**Shape, square

Description automatically generated** **ĐẠI HỌC CÔNG NGHỆ**

**THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN**

**KHOA KHOA HỌC MÁY TÍNH**

-----🙞🙜🕮🙞🙜-----

**Shape

Description automatically generated with medium confidence**

**ĐỒ ÁN THỰC TẬP TỐT NGHIỆP**

**ĐỀ TÀI: XÂY DỰNG WEBSITE ORGANIC ECOMMERCE VỚI MERN STACK**

Sinh viên thực hiện : **TRẦN QUỐC TÍNH - 19IT5**

Giảng viên hướng dẫn: **THS. NGÔ LÊ QUÂN**

***Đà nẵng, tháng 09 năm 2023***

**ĐẠI HỌC CÔNG NGHỆ**

**THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN**

**KHOA KHOA HỌC MÁY TÍNH**

-----🙞🙜🕮🙞🙜-----

**Shape

Description automatically generated with medium confidence**

**ĐỒ ÁN THỰC TẬP TỐT NGHIỆP**

**ĐỀ TÀI: XÂY DỰNG WEBSITE ORGANIC ECOMMERCE VỚI MERN STACK**

Sinh viên thực hiện : **TRẦN QUỐC TÍNH - 19IT5**

Giảng viên hướng dẫn: **THS. NGÔ LÊ QUÂN**

***Đà nẵng, tháng 08 năm 2023***

A close-up of a questionnaire

Description automatically generated

**LỜI MỞ ĐẦU**

Khởi đầu một dự án xây dựng một trang thương mại điện tử bằng MERN stack là một thử thách thú vị và đầy tiềm năng. Trong quá trình này, em đã tìm hiểu và tham khảo nhiều nguồn kiến thức, từ tài liệu hướng dẫn trực tuyến đến các dự án mã nguồn mở tương tự đã được triển khai thành công.

Việc tham khảo các nguồn tài liệu và dự án đã giúp em hiểu rõ hơn về kiến trúc và cấu trúc của MERN stack. Em đã nắm vững nguyên tắc hoạt động của MongoDB, Express.js, React và Node.js, và biết cách kết hợp chúng để tạo ra một ứng dụng web mạnh mẽ.

Em cũng đã tìm hiểu và tham khảo các best practice và quy ước trong việc phát triển ứng dụng web. Từ việc quản lý trạng thái ứng dụng bằng Redux, đến việc sử dụng các thư viện UI như Ant-design hay Bootstrap để tạo giao diện hấp dẫn và đáp ứng, em đã rút ra nhiều kinh nghiệm quý giá từ những nguồn tham khảo này.

Ngoài ra, em đã tham khảo các dự án Ecommerce thực tế đã được triển khai bằng MERN stack. Điều này giúp em hiểu rõ hơn về các chức năng cần có trong một trang web Ecommerce, từ việc quản lý sản phẩm, giỏ hàng, thanh toán cho đến quản lý đơn hàng và tạo tài khoản người dùng. Em đã học hỏi từ những dự án này và áp dụng những phương pháp tốt nhất vào dự án của mình.

Tham khảo nhiều nguồn kiến thức và dự án đã giúp em xây dựng nền tảng vững chắc cho dự án của mình. Em đã áp dụng những phương pháp đã được kiểm chứng và rút ra những bài học quan trọng từ những dự án thành công trước đó. Điều này đã đảm bảo cho sự tiến triển ổn định và hiệu quả của dự án Ecommerce mà em đang thực hiện. Trong quá trình thực hiện đồ án, với khoảng thời gian hạn hẹp và lượng kiến thức có hạn nên đồ án chắc chắn sẽ còn nhiều thiếu sót. Mong nhận được ý kiến nhận xét và góp ý của Thầy Ngô Lê Quân cũng như các thầy cô trong nhà trường để em có thể hoàn thiện đề tài lần này tốt hơn.

**LỜI CẢM ƠN**

Lời cảm ơn đầu tiên, em xin gửi đến ban lãnh đạo Công ty TMA Solutions Bình Định, cùng với tất cả các anh chị trong công ty và đặc biệt là DC20 team, vì đã đồng hành và hỗ trợ em suốt quá trình thực tập tại công ty. Nhờ sự hướng dẫn và chỉ bảo của mọi người, em đã có cơ hội được tiếp cận với các dự án thực tế và phát triển kỹ năng chuyên môn.

Em cũng muốn gửi lời tri ân chân thành đến quý thầy, cô giáo trong Khoa Khoa Học Máy Tính - Trường Đại Học Công Nghệ Thông Tin và Truyền Thông Việt Hàn – Đại học Đà Nẵng, vì đã tận tâm giảng dạy và truyền đạt những kiến thức, kinh nghiệm quý báu cho em. Nhờ những kiến thức này, em đã có nền tảng vững chắc để ứng dụng vào công việc thực tế.

Đặc biệt, em xin cám ơn thầy Ngô Lê Quân, người đã tận tình hướng dẫn em hoàn thành bài báo cáo này. Sự chỉ bảo và những lời khuyên từ Thầy đã giúp em nắm bắt được các khía cạnh quan trọng trong việc thực hiện báo cáo và nâng cao kỹ năng viết.

Cuối cùng, em xin gửi lời cảm ơn chân thành đến anh Phạm Quốc Khánh – là Mentor hướng dẫn em trực tiếp tại công ty TMA Solutions Bình Định. Anh đã dành thời gian và công sức để hướng dẫn, giúp đỡ em trong việc tiếp cận và hiểu rõ hơn về mảng FrontEnd. Nhờ sự hỗ trợ từ anh, em đã có cơ hội áp dụng kiến thức vào thực tế và hoàn thành tốt hơn trong quá trình thực tập.

Em nhận thức rằng bài báo cáo của em vẫn còn hạn chế và không tránh khỏi những thiếu sót. Vì vậy, em rất mong nhận được ý kiến đóng góp từ Công ty, quý thầy cô và các bạn, để em có thể rút kinh nghiệm và hoàn thiện kỹ năng của mình. Em xin chân thành cảm ơn sự giúp đỡ và sự đánh giá từ mọi người!

**LỜI CAM ĐOAN**

Em xin cam đoan đề tài “Xây Dựng Website Thương Mại Điện Tử Với Mern Stack” là kết quả từ quá trình thực tập tại doanh nghiệp của em dưới sự hướng dẫn của Mentor: Phạm Quốc Khánh và giảng viên hướng dẫn: ThS. Ngô Lê Quân.

Đề tài, nội dung báo cáo là sản phẩm mà em đã được thực hành trong quá trình thực tập tại doanh nghiệp. Các dữ liệu, kết quả trình bày trong báo cáo là hoàn toàn trung thực, em xin chịu hoàn toàn trách nhiệm với bộ môn và nhà trường nếu có bất kỳ vấn đề nào xảy ra. Em xin chân thành cảm ơn!

**NHẬN XÉT**

**CỦA GIÁO VIÊN HƯỚNG DẪN**

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

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

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

Đà Nẵng, ngày 30 tháng 08 năm 2023

**Giảng viên hướng dẫn**

**THS.NGÔ LÊ QUÂN**

**MỤC LỤC**

[MỤC LỤC 8](#_Toc145166529)

[DANH SÁCH HÌNH VẼ 11](#_Toc145166530)

[DANH SÁCH BẢNG 11](#_Toc145166531)

[DANH SÁCH TỪ VIẾT TẮT 12](#_Toc145166532)

[LỜI MỞ ĐẦU 13](#_Toc145166533)

[CHƯƠNG 1. TỔNG QUAN VỀ CÔNG TY TMA SOLUTIONS BÌNH ĐỊNH VÀ CƠ SỞ LÝ THUYẾT VỀ FRONT – END DEVELOPER 16](#_Toc145166534)

[1.1 Giới thiệu tổng quan về công ty TMA Solutions Bình Định 16](#_Toc145166535)

[1.1.1 Tổng quan về công ty 16](#_Toc145166536)

[1.1.2 Tầm nhìn và sứ mệnh 16](#_Toc145166537)

[1.2 Tổng quan về vị trí Front – End Developer 16](#_Toc145166538)

[1.2.1 Giới thiệu về Front - End Developer 16](#_Toc145166539)

[1.2.2 Định hướng về Front – End Developer 17](#_Toc145166540)

[Chương 2 TỔNG QUAN VỀ MERN STACK 18](#_Toc145166541)

[2.1 Phương pháp nghiên cứu 18](#_Toc145166542)

[2.1.1 MongoDB 18](#_Toc145166543)

[2.1.2 Express 19](#_Toc145166544)

[2.1.3 ReactJS 19](#_Toc145166545)

[2.1.4 NodeJS 20](#_Toc145166546)

[2.1.5 Redux – Redux Toolkit 21](#_Toc145166547)

[2.1.6 Mô hình MVC 21](#_Toc145166548)

[2.2 Công cụ thực hiện 22](#_Toc145166549)

[2.1.1 Visual Studio Code 22](#_Toc145166550)

[2.1.2 Postman 22](#_Toc145166551)

[2.1.3 MongoDB Compass 23](#_Toc145166552)

[Chương 3 PHÂN TÍCH THIẾT KẾ HỆ THỐNG 24](#_Toc145166553)

[3.1 Danh sách các tác nhân 24](#_Toc145166554)

[3.1.1 Yêu cầu hệ thống 24](#_Toc145166555)

[3.1.2 Phân tích hệ thống 24](#_Toc145166556)

[Chương 4 XÂY DỰNG BACKEND 27](#_Toc145166557)

[4.1 Cấu hình môi trường phát triển 27](#_Toc145166558)

[4.2 Xây dựng các router và controllers 29](#_Toc145166559)

[Chương 5 XÂY DỰNG GIAO DIỆN NGƯỜI DÙNG VỚI REACT 38](#_Toc145166560)

[5.1 Tạo các component và cấu trúc UI 38](#_Toc145166561)

[Chương 6 TÍCH HỢP VÀ KIỂM THỬ 46](#_Toc145166562)

[6.1 Tích hợp frontend và backend 46](#_Toc145166563)

[6.1.1 Chức năng đăng ký 46](#_Toc145166564)

[6.1.2 Chức năng đăng nhập, đăng xuất 48](#_Toc145166565)

[6.1.3 Chức năng thêm sản phẩm vào giỏ hàng 50](#_Toc145166566)

[6.1.4 Chức năng thanh toán 52](#_Toc145166567)

[6.1.5 Chức năng tìm kiếm sản phẩm 54](#_Toc145166568)

[6.1.6 Chức năng đánh giá sản phẩm 55](#_Toc145166569)

[6.1.7 Chức năng đăng nhập 58](#_Toc145166570)

[6.1.8 Chức năng đăng xuất 61](#_Toc145166571)

[6.1.9 Chức năng thêm sản phẩm 61](#_Toc145166572)

[6.1.10 Chức năng thêm, sửa xoá brand 64](#_Toc145166573)

[6.2 Sửa lỗi và cải thiện hiệu suất 68](#_Toc145166574)

[Chương 7 KẾT LUẬN 69](#_Toc145166575)

[7.1 Tổng kết 69](#_Toc145166576)

[7.1.1 Những điều đạt được 69](#_Toc145166577)

[7.1.2 Kết luận 69](#_Toc145166578)

[TÀI LIỆU THAM KHẢO 70](#_Toc145166579)

**DANH SÁCH HÌNH VẼ**

[Hình 1 Công ty TMA solutions Bình Định 16](#_Toc145166580)

[Hình 2 MongoDB logo 19](#_Toc145166581)

[Hình 3 ExpressJS logo 19](#_Toc145166582)

[Hình 4 ReactJS logo 20](#_Toc145166583)

[Hình 5 NodeJS logo 20](#_Toc145166584)

[Hình 6 Redux Toolkit logo 21](#_Toc145166585)

[Hình 7 Mô hình MVC 21](#_Toc145166586)

[Hình 8 Visual Studio Code logo 22](#_Toc145166587)

[Hình 9 Postman logo 23](#_Toc145166588)

[Hình 10 Usecase admin 24](file:///C:\Users\Admin\Desktop\Intern_Report\TTTN_Ecommerce-Tran-Quoc-Tinh_19IT358.docx#_Toc145166589)

[Hình 11 Usecase customer 25](#_Toc145166590)

[Hình 12 Biểu đồ lớp 26](#_Toc145166591)

**DANH SÁCH BẢNG**

[Bảng 1 Các package hỗ trợ cho việc xây dựng FrontEnd 26](#_Toc144393909)

[Bảng 2 Các package hỗ trợ cho việc xây dựng BackEnd 27](#_Toc144393910)

**DANH SÁCH TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| DOM | Document Object Model |
| MVC | Model-View-Controller |
| IDE | Integrated Development Environment |
| API | Application Programming Interface |
| SQL | Structured Query Language |
| HTTP | Hyper Text Transfer Protocol |
| URL | Uniform Resource Locator |

# LỜI MỞ ĐẦU

###### 1. Lý do chọn đề tài

Dự án Organic Ecommerce được chọn với mục tiêu xây dựng một trang web thương mại điện tử đặc biệt tập trung vào việc bán các sản phẩm hữu cơ. Lý do chọn đề tài này đến từ nhận thức ngày càng tăng về lợi ích của việc tiêu dùng các sản phẩm hữu cơ cho sức khỏe và môi trường.

Trang web ecommerce là một sự lựa chọn lý tưởng để xây dựng nền tảng kinh doanh trực tuyến cho các sản phẩm hữu cơ. Với mô hình này, khách hàng có thể dễ dàng duyệt qua danh mục sản phẩm, tìm kiếm, đặt hàng và thanh toán trực tuyến một cách thuận tiện và an toàn.

Bên cạnh đó, việc xây dựng một trang web Ecommerce từ đầu sẽ đòi hỏi sự nắm vững kiến thức về thiết kế cơ sở dữ liệu, xây dựng API, quản lý trạng thái ứng dụng và giao diện người dùng. Thông qua dự án này, em có cơ hội rèn luyện và phát triển những kỹ năng quan trọng này, từ việc lựa chọn công nghệ phù hợp, xử lý vấn đề, đến việc tối ưu hóa hiệu suất và cải thiện trải nghiệm người dùng.

###### 2. Mục tiêu của đề tài

Dự án Organic Ecommerce mục tiêu cung cấp một trải nghiệm mua sắm trực tuyến tốt nhất cho khách hàng, cung cấp thông tin chi tiết về các sản phẩm hữu cơ, đảm bảo tính chính xác và đáng tin cậy. Giao diện người dùng sẽ được thiết kế thân thiện và dễ sử dụng, giúp khách hàng dễ dàng tìm kiếm, so sánh và mua các sản phẩm hữu cơ mà họ quan tâm.

Dự án sẽ tập trung vào việc phát triển các tính năng quản lý sản phẩm, giỏ hàng, thanh toán, đánh giá sản phẩm và xử lý đơn hàng. Đồng thời, các biện pháp bảo mật và quản lý dữ liệu sẽ được áp dụng để đảm bảo an toàn và bảo mật thông tin khách hàng.

###### 3. Đối tượng và phạm vi nghiên cứu

Để thực hiện dự án này, em sẽ tiến hành các công việc sau:

Thiết kế và xây dựng dữ liệu (BackEnd): Đầu tiên, em sẽ tập trung vào thiết kế và xây dựng phần backend của trang web. Sử dụng Express.js và Node.js, em sẽ xây dựng các API để quản lý dữ liệu như sản phẩm, đơn hàng và người dùng.

Xây dựng giao diện người dùng (FrontEnd): Tiếp theo, em sẽ sử dụng ReactJs kết hợp với Redux để xây dựng giao diện người dùng cho trang web. Em sẽ tạo các trang và thành phần cần thiết để hiển thị thông tin sản phẩm, giỏ hàng và quá trình thanh toán. Giao diện người dùng sẽ được thiết kế sao cho trực quan và dễ sử dụng.

Tạo trang quản lý cho admin (Dashboard): Sau khi hoàn thành phần giao diện người dùng, em sẽ tạo một trang quản lý riêng dành cho quản trị viên. Trang này sẽ cung cấp các chức năng quản lý sản phẩm như thêm, sửa, xóa sản phẩm, quản lý đơn hàng và xem báo cáo thống kê. Điều này giúp quản trị viên dễ dàng quản lý và điều hành các hoạt động của trang web Ecommerce.

Kiểm thử và tối ưu: Tối ưu hóa hiệu suất của trang web bằng cách sử dụng các kỹ thuật như caching, tối ưu tải trang và tăng tốc độ phản hồi.

* Chương 1: Tổng quan về công ty TMA Solutions Bình Định
* Chương 2: Tổng quan về MERN stack
* Chương 3: Phân tích thiết kế hệ thống
* Chương 4: Xây dựng BackEnd
* Chương 5: Xây dựng FrontEnd (Giao diện người dùng và Dashboard)
* Chương 6: Tích hợp và kiểm thử
* Chương 7: Kết luận

###### 4. Giới hạn của đề tài

Với thời gian có hạn nên em chỉ thực hiện nghiên cứu các vấn đề cơ bản trong phạm vi nhất định.

Sẽ có 2 loại người dùng đó là Customer và Admin:

* Customer: Người dùng sẽ có thể xem các sản phẩm, thêm sản phẩm vào danh sách yêu thích, giỏ hàng, mua sản phẩm, đánh giá và bình luận về sản phẩm, đọc blog
* Quản trị viên: Quản lý thông tin người dùng, sản phẩm, blog, category, …

1. **TỔNG QUAN VỀ CÔNG TY TMA SOLUTIONS BÌNH ĐỊNH VÀ CƠ SỞ LÝ THUYẾT VỀ FRONT – END DEVELOPER**

## Giới thiệu tổng quan về công ty TMA Solutions Bình Định

### 1.1.1 Tổng quan về công ty

TMA Bình Định là dự án đầu tư quy mô lớn của TMA Solutions. Với bề dày 25 năm phát triển vững mạnh của TMA Solutions thì TMA Bình Định đang dần khẳng định vị thế của mình đối với nền CNTT tỉnh Bình Định, góp sức cùng với tỉnh xây dựng Thung Lũng Sáng tạo Quy Nhơn trở thành trung tâm khoa học, công nghệ cao tại miền Trung.



Hình 1 Công ty TMA solutions Bình Định

### 1.1.2 Tầm nhìn và sứ mệnh

Tầm nhìn: Trở thành một trong những công ty hàng đầu về cung cấp giải pháp phần mềm tại Việt Nam và trong khu vực.

Sứ mệnh: Tôn trọng và mang đến cho khách hàng những sản phẩm, giải pháp phần mềm tốt nhất với chi phí hợp lí nhất. Đồng thời xây dựng mối quan hệ tin cậy, uy tín, hợp tác cùng phát triển với các đối tác trong lĩnh vực công nghệ thông tin.

## Tổng quan về vị trí Front – End Developer

### 1.2.1 Giới thiệu về Front - End Developer

Front-end Developer là người chịu trách nhiệm xây dựng giao diện người dùng (UI) và trải nghiệm người dùng (UX) trên các ứng dụng web. Họ là các nhà phát triển chuyên về phần giao diện của một trang web và làm việc với các ngôn ngữ lập trình như HTML, CSS, JavaScript và các framework, library như ReactJS, VueJS, Angular, ... để tạo ra các trang web tương tác và hấp dẫn.

### 1.2.2 Định hướng về Front – End Developer

* Fresher/Intern Front-end Developer:
* Được đào tạo cơ bản về HTML, CSS và JavaScript.
* Học cách sử dụng các công cụ và framework phổ biến như React, Angular hoặc Vue.js.
* Tham gia vào các dự án nhỏ để rèn kỹ năng và hiểu cách làm việc trong môi trường phát triển Front-end.
* Junior Front-end Developer:
* Xây dựng và duy trì các giao diện người dùng cơ bản trên các dự án.
* Nắm vững kiến thức về responsive design và tối ưu hóa trải nghiệm người dùng trên các thiết bị khác nhau.
* Học cách sử dụng công cụ quản lý trạng thái như Redux hoặc MobX để quản lý dữ liệu trên ứng dụng.
* Intermediate Front-end Developer:
* Tham gia xây dựng các tính năng phức tạp và tương tác trên giao diện người dùng.
* Nắm vững các nguyên tắc thiết kế giao diện người dùng và trải nghiệm người dùng.
* Học cách tối ưu hóa hiệu suất và tăng cường bảo mật trên trang web.
* Senior Front-end Developer:
* Trở thành chuyên gia trong một số lĩnh vực cụ thể như animation, performance optimization hoặc accessibility.
* Dẫn dắt và hướng dẫn các thành viên mới trong đội, tham gia vào việc đưa ra quyết định thiết kế quan trọng.
* Đóng góp vào việc xây dựng kiến trúc Front-end và thiết lập tiêu chuẩn phát triển trong dự án.

# TỔNG QUAN VỀ MERN STACK

2.1 Phương pháp nghiên cứu

### 2.1.1 MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) phát triển bởi MongoDB Inc. Nó cung cấp một cách tiếp cận khác biệt so với SQL (Structured Query Language), ngôn ngữ truy vấn có cấu trúc sử dụng trong các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS).

Trong thực tế, MongoDB không sử dụng SQL để tương tác với dữ liệu. Thay vào đó, MongoDB sử dụng một ngôn ngữ truy vấn linh hoạt gọi là MongoDB Query Language (MQL). MQL cho phép người dùng truy vấn dữ liệu dựa trên các điều kiện và phương thức tìm kiếm phù hợp với mô hình dữ liệu tài liệu của MongoDB.

Một điểm khác biệt quan trọng giữa MongoDB và cơ sở dữ liệu quan hệ là cách dữ liệu được tổ chức. Trong MongoDB, dữ liệu được lưu trữ dưới dạng các tài liệu BSON (Binary JSON), mỗi tài liệu là một bản ghi độc lập và có thể chứa các trường và giá trị khác nhau. Thay vì sử dụng bảng và mối quan hệ như SQL, MongoDB cho phép lưu trữ dữ liệu phong phú và phức tạp trong một tài liệu duy nhất.

Với MongoDB, bạn có thể truy vấn dữ liệu bằng cách sử dụng các phương thức như find(), aggregate() và update(). Các truy vấn có thể được lọc, sắp xếp, nhóm hóa và thực hiện các phép toán phức tạp khác để truy xuất và xử lý dữ liệu.



Hình 2 MongoDB logo

### 2.1.2 Express

Express là một framework giành cho nodejs. Nó cung cấp cho chúng ta rất nhiều tính năng mạnh mẽ trên nền tảng web cũng như trên các ứng dụng di động. Express hỗ rợ các phương thức HTTP và midleware tạo ra môt API vô cùng mạnh mẽ và dễ sử dụng. Có thể tổng hợp một số chức năng chính của express như sau:

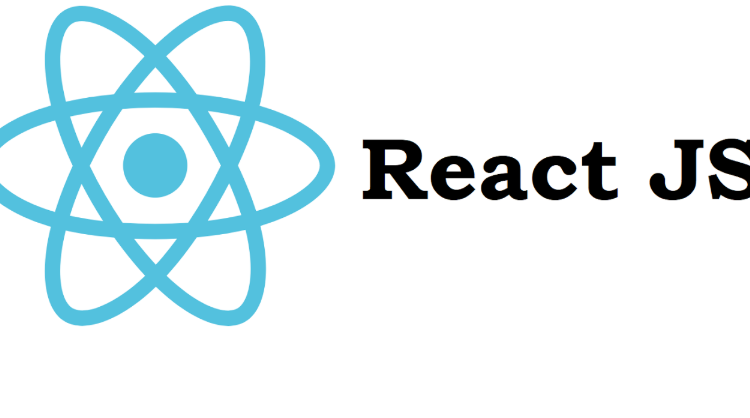
* Thiết lập các lớp trung gian để trả về các HTTP request
* Định nghĩa router cho phép sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL
* Cho phép trả về các trang HTML dựa vào các tham số.



Hình 3 ExpressJS logo

### 2.1.3 ReactJS

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

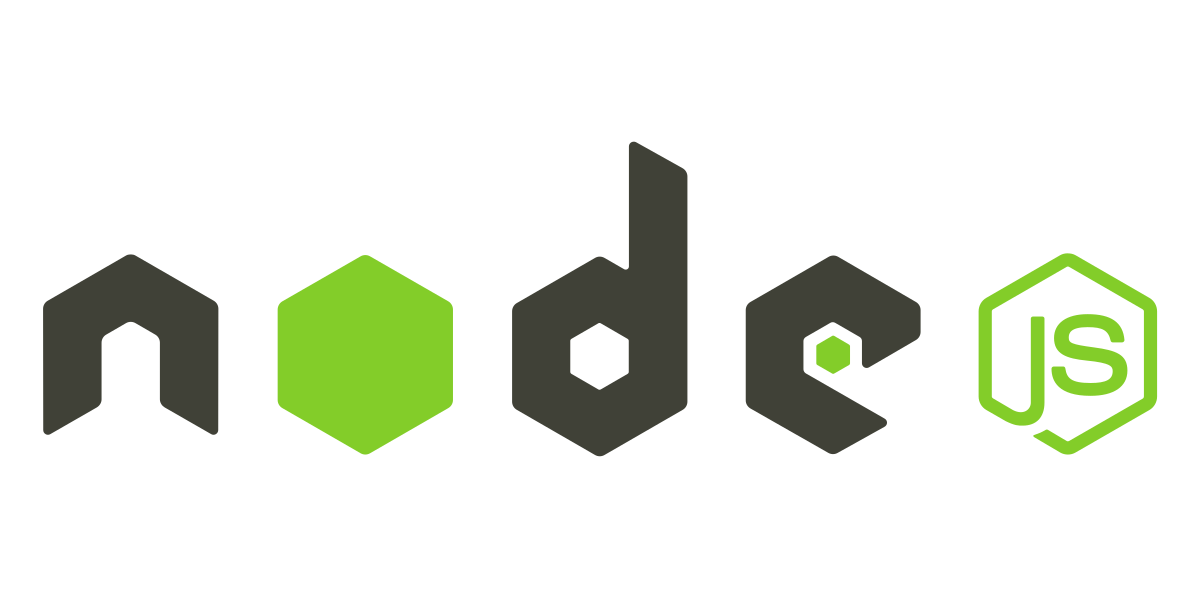


Hình 4 ReactJS logo

### 2.1.4 NodeJS

Nodejs là một nền tảng (Platform) phát triển độc lập được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript giúp chúng ta có thể xây dựng được các ứng dụng web như các trang video clip, các forum và đặc biệt là trang mạng xã hội phạm vi hẹp một cách nhanh chóng và dễ dàng mở rộng.

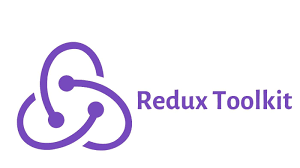
NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ Window cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất.



Hình 5 NodeJS logo

### 2.1.5 Redux – Redux Toolkit

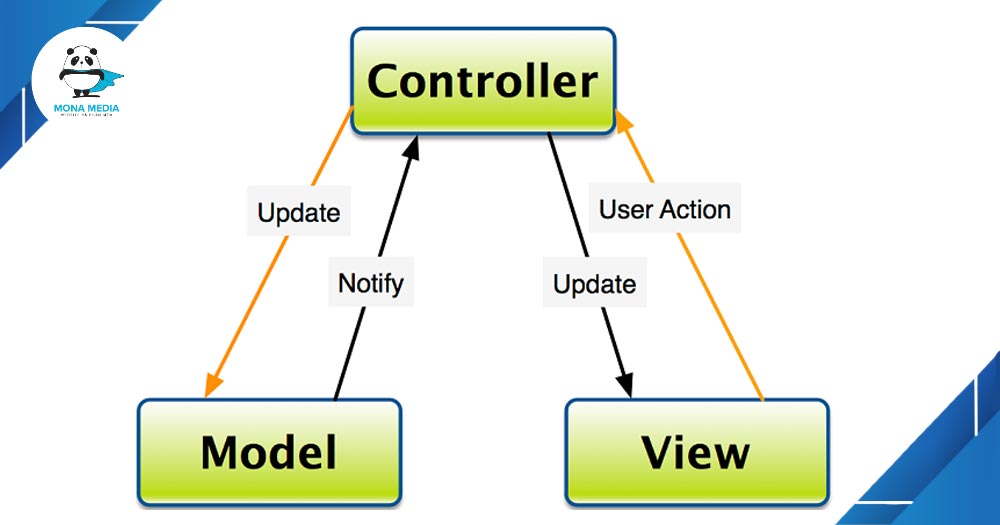
Redux là một thư viện quản lý trạng thái cho ứng dụng JavaScript. Nó được sử dụng chủ yếu trong các ứng dụng React để quản lý trạng thái của ứng dụng một cách dễ dàng và dự đoán được. Redux giúp quản lý trạng thái toàn cục của ứng dụng một cách hiệu quả, giúp tránh tình trạng truyền trạng thái qua nhiều thành phần con và làm cho việc theo dõi và điều khiển trạng thái trở nên dễ dàng hơn.



Hình 6 Redux Toolkit logo

### 2.1.6 Mô hình MVC

MVC là viết tắt của Model-View-Controller. Trong đó, Model xử lý dữ liệu. Controller xử lý logic. Còn View là phần hiển thị và tiếp nhận request từ phía User (người dùng).



Hình 7 Mô hình MVC

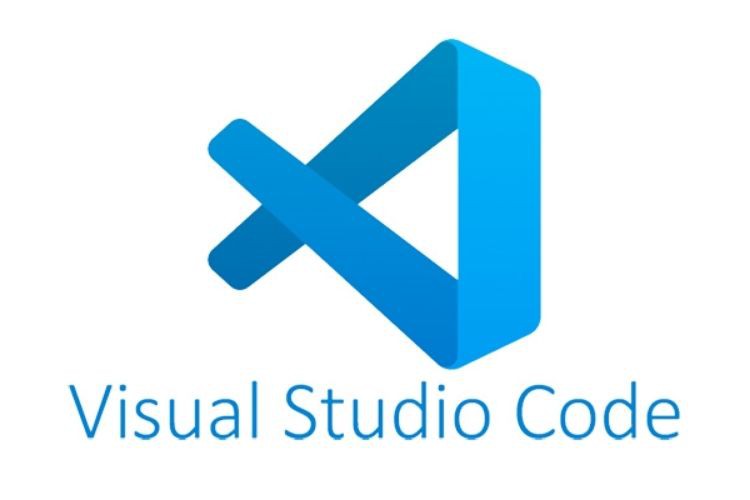
Trong MVC, Controller đóng vai trò là cầu nối giữa Model và View. Giữa Controller-View và Controller-Model đều là tương tác 2 chiều. Mỗi phàn trong đó sẽ bao gồm các đoạn code xử lý độc lập theo vai trò của mình. Mục đích chính của mô hình này nhằm chia nhỏ code để dễ phát triển và bảo trì.

Khi có một action từ phía user, ví dụ như submit form, action đó sẽ đi qua một Controller chính. Controller này gọi đến các Controller phụ (Service) và các Model cần thiết để xử lý. Sau đó, nó sẽ quyết định gọi đến phần View nào cần hiển thị và cập nhật lại. Đặc tính này cũng nâng cao tính tái sử dụng của code. Trong đó, View và Model không cần phải quan tâm mình được gọi như thế nào và ở đâu.

2.2 Công cụ thực hiện

### 2.1.1 Visual Studio Code

Visual Studio Code là trình soạn thảo, biên tập lập trình mã nguồn miễn phí được sử dụng trên 3 nền tảng đó là: Windows, macOS và Linux được xây dựng, phát triển bởi Microsoft. Visual Studio Code được các chuyên gia công nghệ thông tin đánh giá cao, nó là sự kết hợp hoàn hảo giữa IDE và CODE Editor.



Hình 8 Visual Studio Code logo

### 2.1.2 Postman

Postman là một ứng dụng desktop và dịch vụ trực tuyến được sử dụng để thử nghiệm, gỡ lỗi và phân tích các API (Application Programming Interface). Nó cho phép nhà phát triển tạo và gửi các yêu cầu HTTP / HTTPS đến các API để kiểm tra hoạt động của chúng và nhận phản hồi từ API đó.

Ảnh có chứa vòng tròn, hình mẫu, Đồ họa, minh họa

Mô tả được tạo tự động

Hình 9 Postman logo

### 2.1.3 MongoDB Compass

MongoDB Compass là một công cụ giao diện đồ họa cho phép bạn truy vấn, khám phá và quản lý cơ sở dữ liệu MongoDB trên máy tính cá nhân của bạn. Bạn có thể sử dụng MongoDB Compass để làm việc với cơ sở dữ liệu MongoDB local trên laptop của mình một cách dễ dàng và thuận tiện.

# PHÂN TÍCH VÀ THIẾT KẾ ỨNG DỤNG

3.1 Danh sách các tác nhân

### 3.1.1 Yêu cầu hệ thống

Yêu cầu về thiết kế chức năng tối ưu trải nghiệm của người dùng

* Chức năng đơn giản, thân thiện , dễ dàng sử dụng
* Đảm bảo các tính năng cơ bản của một trang web ecommerce

Admin: Đăng nhập / Đăng xuất, thêm sửa xoá product, category, blog, color, xem thông tin người dùng, các request của khách hàng (enquiry)

Customer: Đăng ký / Đăng nhập / Đăng xuất, xem chi tiết sản phẩm, thêm sản phẩm vào giỏ hàng, thanh toán, đánh giá sản phẩm, đọc blog

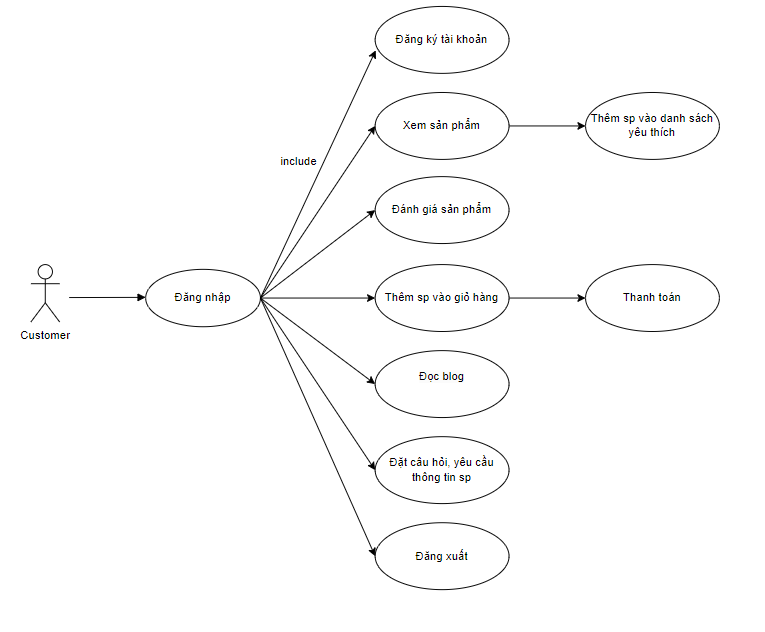
### 3.1.2 Phân tích hệ thống

#### 3.1.2.1 Biểu đồ UseCase tổng quan

A diagram of a diagram

Description automatically generated

Hình 10 Usecase admin



Hình 11 Usecase customer

#### 3.1.2.2 Biểu đồ lớp

A screenshot of a computer

Description automatically generated

Hình 12 Biểu đồ lớp

#### 3.1.2.3 Yêu cầu đặc tả

a) Yêu cầu chức năng:

* Người dùng có thể đăng nhập vào hệ thống bằng tên người email và mật khẩu mà trước đó đã đăng ký
* Hệ thống phải cung cấp một token cho người dùng hoặc admin khi đăng nhập thành công, để verify có quyền truy cập các chức năng được cấp phép

b) Yêu cầu phi chức năng:

* Hiệu suất: Hệ thống phải đáp ứng trong vòng 3 giây khi thực hiện các hoạt động xem, thêm, sửa đổi hoặc xóa các dữ liệu.
* Bảo mật: Hệ thống phải sử dụng jsonwebtoken để xác thực và phân quyền người dùng, đảm bảo chỉ người dùng có quyền mới được phép truy cập vào các chức năng liên quan.
* Độ tin cậy: Hệ thống cần đảm bảo tính ổn định.

# XÂY DỰNG BACKEND

4.1 Cấu hình môi trường phát triển

* Khởi tạo dự án React mới sử dụng template Redux
* npx create-react-app frontend --template redux
* cd frontend : truy cập vào dự án
* code . : Sau khi vào visual studio code thì mở terminal lên và npm start ứng dụng
* Khởi tạo dự án React
* npx create-react-app dashboard
* cd dashboard : truy cập vào dự án
* code . : Sau khi vào visual studio code thì mở terminal lên và npm start ứng dụng
* Các package hỗ trợ cho việc xây dựng FrontEnd:

Bảng 1 Các package hỗ trợ cho việc xây dựng FrontEnd

|  |  |  |
| --- | --- | --- |
| STT | Tên package | Lệnh cài đặt |
| 1 | ant-design | npm install antd |
| 2 | bootstrap | npm install bootstrap |
| 3 | react icons | npm install react-icons |
| 4 | react-toastify | npm install react-toastify |
| 5 | formik + yup | npm install formik yub |
| 6 | axios | npm install axios |
| 7 | react router dom | npm install react-router-dom |
| 8 | react Fast Marquee | npm install react-fast-marquee |
| 9 | react-image-zoom | npm install react-image-zoom |
| 10 | react-rating-stars-component | npm install react-rating-stars-component |
| 11 | moment | npm install moment |

* Khởi tạo dự án BackEnd với NodeJs
* Tạo 1 folder sau đó mở lên bằng visual studio code
* npm innit : Khởi tạo dự án
* Sau đó tạo build các folder : config, controllers, moldes, middwares, routes, file index.js và cài đặt các packages cho dự án
* Câu lệnh cài đặt: npm i express mongoose bcrypt body-parser dotenv
* body-parser : Thực chất là một middleware trong NodeJS, được sử dụng để xử lý các yêu cầu HTTP với dữ liệu gửi đến từ phía client trong các định dạng như JSON, URL\_encoded. Nó cho phép chúng ta truy cập các dữ liệu được gửi đến từ phía client thông qua các đối tượng request của Express. Để có thể viết API được, phía server NodeJS giúp tối giản các tham số từ client lên
* Sau đó cấu hình ở file index.js để khởi động con server và lắng nghe các kết nối đến từ client trên port 5000 được lưu ở file môi trường (.env). Kết hợp với công cụ nodemon để reload lại web khi có thay đổi. Cần phải vào file package.json để sửa lại dòng start và server

A black background with white text

Description automatically generated

* Bây giờ khi muốn chạy server, chỉ cần chạy dòng lệnh : **npm run server**
* Các package hỗ trợ cho việc xây dựng BackEnd:

Bảng 2 Các package hỗ trợ cho việc xây dựng BackEnd

|  |  |  |
| --- | --- | --- |
| STT | Tên package | Lệnh cài đặt |
| 1 | bcrypt | npm install bcrypt |
| 2 | mongoose | npm install mongoose |
| 3 | express | npm install express |
| 4 | body-parser | npm install body-parser |
| 5 | cloudinary | npm install cloudinary |
| 6 | cors | npm install cors |
| 7 | dotenv | npm install dotenv |
| 8 | jsonwebtoken | npm install jsonwebtoken |
| 9 | morgan | npm install morgan |
| 10 | multer | npm install multer |
| 11 | moment | npm install moment |
| 12 | sharp | npm install sharp |
| 13 | uniqid | npm install uniqid |
| 14 | razorpay | npm install razorpay |
| 15 | nodemon | npm install nodemon |

4.2 Xây dựng các router và controllers

* Tạo 1 repository trên git cá nhân.
* Xây dựng modal
* Modal :

A screenshot of a computer program

Description automatically generated

* **dotenv :** giúp chúng ta có thể lấy được các tham số khai báo trong file môi trường
* Config folder
* **dbConnect.js :** Có mục đích kết nối với cơ sở dữ liệu MongoDB bằng sử dụng thư viện Mongoose trong Node.js.

A computer screen shot of a program code

Description automatically generated

* **jwtToken.js :** tạo và trả về một JWT (JSON Web Token) sau khi người dùng đăng nhập thành công

A computer screen shot of a program code

Description automatically generated

* refreshToken :

A computer screen with text on it

Description automatically generated

* Tránh sử dụng JWT vô thời hạn, để tăng cường bảo mật ta phải yêu cầu người dùng đăng nhập lại sau 3 ngày sau khi JWT hết hạn.
* Nếu khi user đăng nhập thành công, ta cần phải res 1 cái token cho user. Khi gửi các yêu cầu tiếp theo đến server, client sẽ gửi token đó kèm theo trong yêu cầu để xác thực người dùng. Server sẽ kiểm tra tính hợp lệ của token và xác định quyền truy cập của người dùng dựa trên thông tin được lưu trữ trong token.
* Ví dụ khi register :
* A computer code on a white background

  Description automatically generated
* Ví dụ khi login :

A computer screen shot of a computer screen

Description automatically generated

* Xây dựng các model
* **Model folder:** Chứa các thành phần xử lý dữ liệu và logic của ứng dụng. Tạo các model tương ứng trong Mongoose

A screenshot of a computer

Description automatically generated

* **blogCatModel.js** : Đây là danh mục blog, gồm có tiêu đề và timestamp
* **blogModel.js** : Đây là model blog, gồm có tiêu đề, mô tả bài viết, author tác giả của bài viết, hình ảnh, danh mục, trạng thái của bài viết
* Tương tự như vậy cho các model khác gồm có các trường cơ bản đầy đủ để phù hợp với website
* Đặc biệt ở **userModel**, ngay dưới server em định nghĩa các phương thức như:
* **Hash password:** Trước khi lưu (pre-save hook) xuống CSDL, em sẽ thực hiện một bước mã hoá password bằng cách sử dụng thư viện **bcrypt** để tạo salt và mã hoá mật khẩu.

A computer screen shot of a program code

Description automatically generated

* **Compare password:** Việc cần làm tiếp theo là check xem người dùng có đúng là email và password đó không. Ta sử dụng cái method được định nghĩa ở dưới isPasswordMatched và truyền password vào để kiểm tra

A screen shot of a computer screen

Description automatically generated

* **Reset token:** Ở đây em dùng module crypto trong NodeJS để tạo ra chuỗi ngẫu nhiên có độ dài 32 byte rồi chuyển đổi nó sang định dạng hex (hệ thập lục phân), sau đó băm nó bằng thuật toán SHA-256. Sau đó tính toán thời gian hết hạn của token sau 30 phút

A computer screen shot of a program code

Description automatically generated

* Xây dựng các router và controllers
* **Controller folder** **:** Chứa các file module Controller để xử lý các logic và hành động liên quan đến Model tương ứng. Nhận request từ người dùng và thực hiện xử lý sau đó trả respone lại cho người dùng (như CRUD)

A screenshot of a computer

Description automatically generated

* **Routes folder :** Định nghĩa các route cho ứng dụng. Mỗi file route đại diện cho mỗi nhóm route liên quan đến mỗi phần của ứng dụng.

A screenshot of a computer

Description automatically generated

* authRoute : chứa các route liên quan đến xác thực, quản lý người dùng, giỏ hàng, đơn hàng
* blogCatRoute : các route quản lý danh mục blog
* blogRoute : quản lý các bài đăng trên blog
* brandRoute : quản lý các brand
* colorRoute : quản lý các màu sắc
* contactRoute : quản lý các liên hệ
* couponRoute : quản lý các mã giảm giá
* enqRoute : quản lý các yêu cầu thông tin và câu hỏi của người dùng
* prodcategoryRoute : quản lý danh mục sản phẩm
* productRoute : quản lý sản phẩm
* uploadRoute : quản lý hình ảnh
* Quản lý xác thực và xử lý module hoá
* **Middlewares folder :** Chứa các middleware xử lý các tác vụ, tăng tính module hoá và quản lý ứng dụng

A screenshot of a computer error

Description automatically generated

* **authMiddleware.js :** Định nghĩa các hàm liên quan đến xác thực người dùng. Mục đích là để xem req này được uỷ quyền hay không. Nó sẽ trích xuất từ header xem có chứa chuỗi “Bearer” hay không.
* Chuỗi Bearer ở đây là một phần của chuẩn xác thực Oauth 2.0, được sử dụng để định dang và xác thực người dùng trong các ứng dụng web và di động
* Sau đó tách chuỗi, dùng split để tách chuỗi thành mảng các chuỗi con bằng khoảng trắng ‘ ‘, [1] để lấy token phía sau
* Tiếp theo, nó sẽ xem cái token có hợp lệ hay k, bằng cách dùng **jwt.verify()** để xác thực và giải mã. Nếu token hợp lệ nó sẽ lấy ra id người dùng đó, sau đó gắn vào đối tượng user thông qua req.user rồi next() đến middleware hoặc route tiếp theo
* isAdmin? – xác minh và kiểm tra xem người dùng có phải là admin hay không ? Rồi sau thực hiện một số validate cho Mongodb Id, sau đó tạo unlock hay block func cho admin để chặn người dùng => Verify the JWT token
* **handleError.js :** Kiểm tra và xử lý lỗi
* **uploadImages.js :** Xử lý ảnh tải lên bằng các thư viện : multer, sharp, path và fs
* Utils folder :

A black background with white text

Description automatically generated

* **cloudinary :** sử dụng thư viện Cloudinary để tải lên và xóa hình ảnh từ ứng dụng của mình.
* **validateMongodbId :** Tại sao lại phải cần Validate ? Để kiểm tra tính hợp lệ của mỗi chuỗi định danh ObjectId của MongoDB. Ví dụ, khi muốn update user hay get user, ta phải cần kiểm tra tính hợp lệ của ObjectId để đảm bảo rằng ta đang truy vấn đúng tài khoản người dùng mà ta muốn cập nhật hay lấy thông tin. Sử dụng phương thức .isValid() trong ObjectId
* Cấu hình và khởi động
* **Index.js :** Đây là 1 file entry point. Khi chạy file này, nó sẽ tạo ra một instance của ứng dụng Express và cấu hình các middleware, router và các cài đặt khác để ứng dụng có thể hoạt động. Mỗi route trong file này tương ứng mỗi nhóm route tương ứng với một phần của ứng dụng.

A screen shot of a computer program

Description automatically generated

# XÂY DỰNG GIAO DIỆN NGƯỜI DÙNG VỚI REACT

5.1 Tạo các component và cấu trúc UI

* Tạo 1 repository trên git cá nhân.
* Xây dựng khung sườn của dự án

A screenshot of a computer program

Description automatically generated

* Bây giờ hãy đi sâu vào việc tìm hiểu mục đích của từng folder !
* Đây là cách tổ chức dự án và triển khai quản lý trạng thái sử dụng Redux Toolkit.
* **features folder :** Đây là nơi chứa các tệp tin liên quan đến các tính năng của trang web.
* Mỗi file **service** là một module chứa logic giao tiếp với dịch vụ hoặc API liên quan đến đối tượng nào đó (vd: người dùng, product). Trong file này em định nghĩa các hàm để thực hiện các yêu cầu HTTP, xử lý dữ liệu và gửi các hành động (actions) đến file **slice** tương ứng để cập nhật trạng thái của đối tượng.
* Còn mỗi file **slice** là module chứa reducer và các hành động liên quan đến phần trạng thái của 1 đối tượng cụ thể, cập nhật thông qua các hành động và xử lý các yêu cầu từ file **service**.

A screenshot of a computer program

Description automatically generated

* **app folder :** Ở đây chính nơi lưu trữ store của trang web (store.js). Store là gì ? Là nơi lưu trữ trạng thái toàn cục của ứng dụng một cách có cấu trúc và dễ bảo trì

A screen shot of a computer

Description automatically generated

* reducer được cung cấp cho configureStore thông qua thuộc tính reducer. Mỗi reducer này tương ứng với các slice khác nhau trong ứng dụng
* Đầu tiên, cần xây dựng các component cần thiết cho ứng dụng. Việc chia tách từng phần website ra từng các component giúp cho việc code trở nên dễ dàng hơn, dễ bảo trì và có thể tái sử dụng nhiều lần

A screenshot of a computer

Description automatically generated

* Ở trong này, em định nghĩa một cái file main **Layout**, sử dụng thành phần <Outlet /> của react router để định vị và hiển thị các thành phần của trang web. Và em sẽ đặt <Outlet /> trong giữa header và footer dùng chung

A screen shot of a computer program

Description automatically generated

* Tiếp theo là xây dựng các pages tương ứng. Ở phía giao diện người dùng hiện tại, website có các trang như sau : Trang home, our store, my orders, blogs, contact, register/login/logout, wishlist, detail product, cart, check out, view profile

A screenshot of a computer

Description automatically generated

* Để chuyển trang trên trang web, ta cần định nghĩa các route tương ứng với mỗi trang

A screen shot of a computer program

Description automatically generated

* Bây giờ sau khi có bộ khung rồi thì ta có thể dễ dàng để xây dựng trang web của mình. Chúng ta sẽ xây dựng từng thành phần nội dung cho mỗi trang tương ứng
* Ngoài ra còn có folder routing, gồm có 2 file là PrivateRoutes và PublicRoutes. Hai file này có nhiệm vụ là quản lý quyền truy cập và điều hướng trong ứng dụng một cách bảo mật và có tổ chức, chỉ khi người dùng đăng nhập mới được phép truy cập vào các trang như là giỏ hàng, thanh toán, … Để làm được điều này thì đầu tiên em sẽ getItem ‘token’ ở trên localStorage về, nếu nó bằng null tức là không có token, vậy thì em sẽ chuyển hướng người dùng sang những trang chỉ cho phép xem mà không cần đăng nhập (home, our store, blog)

A computer screen shot of a program code

Description automatically generated

* Ngược lại, nếu có token (tức là !== null) thì nó sẽ render children (các thành phần con được truyền vào PrivateRoutes). Điều này cho phép người dùng truy cập vào các privateRoute.

A screen shot of a computer code

Description automatically generated

* Cuối cùng utils folder, ở đây em có thực hiện các việc như là lưu các data tĩnh (không thay đổi trên trang web) và hàm config. Hàm này có nhiệm vụ là chỉ để xác thực người dùng bằng cách sử dụng axios để thực hiện các yêu cầu HTTP. (Ở các file Service)

A computer screen shot of a program

Description automatically generated

# TÍCH HỢP VÀ KIỂM THỬ

6.1 Tích hợp frontend và backend

* Kiểm thử chức năng và giao diện phía User

### 6.1.1 Chức năng đăng ký

* FrontEnd
* Link code : <https://tinyurl.com/yxtwkr7s>
* Giao diện :

A screenshot of a login form

Description automatically generated

* Hàm Service :

A screen shot of a computer code

Description automatically generated

* Hàm Slice :

A screen shot of a computer code

Description automatically generated

* BackEnd

A computer screen shot of a program code

Description automatically generated

* Mô tả : Khi người dùng đăng ký, action registerUser sẽ gửi một request thông qua hàm register (service), gửi kèm userData. Ở dưới server nó sẽ thực hiện truy vấn xem email đó đã được đăng ký hay chưa. Nếu không tìm thấy thì tạo mới còn tìm thấy thì hiện thị thông báo lỗi. Nếu tất cả điều kiện đúng, dữ liệu người dùng được lưu trữ trong localStorage với key "customer" và giá trị là một chuỗi JSON.

### 6.1.2 Chức năng đăng nhập, đăng xuất

* FrontEnd
* Link code : <https://tinyurl.com/mwc73dxd>
* Giao diện :

A screenshot of a login page

Description automatically generated

* Hàm Service :

A computer screen shot of a code

Description automatically generated

* Hàm Slice :

A screen shot of a computer code

Description automatically generated

* BackEnd

A screen shot of a computer program

Description automatically generated

* Mô tả : Khi người dùng đăng nhập, thì action loginUser sẽ gửi yêu cầu thông qua hàm login, nó sẽ gửi một request xuống dưới server kèm theo data. Sau đó ở dưới server nó sẽ lấy tìm email đó xem tồn tại hay không, và xem password có match hay không. Nếu tất cả là đúng thì nó sẽ trả về thông tin của người dùng, ngược lại thì trả về lỗi.
* Khi người dùng muốn đăng xuất, ta chỉ cần định nghĩa một cái hàm handleLogout có nhiệm vụ là clear đi localStorage và reload lại trang.

### 6.1.3 Chức năng thêm sản phẩm vào giỏ hàng

* FrontEnd
* Link code : <https://tinyurl.com/4wmsr934>
* Giao diện :

A screenshot of a computer

Description automatically generated

* Hàm Service :

A computer screen shot of a code

Description automatically generated

* Hàm Slice :

A screen shot of a computer program

Description automatically generated

* BackEnd

A screen shot of a computer program

Description automatically generated

* Mô tả : action addCartToProduct sẽ gửi một yêu cầu xuống backend thông qua hàm service addToCart (hàm này có gắn token để verify hợp lệ). Ở phía backend nó sẽ thực hiện một bước xác nhận id của người dùng xem có hợp lệ hay không trước thêm sản phẩm vào giỏ hàng.

### 6.1.4 Chức năng thanh toán

* FrontEnd
* Link code : <https://tinyurl.com/mr25e6um>
* Giao diện :

A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

* Hàm Service :

A computer screen shot of a program code

Description automatically generated

* Hàm Slice :

A screen shot of a computer code

Description automatically generated

* BackEnd

A screen shot of a computer

Description automatically generated

* Mô tả : action createAnOrder sẽ gửi một yêu cầu kèm theo chi tiết của đơn hàng xuống dưới backend thông qua hàm service createOrder để tạo một đơn hàng mới. Sau đó ở phía backend sẽ tạo một đơn hàng mới dựa trên các thông tin lấy từ req.body và thông tin về người dùng (\_id) từ req.user.

### 6.1.5 Chức năng tìm kiếm sản phẩm

* FrontEnd
* Link code : <https://tinyurl.com/78cwr9ft>
* Giao diện :

A screenshot of a website

Description automatically generated

* Sử dụng Typeahead – đây là một thành phần UI, có chức năng cung cấp gợi ý tự động và hoàn thành tự động trong trường nhập liệu.

A screen shot of a computer screen

Description automatically generated

Mô tả : Khi nhập vào ô tìm kiếm, dựa vào option là productOpt (chứa các sản phẩm). Khi sự kiện onChange thay đổi, nó sẽ điều hướng người dùng đến trang /product/{selected[0]?.prod} (với selected[0]?.prod là giá trị của thuộc tính prod trong tùy chọn đã chọn) và gửi một yêu cầu để lấy thông tin chi tiết về sản phẩm bằng cách gọi hàm dispatch(getAProduct(selected[0]?.prod)).

### 6.1.6 Chức năng đánh giá sản phẩm

* FrontEnd
* Link code : <https://tinyurl.com/apw88zbw>
* Giao diện :

A screenshot of a computer

Description automatically generated

* Hàm Service :

A screen shot of a computer program

Description automatically generated

* Hàm Slice :

A screen shot of a computer program

Description automatically generated

* BackEnd :

A screen shot of a computer program

Description automatically generated

A computer screen shot of a program code

Description automatically generated

* Mô tả : action addRating này sẽ gửi một resquest xuống backend thông qua hàm service rateProduct. Đầu tiên, lấy \_id của người dùng, star và prodId từ req.body. Rồi tìm kiếm sản phẩm tương ứng với prodId. Sau đó kiểm tra xem sản phẩm đã được người dùng đánh giá trước đó hay chưa (bằng cách sử dụng pt find() để tìm pt trong mảng ratings của sản phẩm có postedby (được sử dụng để xác định người dùng nào đã đánh giá sản phẩm) trùng với \_id của người dùng. Nếu sản phẩm đã được đánh giá, tiến hành cập nhật đánh giá của người dùng với pt updateOne() (sd $elemMatch để tìm kiếm phần tử trong mảng ratings của sản phẩm có postedby trùng với \_id của người dùng và cập nhật giá trị star của phần tử này bằng star được gửi lên từ req.body.).
* Kiểm thử chức năng và giao diện phía Admin

### 6.1.7 Chức năng đăng nhập

* FrontEnd
* Link code : <https://tinyurl.com/ye229ve6>
* Giao diện :

A screenshot of a login screen

Description automatically generated

* Hàm Service :

A computer screen shot of a program code

Description automatically generated

* Hàm Slice :

A screen shot of a computer code

Description automatically generated

* BackEnd

A screen shot of a computer program

Description automatically generated

Mô tả : Khi người dùng đăng nhập, thì action login sẽ gửi yêu cầu thông qua service, nó sẽ gửi một request POST xuống dưới server kèm theo data. Sau đó ở dưới server nó sẽ lấy email và password đó, sau đó kiểm tra đó có phải là admin hay không. Nếu tất cả đều đúng, thì nó sẽ tạo 1 token và refreshToken mới, sau đó trả kèm theo token và thông tin tương ứng trong respone. Sau đó trả ngược lại cho thằng slice, nếu thành công thì nó trả về dữ liệu (thông qua redux state) còn không thì trả về giá trị lỗi. Cuối cùng, lưu trữ thông tin nhận được từ phản hồi vào localStorage với tên cột là ‘user‘

### 6.1.8 Chức năng đăng xuất

**A computer screen shot of a program code

Description automatically generated**

* Mô tả : Vì lưu thông tin người dùng ở trên localStorage nên lúc muốn logout thì chỉ cần định nghĩa một hàm như trên. Khi onClick vào thì nó sẽ remove đi item có tên là toekn và sẽ điều hướng người dùng tới trang login và hiện thị thông báo đã đăng xuất thành công

### 6.1.9 Chức năng thêm sản phẩm

* FrontEnd
* Link code : <https://tinyurl.com/yckrxas8>
* Giao diện :

A screenshot of a computer

Description automatically generated

* Hàm Service :

A computer screen shot of a code

Description automatically generated

* Upload Service :

A computer screen shot of a code

Description automatically generated

* Hàm Slice :

A screen shot of a computer code

Description automatically generated

* Upload Slice :

A computer screen with colorful text

Description automatically generated

* BackEnd

A screen shot of a computer program

Description automatically generated

* Các hàm liên quan : productImgResize, uploadImgs (Sử dụng dịch vụ Cloudinary để lưu trữ và quản lý hình ảnh)
* Mô tả : action createProducts (slice) gửi yêu cầu tạo sản phẩm từ frontend tới backend thông qua hàm createProduct (service). Backend xử lý yêu cầu, tạo sản phẩm mới và trả về kết quả cho frontend.

### 6.1.10 Chức năng thêm, sửa xoá brand

* FrontEnd
* Link code thêm brand : <https://tinyurl.com/armruxsj>
* Link code sửa và xoá brand: <https://tinyurl.com/4cmc7y44>
* Giao diện :

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* Hàm Service :

A screen shot of a computer program

Description automatically generated

* Hàm Slice :

A screen shot of a computer program

Description automatically generated

* BackEnd

A screen shot of a computer program

Description automatically generated

* Mô tả : Các action createBrand, updateABrand, deleteABrand (slice) gửi yêu cầu tạo tên brand, chỉnh sửa và xoá brand từ frontend tới backend thông qua hàm các hàm tương ứng createBrand, deleteBrand, updateBrand (service). Backend xử lý yêu cầu sau đó trả về kết quả cho frontend.
* Các chức năng thêm, xoá, sửa Category, Color, Blog, Blog Category, Coupon cũng tương tự như trên.
* Ở phía admin, lúc nào ta gửi request xuống server thì phải gửi kèm hàm config. Vì chỉ có admin mới có quyền thêm, sửa, xoá các thành phần trên giao diện.

6.2 Sửa lỗi và cải thiện hiệu suất

Hiện website mặc dù đã có những chức năng cơ bản. Tuy nhiên việc nâng cấp website bằng cách thêm các chức năng nâng cao, xử lý trải nghiệm đang gặp một số vấn đề liên quan đến việc thiết kế cơ sở lúc đầu chưa được tối ưu, dẫn đến việc thực hiện các thay đổi trên website gây ảnh hưởng đến trải nghiệm người dùng. Giải pháp mà em nghĩ ra đó là, xem lại cơ sở dữ liệu, xử dụng các kỹ thuật công nghệ mới nhằm tối ưu hiệu năng trang web, xác định và xử lý các vấn đề liên quan đến việc xác thực và quản lý token người dùng. Qua đó, sẽ nâng cấp các tác nhân và chức năng cho trang web trong tương lai.

# KẾT LUẬN

7.1 Tổng kết

### 7.1.1 Những điều đạt được

* Hiểu được quy trình xây dựng một dự án fullstack từ đầu đến cuối
* Nắm được kiến thức Database, các Relationship và sử dụng Restful API để thực hiện các thao tác GET, POST, PUT, DELETE.
* Nâng cao khả năng tự học, tiếp thu kiến thức một cách chủ động.
* Cải thiện kỹ năng giao tiếp, nâng cao hiểu biết của bản thân trong môi trường làm việc chuyên nghiệp của doanh nghiệp.
* Được training những kỹ năng chuyên môn và một số kỹ năng mềm như: bảo mật trong doanh nghiệp, trao đổi trong các buổi seminar,…
* Nâng cao kỹ năng viết email, báo cáo, cũng như báo cáo trong các buổi họp team hàng tuần..

### 7.1.2 Kết luận

Khởi đầu một dự án xây dựng một trang web Ecommerce bằng MERN stack là một thử thách thú vị. Trong quá trình này, em đã tìm hiểu và tham khảo nhiều nguồn kiến thức, từ tài liệu hướng dẫn trực tuyến đến các dự án mã nguồn mở tương tự đã được triển khai thành công.

Việc tham khảo các nguồn tài liệu và dự án đã giúp em hiểu rõ hơn về kiến trúc và cấu trúc của MERN stack. Em đã nắm được nguyên tắc hoạt động của MongoDB, Express.js, React và Node.js, và biết cách kết hợp chúng để tạo ra một ứng dụng web mạnh mẽ.

Em cũng đã tìm hiểu và tham khảo các best practice và quy ước trong việc phát triển ứng dụng web. Từ việc quản lý trạng thái ứng dụng bằng Redux, đến việc sử dụng các thư viện UI như Ant-design hay Bootstrap để tạo giao diện hấp dẫn và đáp ứng, em đã rút ra nhiều kinh nghiệm quý giá từ những nguồn tham khảo này.

# TÀI LIỆU THAM KHẢO

[1] https://www.w3schools.com/react/default.asp

[2] https://redux-toolkit.js.org/introduction/getting-started

[3] https://expressjs.com/en/guide/routing.html