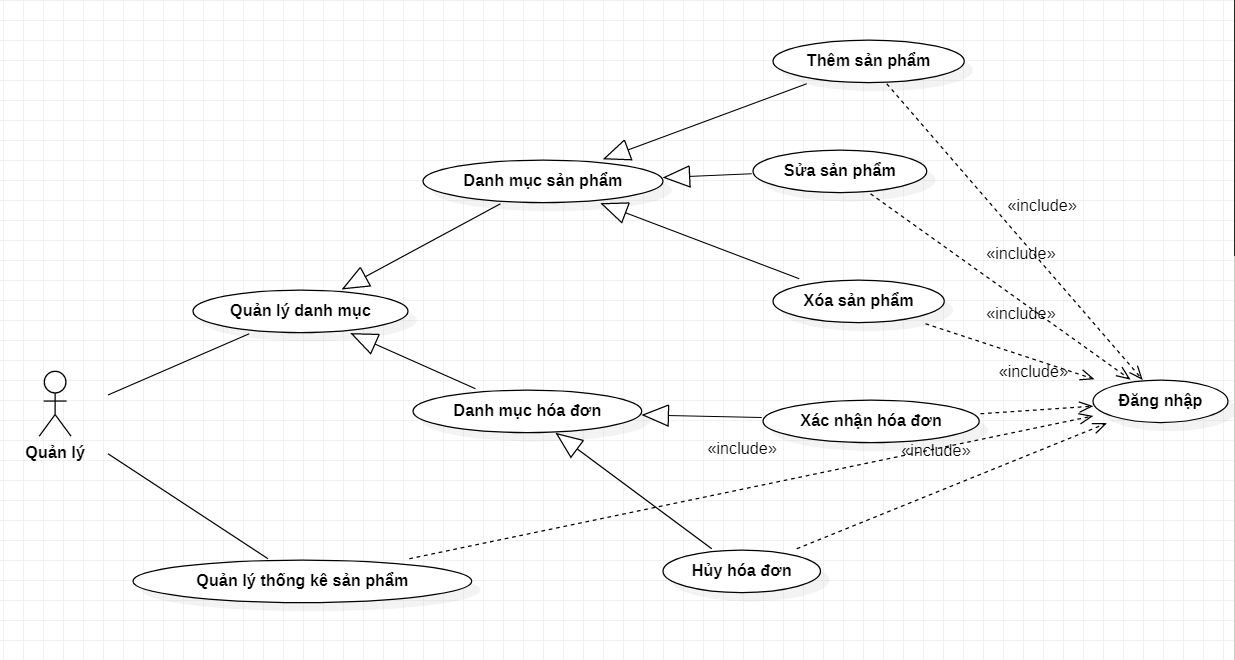
Bảng 5: Use case mua sản phẩm

#### 2.2.5 Thêm sản phẩm vào giỏ hàng.

|  |  |
| --- | --- |
| **Tên use case** | Thêm sản phẩm vào giỏi hàng |
| **Mô tả** | Cho phép khách hàng lưu trữ sản phẩm tại giỏ hàng tránh tìm kiếm lần 2 |
| **Tiền điều kiện** | Khách hàng đã đăng nhập |
| **Tác nhân chính** | Khách hàng |
| **Cách kích hoạt** | Click vào nút giỏ hàng tại sản phẩm cần thêm |

Bảng 6: Use case thêm sản phẩm vào hàng

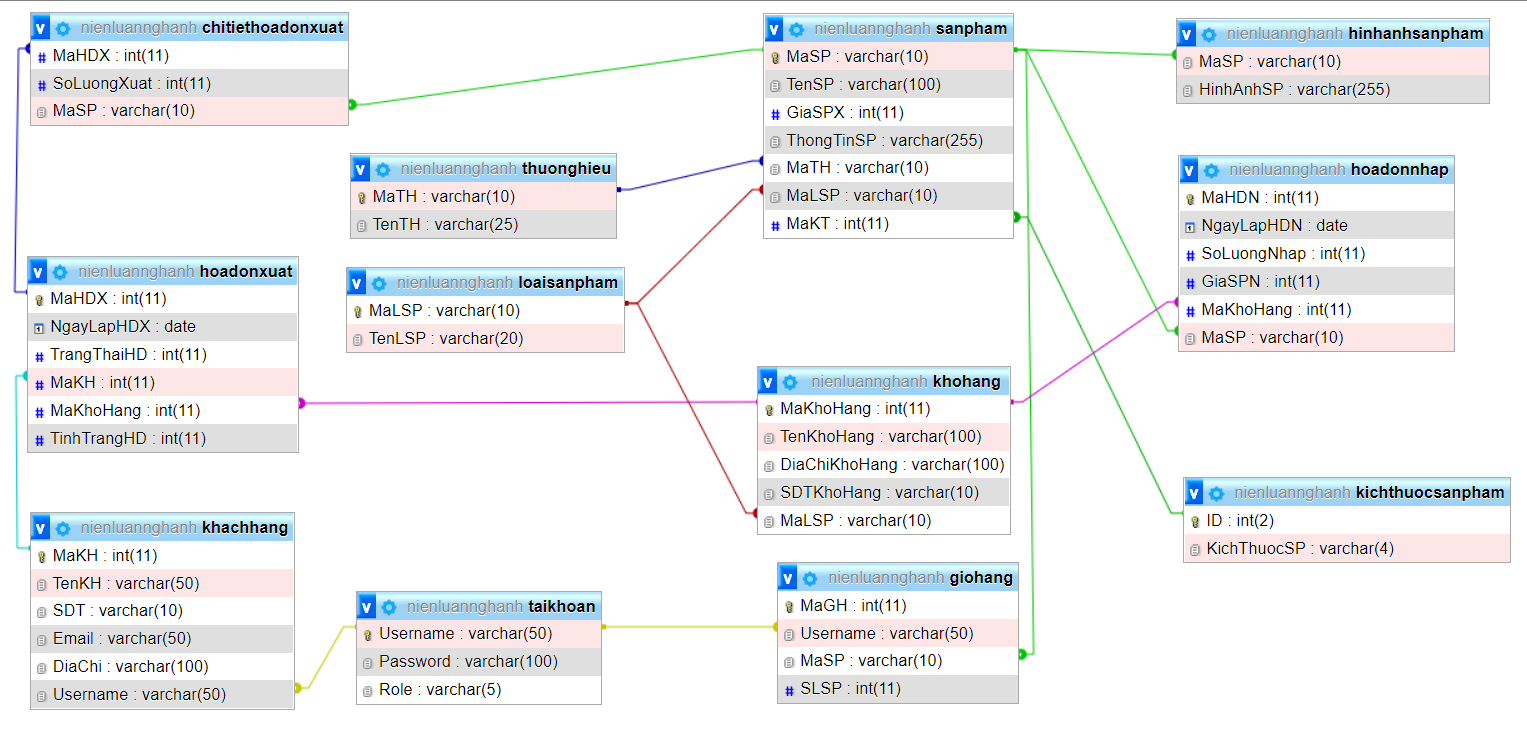
### 2.3 Sơ đồ use case quản trị



Hình 2: Sơ đồ use case quản lý

## 3. Mô hình cơ sở dữ liệu và đặc tả cơ sở dữ liệu

### 3.1 Mô hình cơ sở dữ liệu

****

Hình 3: Mô hình cơ sở dữ liệu

### 3.2 Đặc tả cơ sở dữ liệu

**- Bảng khách hàng**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KhachHang** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaKH | Int(11) | x | x | Mã khách hàng |
| 2 | TenKH | Varchar(50) |  |  | Tên khách hàng |
| 3 | SDT | Varchar(10) |  |  | Số điện thoại khách hàng |
| 4 | Email | Varchar(50) |  |  | Email khách hàng |
| 5 | DiaChi | Varchar(100) |  |  | Địa chỉ khách hàng |

Bảng 7: Bảng khách hàng

**- Bảng tài khoản**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TaiKhoan** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | Username | Varchar(50) | x | x | Tên tài khoản |
| 2 | Password | Varchar(100) |  |  | Mật khẩu tài khoản |
| 3 | Role | Varchar(5) |  |  | Quyền của tài khoản |

Bảng 8: Bảng tài khoản

**- Bảng giỏ hàng**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **GioHang** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaGH | Int(11) | x | x | Mã giỏ hàng |
| 2 | SLSP | Int(11) |  |  | Số lượng sản phẩm |

Bảng 9: Bảng giỏ hàng

**- Bảng kho hàng**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KhoHang** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaKhoHang | Int(11) | x | x | Mã kho hàng |
| 2 | TenKhohang | Varchar(100) |  |  | Tên kho hàng |
| 3 | DiaChiKhoHang | Varchar(100) |  |  | Địa chỉ kho hàng |
| 4 | SDTKhoHang | Varchar(10) |  |  | Số điện thoại kho hàng |

Bảng 10: Bảng kho hàng

**- Bảng sản phẩm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SanPham** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaSP | Varchar(10) | x | x | Mã sản phẩm |
| 2 | TenSP | Varchar(100) |  |  | Tên sản phẩm |
| 3 | GiaSPX | Int(10) |  |  | Giá sản phẩm xuất |
| 4 | ThongTinSP | Varchar(255) |  |  | Thông tin sản phẩm |

Bảng 11: Bảng sản phẩm

**- Bảng loại sản phẩm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LoaiSanPham** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaLSP | Varchar(10) | x | x | Mã loại sản phẩm |
| 2 | TenLSP | Varchar(20) |  |  | Tên loại sản phẩm |

Bảng 12: Bảng loại sản phẩm

**- Bảng thương hiệu**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ThuongHieu** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaTH | Int(11) | x | x | Mã thương hiệu |
| 2 | TenTH | Varchar(100) |  |  | Tên thương hiệu |

Bảng 13: Bảng thương hiệu

**- Bảng kích thước sản phẩm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KichThuocSanPham** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | ID | Int(11) | x | x | Mã kích thước |
| 2 | KichThuocSP | Varchar(100) |  |  | Kích thước sản phẩm |

Bảng 14: Bảng kích thước sản phẩm

**- Bảng hình ảnh sản phẩm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **HinhAnhSanPham** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | HinhAnhSP | Int(11) |  |  | Hình ảnh sản phẩm |

Bảng 15: Bảng hình ảnh sản phẩm

**- Bảng hóa đơn nhập**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **HoaDonNhap** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaHDN | Int(11) | x | x | Mã hóa đơn nhập |
| 2 | NgayLapHDN | Date |  |  | Ngày lập hóa đơn nhập |
| 3 | SoLuongNhap | Int(11) |  |  | Sô lượng nhập |
| 4 | GiaSPN | Int(11) |  |  | Giá sản phẩm nhập |

Bảng 16: Bảng hóa đơn nhập

**- Bảng hóa đơn xuất**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **HoaDonXuat** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | MaHDX | Int(11) | x | x | Mã hóa đơn xuất |
| 2 | NgayLapHDX | Varchar(100) |  |  | Ngày lập hóa đơn xuất |
| 3 | TrangThaiHD | Varchar(100) |  |  | Trạng thái hóa đơn |
| 4 | TinhTrangHD | Varchar(10) |  |  | Tình trạng hóa đơn |

Bảng 17: Bảng hóa đơn xuất

**Bảng chi tiết hóa đơn xuất**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KhoHang** | | | | | |
| **STT** | **Tên trường** | **Kiểu** | **Khóa chính** | **Khóa ngoại** | **Mô tả** |
| 1 | SoLuongXuat | Int(11) |  |  | Số lượng xuất |

Bảng 18: Bảng chi tiết hóa đơn xuất

**Định dạng URI**

Các endpoint của REST API chính là URI. Định dạng URI chung cho các tác vụ CRUD được thiết kế theo quy tắc /api/model/.

Trong đó:

− model: là danh từ chỉ model sẽ được thực hiện tác vụ CRUD. Ví dụ như thao tác CRUD với user thì model lúc này là “user”.

Các request gửi tới các endpoint có thể kèm theo header là những dữ liệu định dạng json hoặc form-multipart là access token.

Ví dụ để tạo một cặp access và refresh token thì thông tin cần thiết của một request bao gồm:

− URI: [host-address:port]/api/

− HTTP method: POST

− Content Type: JSON

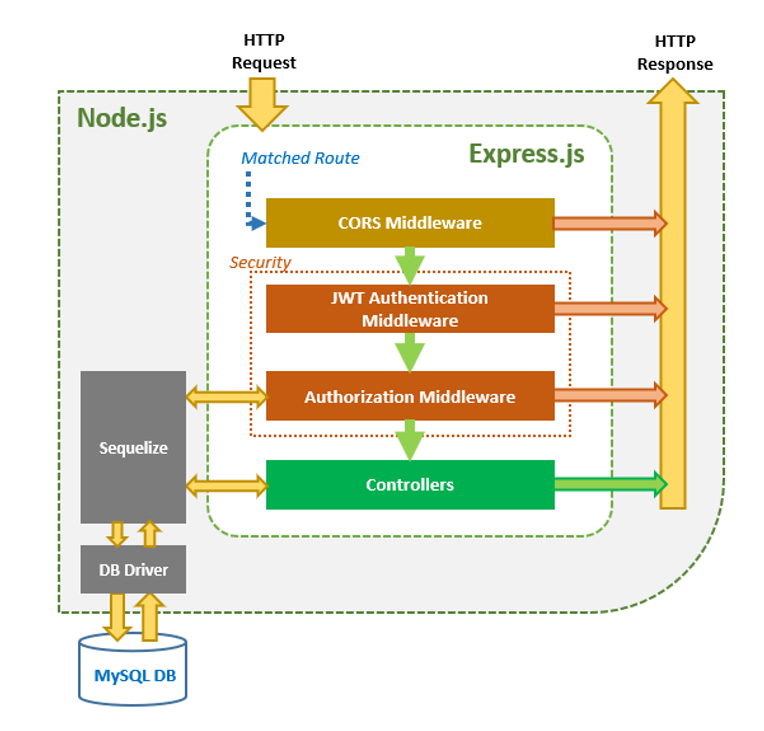
− Headers: { “x-access-token”: “token” }

Dưới đây là danh sách một số status code thông dụng được trả về từ response:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Code** | **TextCode** | **Mô tả** |
| 1 | 200 | OK | Thành công |
| 2 | 201 | Created | Đã tạo ra một tài nguyên thành công |
| 3 | 204 | No Content | Thành công nhưng response không chứa nội dung |
| 4 | 400 | Bad Request | Lỗi server không hiểu yêu cầu |
| 5 | 401 | Bad Request | Lỗi yêu cầu không được xác thực hợp lệ |
| 6 | 403 | Forbidden | Lỗi yêu cầu không được phép |
| 7 | 404 | Not Found | Lỗi yêu cầu không được tìm thấy |
| 8 | 405 | Method Not Allow | Lỗi server từ chối tiếp nhận loại phương thức HTTP cụ thể |
| 9 | 500 | Server Internal Error | Lỗi không mong muốn trên server nhằm từ chối việc thực hiện yêu cầu |

Bảng 19: Danh sách một số status code thông dụng được trả về từ response

**Kiến trúc các thành phần của REST API server**



Hình 4: Kiến trúc các thành phần của REST API

Sử dụng Nodejs và ExpressJS framework để xây dựng REST API và dùng JWT để xác thực cấp quyền cho request. Chuẩn dữ liệu để trao đổi giữa server và client là JSON.

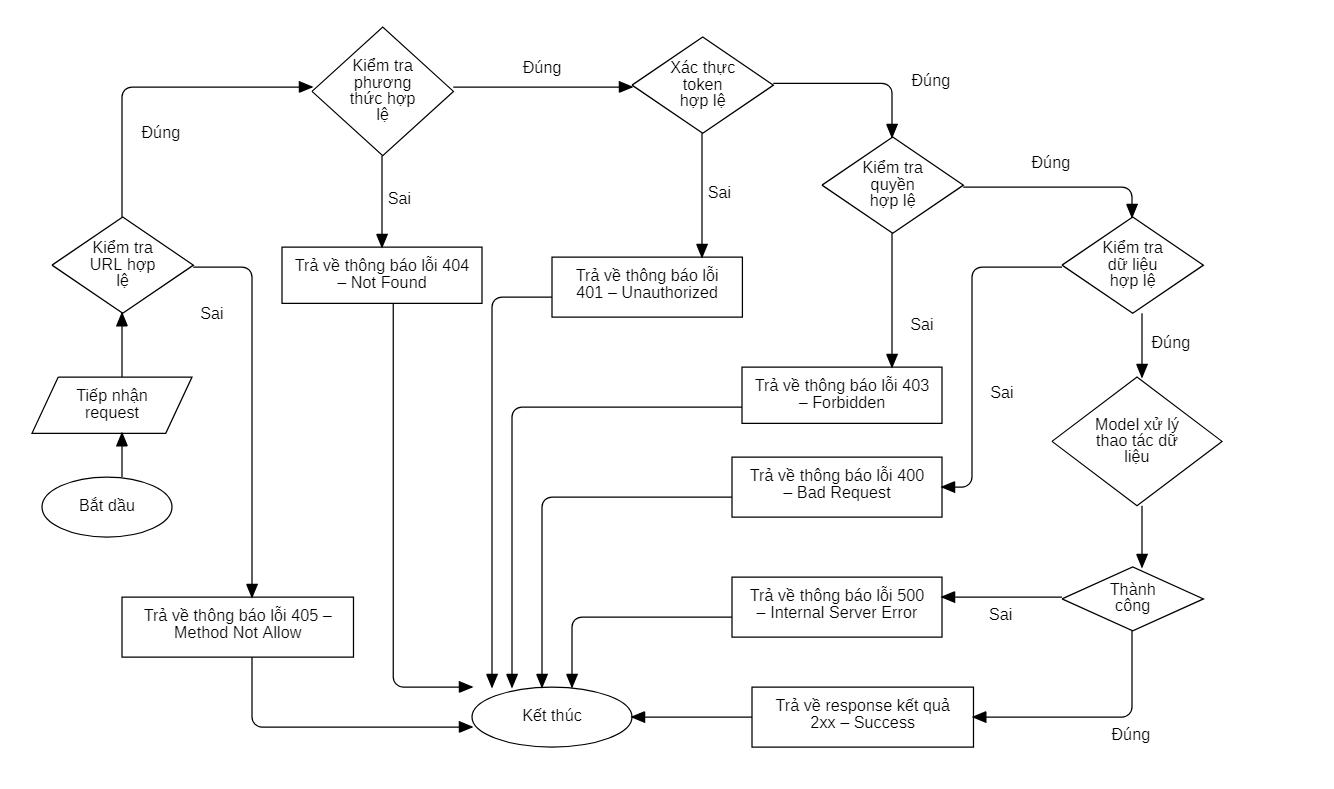
Với các bộ định tuyến(routes) của Express, **HTTP request** sẽ khớp với bộ định tuyến thông qua **CORS Middleware** trước khi đến lớp bảo mật(security)

Lớp bảo mật bao gồm:

- JWT Authentication Middleware: verify Sign Up, verify token.

- Authorization Middleware: check IsAdmin .

Nếu **Middleware** gặp bất cứ lỗi nào thông báo sẽ được gửi dưới dạng phản hồi HTTP (HTTP response).



Hình 5: Lưu đồ xử lý yêu cầu của REST API server

# PHẦN 3: CƠ SỞ LÝ THUYẾT

## I. NGÔN NGỮ PHÍA CLIENT

### 1. HTML

****

Hình 6: Logo html

HTML (tiếng Anh, viết tắt cho HyperText Markup Language, hay là "Ngôn ngữ Đánh dấu Siêu văn bản") là một ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web với các mẩu thông tin được trình bày trên World Wide Web. Cùng với CSS và JavaScript, HTML tạo ra bộ ba nền tảng kỹ thuật cho World Wide Web. HTML được định nghĩa như là một ứng dụng đơn giản của SGML và được sử dụng trong các tổ chức cần đến các yêu cầu xuất bản phức tạp. HTML đã trở thành một chuẩn Internet do tổ chức World Wide Web Consortium (W3C) duy trì. Phiên bản chính thức mới nhất của HTML là HTML 4.01 (1999). Sau đó, các nhà phát triển đã thay thế nó bằng XHTML.

### 2. CSS

****

Hình 7: Logo css

CSS là tập tin định kiểu theo tầng – dịch từ tiếng Anh là Cascading Style Sheets (CSS) – được dùng để miêu tả cách trình bày các tài liệu viết bằng ngôn ngữ HTML và XHTML. Ngoài ra ngôn ngữ định kiểu theo tầng cũng có thể dùng cho XML, SVG, XUL. Các đặc điểm kỹ thuật của CSS được duy trì bởi World Wide Web Consortium (W3C).

Hạn chế tối thiểu việc làm rối mã [HTML](https://vi.wikipedia.org/wiki/HTML) của trang [Web](https://vi.wikipedia.org/wiki/World_Wide_Web) bằng các thẻ quy định kiểu dáng (chữ đậm, chữ in nghiêng, chữ có gạch chân, chữ màu), khiến mã nguồn của trang Web được gọn gàng hơn, tách nội dung của trang [Web](https://vi.wikipedia.org/wiki/World_Wide_Web) và định dạng hiển thị, dễ dàng cho việc cập nhật nội dung.

Tạo ra các kiểu dáng có thể áp dụng cho nhiều trang Web, giúp tránh phải lặp lại việc định dạng cho các trang [Web](https://vi.wikipedia.org/wiki/World_Wide_Web) giống nhau.

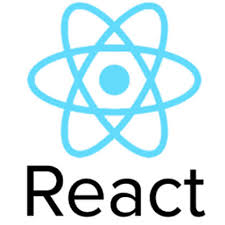
### 3. Javascript

****

Hình 8: Logo javascript

JavaScript, theo phiên bản hiện hành, là một ngôn ngữ lập trình kịch bản dựa trên đối tượng được phát triển từ các ý niệm nguyên mẫu. Ngôn ngữ này được dùng rộng rãi cho các trang web, nhưng cũng được dùng để tạo khả năng viết script sử dụng các đối tượng nằm sẵn trong các ứng dụng. Nó vốn được phát triển bởi Brendan Eich tại Hãng truyền thông Netscape với cái tên đầu tiên Mocha, rồi sau đó đổi tên thành LiveScript, và cuối cùng thành JavaScript. Giống Java, JavaScript có cú pháp tương tự C, nhưng nó gần với Self hơn Java. .js là phần mở rộng thường được dùng cho tập tin mã nguồn JavaScript. Phiên bản mới nhất của JavaScript là phiên bản 1.5, tương ứng với ECMA-262 bản 3. ECMAScript là phiên bản chuẩn hóa của JavaScript. Trình duyệt Mozilla phiên bản 1.8 beta 1 có hỗ trợ không đầy đủ cho E4X - phần mở rộng cho JavaScript hỗ trợ làm việc với XML, được chuẩn hóa trong ECMA-357.

### 4. ReactJS

****

Hình 9: Logo reactjs

ReactJS (hay React, React.js) là một thư viện mã nguồn mở (MIT License) dùng để xây dựng giao diện người dùng được viết bằng JavaScript. React được phát hành lần đầu tiên vào năm 2013 bởi Facebook. React hiện nay được duy trì bởi Facebook và cộng đồng lập trình viên. React được sử dụng bởi những doanh nghiệp công nghệ hàng đầu như Facebook, Twitter, Instagram.

Sức mạnh của React nằm ở việc tự động cập nhật lại UI (user interfaces) khi dữ liệu thay đổi với chi phí ít nhất mà không cần phải tải lại toàn bộ trang web. Điều này làm cho ứng dụng phản hồi lại nhanh hơn, tiết kiệm băng thông, tăng tốc ứng dụng và tăng trải nghiệm người dùng. Trong React sẽ có một số khái niệm cơ bản về JSX, components, props, state và hook.

##### 4.1 JavaScript XML

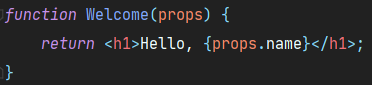
**JSX** (JavaScript XML) là một cú pháp mở rộng của JavaScript cho phép định nghĩa các thành phần HTML trong React. Nói một cách đơn giản, cú pháp này cho phép ta viết các HTML tag trực tiếp trong JavaScript.

Trên thực tế, các trình duyệt ngay cả các trình duyệt mới nhất cũng không hỗ trợ cú pháp của JSX. Do đó mã nguồn sử dụng JSX cần được chuyển về JavaScript thông qua một thư viện có tên là Babel (một JavaScript compiler).

#### 4.2 Component

**Components** là những thành phần UI được chia nhỏ ra, độc lập và có thể tái sử dụng. Component có thể là những function (stateless) hoặc class (stateful) trong JS.

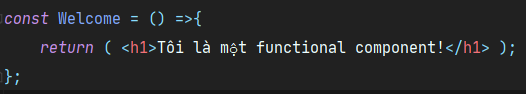
**Functional component:** là một hàm Javascript (hoặc ES6) trả về một React element. Theo tài liệu chính thức của React, hàm dưới đây là một component hợp lệ.

****

Hình 10: Function components hợp lệ

Function này là một component React hợp lệ vì nó nhận một "props" làm tham số và trả về 1 React element.

Hoặc theo ES6 arrow function:



Hình 11: Function components sử dụng arrow function

**Functional component** cũng được biết tới với cái tên là stateless components. Bởi vì chúng không thể làm nhiều thứ phức tạp như quản lý React State (data) hoặc xử lý vấn đề liên quan tới life-cycle trong functional components.

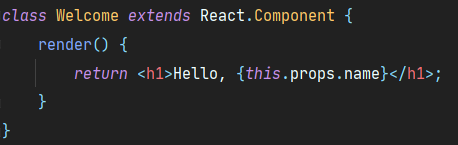
Tuy nhiên, từ phiên bản React 16.8, nhà phát hành giới thiệu tính năng React Hooks. Với Hooks, chúng ta có thể sử dụng state và những features khác trong functional components.

**Class components:** Các Class components là những class ES6. Chúng phức tạp hơn functional components.

Class components còn có:

- Phương thức khởi tạo, có hàm về vòng đời component, hàm render().

- State (dữ liệu ứng dụng).



Hình 12: Ví dụ class components

- Tóm lại, một class components là:

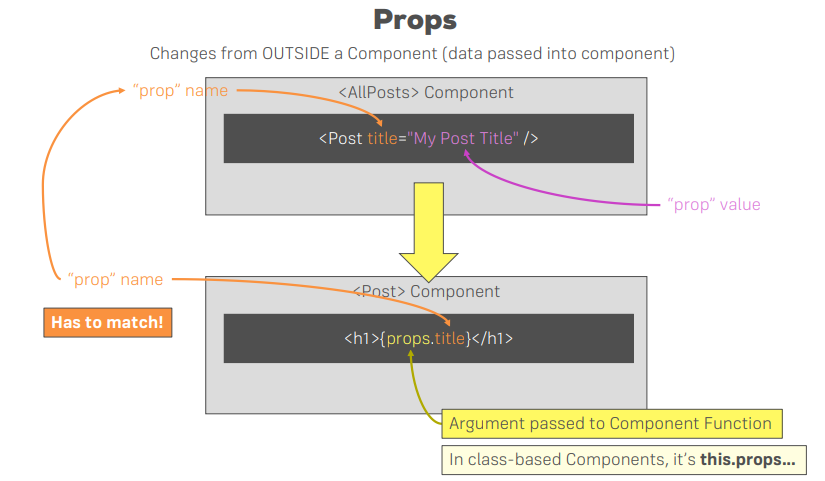
Là một class kế thừa từ React.Component.

Có thể nhận props (trong hàm khởi tạo) nếu cần.

Phải có hàm render() và trong đó trả về 1 React element hoặc NULL.

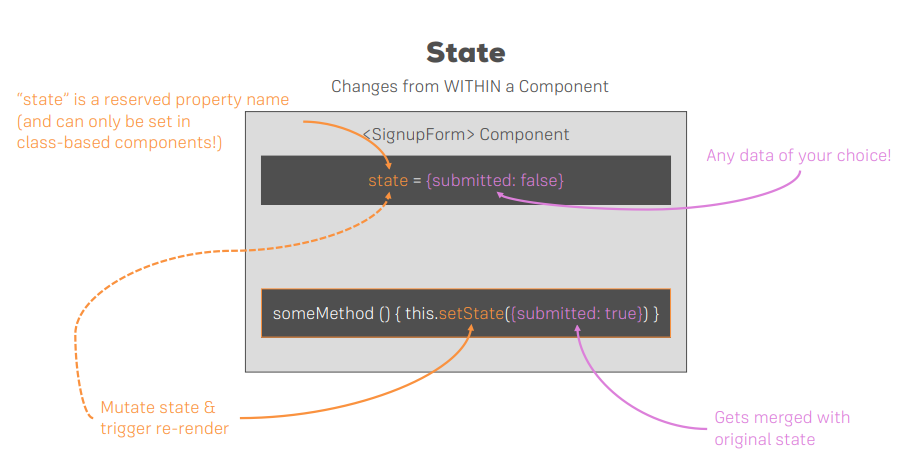
#### 4.3 Props và state

**Props** là những thuộc tính được truyền vào một component và chỉ có thể đọc. Ví dụ như thẻ Post có thuộc tính được truyền vào là title. Truy xuất bằng cú pháp props.title sẽ cho giá trị là “My Post Title”.

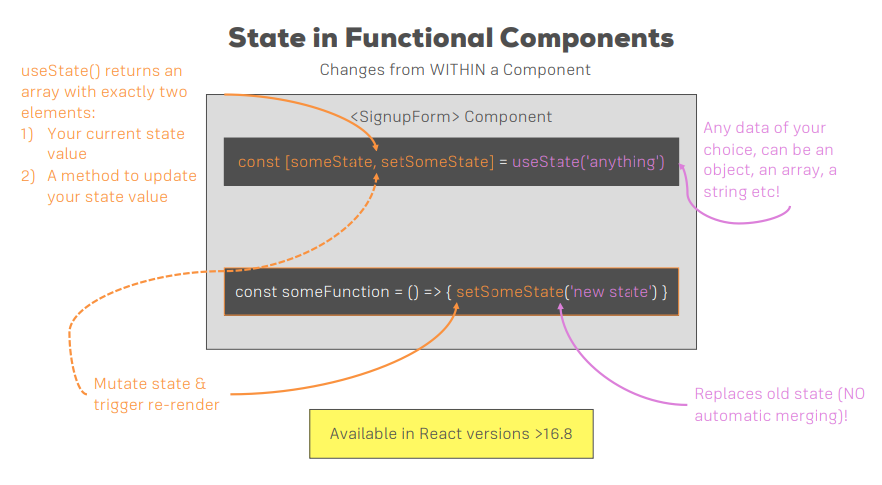
****

Hình 13: Props

**State** là trạng thái thuộc về chính component đó, được quản lý bởi chính nó và không được truy xuất từ bên ngoài. Chỉ có thể sử dụng state khi dùng stateful component.



Hình 14: State trong class components



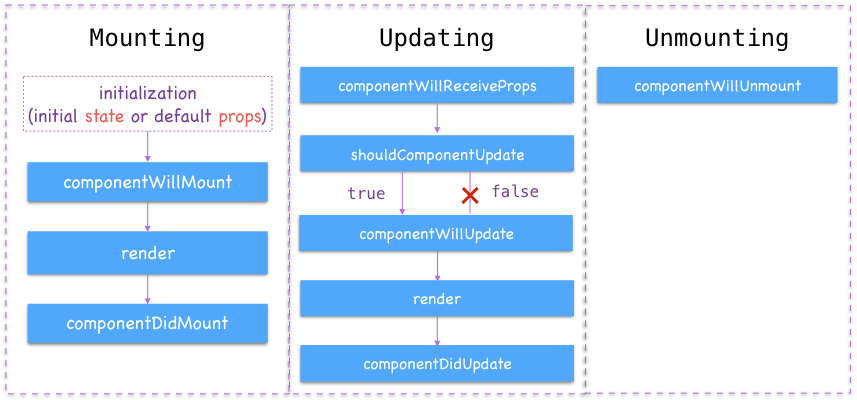
Hình 15: State trong function components

#### 4.4 Lifecycle

**LifeCycle** là một vòng đời của một React component từ lúc được render lần đầu tiên và mỗi lần render lại (mounting) và khi gỡ bỏ component (unmounting).

Có 2 phương thức được tự động gọi khi sự kiện mounting (component được render lần đầu tiên hoặc được render lại) và unmounting (component bị gỡ bỏ) xảy ra lần lượt là componentDidMount và componentWillUnmount.

Có thể ghi đè 2 phương thức này khi sử dụng stateful component (class).



Hình 16: LifeCycle trong ReactJS

#### 4.5 Hook

**Hook** là tính năng được thêm vào React ở phiên bản 16.8. Cho phép sử dụng một số tính năng chỉ có ở stateful component (class) khi dùng stateless component (function) như state (useState), life cycle (useEffect),...

### 5. Thư viện axios

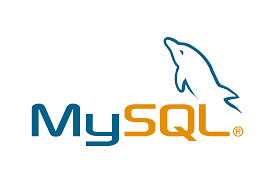
Axios là một thư viện HTTP client dựa trên Promise dùng để gửi các request HTTP bất đồng bộ đến REST endpoint để sử dụng các dịch vụ CRUD.

Promise là một cải tiến để loại bỏ try catch, callback rườm rà trong xử lý bất đồng bộ, thay vào đó là then – catch. Hiểu đơn giản là khi thực hiện một tác vụ bất đồng bộ, sau khi thực hiện thành công thì “then” sẽ được gọi, ngược lại khi có lỗi phát sinh thì “catch” sẽ được gọi. Tương tự như promise, axios cũng có thể sử dụng theo kiểu gửi request lồng nhau.

Ngoài ra, axios còn hỗ trợ interceptor dùng để thực hiện một số công việc khác trước khi gửi request hoặc ngay khi nhận được response. Ví dụ như tạo mới access token khi nhận được response thông báo lỗi “401 – Unauthorized”.

## II. NGÔN NGỮ PHÍA SERVER

### 1. Cơ sở dữ liệu MySQL

****

Hình 17: Logo mysql

MySQL là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng. Vì MySQL là cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh. Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên internet. MySQL miễn phí hoàn toàn cho nên bạn có thể tải về MySQL từ trang chủ. Nó có nhiều phiên bản cho các hệ điều hành khác nhau: phiên bản Win32 cho các hệ điều hành dòng Windows, Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS,…

### 2. NodeJS

****

Hình 18: Logo nodejs

[NodeJS](https://quantrimang.com/NodeJS) là một nền tảng Server side được xây dựng dựa trên Javascript Engine Node.js được phát triển bởi Ryan Dahl năm 2009 và phiên bản cuối cùng là v0.10.36. Định nghĩa NodeJs bởi tài liệu chính thức như sau:

Nền tảng Node.js dựa trên Chrome Javascript runtime để xây dựng các ứng dụng nhanh, có độ lớn. Node.js sử dụng các phần phát sinh các sự kiện (event-driven), mô hình non-blocking I/O để tạo ra các ứng dụng nhẹ và hiệu quả cho các ứng dụng về dữ liệu thời gian thực chạy trên các thiết bị phân tán.

NodeJs là một mã nguồn mở, đa nền tảng cho phát triển các ứng dụng phía Server và các ứng dụng liên quan đến mạng. Ứng dụng Node.js được viết bằng Javascript và có thể chạy trong môi trường Node.js trên hệ điều hành Window, Linux...

Node.js cũng cung cấp cho chúng ta các module Javascript đa dạng, có thể đơn giản hóa sự phát triển của các ứng dụng web sử dụng Node.js với các phần mở rộng.

### 3. ExpressJS



Hình 19: Logo expressjs

**Expressjs** là một framework được xây dựng trên nền tảng của **Nodejs**. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. **Expressjs** hỗ trợ các method HTTP và middleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng.**Định tuyến (Routing)**

Routing trong Node.js là một khái niệm nói đến việc xác định ứng dụng sẽ đáp ứng như thế nào khi người dùng tạo một request đến một endpoint (Điểm cuối) cụ thể nào đó. Điểm cuối đó thường là một URI hoặc một đường dẫn (Path) với một Request method (POST, PUT, GET, ... ) cụ thể.

**Cấu trúc định tuyến cơ bản:**

app.METHOD (Path, Handler...)

Trong đó:

* **app** : là một instance của express
* **METHOD**: là một HTTP Method

Express hỗ trợ rất nhiều loại HTTP methods khác nhau, bao gồm: get, post, put, head, delete, options, trace, copy, lock, mkcol, move, purge, unlock, report, mkactivity, checkout, merge, m-search, notify, subscribe, unsubscribe, patch.

* **Path**: là một đường dẫn trên máy chủ

Route path có thể là một chuỗi thông thường (String) hoặc là một chuỗi có ký hiệu biểu thức chính quy (string patterns) hoặc là một biểu thức chính quy (regular expressions)

* **Handler** : là một function sẽ thực thi khi một **route** được trùng khớp

Đơn giản là một hoặc nhiều function sẽ được gọi khi một route trùng khớp để đáp ứng một yêu cầu nào đó. Lưu ý các handler sẽ được gọi đúng theo thứ tự truyền vào.



Hình 20: Ví dụ cấu trúc route

**Route parameters**

Route parameters là những vị trí trên URL được đánh dấu bằng cách đặt tên, mục đích là để lấy ra các giá trị tương ứng. Tất cả cá giá trị đối số sẽ được đặt vào đối tượng **req** trong thuộc tính **params**. Với tên thuộc tính trùng khớp với từ khóa được xác định trên URL.

Ví dụ, chúng ta định nghĩa một path là /api/color/:colorId . Thì ở đây :colorId chính là một route param. Khi đó nếu người dùng truy cập đường dẫn như là /api/color/44 ta lấy ra được:colorId = 44 và giá trị này sẽ nằm ở req.params.colorId

**Response methods**

Sau việc tiếp nhận và xử lý, thì việc tiếp theo đó là đáp ứng (Response). Trong express định nghĩa sẵn một số phương thức hỗ trợ hay dùng nhất là:

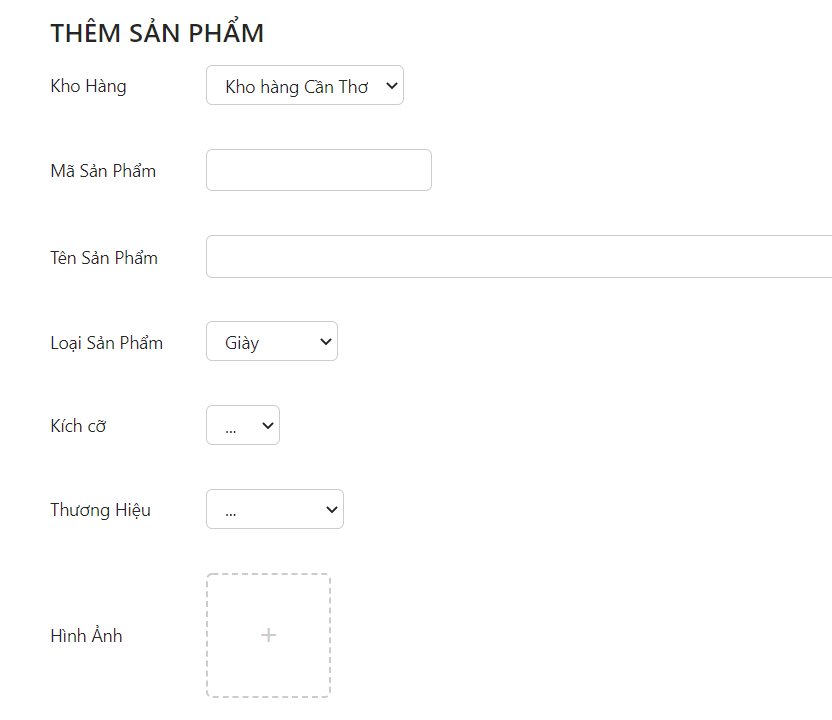
|  |  |
| --- | --- |
| **Tên phương thức** | **Ý nghĩa** |
| res.json() | Trả về một dữ liệu dạng JSON |
| res.redirect() | Chuyển hướng đến một đường dẫn nào đó |
| res.render() | Trả về một view template |
| res.send() | Gửi dữ liệu dạng text |

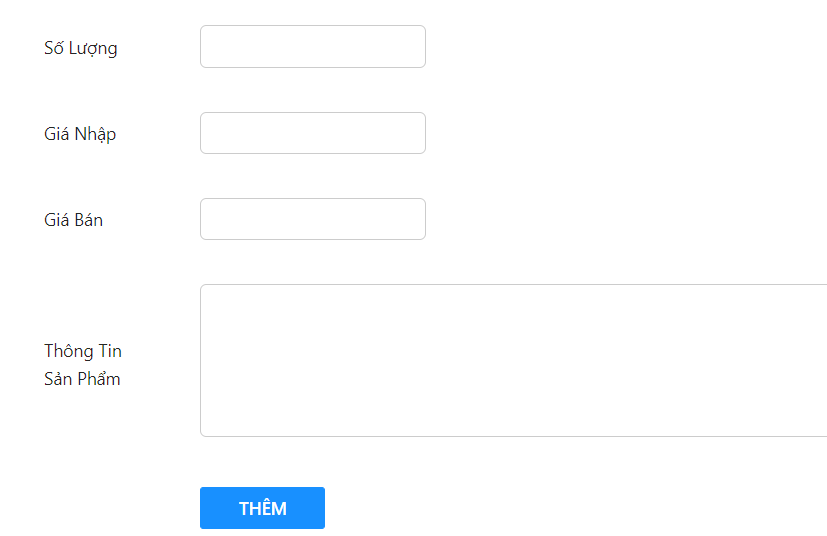
Hình 21: Một số phương thức (response methods) hỗ trợ hay dùng nhất

# PHẦN 4: XÂY DỰNG HỆ THỐNG

## I. XÂY DỰNG CHỨC NĂNG CHO NGƯỜI QUẢN LÝ

### 1. Chức năng thêm sản phẩm

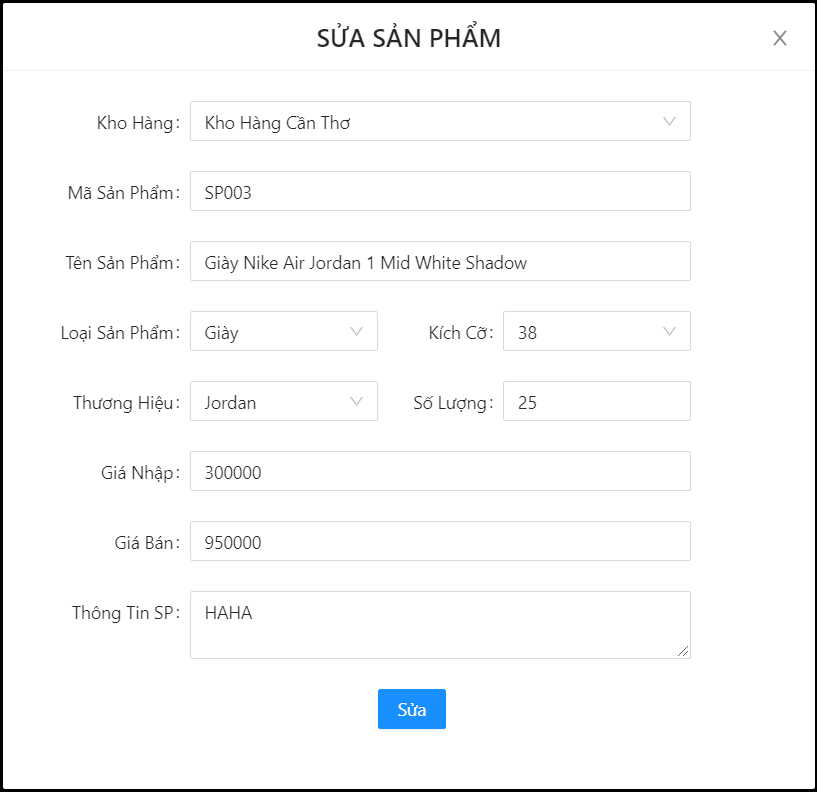




Hình 22: Giao diện thêm sản phẩm

- Chức năng thêm sản phẩm cho phép người quản lý nhập thông tin cần thiết để tạo mới một sản phẩm mới như: chọn kho hàng, loại sản phẩm, tạo mã sản phẩm, tên sản phẩm, kích cỡ, thương hiệu, hình ảnh, số lượng sản phẩm, giá nhập, giá bán và thông tin sản phẩm.

### 2. Chức năng sửa sản phẩm



Hình 23: Giao diện sửa sản phẩm

- Sau khi thêm sản phẩm, nếu sản phẩm bị sai thông tin, chức năng sửa sản phẩm có thể giúp người quản lý cập nhật lại thông tin sản phẩm mới chính xác dựa trên thông tin sản phẩm đã tạo. Các thông tin tương tự chức năng thêm sản phẩm.

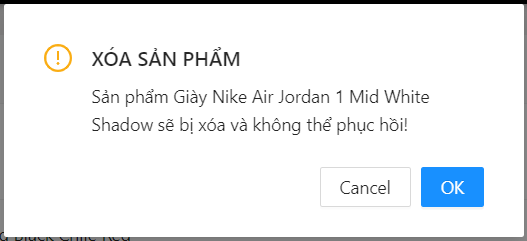
### 3. Chức năng xem thông tin sản phẩm



Hình 24: Giao diện xem thông tin sản phẩm

- Chức năng xem thông tin sản phẩm cho phép người quản lý kiểm tra lại các thông tin của sản phẩm sau khi tạo.

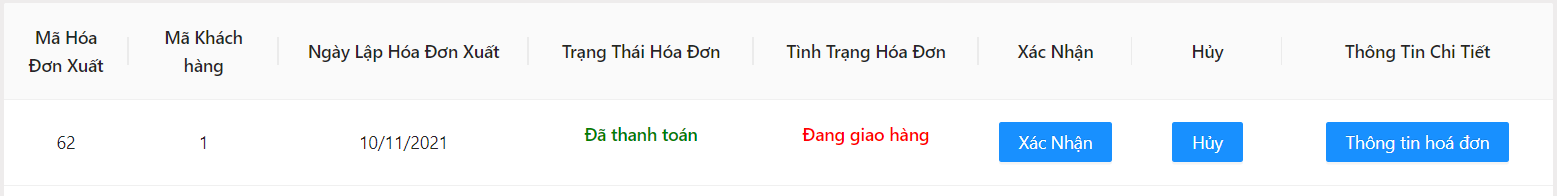
### 4. Chức năng xóa sản phẩm



Hình 25: Giao diện xóa sản phẩm

- Chức năng xóa sản phẩm cho phép gười quản lý xóa tất cả thông tin của sản phẩm.

### 5. Chức năng xác nhận hóa đơn và hủy hóa đơn

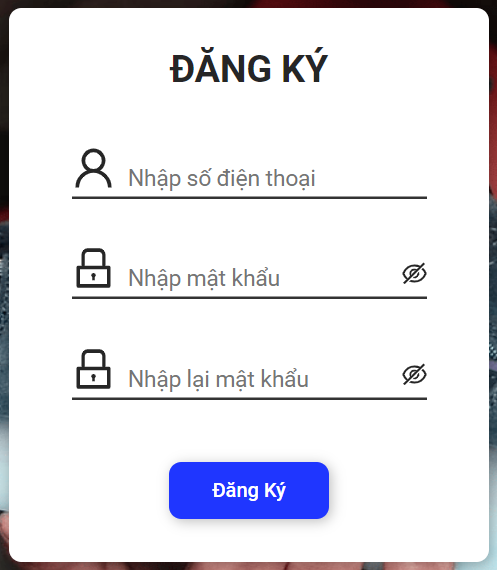


Hình 26: Giao diện xác nhận đơn và hủy đơn hàng

- Chức năng này cho phép người quản lý xác nhận hóa đơn khi sản phẩm đã đến tay khách hàng, và có thể hủy hóa đơn nếu khách hàng không muốn mua sản phẩm nữa.

## II. XÂY DỰNG CHỨC NĂNG CHO KHÁCH HÀNG

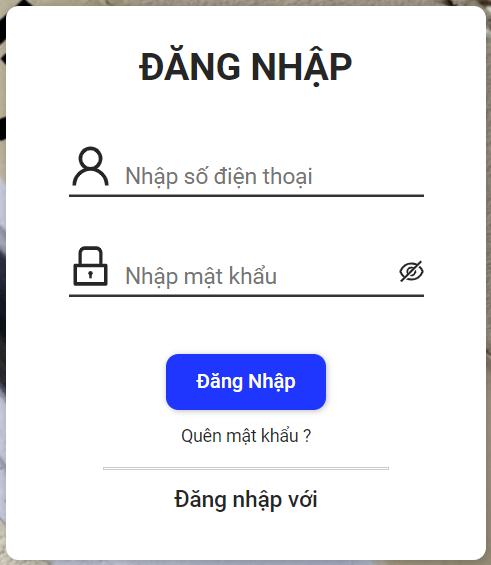
### 1. Chức năng đăng ký



Hình 27: Giao diện đăng ký

- Chức năng đăng ký cho phép khách hàng tạo một tài khoản của riêng mình với 2 thông tin là số điện thoại và mật khẩu.

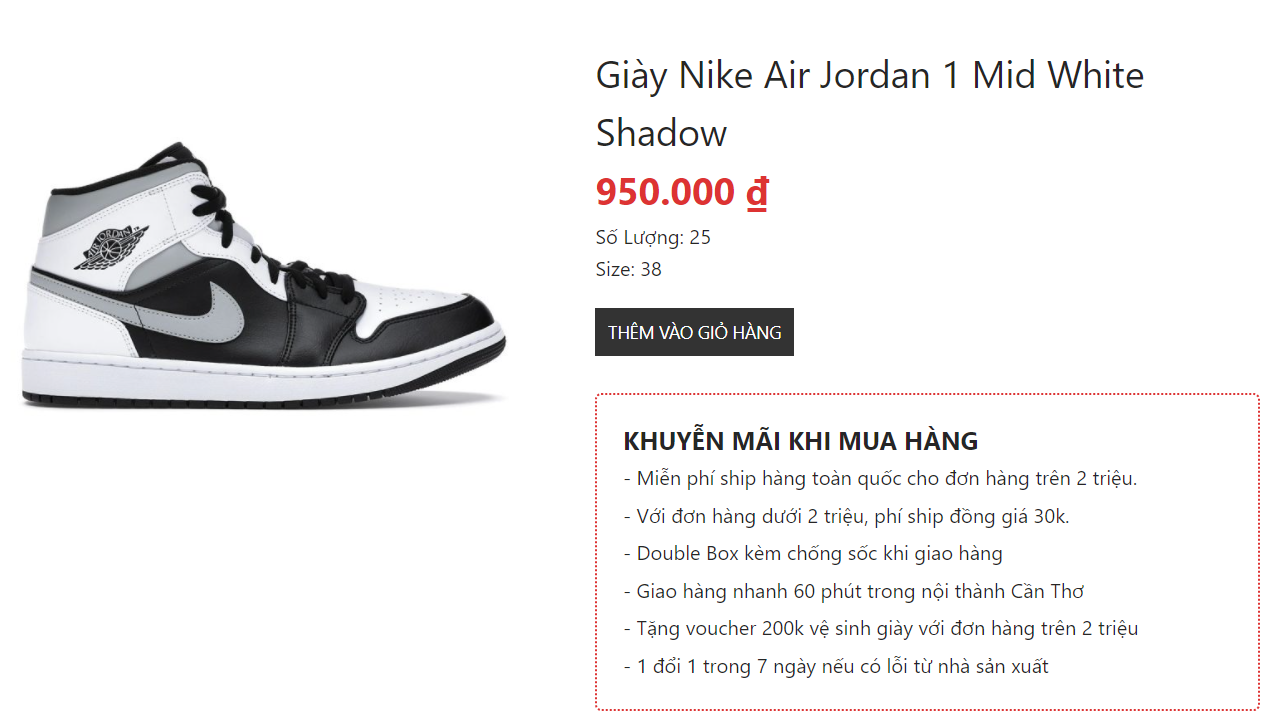
### 2. Chức năng đăng nhập



Hình 28: Giao diện đăng nhập

- Chức năng đăng nhập cho phép khách hàng truy cập vào tài khoản mua hàng của mình bằng thông tin đã dùng để đăng ký trước đó là số điện thoại và mật khẩu.

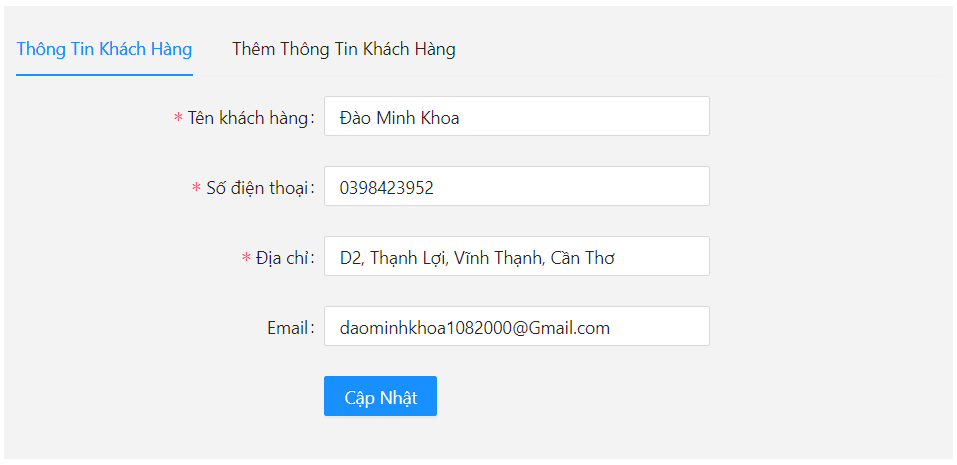
### 3. Chức năng xem thông tin sản phẩm



Hình 29: Giao diện xem thông tin sản phẩm

- Chức năng này cho phép khách hàng xem thông tin chi tiết của một sản phẩm bằng cách chọn một sản phẩm trong danh sách sản phẩm được hiển thị. Trang chi tiết sản phẩm sẽ hiển thị thông tin của một sản phẩm như: tên sản phẩm, hình ảnh sản phẩm, giá sản phẩm, số lượng, và kích thước của sản phẩm, cùng với đó là một số khuyễn mãi từ nhà cung cấp.

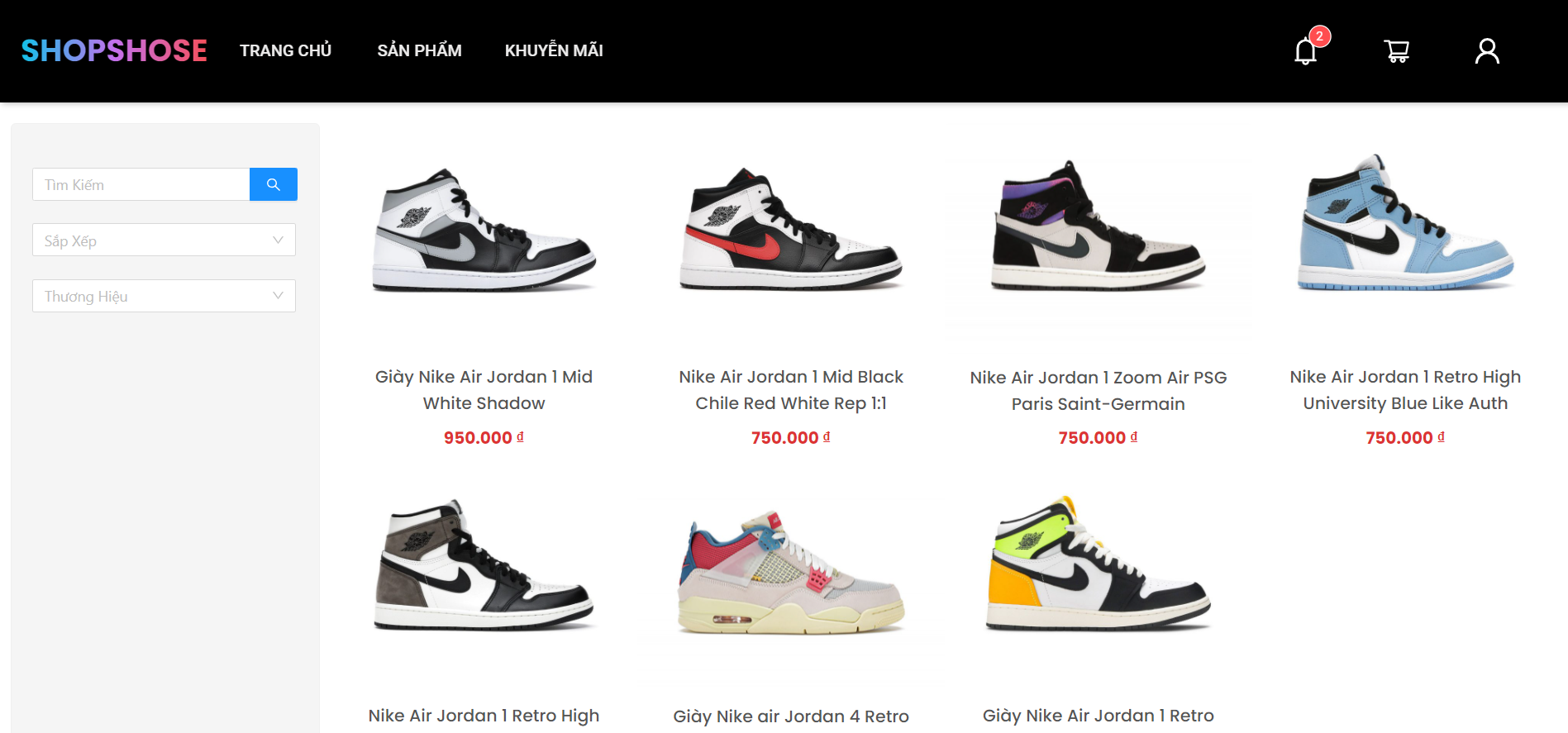
### 4. Chức năng thêm thông tin khách hàng



Hình 30: Giao diện thêm thông tin khách hàng

- Để có thể mua sản phẩm, khách hàng cần thêm thông tin của mình như: tên khách hàng, số điện thoại, địa chỉ, email, để sản phẩm có thể được chuyển tới tay khách hàng.

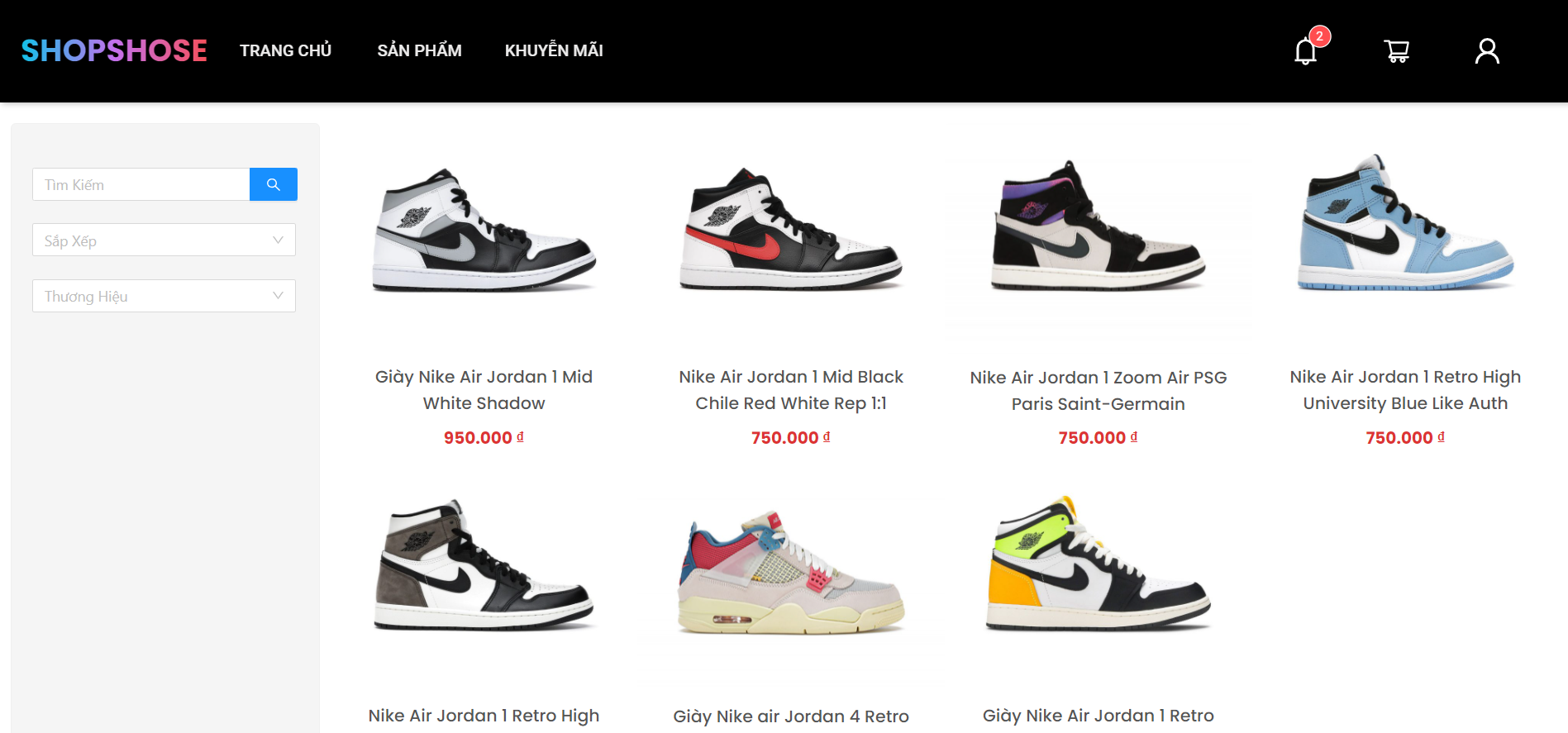
### 5. Chức năng tìm kiếm sản phẩm



Hình 31: Giao diện tìm kiếm sản phẩm

- Chức năng tìm kiếm chó phép khách hàng tìm sản phẩm theo nhu cầu của mình.

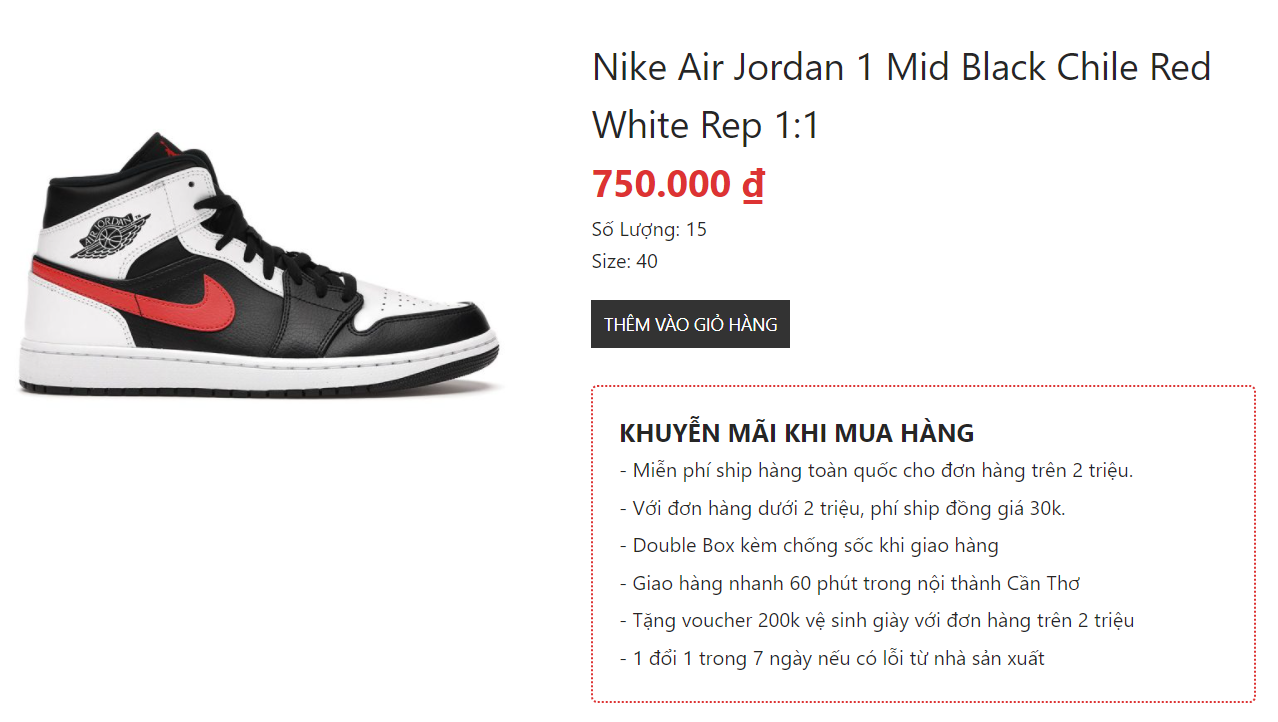
### 6. Chức năng lọc sản phẩm



Hình 32: Giao diện lọc sản phẩm

- Chức năng lọc sản phẩm cho phép khách hàng lọc ra những sản phẩm theo nhu cầu của mình một cách chính xác nhất. có 2 cách lọc là: lọc theo giá và lọc theo thương hiệu.

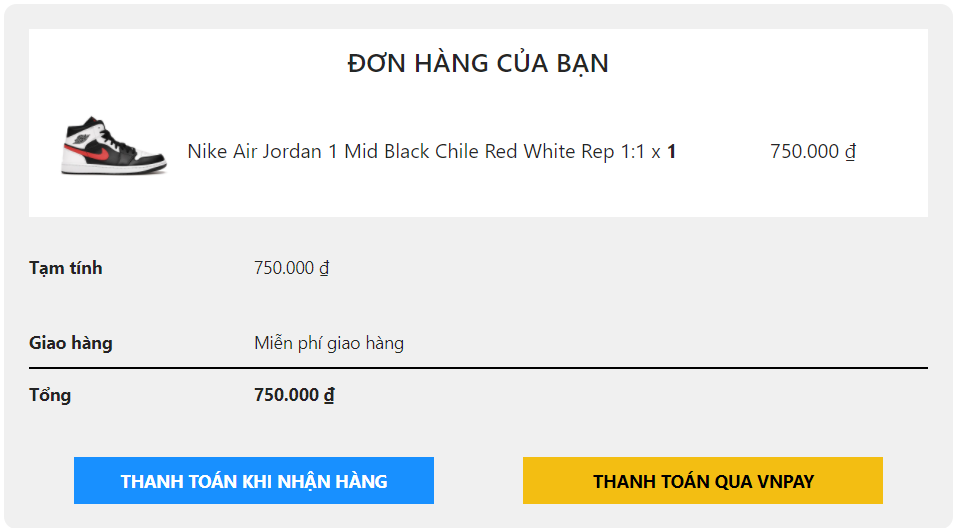
### 7. Chức năng thêm sản phẩm vào giỏ hàng



Hình 33: Giao diện thêm sản phẩm vào giỏ hàng

- Chức năng thêm sản phẩm vào giỏ hàng cho phép khách hàng đánh dấu những sản phẩm muốn mua hoặc đang chờ vào giỏ hàng. Lưu ý: để thêm san phẩm vào giỏ hàng, khách hàng cần đăng nhập tài khoản trước.

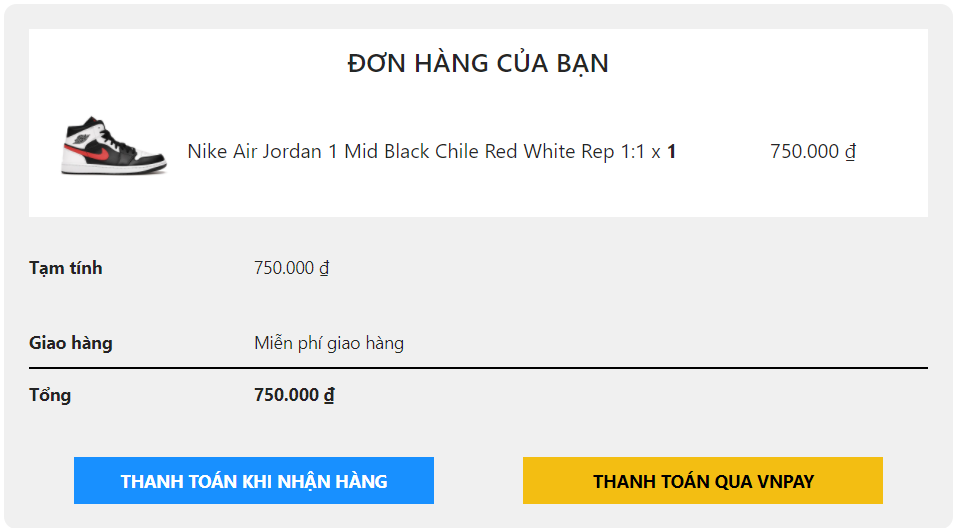
### 8. Chức năng thanh toán khi nhận hàng



Hình 34: Giao diện thanh toán khi nhận hàng

- Chứa năng này cho phép khách hàng thực hiện ngay việc đặt hàng và thanh toán khi đã nhận được sản phẩm.

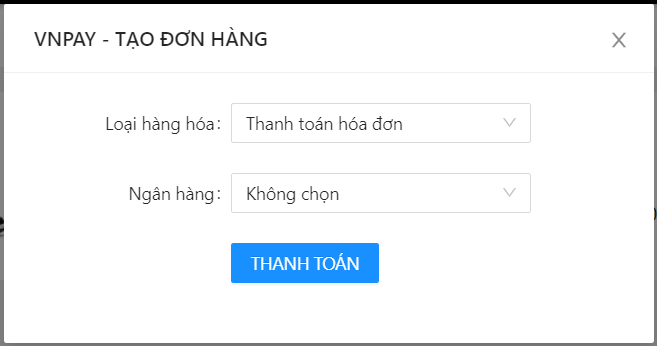
### 9. Chức năng thanh toán qua vnpay



Hình 35: Giao diện thanh toán qua vnpay

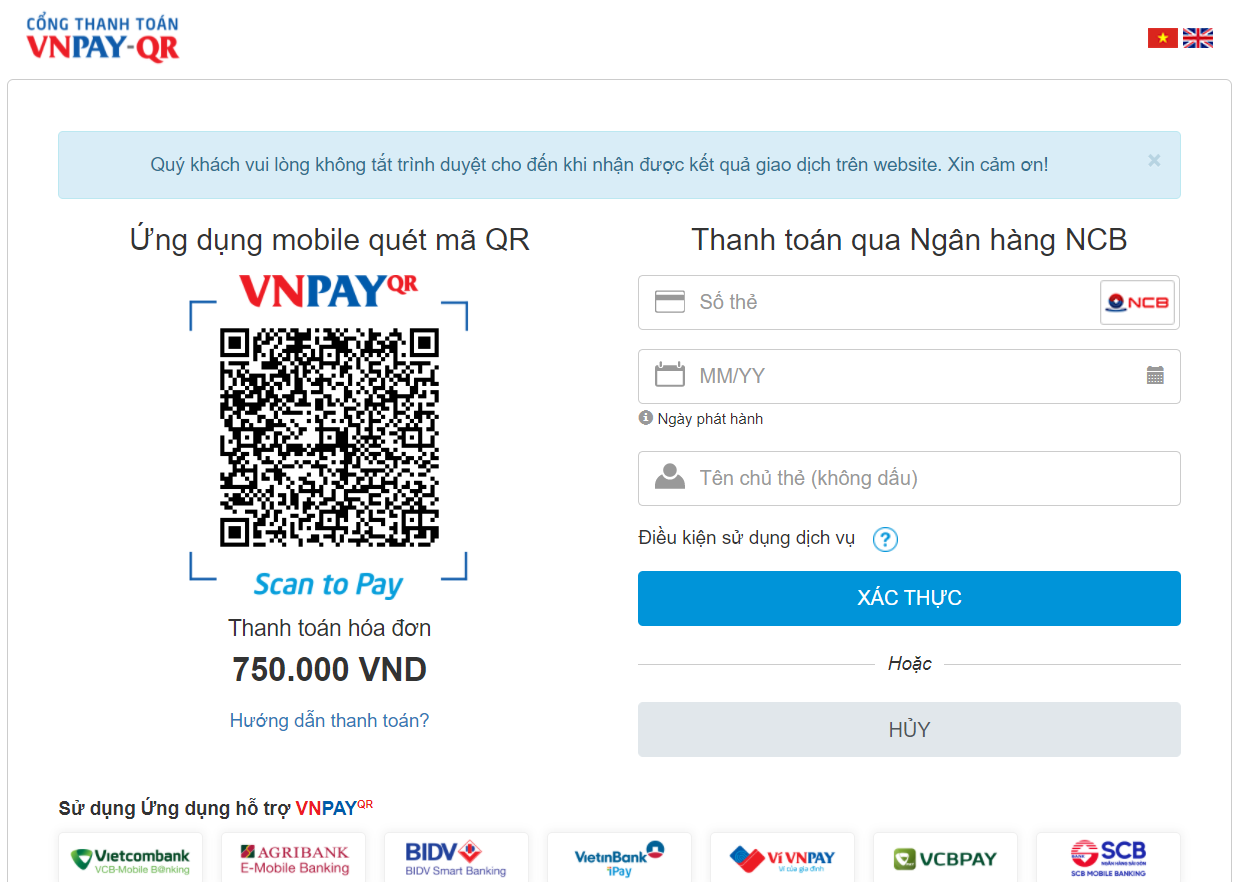
- Chức năng này cho phép khách hàng trả tiền trước trước khi nhận hàng.

Sau khi chọn thanh toán qua vnpay, một hộp thoại sẽ mở ra để khách hàng có thể chọn hình thức thanh toán và ngân hàng sẽ thanh toán cho sản phẩm



Hình 36: Giao diện chọn hình thức thanh toán

Sau khi đã chọn hình thức thanh toán và ngân hàng thanh toán, website sẽ chuyển hướng đến trang thanh toán của vnpay. Khách hàng cần nhập thông tin ngân hàng của mình để thực hiện thanh toán. Cuối cùng khách hàng sẽ được chuyển hướng về trang hóa đơn của mình.



Hình 37: Giao diện nhập thông tin thanh toán

### 10. Chức năng theo dõi đơn hàng



Hình 38: Giao diện đơn hàng

- Chức năng theo dõi đơn hàng cho phép khách hàng xem tình trạng và trạng thái đơn hàng của mình.

# PHẦN 5: KẾT LUẬN

## 1. Kết quả đạt được

Qua 5 tháng học hỏi , tìm tòi, được sự giúp đỡ từ bạn bè, tham khảo trên các diễn đàn thì đồ án đã hoàn thành và mang lại cho em nhiều kiến thức mới, có cái nhìn khách quan hơn về cách framework mới, cùng với đó là cách tư duy sáng tạo sản phẩm của mình. Giúp em hiểu được cách thức xây dựng website bằng Reactjs, Nodejs,.. cũng với đó là ôn lại các kiến thức cơ bản mà em đã học ở môn lập trình web.

## 2. Hướng phát triển

- Xây dựng thêm một số chức năng mới: thêm các bộ lọc mới,

- Tối ưu hóa lại website.

- Thêm một số API mới: Google Map, Các loại thanh toán nước ngoài như paypal

## 3. Tài liệu tham khảo

[1]. ReacJS [online] 9/9/2021 : <https://reactjs.org/docs/getting-started.html>

[2]. React-Boostrap [online] 17/10/2021:

<https://react-bootstrap.github.io/getting-started/introduction/>

[3]. NodeJS [online] 20/10/2021: <https://www.w3schools.com/nodejs/>

[4]. Axios [online] 28/10/2021: <https://axios-http.com/docs/req_config>

[5]. React-Form-Hook [online] 10/11/2021: <https://react-hook-form.com/>

[6]. React router [online] 20/11/2021: <https://reactrouter.com/web/guides/quick-start>

[7]. React Saga [online] 30/11/2021:

<https://redux-saga.js.org/docs/introduction/GettingStarted>

[8]. MD5 [online] 05/12/2021: <https://www.npmjs.com/package/md5>

[9]. vnPay [online] 06/12/2021: <https://sandbox.vnpayment.vn/apis/docs/huong-dan-tich-hop/>