|  |
| --- |
|  |
| **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **KHOA: CÔNG NGHỆ THÔNG TIN**  **---------------------------------------** |
|  |
| BÁO CÁO BÀI TẬP LỚN HỌC PHẦN PHÁT TRIỂN PHẦN MỀM HƯỚNG FRAMEWORK |
|  |
| **TÊN CHỦ ĐỀ NGHIÊN CỨU** |
| **XÂY DỰNG WEBSITE BÁN DÉP BẰNG THƯ VIỆN REACTJS VÀ FRAMEWORK LARAVEL PHP** |
|  |
| **GVHD:ThS. Nguyễn Thái Cường**  **Nhóm: 3** |
| **Sinh viên: Đoàn Tuấn Anh**  **Đàm Thị Kim Phượng**  **Nguyễn Ngọc Phương**  **Tô Anh Quân**  **Nguyễn Đắc Quang** |
| **Lớp: 202020503146003. Khóa: 12** |
|  |
|  |
|  |
| **Hà Nội – Năm 2021** |

MỤC LỤC

[Chương 1: Tổng quan về ReactJS và Laravel](#_heading=h.30j0zll) **3**

[1.1 Tổng quan về ReactJS](#_heading=h.1fob9te) 3

[1.1.1 Khái niệm về ReactJS](#_heading=h.3znysh7) 3

[1.1.2 Lịch sử phát triển của ReactJS](#_heading=h.2et92p0) 3

[1.1.3 Các thành phần của ReactJS](#_heading=h.tyjcwt) 4

[1.1.3.1 Components](#_heading=h.3dy6vkm) 4

[1.1.3.2 JSX](#_heading=h.1t3h5sf) 5

[1.1.3.3 Redux](#_heading=h.4d34og8) 6

[1.1.3.4 Virtual DOM](#_heading=h.2s8eyo1) 8

[1.1.4 Tính năng nổi bật của ReactJS](#_heading=h.17dp8vu) 10

[1.1.5 Lợi ích của việc sử dụng ReactJS](#_heading=h.3rdcrjn) 11

[1.1.6 Tồn tại của ReactJS](#_heading=h.2jxsxqh) 12

[1.1.7 Cài đặt 1 project ReactJS](#_heading=h.z337ya) 12

[1.2 Tổng quan về Laravel](#_heading=h.3j2qqm3) 15

[1.2.1 Khái niệm về Laravel](#_heading=h.1y810tw) 15

[1.2.2 Lịch sử phát triển của Laravel](#_heading=h.4i7ojhp) 15

[1.2.3 Đặc tính và tính năng nổi bật của Laravel](#_heading=h.2xcytpi) 17

[1.2.4 Cấu trúc của Laravel](#_heading=h.1ci93xb) 19

[1.2.5 Ưu điểm và nhược điểm của Laravel](#_heading=h.3whwml4) 21

[1.2.6 Cài đặt Laravel và chạy Laravel](#_heading=h.2bn6wsx) 22

[Chương 2: Phân tích thiết kế hệ thống](#_heading=h.v355htb5p9lc) **23**

[2.1 Sơ](#_heading=h.1fob9te) đồ ca sử dụng 23

[2.1.1 Sơ](#_heading=h.3znysh7) đồ ca sử dụng của khách hàng 23

[2.1.2 Sơ](#_heading=h.3znysh7) đồ ca sử dụng của người quản trị 25

[2.1.3 Đặc](#_heading=h.3znysh7) tả use case người quản trị 27

[2.1.4 Đặc](#_heading=h.3znysh7) tả use case khách hàng 37

[2.2 Sơ](#_heading=h.1fob9te) đồ lớp 47

[2.3 Biểu](#_heading=h.1fob9te) đồ trình tự 48

Chương 3 : Thiết kế giao diện

[3.1 Giao](#_heading=h.1fob9te) diện login 53

[3.2 Giao](#_heading=h.1fob9te) diện admin 53

[3.3 Giao](#_heading=h.1fob9te) diện trang chủ 57

[3.4 Giao](#_heading=h.1fob9te) diện giỏ hàng 58

Đánh giá và Kết Luận 59

# Chương 1: Tổng quan về ReactJS và Laravel

## 1.1 Tổng quan về ReactJS

### 1.1.1 Khái niệm về ReactJS

React là một thư viện Javascript mã nguồn mở được phát triển tại Facebook để hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái (state) và có thể sử dụng lại được.

React JS là một framework hiển thị View chú ý đến hiệu năng. Rất nhiều đối thủ nặng ký về framework MVVM (Model-View-ViewModel) mất một thời gian lớn để hiển thị những lượng data lớn, như trong trường hợp những danh sách (list) và tương tự. Nhưng React đó không còn là vấn đề, vì nó chỉ hiển thị những gì thay đổi.

### 1.1.2 Lịch sử phát triển của ReactJS

React được tạo ra bởi Jordan Walke, một kỹ sư phần mềm tại Facebook, người đã phát hành một nguyên mẫu đầu tiên của React được gọi là "FaxJS". Ông bị ảnh hưởng bởi XHP, một thư viện thành phần HTML cho PHP. Nó được triển khai lần đầu tiên trên News Feed của Facebook vào năm 2011 và sau đó là Instagram vào năm 2012. Nó có nguồn mở tại JSConf US vào tháng 5 năm 2013.

React cho phép phát triển Android, iOS và UWP gốc với React, đã được công bố tại Facebook's React Conf vào tháng 2 năm 2015 và có nguồn mở vào tháng 3 năm 2015.

Vào ngày 18 tháng 4 năm 2017, Facebook đã công bố React Fiber, một thuật toán cốt lõi mới của thư viện React để xây dựng giao diện người dùng. React Fiber đã trở thành nền tảng của bất kỳ cải tiến nào trong tương lai và phát triển tính năng của thư viện React.

Vào ngày 26 tháng 9 năm 2017, React 16.0 đã được phát hành ra công chúng.

Vào ngày 16 tháng 2 năm 2019, React 16.8 đã được phát hành ra công chúng. Bản phát hành đã giới thiệu React Hooks.

Vào ngày 10 tháng 8 năm 2020, nhóm React đã công bố ứng cử viên phát hành đầu tiên cho React v17.0, đáng chú ý là bản phát hành lớn đầu tiên không có thay đổi lớn đối với API dành cho nhà phát triển React.

### 1.1.3 Các thành phần của ReactJS

#### 1.1.3.1 Components

Trong React, các component hoạt động giống như các hàm trả về các thành phần HTML. Các component là các thành phần độc lập và có thể sử dụng lại.

Các component thực hiện công việc giống như các functions trong JavaScript nhưng chúng độc lập và nhiệm vụ chính là trả về HTML thông qua hàm render. Nó giúp phân chia các UI (giao diện người dùng) thành các phần nhỏ để dễ dàng quản lý và tái sử dụng. Giả sử mình có một website gồm nhiều phần bố cục khác nhau và mình muốn chia nhỏ các phần ra để dễ quản lý.

Có 2 loại component là Function Component và Class Component:

- Function Component là một hàm JavaScript bình thường nhưng giá trị return là một object React Element. Object này được tạo ra bởi hàm React.createElement hoặc React.cloneElement.



Hình 1.1 Ví dụ một Function Component

- Class Component là một class JavaScript thừa kế từ React.Component hoặc React.PureComponent hay một Class component khác. Class này phải implement hàm render () có giá trị return là một object React Element, nội dung hàm render () này cũng giống như Functional component.



Hình 1.2 Ví dụ một Class Component

#### 1.1.3.2 JSX

JSX là viết tắt là Javascript XML, một template languages nhưng nó lại mang hầu hết tính năng của Javascript. Nó cho phép bạn viết các đoạn mã HTML trong React một cách dễ dàng và có cấu trúc hơn. Code JSX tương tự như HTML nhưng thật sự nó là một sự pha trộn giữa JavaScript và HTML.

JSX giúp cho việc xây dựng các ứng dụng React một cách nhanh hơn, dễ tối ưu trong việc compile code sang javascript. JSX nhanh hơn JavaScript vì nó thực hiện tối ưu hóa khi biên dịch mã nguồn. JSX cũng được khuyên dùng hơn vì nó dễ sử dụng hơn so với code thuần JS.

Việc sử dụng JSX trong ReactJS là không bắt buộc. Bạn có thể sử dụng chỉ JS thuần thôi. Nhưng có rất nhiều lý do cho việc nên sử dụng JSX trong ReactJS. Cấu trúc cây khi biểu thị các attributes, điều đó giúp ta dễ dàng định nghĩa, quản lý được các component phức tạp, thay vì việc phải định nghĩa và gọi ra nhiều hàm hoặc object trong javascript. Khi nhìn vào cấu trúc đó cũng dễ dàng đọc hiểu được ý nghĩa của các component.

Code JSX ngắn hơn, dễ hiểu hơn code JS. JSX rất dễ xem các lỗi trong quá trình triển khai bởi hầu hết các lỗi sẽ được hiển thị trong quá trình compile, không như các đoạn mã HTML có thể thừa thiếu các thẻ div khiến giao diện bị hiển thị sai. JSX lại hoàn toàn ngược lại, khi bạn quên đóng div chẳng hạn thì nó lập tực sẽ hiển thị lỗi.

#### 1.1.3.3 Redux

Redux là một predictable state management tool cho các ứng dụng Javascript. Nó giúp bạn viết các ứng dụng hoạt động một cách nhất quán, chạy trong các môi trường khác nhau (client, server, and native) và dễ dàng để test. Redux ra đời lấy cảm hứng từ tư tưởng của ngôn ngữ Elm và kiến trúc Flux của Facebook. Do vậy Redux thường dùng kết hợp với React.

Redux ra đời là vì do yêu cầu cho các ứng dụng single-page sử dụng Javascript ngày càng trở lên phức tạp thì code của chúng ta phải quản lý nhiều state hơn. Với Redux, state của ứng dụng được giữ trong một nơi gọi là store và mỗi component đều có thể access bất kỳ state nào mà chúng muốn từ chúng store này.

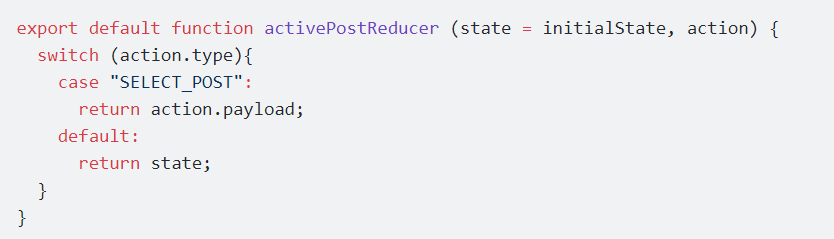
Có 4 thành phần chính của Redux: Actions Creators, Store, Reducers, View:

- Actions Creators đơn giản là các events. Chúng là cách mà chúng ta send data từ app đến Redux store. Những data này có thể là từ sự tương tác của user và app, API calls hoặc cũng có thể là từ form submission.



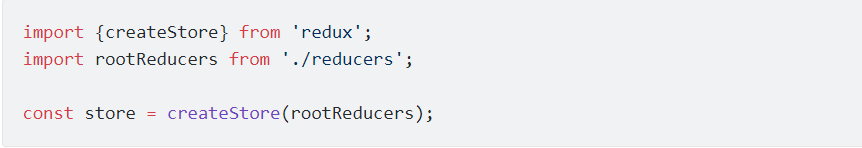
Hình 1.3 Actions Creators

- Reducers là các function nguyên thủy chúng lấy state hiện tại của app, thực hiện một action và trả về một state mới. Những states này được lưu như những objects và chúng định rõ các state của một ứng dụng thay đổi trong việc phản hồi một action được gửi đến store.



Hình 1.4 Reducers

- Store là nơi tạo ra các action dùng để mô tả event do người dùng tạo ra. Nó lưu trạng thái ứng dụng và nó là duy nhất trong bất kỳ một ứng dụng Redux nào, cho phép truy cập State qua getState (), update State qua dispatch(action). Bạn có thể access các state được lưu, update state, và đăng ký or hủy đăng ký các listeners thông qua helper methods.



Hình 1.5 Store

- View: Hiển thị dữ liệu được cung cấp bởi Store.

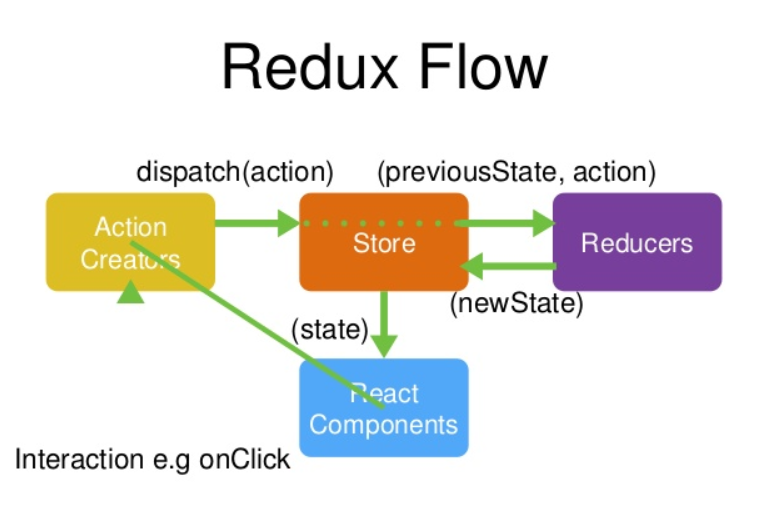
Nguyên lý hoạt động của Redux:

Bước 1: Khi có 1 sự kiện (event) như là GET, POST, UPDATE, DELETE... thì thằng action creators sẽ sinh ra 1 action mô tả những gì đang xảy ra.

Bước 2: Action sẽ thực hiện điều phối Reducer xử lý event thông qua hàm dispatch(action).

Bước 3: Reducer dựa vào những mô tả của Action để biết cần thực hiện thay đổi gì trên State và thực hiện update.

Bước 4: Khi State được update thì các trigger đang theo dõi state đó sẽ nhận được thông tin update và tiến hành render lại phần view để hiển thị ra cho người dùng.



Hình 1.6 Nguyên lý hoạt động của Redux

Các nguyên tắc trong Redux:

- Store luôn là nguồn dữ liệu đúng và tin cậy duy nhất.

- State chỉ được phép đọc, cách duy nhất để thay đổi State là phát sinh một Action, và để Reducer thay đổi State.

- Các function Reducer phải là Pure function (với cùng 1 đầu vào chỉ cho ra 1 đầu ra duy nhất).

#### 1.1.3.4 Virtual DOM

DOM (Mô hình đối tượng tài liệu) là một bản tóm tắt của văn bản có cấu trúc, có nghĩa là nó được làm bằng mã HTML và css. Các phần tử HTML này trở thành các nút trong DOM. Có những hạn chế đối với các phương pháp thao tác DOM trước đây.

Virtual DOM là một bản tóm tắt của HTML DOM theo nghĩa đen được tạo ra trước khi React được tạo hoặc sử dụng, nhưng với mục đích của chúng tôi, chúng tôi sẽ sử dụng nó trong buổi hòa nhạc với ReactJS. Virtual DOM rất nhẹ và tách rời khỏi việc triển khai DOM trong trình duyệt.

Virtual DOM về cơ bản là một ảnh chụp màn hình (hoặc bản sao) của DOM tại một thời điểm nhất định. Một cách để xem xét nó từ góc độ nhà phát triển là DOM là môi trường sản xuất và Virtual DOM là môi trường (dev) cục bộ. Mỗi lần dữ liệu thay đổi trong ứng dụng React, một đại diện DOM ảo mới của giao diện người dùng được tạo.

Phương thức cơ bản nhất cần thiết để tạo thành phần tĩnh trong ReactJS là: Bạn phải trả lại mã từ phương thức kết xuất. Bạn phải chuyển đổi mọi lớp thành className vì lớp được dành riêng từ trong JavaScript. Ngoài những thay đổi lớn hơn, có hai sự khác biệt nhỏ giữa hai DOM bao gồm ba thuộc tính xuất hiện trong Virtual DOM nhưng không có trong DOM DOM (khóa, ref và nguy hiểm SetInnerHTML).

Một điều quan trọng cần hiểu khi làm việc với Virtual DOM là sự khác biệt giữa ReactEuity và ReactComponent:

- ReactEuity:

+ Là một đại diện ảo nhẹ, không trạng thái, bất biến, của một phần tử DOM.

+ Đây là loại chính trong React và nằm trong Virtual DOM. ReactElements có thể được kết xuất thành HTML DOM.

+ JSX biên dịch các thẻ HTML thành ReactElements.

- ReactComponent:

+ ReactComponent là những thành phần có trạng thái.

+ React.createClass được coi là ReactComponent.

+ Bất cứ khi nào nhà nước thay đổi thành phần được đăng ký lại.

Bất cứ khi nào ReactComponent có thay đổi trạng thái, chúng tôi muốn có ít thay đổi nhất đối với DOM. DOM càng tốt để ReactComponent được chuyển đổi thành ReactEuity sau đó có thể được chèn vào DOM ảo, được so sánh và cập nhật nhanh chóng và dễ dàng. Khi React biết khác biệt - nó được chuyển đổi thành mã mức thấp (HTML DOM), được thực thi trong DOM.

### 1.1.4 Tính năng nổi bật của ReactJS

Viết ứng dụng trực tiếp trên JavaScript.

Phá vỡ các cấu trúc UI phức tạp và biến chúng thành các components độc lập.

Chuyển các dữ liệu đã đc tùy biến đến một UI component cụ thể. Thay đổi trạng thái cho nhiều component trên ứng dụng nhưng không ảnh hưởng tới component gốc.

Sử dụng JSX: Trong React, thay vì thường xuyên sử dụng JavaScript để thiết kế bố cục trang web thì sẽ dùng JSX. JSX được đánh giá là sử dụng đơn giản hơn JavaScript và cho phép trích dẫn HTML cũng như việc sử dụng các cú pháp thẻ HTML để render các subcomponent. JSX tối ưu hóa code khi biên soạn, vì vậy nó chạy nhanh hơn so với code JavaScript tương đương.

Single-way data flow (Luồng dữ liệu một chiều): ReactJS không có những module chuyên dụng để xử lý data, vì vậy ReactJS chia nhỏ view thành các component nhỏ có mối quan hệ chặt chẽ với nhau. Luồng truyền dữ liệu trong ReactJS là luồng dữ liệu một chiều từ cha xuống con. Việc ReactJS sử dụng one-way data flow có thể gây ra một chút khó khăn cho những người muốn tìm hiểu và ứng dụng vào trong các dự án. Tuy nhiên, cơ chế này sẽ phát huy được ưu điểm của mình khi cấu trúc cũng như chức năng của view trở nên phức tạp thì ReactJS sẽ phát huy được vai trò của mình.

Sử dụng Virtual DOM: Những Framework sử dụng Virtual-DOM như ReactJS khi Virtual-DOM thay đổi, chúng ta không cần thao tác trực tiếp với DOM trên View mà vẫn phản ánh được sự thay đổi đó. Do Virtual-DOM vừa đóng vai trò là Model, vừa đóng vai trò là View nên mọi sự thay đổi trên Model đã kéo theo sự thay đổi trên View và ngược lại. Có nghĩa là mặc dù chúng ta không tác động trực tiếp vào các phần tử DOM ở View nhưng vẫn thực hiện được cơ chế Data-binding. Điều này làm cho tốc độ ứng dụng tăng lên đáng kể – một lợi thế không thể tuyệt vời hơn khi sử dụng Virtual-DOM.

### 1.1.5 Lợi ích của việc sử dụng ReactJS

\* Dễ sử dụng:

React là một thư viện GUI nguồn mở JavaScript tập trung vào một điều cụ thể; hoàn thành nhiệm vụ UI hiệu quả. Nó được phân loại thành kiểu “V” trong mô hình MVC (Model-View-Controller).

Là lập trình viên JavaScript, bạn sẽ dễ dàng hiểu được những điều cơ bản về React. Bạn thậm chí có thể bắt đầu phát triển các ứng dụng dựa trên web bằng cách sử dụng react chỉ trong vài ngày.

Để củng cố hiểu biết của mình, bạn hãy thử khám phá thêm nhiều hướng dẫn về React. Chúng mang đến nhiều thông tin về cách sử dụng công cụ: videos, hướng dẫn và dữ liệu làm phong phú góc nhìn của bạn.

\* Nó hỗ trợ Reusable Component trong Java:

React cho phép bạn sử dụng lại components đã được phát triển thành các ứng dụng khác có cùng chức năng. Tính năng tái sử dụng component là một lợi thế khác biệt cho các lập trình viên.

\* Viết component dễ dàng hơn:

React component dễ viết hơn vì nó sử dụng JSX, mở rộng cú pháp tùy chọn cho JavaScript cho phép bạn kết hợp HTML với JavaScript.

JSX là một sự pha trộn tuyệt vời của JavaScript và HTML. Nó làm rõ toàn bộ quá trình viết cấu trúc trang web. Ngoài ra, phần mở rộng cũng giúp render nhiều lựa chọn dễ dàng hơn.

JSX có thể không là phần mở rộng cú pháp phổ biến nhất, nhưng nó được chứng minh là hiệu quả trong việc phát triển components đặc biệt hoặc các ứng dụng có khối lượng lớn.

\* Hiệu suất tốt hơn với Virtual DOM:

React sẽ cập nhật hiệu quả quá trình DOM (Document Object Model – Mô hình đối tượng tài liệu). Như bạn có thể biết, quá trình này có thể gây ra nhiều thất vọng trong các dự án ứng dụng dựa trên web. May mắn là React sử dụng virtual DOMs, vì vậy bạn có thể tránh được vấn đề này.

Công cụ cho phép bạn xây dựng các virtual DOMs và host chúng trong bộ nhớ. Nhờ vậy, mỗi khi có sự thay đổi trong DOM thực tế, thì virtual sẽ thay đổi ngay lập tức.

Hệ thống này sẽ ngăn DOM thực tế để buộc các bản cập nhật được liên tục. Do đó, tốc độ của ứng dụng sẽ không bị gián đoạn.

\* Thân thiện với SEO

React cho phép bạn tạo giao diện người dùng có thể được truy cập trên các công cụ tìm kiếm khác nhau. Tính năng này là một lợi thế rất lớn vì không phải tất cả các khung JavaScript đều thân thiện với SEO.

Ngoài ra, vì React có thể tăng tốc quá trình của ứng dụng nên có thể cải thiện kết quả SEO. Cuối cùng tốc độ web đóng một vai trò quan trọng trong tối ưu hóa SEO.

Tuy nhiên, bạn cần lưu ý rằng React chỉ là một thư viện JavaScript. Nghĩa là nó không thể tự làm mọi thứ. Sử dụng các thư viện bổ sung có thể cần thiết cho các mục tiêu quản lý, định tuyến và tương tác.

### 1.1.6 Tồn tại của ReactJS

Reactjs chỉ phục vụ cho tầng View. React chỉ là View Library nó không phải là một MVC framework như những framework khác. Đây chỉ là thư viện của Facebook giúp render ra phần view. Vì thế React sẽ không có phần Model và Controller, mà phải kết hợp với các thư viện khác.

Tích hợp Reactjs vào các framework MVC truyền thống yêu cầu cần phải cấu hình lại.

React khá nặng nếu so với các framework khác React có kích thước tương tương với Angular (Khoảng 35kb so với 39kb của Angular). Trong khi đó Angular là một framework hoàn chỉnh.

Khó tiếp cận cho người mới học Web.

### 1.1.7 Cài đặt 1 project ReactJS

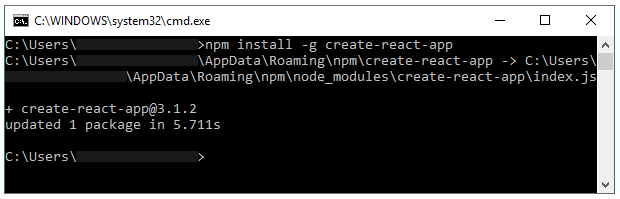
Bước 1 - Cài đặt môi trường Node.js:

Truy cập vào link https://nodejs.org/en/ để download và cài đặt theo hướng dẫn.

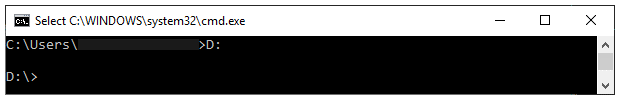
Bước 2 - Cài đặt bộ cài app React:

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

npm install -g create-react-app



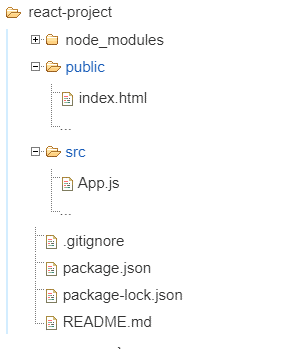
Bước 3 - Tạo project React.js trong ổ đĩa D với tên react-project:



Chạy lệnh tạo project với tên react-project (tên project tùy ý) như bên dưới:

create-react-app react-project

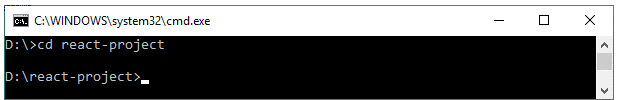
Khi này sẽ xuất hiện một thư mục react-project bên trong ổ đĩa D, chứa các file và thư mục như sau:



Bước 4 – Thực thi lệnh chạy:

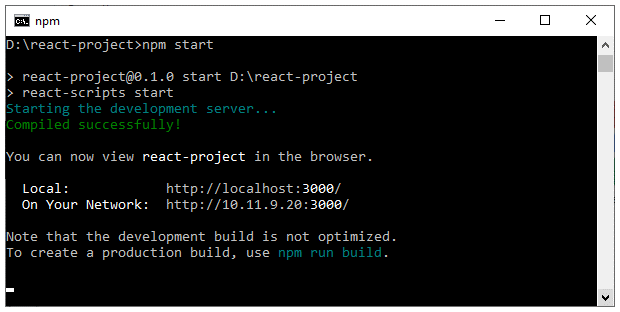
Ta lần lượt tiến hành: trỏ vào thư mục react-project với lệnh cmd sau:

cd react-project

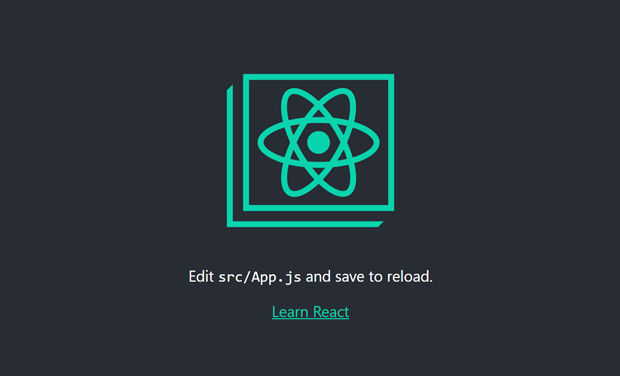


Thực thi chạy project React bằng lệnh:

npm start



Khi này ta thấy ứng dụng React sẽ tự động mở một trang chạy trên trình duyệt với đường dẫn: http://localhost:3000/, kết quả hiển thị như bên dưới là bạn đã cài đặt thành công.



## 1.2 Tổng quan về Laravel

### 1.2.1 Khái niệm về Laravel

Laravel là một PHP Framework mã nguồn mở và miễn phí, được phát triển bởi Taylor Otwell và nhắm mục tiêu hỗ trợ phát triển các ứng dụng web theo cấu trúc model- view- controller (MVC). Những tính năng nổi bật của Laravel bao gồm cú pháp dễ hiểu- rõ ràng, một hệ thống đóng gói Modular và quản lý gói phụ thuộc, nhiều cách khác nhau để truy cập vào các cơ sở dữ liệu quan hệ, nhiều tiện ích khác nhau hỗ trợ việc triển khai vào bảo trì ứng dụng.

Laravel là một framework rõ ràng và ưu việt cho việc phát triển web PHP. Giải thoát bạn khỏi mã spaghetti, nó giúp bạn tạo ra những ứng dụng tuyệt vời, sử dụng syntax đơn giản. Phát triển là thích thú những trải nghiệm sáng tạo chứ không phải là sự khó chịu.

### 1.2.2 Lịch sử phát triển của Laravel

Bản Laravel beta đầu tiên được phát hành vào ngày 9/6/2011, tiếp đó là Laravel 1 phát hành trong cùng tháng. Laravel 1 bao gồm các tính năng như xác thực, bản địa hóa, model, view, session, định tuyến và các cơ cấu khác, nhưng vẫn còn thiếu controller, điều này làm nó chưa thật sự là một MVC framework đúng nghĩa.

Laravel 2 được phát hành vào tháng 9 năm 2011, mang đến nhiều cài tiến từ tác giả và cộng đồng. Tính năng đáng kể bao gồm hỗ trợ controller, điều này thực sự biến Laravel 2 thành một MVC framework hoàn chỉnh, hỗ trợ Inversion of Control (IoC), hệ thống template Blade. Bên cạnh đó, có một nhược điểm là hỗ trợ cho các gói của nhà phát triển bên thứ 3 bị gỡ bỏ.

Laravel 3 được phát hành vào tháng 2 năm 2012, với một tấn tính năng mới bao gồm giao diện dòng lệnh (CLI) tên “Artisan”, hỗ trợ nhiều hơn cho hệ thống quản trị cơ sở dữ liệu, chức năng ánh xạ cơ sở dữ liệu Migration, hỗ trợ “bắt sự kiện” trong ứng dụng và hệ thống quản lý gói gọi là “Bundles”. Lượng người dùng và sự phổ biến tăng trưởng mạnh kể từ phiên bản Laravel 3.

Laravel 4, tên mã “Illuminate”, được phát hành vào tháng 5 năm 2013. Lần này thực sự là sự lột xác của Laravel framework, di chuyển và tái cấu trúc các gói hỗ trợ vào một tập được phân phối thông qua Composer, một chương trình quản lý gói thư viện phụ thuộc độc lập của PHP. Bố trí mới như vậy giúp khả năng mở rộng của Laravel 4 tốt hơn nhiều so với các phiên bản trước. Ra mắt lịch phát hành chính thức mỗi sáu tháng một phiên bản nâng cấp nhỏ.

Laravel 5 được phát hành trong tháng 2 năm 2015, như một kết quả thay đổi đáng kể cho việc kết thúc vòng đời nâng cấp Laravel lên 4.3. Bên cạnh một loạt tính năng mới và các cải tiến như hiện tại, Laravel 5 cũng giới thiệu cấu trúc cây thư mục nội bộ cho phát triển ứng dụng mới. Những tính năng mới của Laravel 5 bao gồm hỗ trợ lập lịch định kỳ thực hiện nhiệm vụ thông qua một gói tên là “Scheduler”, một lớp trừu tượng gọi là “Flysystem” cho phép điều khiển việc lưu trữ từ xa đơn giản như lưu trữ trên máy local – dễ thấy nhất là mặc định hỗ trợ dịch vụ Amazon S3, cải tiến quản lý assets thông qua “Elixir”, cũng như đơn giản hóa quản lý xác thực với các dịch vụ bên ngoài bằng gói “Socialite”.

Laravel 5.1 phát hành vào tháng 6 năm 2015, là bản phát hành đầu tiên nhận được hỗ trợ dài hạn (LTS) với một kế hoạch fix bug lên tới 2 năm vào hỗ trợ vá lỗi bảo mật lên tới 3 năm. Các bản phát hành LTS của Laravel được lên kế hoạch theo mỗi 2 năm.

Laravel 5.3, được phát hành vào ngày 23 tháng 8 năm 2016. Các tính năng mới trong 5.3 tập trung vào việc cải thiện tốc độ phát triển bằng cách bổ sung thêm các cải tiến cho các tác vụ phổ biến.

Laravel 5.4 Phiên bản này có nhiều tính năng mới, như Laravel Dusk, Laravel Mix, Blade Components và Slots, Markdown Emails, Automatic Facades, Route Improvements, Higher Order Messaging cho Collections, và nhiều thứ khác.

Laravel 5.5, phát hành vào ngày 30 tháng 8 năm 2017 là phiên bản LTS thứ 2

Laravel 5.6, phát hành vào ngày 7 tháng 2 năm 2018.

Laravel 5.7, phát hành vào ngày 4 tháng 9 năm 2018 với những cập nhật: Cải thiện thông báo lỗi, Callable Action URLs, Email Verification.

Laravel 6, phát hành vào ngày 3 tháng 9 năm 2019. Đây là version LTS với những cập nhật mới: Đổi versioning scheme sang Semantic Versioning, Cải thiện Exceptions thông qua Ignition, Cải thiện Authorization Responses.

Laravel 7, ra mắt ngày 3 tháng 3 năm 2020 với nhiều tính năng cũng như cải thiện tốc độ.

### 1.2.3 Đặc tính và tính năng nổi bật của Laravel

Laravel ứng dụng mô hình MVC. MVC (Model-View-Controller) là mẫu kiến trúc phần mềm trên máy tính nhằm mục đích tạo lập giao diện cho người dùng. Theo đó, hệ thống MVC được chia thành ba phần có khả năng tương tác với nhau và tách biệt các nguyên tắc nghiệp vụ với giao diện người dùng. Ba thành phần ấy bao gồm:

- Controller: Giữ nhiệm vụ nhận điều hướng các yêu cầu từ người dùng và gọi đúng những phương thức xử lý chúng.

- Model: Là thành phần chứa tất cả các nghiệp vụ logic, phương thức xử lý, truy xuất database, đối tượng mô tả dữ liệu như các Class, hàm xử lý…

- View: Đảm nhận việc hiển thị thông tin, tương tác với người dùng, nơi chứa tất cả các đối tượng GUI như textbox, images…

Laravel có 3 đặc tính nổi trội:

- Đơn giản: các chức năng của Laravel rất dễ hiểu và thực hiện.

- Ngắn gọn: hầu hết các chức năng của Laravel hoạt động liên tục với cấu hình rất nhỏ, dựa vào các quy tắc chuẩn để giảm bớt code-bloat

- Trình bày hợp lý: hướng dẫn sử dụng Laravel rất đầy đủ và luôn cập nhật. Nhà lập trình, người tạo ra framework luôn cập nhật tài liệu trước khi cho ra một phiên bản mới, đảm bảo những người học lập trình luôn luôn có những tài liệu mới nhất.

Những tính năng tuyệt vời có thể kể đến của Laravel là:

- Composer: sử dụng để nâng cấp, cài đặt…

- Eloquent ORM: thao tác với cú pháp đẹp mắt và đơn giản.

- Restful API: hỗ trợ biến Laravel thành một web service API.

- Artisan: cung cấp các lệnh cần thiết để phát triển ứng dụng.

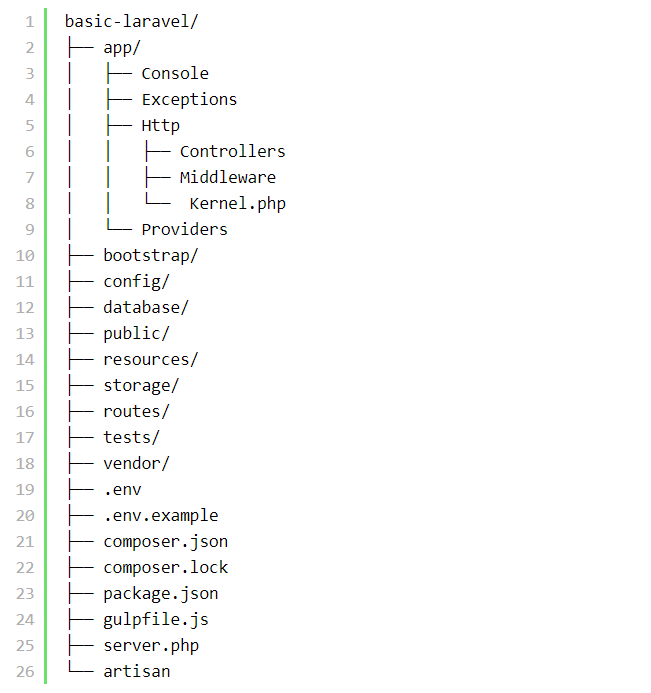
- View: giúp code sạch sẽ hơn rất nhiều.

- Migrations: hỗ trợ tạo các trường trong cơ sở dữ liệu, thêm các cột trong bảng, tạo mối quan hệ giữa các bảng, hỗ trợ quản lý cơ sở dữ liệu.

- Authentication: cung cấp sẵn các tính năng đăng nhập, đăng ký, quên mật khẩu…

- Unit Testing: hỗ trợ test lỗi để sửa chữa.

### 1.2.4 Cấu trúc của Laravel



Hình 1.7 Cấu trúc của Laravel

**app** là thư mục chứa tất cả các thư mục, các tập tin php, các class php, thư viện, models để xây dựng project của bạn:

**Console** thư mục chứa các tập tin định nghĩa các lệnh thực thi trên Artisan.

**Exception** thư mục chứa các tập tin quản lý, điều hướng lỗi.

**Http:**

**- Controllers** là thư mục chứa các tập tin controllers

**- Middleware** là thư mục chứa các tập tin lọc và ngăn chặn các requests.

**- Kernel.php** là tập tin cấu hình, định nghĩa Middleware hoặc nhóm Middleware.

**Providers** chứa các providers mình sẽ nói rõ về nó trong các bài nâng cao.

**bootstrap** thư mục chứa tập tin điều hướng khởi động hệ thống, thường thì chúng ta sẽ không làm gì đến nó.

**config** chứa mọi tập tin cấu hình của Laravel

**database** chứa các thư mục tập tin về CSDL:

**migrations** chứa các tập tin định nghĩa khởi tạo và sử bảng.

**seeds** chứa các tập tin định nghĩa dữ liệu insert vào database.

**factories** chứa các tập tin định nghĩa các cột bảng dữ liệu để tạo ra các dữ liệu ảo phục vụ cho tests.

**public** chính là webroot người dùng sẽ truy cập vào đây, đây cũng là nơi chứa các tập tin css, js, image.

**resources** chứa các tập tin giao diện (js, css, less, sass, coffeescript,...), views, ngôn ngữ.

**storage** chứa các tập tin hệ thống như upload, cache, session, cookie, log...

**routes** là thư mục chứa các tập tin định nghĩa các router, xử lý router hoặc điều hướng router (tức là URL, laravel không tự đặt url theo kiểu example.com/controller/action/value mà chúng ta phải tự định nghĩa chúng) bao gồm 3 loại là web, api và console.

**tests** chứa các tập tin định nghĩa tests.

**vendor** thư mục của composer.

**.env** và **.env.example** là 2 tập tin cấu hình chính của laravel như key app, tên app, url app, email, env mode, CSDL hay bật tắt debug.

**composer.json**, **composer.lock** tập tin của composer.

**package.js** tập tin cấu hình của nodejs chứa các package cần thiết cho projects.

**gulpfile.js** là tập tin gulp builder.

**phpunit.xml** là tập tin xml của phpunit dùng để testing project.

**server.php** là tập tin để artisan trỏ đến tạo server khi gõ lệnh php artisan serve

**artisan** tập tin thực thi lệnh của Laravel, cũng là tập tin mà chúng ta tương tác nhiều nhất.

### 1.2.5 Ưu điểm và nhược điểm của Laravel

\* Ưu điểm:

Sử dụng các tính năng mới nhất của PHP: sử dụng Laravel 5 giúp các lập trình viên tiếp cận những tính năng mới nhất mà PHP cung cấp, nhất là đối với Namespaces, Interfaces, Overloading, Anonymous functions và Shorter array syntax.

Nguồn tài nguyên vô cùng lớn và sẵn có: nguồn tài nguyên của Laravel rất thân thiện với các lập trình viên với đa dạng tài liệu khác nhau để tham khảo. Các phiên bản được phát hành đều có nguồn tài liệu phù hợp với ứng dụng của mình.

Tích hợp với dịch vụ email: Laravel là framework được trang bị API sạch trên thư viện SwiftMailer, do đó, bạn có thể gửi thư qua các dịch vụ dựa trên nền tảng đám mây hoặc local.

Tốc độ xử lý nhanh: Laravel hỗ trợ hiệu quả cho việc tạo lập website hay các dự án lớn trong thời gian ngắn. Vì vậy, nó được các công ty công nghệ và lập trình viên sử dụng rộng rãi để phát triển các sản phẩm của họ.

Dễ sử dụng: Laravel được đón nhận và trở nên sử dụng phổ biến vì nó rất dễ sử dụng. Thường chỉ mất vài giờ, bạn có thể bắt đầu thực hiện một dự án nhỏ với vốn kiến thức cơ bản nhất về lập trình với PHP.

Tính bảo mật cao: Laravel cung cấp sẵn cho người dùng các tính năng bảo mật mạnh mẽ để người dùng hoàn toàn tập trung vào việc phát triển sản phẩm của mình:

- Sử dụng PDO để chống lại tấn công SQL Injection.

- Sử dụng một field token ẩn để chống lại tấn công kiểu CSRF.

Mặc định đều được Laravel escape các biến được đưa ra view mặc định, do đó có thể tránh được tấn công XSS.

\* Nhược điểm:

So với các PHP framework khác, Laravel bộc lộ khá ít nhược điểm. Vấn đề lớn nhất có thể kể đến của framework này là thiếu sự liên kết giữa các phiên bản, nếu cố cập nhật code, có thể khiến cho ứng dụng bị gián đoạn hoặc phá vỡ.

Bên cạnh đó, Laravel cũng quá nặng cho ứng dụng di động, khiến việc tải trang trở nên chậm chạp.

### 1.2.6 Cài đặt Laravel và chạy Laravel

\* Cách 1: Thông qua Laravel Installer

Đây là một thư viện của Laravel viết ra giúp bạn cài đặt Laravel bởi 1 câu lệnh duy nhất, các bạn cài đặt thông qua Composer nhé. Bạn mở Terminal (CMD hoặc Git Bash) ở bất kỳ vị trí nào và gõ dòng lệnh sau:

composer global require "laravel/installer"

Khi cài đặt xong bạn cần chắc chắn rằng đường dẫn tập tin thực thi của Composer đã được thêm vào Windows Environment Variables Path.Đối với Windows, đường dẫn đó là "%appdata%\Composer\vendor\bin" và đối với macOS và Linux thì nó ở "~/.composer/vendor/bin".

Sau khi cài đặt xong, chúng ta sẽ di chuyển vào thư mục htdocs của XAMPP, tại đây các bạn mở cửa sổ lệnh (đối với windows thì nhấn Shift + chuột phải và chọn Command Window Here hoặc Git Bash Here) và gõ dòng lệnh sau:

laravel new blog

Trong đó blog chính là tên thư mục laravel project của bạn. Vậy là chúng ta đã cài đặt xong Laravel rồi đấy.

\* Cách 2: Thông qua Composer

Chúng ta sẽ di chuyển thẳng vào thư mục htdocs của XAMPP, tại đây các bạn mở cửa sổ lệnh (như trên) và gõ dòng lệnh sau:

composer create-project --prefer-dist laravel/laravel blog

Trong đó blog chính là tên thư mục laravel project của bạn. Vậy là chúng ta đã cài đặt xong Laravel rồi đấy.

\* Chạy Laravel

Sau khi cài đặt hoàn tất, bạn chỉ cần mở WebServer của bạn lên và chạy đến thư mục public trong thư mục Laravel project của các bạn hoặc thực thi lệnh sau từ thư mục Laravel project của các bạn.

php artisan serve

Khi đó trên màn hình console xuất hiện thông báo:

Laravel development server started on http://localhost:8000/

Tức là Laravel project của bạn đã khởi chạy bạn vào trình duyệt gõ: http://localhost:8000

Và nếu thấy như hình tức là bạn đã chạy Laravel thành công.

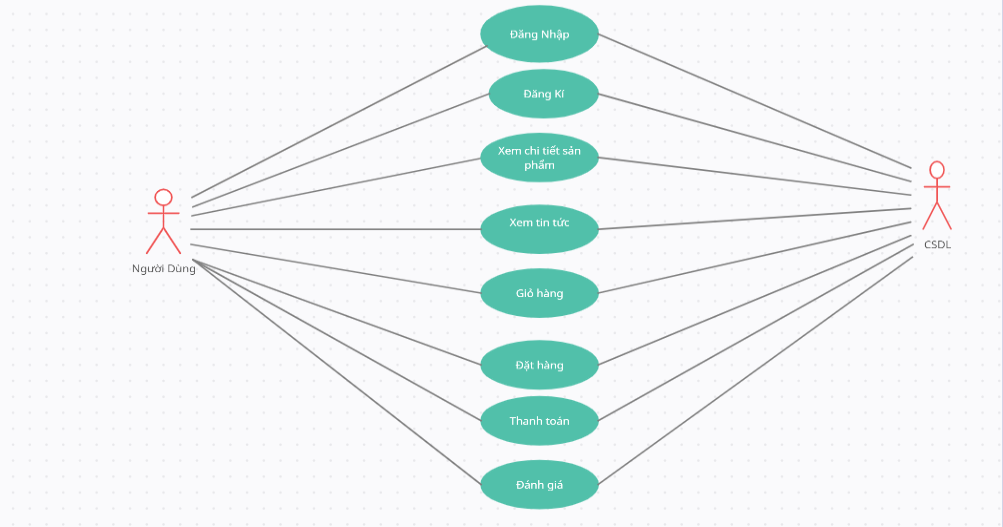
# Chương 2: Cài đặt và xây dựng website bán dép cho cửa hàng Đắc Quang

## **2. 1** : Sơ đồ ca sử dụng :

* Qua khảo sát hiện trạng thực tế, kết hợp với việc tham khảo tài liệu và phân tích ý kiến của người quản trị website bán dụng dép Đắc Quang có các thành phần:
* Tác nhân : Người quản trị, người dùng.
* Các use case gồm có:
* Người quản trị: UC đăng nhập, UC quản trị thông tin hãng sản xuất , UC quản trị danh mục sản phẩm , UC quản trị thông tin sản phẩm, UC quản trị đơn hàng , UC quản trị thông tin khách hàng, UC quản trị tài khoản người dùng, UC quản trị tin tức
* Người dùng: UC đăng nhập, UC đăng ký, UC giỏ hàng, UC thanh toán, UC xem thông tin xem tin tức, UC xem chi tiết sản phẩm.

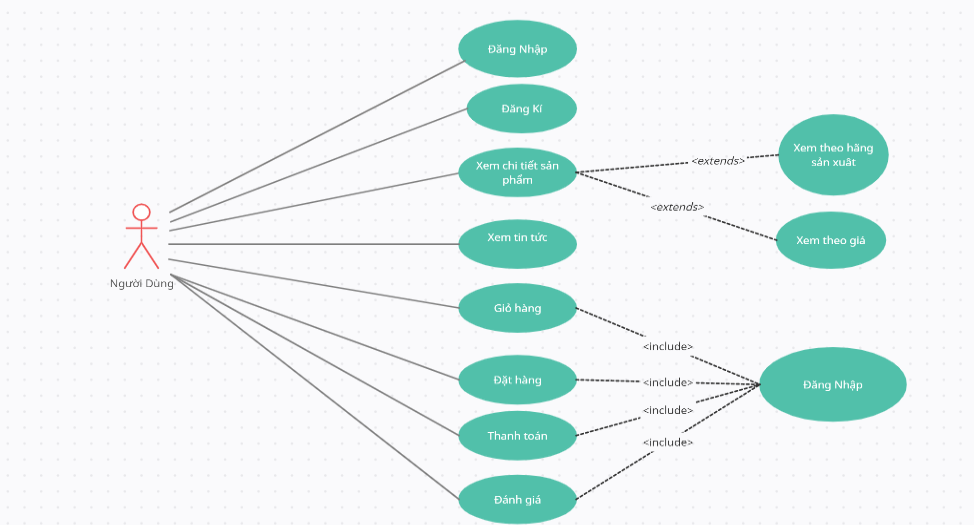
### 2.1.1 Sơ đồ ca sử dụng của khách hàng:

* Mô hình hóa Use Case:

****

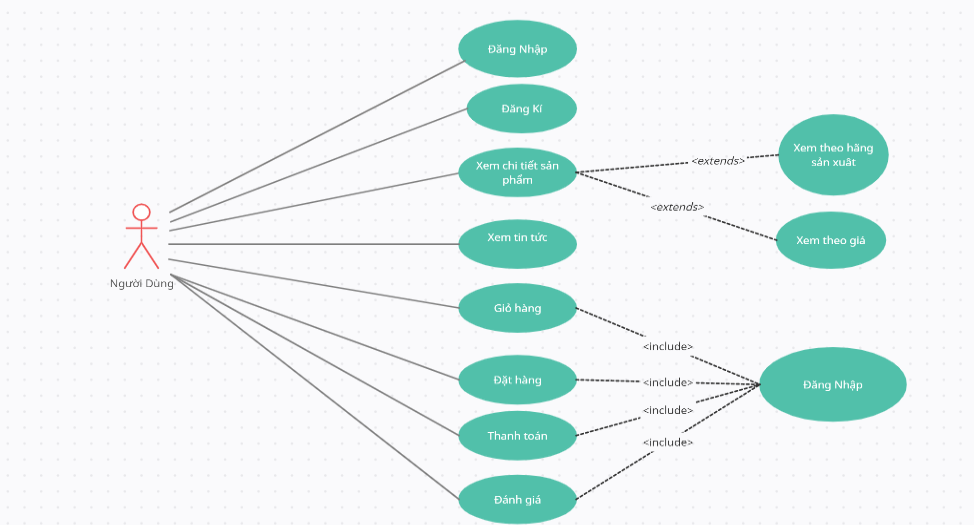
Hình 2.1 : Sơ đồ ca sử dụng của khách hàng

* Phân rã một số usecase
* Quan hệ include



Hình 2.2 : Sơ đồ ca sử dụng của khách hàng ( include)

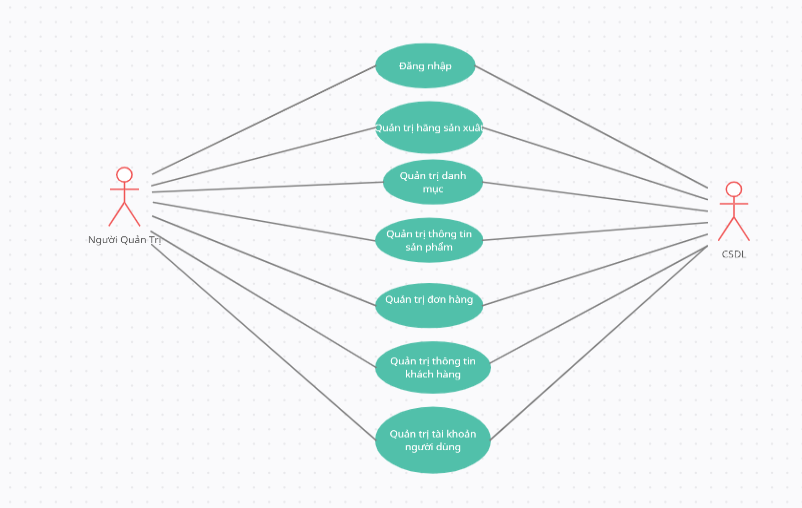
* Quan hệ extend:



Hình 2.3 : Sơ đồ ca sử dụng của khách hàng ( extends)

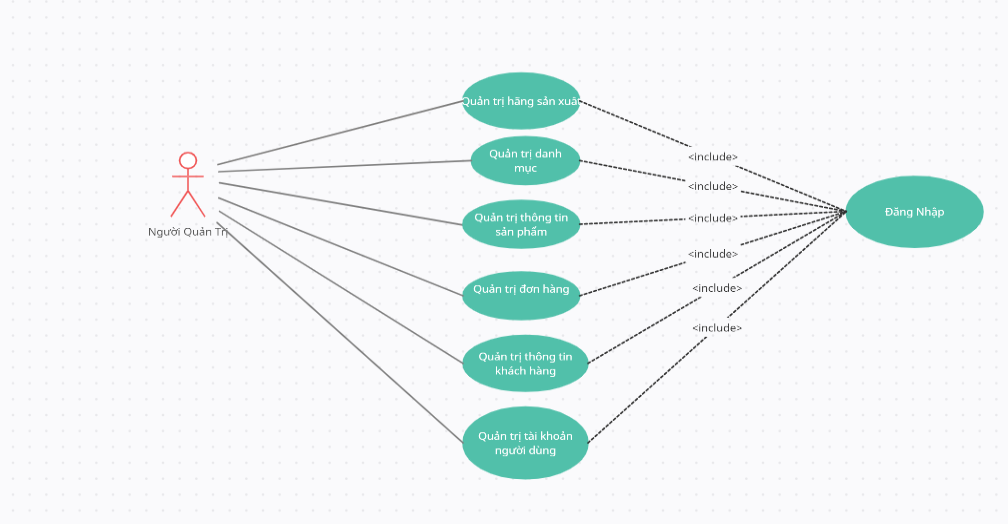
### 2.1.2 Sơ đồ ca sử dụng của người quản trị:

* Mô hình hóa usecase:



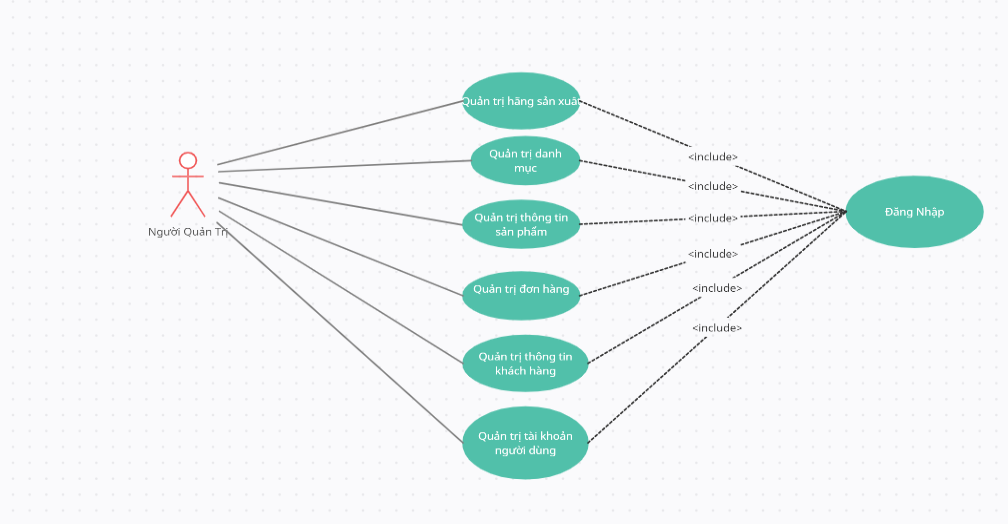
Hình 2.4 : Sơ đồ ca sử dụng của người quản lý

* Phân rã một số usecase
* Quan hệ include



Hình 2.5 : Sơ đồ ca sử dụng của người quản trị( include)

* Quan hệ extends:



Hình 2.6 : Sơ đồ ca sử dụng của người quản trị ( extends)

### Đặc tả usecase người quản trị :

* **Đăng nhập :**
* Usecase này cho phép người quản trị đăng nhập vào trang chủ admin dựa theo phân quyền của tài khoản.
* Luồng sụ kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị muốn truy cập vào website dành cho người quản trị.
* Hệ thống yêu cầu người quản trị nhập email, mật khẩu.
* Hệ thống kiểm tra tính hợp lệ của email, mật khẩu đăng nhập. Dựa vào phân quyền của tài khoản mà điều hướng trang web (Nếu phân quyền admin thì sau khi đăng nhập sẽ điều hướng về trang chủ quản lý của admin. Các phân quyền còn lại không thể đăng nhập mà sẽ điều hướng trở lại trang đăng nhập).
* Luồng rẽ nhánh:
* Nếu trong dòng sự kiện chính, người quản trị cung cấp email, mật khẩu không hợp lệ hoặc quyền truy cập không phải admin, hệ thống sẽ thông báo lỗi. Người quản trị có thể chọn tiếp tục đăng nhập lại theo dòng sự kiện chính hoặc hủy bỏ việc đăng nhập để kết thúc chức năng đăng nhập.
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần phải có tài khoản với phân quyền admin trước đó.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Các mục quản trị
* **Quản trị thương hiệu:**
* Use case này cho phép người quản trị thao tác thêm mới, sửa xóa thông tin của thương hiệu.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “THƯƠNG HIỆU” trên menu quản trị. Hệ thống lấy thông tin branches\_id, branches\_name, branches\_image, branches\_type, branches\_code từ bảng “branches” trong CSDL rồi hiển thị ra màn hình
* Thêm: Người quản trị click “Thêm mới” trên màn hình. Hệ thống hiển thị form thêm mới thương hiệu trên màn hình. Người quản trị điền đầy đủ thông tin của nhãn hiệu sau đó click nút “Cập nhật”. Hệ thống sẽ tạo một bản ghi mới trong bảng “branches” và hiển thị thông báo.
* Sửa: Người quản trị click button “edit” trong cột “Thao tác” của nhãn hàng cần chỉnh sửa trong bảng hiển thị nhãn hàng. Hệ thống kiểm tra id của nhãn hàng muốn chỉnh sửa, lấy thông tin của nhãn hàng đó từ bảng “branches” rồi hiển thị lên màn hình thông qua form. Người quản trị chỉnh sửa thông tin của nhãn hàng rồi click “update”. Hệ thống sẽ kiểm tra thông tin rồi cập nhật thông tin nhãn hàng trong bảng “branches” và hiển thị thông báo.
* Xóa: Người quản trị click button “delete” trong cột “Thao tác” của nhãn hàng cần xóa trong bảng hiển thị nhãn hàng trên màn hình. Hệ thống sẽ xóa nhãn hàng khỏi bảng “branches” và hiển thị lại danh sách đã được cập nhật.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Tại chức năng thêm nhãn hàng, nếu người quản trị điền thiếu thông tin trong form hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi và người quản trị sẽ phải nhập lại thông tin trong form.
* Tại chức năng sửa nhãn hàng, nếu người quản trị điền thiếu thông tin hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi để người quản trị nhập lại thông tin.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Sau khi xóa nhãn hàng, cần hiển thị bảng dữ liệu nhãn hàng sau khi đã cập nhật.
* Sau các chức năng thêm và sửa cần hiển thị thông báo thành công nếu thực hiện thành công, thông báo lỗi nếu có lỗi xảy ra.
* Điểm mở rộng:
* Không có.
* **Quản trị loại sản phẩm:**
* Use case này cho phép người quản trị thao tác xem hiển thị, trạng thái của Loại sản phẩm.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị tại trang chủ click “LOẠI SẢN PHẨM” trên menu quản trị. Hệ thống lấy thông tin categories\_id, categories\_name, categories\_metatitle, categories\_parentid, categories\_displayorder, categories\_showonhome, categories\_status từ bảng “categories” trong CSDL rồi hiển thị ra màn hình
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Quản trị sản phẩm:**
* Use case này cho phép người quản trị thao tác thêm mới, sửa xóa thông tin của sản phẩm.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “SẢN PHẨM” trên menu quản trị. Hệ thống lấy thông tin product\_id, size\_id, product\_name, product\_code, product\_metatitle, product\_description, product\_more\_image, product\_image, product\_promotion, product\_includedvat, product\_price, product\_quantity, product\_category\_id, product\_detail, product\_status, product\_viewcount, product\_rate, product\_material, product\_size, sex từ bảng “products” trong CSDL rồi hiển thị ra màn hình
* Thêm: Người quản trị click “Thêm mới” trên màn hình. Hệ thống hiển thị form “Quản lý sản phẩm” trên màn hình. Người quản trị điền đầy đủ thông tin của sản phẩm sau đó click nút “Tạo mới”. Hệ thống sẽ tạo một bản ghi mới trong bảng “products” và hiển thị thông báo.
* Sửa: Người quản trị click button “edit” trong cột “Action” của sản phẩm cần chỉnh sửa trong bảng hiển thị sản phẩm. Hệ thống kiểm tra id của sản phẩm muốn chỉnh sửa, lấy thông tin của sản phẩm đó từ bảng “products” rồi hiển thị lên màn hình thông qua form. Người quản trị chỉnh sửa thông tin của sản phẩm rồi click “update”. Hệ thống sẽ kiểm tra thông tin rồi cập nhật thông tin sản phẩm trong bảng “products” và hiển thị thông báo.
* Xóa: Người quản trị click button “delete” trong cột “Action” của sản phẩm cần xóa trong bảng hiển thị sản phẩm trên màn hình. Hệ thống sẽ xóa sản phẩm khỏi bảng “products” và hiển thị lại danh sách đã được cập nhật.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Tại chức năng thêm sản phẩm, nếu người quản trị điền thiếu thông tin trong form hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi và người quản trị sẽ phải nhập lại thông tin trong form.
* Tại chức năng sửa sản phẩm, nếu người quản trị điền thiếu thông tin hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi để người quản trị nhập lại thông tin.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Sau khi xóa sản phẩm, cần hiển thị bảng dữ liệu nhãn hàng sau khi đã cập nhật.
* Sau các chức năng thêm và sửa cần hiển thị thông báo thành công nếu thực hiện thành công, thông báo lỗi nếu có lỗi xảy ra.
* Điểm mở rộng:
* Không có.
* **Quản trị đơn hàng:**
* Use case này cho phép người quản trị thao tác xem, xuất Excel thông tin đơn hàng.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “ĐƠN HÀNG” trên menu quản trị. Hệ thống lấy thông tin orders\_id, product\_size\_id, user\_id, shipplace\_id, orders\_status, orders\_quantity, product\_price, created\_at, orders\_type, product\_cost từ bảng “orders” trong CSDL rồi hiển thị ra màn hình
* Xuất Excel: Người quản trị click button “Xuất Excel” trên màn hình. Hệ thống sẽ xuất ra 1 bản Excel dựa trên các sản phẩm đã được người dùng mua.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Sau khi click “Xuất Excel” cần hiển thị thông báo thành công nếu thực hiện thành công, thông báo lỗi nếu có lỗi xảy ra.
* Điểm mở rộng:
* Không có.
* **Quản trị thông tin khách hàng:**
* Use case này cho phép người quản trị thao tác thêm mới, sửa xóa thông tin của khách hàng.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “KHÁCH HÀNG” trên menu quản trị. Hệ thống lấy thông tin id, username, password, groupid, name, address, email, birthday, about\_me, background\_image, hash, phone, token, like\_web, user\_image, sex từ bảng “users” trong CSDL rồi hiển thị ra màn hình
* Thêm: Người quản trị click “Thêm mới” trên màn hình. Hệ thống hiển thị form thêm mới khách hàng trên màn hình. Người quản trị điền đầy đủ thông tin của khách hàng sau đó click nút “Cập nhật”. Hệ thống sẽ tạo một bản ghi mới trong bảng “users” và hiển thị thông báo.
* Sửa: Người quản trị click button “edit” trong cột “Thao tác” của khách hàng cần chỉnh sửa trong bảng hiển thị khách hàng. Hệ thống kiểm tra id của khách hàng muốn chỉnh sửa, lấy thông tin của khách hàng đó từ bảng “users” rồi hiển thị lên màn hình thông qua form. Người quản trị chỉnh sửa thông tin của khách hàng rồi click “update”. Hệ thống sẽ kiểm tra thông tin rồi cập nhật thông tin khách hàng trong bảng “users” và hiển thị thông báo.
* Xóa: Người quản trị click button “delete” trong cột “Thao tác” của khách hàng cần xóa trong bảng hiển thị khách hàng trên màn hình. Hệ thống sẽ xóa khách hàng khỏi bảng “users” và hiển thị lại danh sách đã được cập nhật.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Tại chức năng thêm khách hàng, nếu người quản trị điền thiếu thông tin trong form hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi và người quản trị sẽ phải nhập lại thông tin trong form.
* Tại chức năng sửa khách hàng, nếu người quản trị điền thiếu thông tin hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi để người quản trị nhập lại thông tin.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Sau khi xóa khách hàng, cần hiển thị bảng dữ liệu khách hàng sau khi đã cập nhật.
* Sau các chức năng thêm và sửa cần hiển thị thông báo thành công nếu thực hiện thành công, thông báo lỗi nếu có lỗi xảy ra.
* Điểm mở rộng:
* Không có.
* **Quản trị tài khoản:**
* Use case này cho phép người quản trị thao tác thêm mới, sửa, xóa thông tin của tài khoản.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “TÀI KHOẢN” trên menu quản trị. Hệ thống lấy thông tin id, username, password, groupid, name, address, email, birthday, about\_me, background\_image, hash, phone, token, like\_web, user\_image, sex từ bảng “users” trong CSDL rồi hiển thị ra màn hình
* Thêm: Người quản trị click “Thêm mới” trên màn hình. Hệ thống hiển thị form “thêm mới tài khoản” trên màn hình. Người quản trị điền đầy đủ thông tin của tài khoản sau đó click nút “Cập nhật”. Hệ thống sẽ tạo một bản ghi mới trong bảng “users” và hiển thị thông báo.
* Sửa: Người quản trị click button “edit” trong cột “Action” của tài khoản cần chỉnh sửa trong bảng hiển thị tài khoản. Hệ thống kiểm tra id của tài khoản muốn chỉnh sửa, lấy thông tin của tài khoản đó từ bảng “users” rồi hiển thị lên màn hình thông qua form. Người quản trị chỉnh sửa thông tin của tài khoản rồi click “update”. Hệ thống sẽ kiểm tra thông tin rồi cập nhật thông tin tài khoản trong bảng “users” và hiển thị thông báo.
* Xóa: Người quản trị click button “delete” trong cột “Action” của tài khoản cần xóa trong bảng hiển thị tài khoản trên màn hình. Hệ thống sẽ xóa tài khoản khỏi bảng “users” và hiển thị lại danh sách đã được cập nhật.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Tại chức năng thêm tài khoản, nếu người quản trị điền thiếu thông tin trong form hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi và người quản trị sẽ phải nhập lại thông tin trong form.
* Tại chức năng sửa tài khoản, nếu người quản trị điền thiếu thông tin hoặc thông tin điền vào không hợp lệ thì hệ thống sẽ thông báo lỗi để người quản trị nhập lại thông tin.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Sau khi xóa tài khoản, cần hiển thị bảng dữ liệu tài khoản sau khi đã cập nhật.
* Sau các chức năng thêm và sửa cần hiển thị thông báo thành công nếu thực hiện thành công, thông báo lỗi nếu có lỗi xảy ra.
* Điểm mở rộng:
* Không có.
* **Thống kê:**
* Use case này cho phép người quản trị thống kê những sản phẩm được mua gần nhất , lọc theo ngày mua.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người quản trị ở trang chủ click “THỐNG KÊ” trên menu quản trị. Hệ thống lấy thông tin những sản phẩm được mua trong một khoảng thời gian được chọn , tổng hợp lại số lượng rồi hiển thị ra màn hình một sơ đồ cột tương ứng với số sản phẩm đã mua.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người quản trị cần đăng nhập trước đó.
* Hậu điều kiện:
* Không có
* Điểm mở rộng:
* Không có.
  + 1. **Đặc tả usecase Khách Hàng**
* **Giỏ Hàng**
* Usecase này cho phép người dùng xem giỏ hàng, thêm sản phẩm vào giỏ hàng, sửa số lượng sản phẩm, xóa sản phẩm khỏi giỏ hàng.
* Luồng sự kiện:
* Luồng cơ bản:
* Mở giỏ hàng:

Usecase này bắt đầu khi người dùng đưa con trỏ chuột đến icon giỏ hàng, trang web hover vắn tắt thông tin giỏ hàng cùng chức năng xóa thông tin sản phẩm.

Người dùng click vào nút “View Card”, hệ thống lấy thông tin ảnh, tên sản phẩm, giá, số lượng, tổng tiền được lưu trên session rồi hiển thị ra màn hình.

* Thêm giỏ hàng:

Use case này bắt đầu khi người dùng click vào icon giỏ hàng cảu sản phẩm mình muốn thêm. Hệ thống sẽ lưu thông tin tên sản phẩm, giá và cập nhật lại số lượng của sản phẩm đó trong giỏ hàng lên 1 sản phẩm nữa.

* Cập nhật giỏ hàng:

Khi người dùng đang ở trang xem giỏ hàng, họ có thể chỉnh sửa thông tin số lượng của sản phẩm mình muốn mua. Người dùng điều chỉnh số lượng sau đó nhấn icon save để lưu thông tin cho từng sản phẩm hoặc nhấn nút “Update all” để cập nhật lại số lượng cho toàn bộ sản phẩm có mặt trong giỏ hàng.

* Xóa giỏ hàng:

Có hai hướng để xóa giỏ hàng:

Người dùng có thể hover vào icon giỏ hàng trên trang web, click nút “X” có trong form rút gọn của giỏ hàng. Hệ thống nhận yêu cầu sau đó sẽ xóa thông tin của sản phẩm đó trên session (bao gồm tên sản phẩm, giá sản phẩm, hình ảnh) và cập nhật lại số lướng ản phẩm trong giỏ hàng, cập nhật lại tổng tiền được hiển thị trên form rút gọn.

Người dùng hover vào icon giỏ hàng trên trang web, sau đó click nút “View card” để vào trang giỏ hàng. Người dùng click nút “X” trong form hiển thị thông tin các sản phẩm trong giỏ hàng.

* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Không có.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:

Không có.

* **Xem chi tiết sản phẩm**
* Usecase này cho phép người dùng xem được các thông tin chi tiết của sản phẩm.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người dùng click vào hình ảnh sản phẩm hoặc tên của sản phẩm. Hệ thống sẽ lấy thông tin id, name, id\_type\_product, description, unit\_price, promotion\_price (nếu >0), image, unit của bảng “products” và các image trong bảng “image” hiển thị ra màn hình.
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Không có.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Đăng nhập :**
* Usecase này cho phép người khách hàng đăng nhập vào hệ thống dựa theo phân quyền của tài khoản.
* Luồng sụ kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người khách hàng muốn truy cập vào website.
* Hệ thống yêu cầu người khách hàng nhập email, mật khẩu.
* Hệ thống kiểm tra tính hợp lệ của email, mật khẩu đăng nhập. Dựa vào phân quyền của tài khoản mà điều hướng trang web (Nếu phân quyền admin thì sau khi đăng nhập sẽ điều hướng về trang chủ quản lý của admin. Các phân quyền còn lại trở về trang chủ).
* Luồng rẽ nhánh:
* Nếu trong dòng sự kiện chính, người khách hàng cung cấp email, mật khẩu không hợp . Người khách hàng có thể chọn tiếp tục đăng nhập lại theo dòng sự kiện chính hoặc hủy bỏ việc đăng nhập để kết thúc chức năng đăng nhập.
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Người khách hàng cần phải có tài khoản .
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Đặt hàng , thanh toán , đánh giá , giỏ hàng.
* **Đăng ký :**
* Usecase này cho phép người khách hàng đăng đăng kí một tài khoản vào hệ thống dựa theo phân quyền của tài khoản.
* Luồng sụ kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người khách hàng muốn đăng kí một tài khoản để có thể vào website.
* Hệ thống yêu cầu người khách hàng nhập email mới, mật khẩu mới, xác nhận mật khẩu .
* Hệ thống kiểm tra tính hợp lệ của email, mật khẩu đã được nhập. Nếu hợp lệ sẽ hiển thị thông báo đăng kí thành công , và tài khoản được thêm vào cơ sở dữ liệu. Dựa vào phân quyền của tài khoản mà điều hướng trang web (Nếu phân quyền admin thì sau khi đăng nhập sẽ điều hướng về trang chủ quản lý của admin. Các phân quyền còn lại trở về trang chủ).
* Luồng rẽ nhánh:
* Nếu trong dòng sự kiện chính, người khách hàng cung cấp email, mật khẩu không hợp .Hệ thống sẽ cảnh báo khách hàng nhập lại.
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Không có
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Xem chi tiết sản phẩm**
* Usecase này cho phép người dùng xem được các thông tin chi tiết của sản phẩm.
* Luồng sự kiện:
* Luồng cơ bản:
* Usecase này bắt đầu khi người dùng click vào hình ảnh sản phẩm hoặc tên của sản phẩm. Hệ thống sẽ lấy thông tin product\_id, size\_id, product\_name, product\_code, product\_metatitle, product\_description, product\_more\_image, product\_image, product\_promotion, product\_includedvat, product\_price, product\_quantity, product\_category\_id, product\_detail, product\_status, product\_viewcount, product\_rate, product\_material, product\_size, sex từ bảng “products” trong CSDL rồi hiển thị ra màn hình
* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Không có.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Giỏ Hàng**
* Usecase này cho phép người dùng xem giỏ hàng, thêm sản phẩm vào giỏ hàng, sửa số lượng sản phẩm, xóa sản phẩm khỏi giỏ hàng.
* Luồng sự kiện:
* Luồng cơ bản:
* Mở giỏ hàng:

Usecase này bắt đầu khi người dùng đưa con trỏ chuột đến icon giỏ hàng, trang web hover vắn tắt thông tin giỏ hàng cùng chức năng xóa thông tin sản phẩm.

Người dùng click vào nút “Cart”, hệ thống lấy thông tin ảnh, tên sản phẩm, giá, số lượng, tổng tiền được lưu trên session rồi hiển thị ra màn hình.

* Thêm giỏ hàng:

Use case này bắt đầu khi người dùng click vào icon giỏ hàng cảu sản phẩm mình muốn thêm. Hệ thống sẽ lưu thông tin tên sản phẩm, giá và cập nhật lại số lượng của sản phẩm đó trong giỏ hàng lên 1 sản phẩm nữa.

* Cập nhật giỏ hàng:

Khi người dùng đang ở trang xem giỏ hàng, họ có thể chỉnh sửa thông tin số lượng của sản phẩm mình muốn mua. Người dùng điều chỉnh số lượng sau đó nhấn icon save để lưu thông tin cho từng sản phẩm hoặc nhấn nút “Update all” để cập nhật lại số lượng cho toàn bộ sản phẩm có mặt trong giỏ hàng.

* Xóa giỏ hàng:

Người dùng hover vào icon giỏ hàng trên trang web, sau đó click nút “View card” để vào trang giỏ hàng. Người dùng click nút “X” trong form hiển thị thông tin các sản phẩm trong giỏ hàng.

* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Đăng nhập.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Đặt hàng**
* Usecase này cho phép người dùng đặt một đơn hàng.
* Luồng sự kiện:
* Luồng cơ bản:
* Mở giỏ hàng:

Usecase này bắt đầu khi người dùng đưa con trỏ chuột đến icon giỏ hàng, trang web hover vắn tắt thông tin giỏ hàng cùng chức năng xóa thông tin sản phẩm.

Người dùng click vào nút “Cart”, hệ thống lấy thông tin ảnh, tên sản phẩm, giá, số lượng, tổng tiền được lưu trên session rồi hiển thị ra màn hình.

* Đặt hàng:

Use case này bắt đầu khi người dùng click vào nút đặt hàng của giỏ hàng. Hệ thống sẽ lưu thông tin đơn hàng bao gồm: orders\_id, product\_size\_id, user\_id, shipplace\_id, orders\_status, orders\_quantity, product\_price, created\_at, orders\_type, product\_cost lên cơ sở dữ liệu.

* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Đăng nhập.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.
* **Đánh giá**
* Usecase này cho phép người dùng đánh giá một sản phẩm nào đó.
* Luồng sự kiện:
* Luồng cơ bản:
* Xem thông tin sản phẩm:

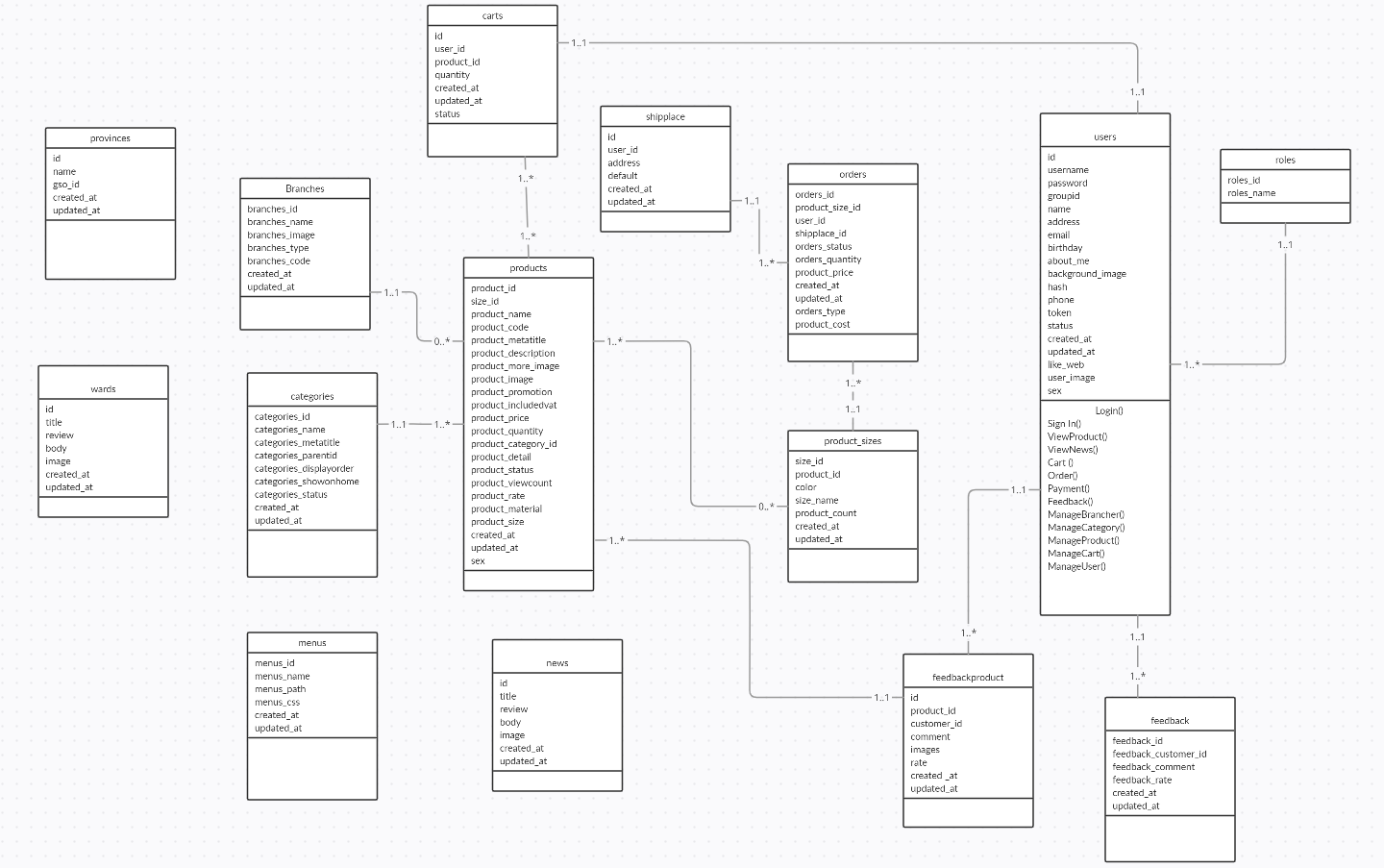
Usecase này bắt đầu khi người dùng click vào hình ảnh sản phẩm hoặc tên của sản phẩm. Hệ thống sẽ lấy thông tin product\_id, size\_id, product\_name, product\_code, product\_metatitle, product\_description, product\_more\_image, product\_image, product\_promotion, product\_includedvat, product\_price, product\_quantity, product\_category\_id, product\_detail, product\_status, product\_viewcount, product\_rate, product\_material, product\_size, sex từ bảng “products” trong CSDL rồi hiển thị ra màn hình

* Đánh giá:

Người khách hàng để lại bình luận , đánh giá về sản phẩm sau khi đã nhận sản phẩm và thanh toán , thông tin khách hàng và nội dung bình luận sẽ được lưu trên cơ sở dữ liệu .

* Luồng rẽ nhánh:
* Tại bất cứ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
* Các yêu cầu đặc biệt:
* Không có.
* Tiền điều kiện:
* Đăng nhập, đơn hàng đã thanh toán.
* Hậu điều kiện:
* Không có.
* Điểm mở rộng:
* Không có.

## Biểu Đồ Lớp:

****

Hình 2.7 : Sơ đồ lớp , cơ sở dữ liệu

## 2.3 Biểu đồ trình tự:

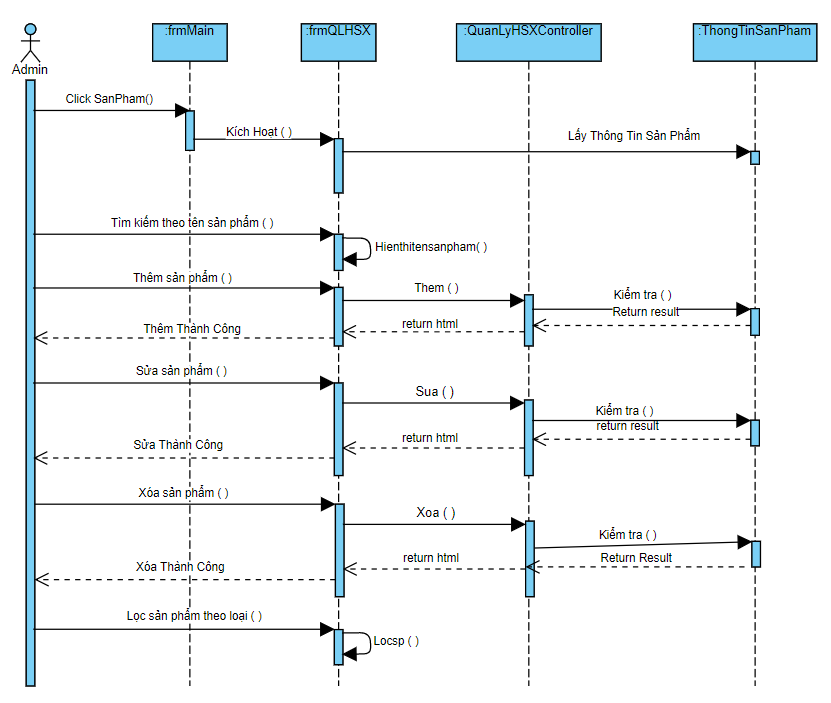
### 2.3.1 Biểu đồ trình tự Đăng Nhập:

Hình 2.8 : Biểu đồ trình tự Đăng Nhập

### 2.3.2 Biểu đồ trình tự Tài Khoản:

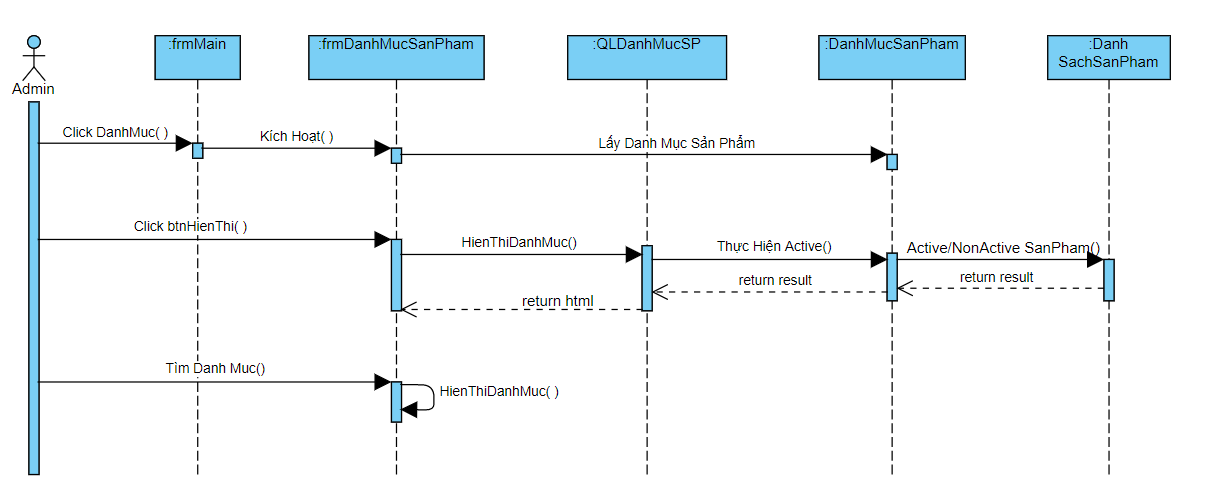
Hình 2.9 : Biểu đồ trình tự Đăng kí

### 2.3.3 Biểu đồ trình tự Sản Phẩm:



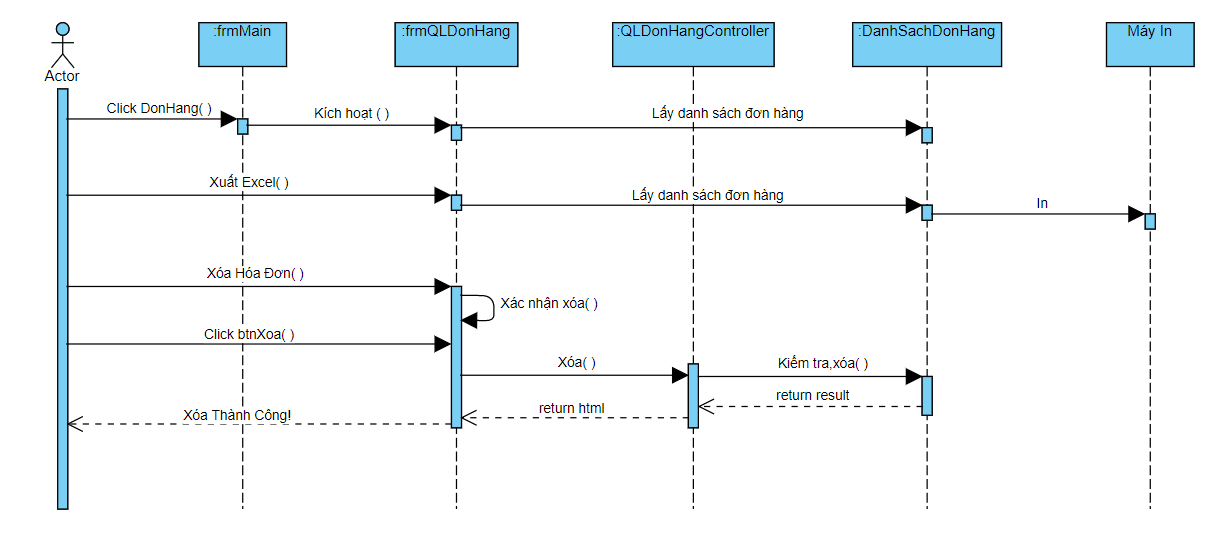
Hình 2.10 : Biểu đồ trình tự quản lý sản phẩm

### 2.3.4 Biểu đồ trình tự Danh Mục Sản Phẩm:



Hình 2.11 : Biểu đồ trình tự Danh mục sản phẩm

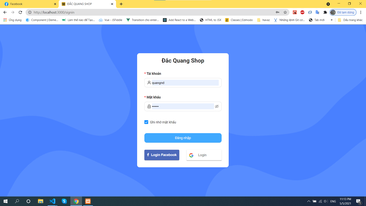
### 2.3.5 Biểu đồ trình tự Đơn Hàng:



Hình 2.11 : Biểu đồ trình tự quản lý đơn hàng

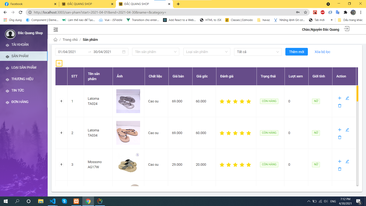
# Chương 3: Thiết kế giao diện

**3.1 Giao diện đăng nhập :**

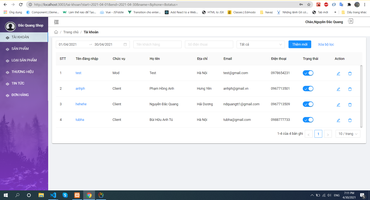
****

Hình 3.1 : Giao diện đăng nhập

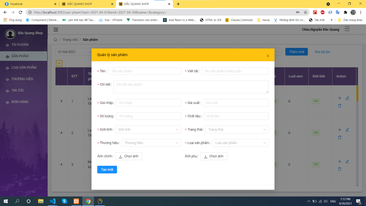
**3.2 Giao diện admin :**

****

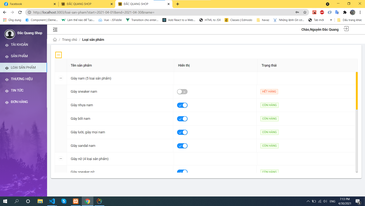
Hình 3.2 : Giao diện admin

****

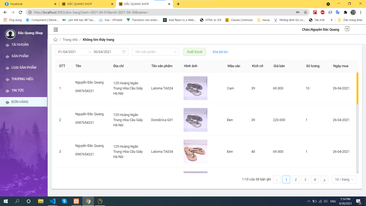
Hình 3.3 : Giao diện admin

****

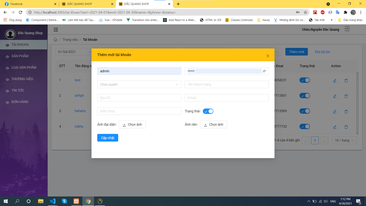
Hình 3.4 : Giao diện admin

****

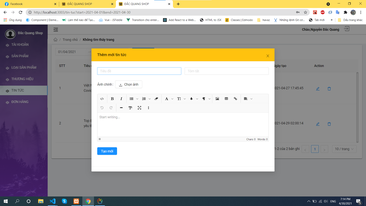
Hình 3.5 : Giao diện admin

****

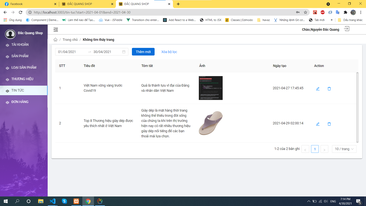
Hình 3.6 : Giao diện admin

****

Hình 3.7 : Giao diện admin

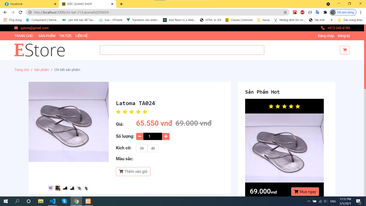
****

Hình 3.7 : Giao diện admin

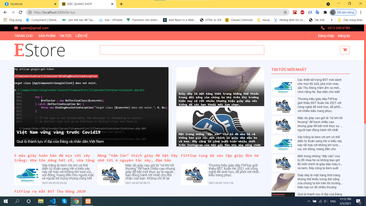
****

Hình 3.8 : Giao diện admin

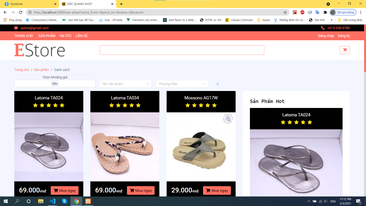
**3.3 Giao diện trang chủ :**



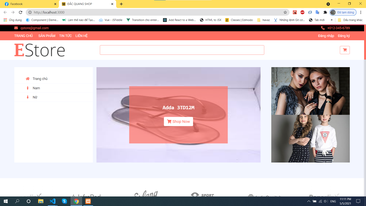
Hình 3.9 : Giao diện trang chủ



Hình 3.10 : Giao diện trang chủ

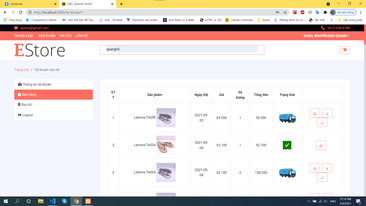


Hình 3.11 : Giao diện trang chủ



Hình 3.12 : Giao diện trang chủ

**3.4 : Giao diện giỏ hàng :**



Hình 3.13 : Giao diện giỏ hàng

# Đánh giá và Kết Luận

* **Đánh giá**

Em đã áp dụng kiến thức về Framework Laravel vào trong project nhằm cải thiện kỹ năng lập trình của bản thân và áp dụng được vào thực tế. Trong khi thực hiện đề tài em có gặp phải 1 số khó khăn như: Kỹ năng phân tích thiết kế hệ thống còn kém, cơ sở dữ liệu và khi code chưa tối ưu. Việc lấy dữ liệu tự động em sẽ cố gắng hoàn thiện trong thời gian sắp tới.

* **Hướng phát triển đề tài**

Trong thời gian tới em sẽ tiếp tục nghiên cứu sâu hơn về các vấn đề của website như sẽ lấy dữ liệu tự động, phát triển thêm 1 số tính năng mới vào website để có thể đáp ứng được nhu cầu người sử dụng như : hiệu ứng animation đa dạng , tìm kiếm bằng giọng nói…. Tối ưu hóa hệ thống cũng như hệ cơ sở dữ liệu để hệ thống có thể chạy nhanh hơn và thân thiện với người sử dụng.

* **Tài liệu tham khảo:**
* Trang chủ Laravel: <https://laravel.com/>
* Tài liệu: <https://toidicode.com/hoc-laravel>
* Trích xuất csdl từ: <https://vi.wikipedia.org/wiki/Wikipedia>