

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN**

Đề tài

**TÌM HIỂU REACT JS VÀ XÂY DỰNG WEBSITE
BÁN QUẦN ÁO**

**Sinh viên: Nguyễn Văn Nhân
Mã số: B1809272
Khóa: K44**

Cần Thơ, 05/2021

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN**

Đề tài

**TÌM HIỂU REACT JS VÀ XÂY DỰNG WEBSITE
BÁN QUẦN ÁO**

Người hướng dẫn
TS. Lâm Nhựt Khang

Sinh viên thực hiện
Nguyễn Văn Nhân
MSSV: B1809272
Khóa: K44

Cần Thơ, 05/2021

LỜI CẢM ƠN

Em xin chân thành gửi lời cảm ơn đến cô Lâm Nhật Khang, cảm ơn vì sự giúp đỡ nhiệt tình của cô trong suốt quá trình thực hiện đề tài. Khi làm việc với cô em học hỏi được rất nhiều điều bổ ích và kỹ năng lập trình của em cũng được cải thiện phần nào. Khi hoàn thành xong Website đã giúp em củng cố rất nhiều kiến thức về lập trình web và biết thêm những kiến thức mới rất có ích cho em trong công việc sau này. Để hoàn thành xong Website ngoài những nỗ lực của bản thân, em còn nhận được sự giúp đỡ nhiệt tình của các anh, chị và bạn bè trong Trường Đại học Cần Thơ.

Em xin chân thành cảm ơn!

Mục lục

PHẦN 1: TỔNG QUAN VỀ ĐỀ TÀI	1
1. Giới thiệu.....	1
2. Mục tiêu đề tài.....	1
3. Đối tượng và phạm vi nghiên cứu	1
3.1. Đối tượng nghiên cứu.....	1
3.2. Phạm vi nghiên cứu.....	1
4. Phương pháp nghiên cứu.....	1
5. Nội dung nghiên cứu	2
PHẦN 2: NỘI DUNG	3
Chương 1: Đặc tả yêu cầu và phân tích thiết kế hệ thống.....	3
1.1. Đặc tả.....	3
1.2. Các nhóm chức năng của hệ thống.....	3
1.3. Các tác nhân chính của hệ thống gồm:.....	4
1.4. Xác định use case.	4
1.4.1. Xác định các use case.....	4
1.4.2. Sơ đồ use case chi tiết	5
1.5. Mô tả chi tiết các use case	6
1.5.1. Đăng ký	6
1.5.2. Đăng nhập.....	6
1.5.3. Xem sản phẩm	6
1.5.4. Cập nhật giỏ hàng.....	7
1.5.5. Thanh toán.....	7
1.5.6. Thay đổi thông tin tài khoản.....	8
1.5.7. Thay đổi mật khẩu.....	8
1.5.8. Xem đơn hàng	9
1.5.9. Hủy đơn hàng	9
1.5.10. Đăng xuất	10
1.5.11. Thêm sản phẩm	10
1.5.12. Sửa thông tin sản phẩm	10
1.5.13. Xóa sản phẩm	11
1.5.14. Các chức năng khác tương tự như trên.....	11
Chương 2: Thiết kế và cài đặt giải pháp.....	12
2.1. Cơ sở lý thuyết	12
2.1.1. Giới thiệu về API và RESTful API.....	12
2.1.2. ReactJS	13
2.1.2.1. JavaScript XML	13
2.1.2.2. Components.....	13

2.1.2.3. Props và State	15
2.1.2.4. LifeCycle	16
2.1.2.5. Hook	16
2.1.3. Thư viện Redux	17
2.1.4. Thư viện Axios	18
2.1.5. Thư viện React UI – Material UI	18
2.1.5. MySQL.....	19
2.1.6. NodeJS.....	19
2.1.6.1. Blocking I/O và Nonblocking I/O	19
2.1.6.2. Synchorous và Asynchorous	20
2.1.6.3. Callback.....	20
2.6.1.4. Event Loop	20
2.6.1.5. Module.....	21
2.1.6.6. NPM - Node Package Manager	21
2.1.7. ExpressJS Framwork.....	22
2.2. Thiết kế hệ thống	24
2.2.1. RESTful API	24
2.2.2. Trang quản lý.....	34
2.2.3. Trang người dùng	41
Chương 3: Đánh giá kiểm thử	48
3.1. Mục tiêu kiểm thử	48
3.2. Kịch bản kiểm thử	48
3.3. Đánh giá kiểm thử	50
PHẦN 3: KẾT LUẬN	55
1. Kết quả đạt được	55
1.1. Kết quả	55
1.2. Hạn chế.....	55
2. Hướng phát triển.....	55

Danh mục hình ảnh

Hình 1. Sơ đồ use case của tác nhân Khách hàng khách và Khách hàng thành viên.....	5
Hình 2. Sơ đồ use case của tác nhân Người quản lý	5
Hình 3. Mô hình hoạt động cơ bản của REST	12
Hình 4. logo thư viện reactjs	13
Hình 5. function components hợp lệ	14
Hình 6. function components sử dụng arrow function	14
Hình 7. Ví dụ class components.....	14
Hình 8. Props.....	15
Hình 9. State trong class components	15
Hình 10. State trong function components	16
Hình 11. LifeCycle trong ReactJS.....	16
Hình 12. logo thư viện redux.....	17
Hình 13. logo của MySQL	19
Hình 14. logo thư viện nodejs	19
Hình 15. logo expressjs framework.....	22
Hình 16. Ví dụ cấu trúc route	23
Hình 17. Mô hình cơ sở dữ liệu.....	24
Hình 18. Kiến trúc các thành phần của REST API	32
Hình 19. Lưu đồ xử lý yêu cầu của REST API server	33
Hình 20. Giao diện màn hình đăng nhập.....	34
Hình 21. Lưu đồ xử lý thực hiện yêu cầu đăng nhập	35
Hình 22. Giao diện quản lý thông tin người dùng.....	35
Hình 23. Lưu đồ xử lý liệt kê danh sách người dùng.....	36
Hình 24. Giao diện quản lý sản phẩm	37
Hình 25. Giao diện tạo mới một sản phẩm.....	37
Hình 26. Lưu đồ xử lý tạo mới sản phẩm.....	37
Hình 27. Giao diện sửa thông tin sản phẩm	38
Hình 28. Giao diện cập nhật ảnh sản phẩm.....	38
Hình 29. Giao diện xóa sản phẩm	39
Hình 30. Giao diện trang quản lý loại sản phẩm	39
Hình 31. Giao diện trang quản lý danh mục size	39
Hình 32. Giao diện quản lý danh mục màu sắc.....	40
Hình 33. Giao diện quản lý chi tiết sản phẩm	40
Hình 34. Giao diện trang quản lý hóa đơn	41
Hình 35. Giao diện trang quản lý đánh giá sản phẩm	41
Hình 36. Kiến trúc các thành phần của web client.....	41
Hình 37. Giao diện trang đăng ký	42
Hình 38. Giao diện trang chủ	42
Hình 39. Giao diện trang chi tiết sản phẩm.....	43
Hình 40. Dialog thanh toán sản phẩm	43
Hình 41. Giao diện trang thanh toán paypal.....	44
Hình 42. Giao diện trang đơn hàng	44
Hình 43. Giao diện giỏ hàng	45
Hình 44. Giao diện trang thông tin cá nhân	45
Hình 45. Giao diện dialog cập nhập thông tin cá nhân	46
Hình 46. Giao diện dialog cập nhập ảnh đại diện	46
Hình 47. Giao diện dialog đổi mật khẩu	46
Hình 48. Giao diện trang đăng nhập khi chưa nhập thông tin.....	50
Hình 49. Giao diện trang đăng ký khi chưa nhập thông tin	51
Hình 50. Giao diện trang đăng ký khi nhập tên đăng nhập đã tồn tại	51

Hình 51. Giao diện trang đổi mật khi nhập mật khẩu cũ sai	52
Hình 52. Giao diện khi cập nhật ảnh đại diện thành công.....	52
Hình 53. Giao diện thanh toán giỏ hàng khi giỏ hàng chưa có sản phẩm	52
Hình 54. Giao diện hủy thanh toán khi thanh toán sản phẩm	53
Hình 55. Giao diện hủy đơn hàng khi chủ cửa hàng chưa xác nhận	53
Hình 56. Giao diện khi xóa một người dùng.....	53
Hình 57. Giao diện khi xác nhận đơn hàng thành công	54
Hình 58. Giao diện khi xóa một sản phẩm thành công	54

Danh mục bảng

Bảng 1. Nội dung nghiên cứu.....	2
Bảng 2. Chi tiết use case đăng ký.....	6
Bảng 3. Chi tiết use case đăng nhập.....	6
Bảng 4. Chi tiết use case xem sản phẩm.....	7
Bảng 5. Chi tiết use case cập nhật giỏ hàng.....	7
Bảng 6. Chi tiết use case thanh toán.....	8
Bảng 7. Chi tiết use case thay đổi thông tin tài khoản.....	8
Bảng 8. Chi tiết use case thay đổi mật khẩu.....	9
Bảng 9. Chi tiết use case xem đơn hàng.....	9
Bảng 10. Chi tiết use case hủy đơn hàng.....	10
Bảng 11. Chi tiết use case đăng xuất.....	10
Bảng 12. Chi tiết use case thêm sản phẩm.....	10
Bảng 13. Chi tiết use case sửa thông tin sản phẩm.....	11
Bảng 14. Chi tiết use case xóa sản phẩm.....	11
Bảng 15. Các tác vụ cơ bản của REST dựa trên phương thức HTTP.....	12
Bảng 16. một số phương thức (response methods) hỗ trợ hay dùng nhất.....	23
Bảng 17. Danh sách các bảng trong cơ sở dữ liệu.....	25
Bảng 18. Profiles.....	25
Bảng 19. user.....	26
Bảng 20. Orders.....	26
Bảng 21. Categories.....	26
Bảng 22. Products.....	27
Bảng 23. orderproducts.....	27
Bảng 24. Sizes.....	27
Bảng 25. Colors.....	28
Bảng 26. Products_has_Sizes.....	28
Bảng 27. Products_has_Colors.....	28
Bảng 28. Reviews.....	29
Bảng 29. danh sách mô tả các REST endpoint.....	31
Bảng 30. danh sách một số status code thông dụng được trả về từ response.....	32
Bảng 31. Kết quả kiểm thử kịch bản.....	50

PHẦN 1: TỔNG QUAN VỀ ĐỀ TÀI

1. Giới thiệu

Sự phát triển của công nghệ thông tin, đặc biệt là sự ra đời và phát triển của mạng internet cùng với dịch vụ website đã tạo ra một thời kỳ quảng cáo thương hiệu, quảng bá sản phẩm. Khách hàng có thể truy cập internet để xem thông tin, lựa chọn sản phẩm, đặt mua sản phẩm ở nhà mà không cần đến cửa hàng, thậm chí có thể mua các sản phẩm trên khắp thế giới. Đồng thời, các nhà sản xuất cũng tiết kiệm phần nào chi phí quảng cáo, khách hàng cũng dễ dàng xem, lựa chọn sản phẩm.

2. Mục tiêu đề tài

- Tìm hiểu lý thuyết về React JS
- Xác định cấu trúc thành phần chính trong React JS
- Xây dựng website bán quần áo
- Quản lý sản phẩm, cập nhật, thêm, sửa, xóa sản phẩm
- Tùy chỉnh giao diện website
- Xem, sửa, quản lý các đơn hàng mà khách hàng đã đặt

3. Đối tượng và phạm vi nghiên cứu

3.1. Đối tượng nghiên cứu

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

- REST API server: xây dựng một server cung cấp các thao tác tạo, thêm, sửa, xóa (CRUD) dữ liệu của hệ thống thông qua HTTP request.
- Web client: xây dựng một ứng dụng web giao tiếp với server thông qua REST API, nhằm mục đích quản lý dữ liệu cốt lõi của hệ thống.

3.2. Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài chủ yếu là làm việc với React JS để tạo ra website bán quần áo. Ngoài ra, sẽ sử dụng thư viện UI (User Interface) của React JS là Material UI để xây dựng giao diện website, Axios để giao tiếp với API server, phía backend sẽ sử dụng NodeJS với framework Express để xây dựng nhanh hệ thống RESTful API

4. Phương pháp nghiên cứu

Nội dung nghiên cứu khá phổ biến, do đó phương pháp nghiên cứu chủ yếu là tìm hiểu thông qua tài liệu trang chủ của các thư viện, thông qua các video trên YouTube và làm các ví dụ trực tiếp. Từ đó, sẽ vận dụng các kiến thức đã tiếp thu được vào thực

hiện đề tài. Tương ứng với mỗi thành phần của hệ thống sẽ có những phương pháp nghiên cứu khác nhau:

API server: Nghiên cứu cơ bản về NodeJS với framework Express để xây dựng một hệ thống RESTful và hệ quản trị cơ sở dữ liệu MySQL.

Web client: Nghiên cứu cơ bản về ứng dụng web sử dụng các thư viện react, redux, axios, tích hợp cổng thanh toán Paypal và material-ui.

5. Nội dung nghiên cứu

Nội dung nghiên cứu được liệt kê dưới đây.

STT	Đối tượng	Nội dung
1	API và RESTful API	Tìm hiểu về API và RESTful API
2	NodeJS – Express	Tìm hiểu NodeJS với framework Express Xây dựng model dữ liệu (các bảng của cơ sở dữ liệu) của đề tài
3	MySQL	Tìm hiểu hệ quản trị cơ sở dữ liệu MySQL Cấu hình MySQL trong NodeJS
4	ReactJS	Tìm hiểu về ReactJS: components, state và props Sử dụng webhook trong react Routing bằng reactjs và chuyển hướng đăng nhập
5	Redux	Tìm hiểu về redux: state, action, container, dispatcher Sử dụng redux vào ứng dụng react
6	Axios	Tìm hiểu về axios Trao đổi với API server thông qua axios
7	Thư viện UI Material	Tìm hiểu về Material và các components mà thư viện cung cấp Xây dựng các components của ứng dụng, kết hợp với redux để chia sẻ các state dùng chung

Bảng 1. Nội dung nghiên cứu

PHẦN 2: NỘI DUNG

Chương 1: Đặc tả yêu cầu và phân tích thiết kế hệ thống

1.1. Đặc tả

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

Người quản trị sẽ đưa các sản phẩm lên website với đầy đủ các thông tin về sản phẩm đó như: kiểu dáng, màu sắc, size, ... Khi khách hàng truy cập vào website của cửa hàng, khách hàng sẽ lựa chọn sản phẩm mình cần mua và khách hàng đặt mua thì sản phẩm đó sẽ được đưa vào giỏ hàng. Trước khi thêm vào giỏ hàng khách hàng phải đăng nhập nếu chưa đăng nhập hoặc đăng ký tài khoản nếu chưa có tài khoản.

Sau khi khách hàng chọn được sản phẩm và số lượng cần mua, khách hàng có thể vào lại giỏ hàng để xem lại các mặt hàng đã đặt mua, nếu có mặt hàng nào không vừa ý khách hàng không muốn mua thì khách hàng có thể thực hiện thao tác xóa sản phẩm đó ra khỏi giỏ hàng.

Sau khi khách hàng đã lựa chọn xong, khách hàng sẽ tiến hành đặt hàng và thanh toán qua Paypal. Trong quá trình thực hiện các bước gửi yêu cầu nếu khách hàng không muốn mua hàng nữa thì khách hàng có thể hủy đơn hàng đó nếu chủ cửa hàng chưa xác nhận đơn hàng. Khi khách hàng gửi yêu cầu đặt hàng hệ thống sẽ kiểm tra thông tin khách hàng như: địa chỉ, số điện thoại, ... Nếu chưa có thì yêu cầu khách hàng cập nhật lại. Sau khi thông tin được gửi đi, người quản lý sẽ tiếp nhận đơn hàng kiểm tra thông tin khách hàng có hợp lệ không nếu không hợp lệ thì người quản trị liên hệ lại khách hàng hoặc hủy đơn hàng đó. Ngược lại, thông tin đơn hàng sẽ được cập nhật trong trạng thái đơn hàng của khách hàng. Nếu khách hàng có yêu cầu hủy đơn hàng thì đơn hàng đó sẽ được hoàn tiền lại và cập nhật trạng thái đơn hàng thao tác hoàn tiền do chủ cửa hàng thực hiện.

Khi khách hàng đã thanh toán sản phẩm trong cửa hàng thì có thể bình luận đánh giá sản phẩm đó.

Lưu ý: Việc xác nhận thông tin khách hàng dựa vào ban quản trị website của cửa hàng. Việc này không thông qua website.

1.2. Các nhóm chức năng của hệ thống

Chức năng của hệ thống được chia thành các nhóm chức năng chính như sau:

Nhóm chức năng đăng ký, đăng nhập thành viên.

Nhóm chức năng xem thông tin, bao gồm xem thông tin giỏ hàng, xem thông tin đơn hàng, xem thông tin sản phẩm, xem thông tin cá nhân.

Nhóm chức năng quản lý thông tin, bao gồm quản lý thông tin cá nhân, quản lý danh sách thành viên, quản lý danh mục sản phẩm.

Nhóm chức năng mua hàng, tiếp nhận và xử lý đơn hàng.

1.3. Các tác nhân chính của hệ thống gồm:

KHÁCH HÀNG: là người giao dịch với hệ thống thông qua các đơn đặt hàng, khách hàng có thể xem, lựa chọn các sản phẩm, chọn địa điểm giao hàng. Khách hàng có thể đăng ký thành viên của hệ thống, thay đổi thông tin cá nhân và mật khẩu.

NGƯỜI QUẢN LÝ: là người điều hành, quản lý và theo dõi mọi hoạt động của hệ thống.

1.4. Xác định use case.

1.4.1. Xác định các use case

- Tác nhân người dùng khách có các use case sau:

- Đăng ký
- Xem sản phẩm
- Xem chi tiết sản phẩm

- Tác nhân Khách hàng có các use case sau:

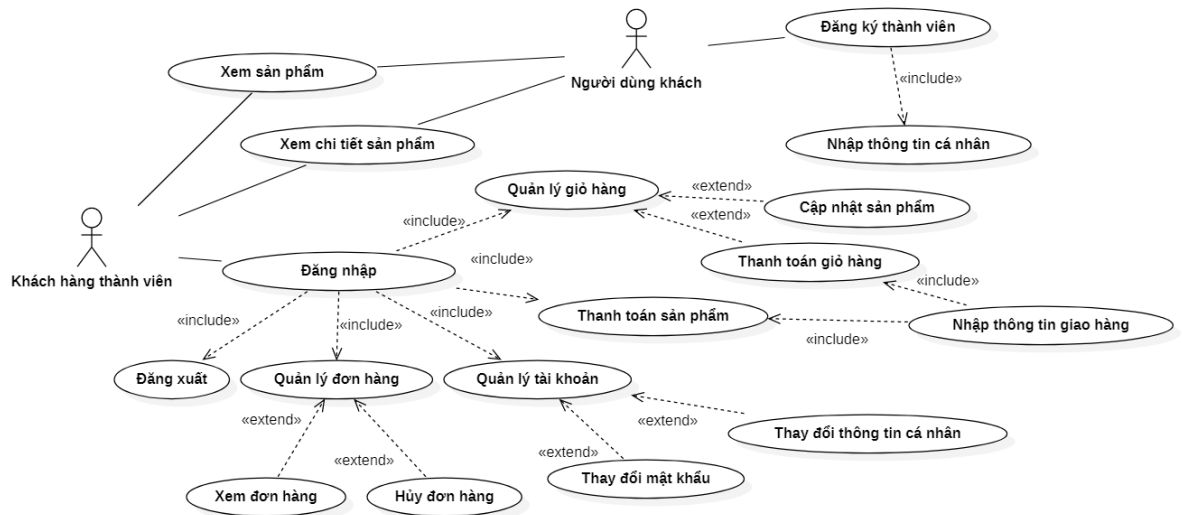
- Đăng nhập
- Xem sản phẩm
- Xem chi tiết sản phẩm
- Quản lý giỏ hàng
- Quản lý tài khoản
- Quản lý đơn hàng
- Đăng xuất

- Tác nhân Người quản lý có các use case sau:

- Đăng nhập
- Quản lý khách hàng
- Quản lý sản phẩm
- Quản lý đơn hàng
- Quản lý danh mục
- Thống kê
- Đăng xuất

1.4.2 Sơ đồ use case chi tiết

- Sơ đồ use case của tác nhân Khách hàng khách và Khách hàng thành viên



Hình 1. Sơ đồ use case của tác nhân Khách hàng khách và Khách hàng thành viên

- Sơ đồ use case của tác nhân Người quản lý



Hình 2. Sơ đồ use case của tác nhân Người quản lý

1.5. Mô tả chi tiết các use case

1.5.1. Đăng ký

Tên Use case	Đăng ký
Mô tả	Cung cấp chức năng cho phép đăng ký khách hàng thành viên
Tác nhân chính	Người dùng khách
Tiền điều kiện	Không cần đăng nhập
Cách kích hoạt	Bấm nút “Đăng ký” trên giao diện website
Luồng xử lý	1. Nhập thông tin đăng ký 2. Bấm nút “Đăng ký” 3. Hệ thống chuyển về trang đăng nhập cho khách hàng đăng nhập
Dòng sự kiện lỗi	1. Hệ thống sẽ yêu cầu khách hàng nhập lại

Bảng 2. Chi tiết use case đăng ký

1.5.2. Đăng nhập

Tên Use case	Đăng nhập
Mô tả	Cung cấp chức năng cho phép khách hàng đăng nhập hệ thống
Tác nhân chính	Người quản lý, khách hàng
Tiền điều kiện	Khách hàng đã đăng ký vào hệ thống
Cách kích hoạt	Bấm nút “Đăng nhập” trên giao diện website
Luồng xử lý	1. Khách hàng nhập thông tin tài khoản vào các trường tương ứng 2. Bấm nút “Đăng nhập” 3. Hệ thống xác thực việc đăng nhập
Dòng sự kiện lỗi	3. Hệ thống thông báo lỗi, yêu cầu khách hàng đăng nhập lại

Bảng 3. Chi tiết use case đăng nhập

1.5.3. Xem sản phẩm

Tên Use case	Xem sản phẩm
Mô tả	Cung cấp chức năng cho phép khách hàng xem các sản phẩm của cửa hàng
Tác nhân chính	Tất cả các tác nhân
Tiền điều kiện	
Cách kích hoạt	Truy cập website

Luồng xử lý	1. Khách hàng truy cập vào website xem sản phẩm
Dòng sự kiện lỗi	

Bảng 4. Chi tiết use case xem sản phẩm

1.5.4. Cập nhật giỏ hàng

Tên Use case	Cập nhật giỏ hàng
Mô tả	Cung cấp chức năng cho phép khách hàng thêm, xóa các sản phẩm trong giỏ hàng của mình
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn biểu tượng giỏ hàng để thêm sản phẩm vào giỏ hàng. Vào giỏ hàng nhấn nút xóa của sản phẩm trong giỏ hàng để xóa sản phẩm ra khỏi giỏ hàng.
Luồng xử lý	1. Khách hàng truy cập vào website 2. Nhấn biểu tượng giỏ hàng để thêm sản phẩm vào giỏ hàng hoặc vào giỏ hàng nhấn nút “xóa” của sản phẩm trong giỏ hàng để xóa sản phẩm ra khỏi giỏ hàng. 3. Hệ thống cập nhật thông tin giỏ hàng
Dòng sự kiện lỗi	

Bảng 5. Chi tiết use case cập nhật giỏ hàng

1.5.5. Thanh toán

Tên Use case	Thanh toán
Mô tả	Cung cấp chức năng cho phép khách hàng thanh toán các sản phẩm trong giỏ hàng của mình qua cổng thanh toán Paypal
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn nút “Thanh toán” để thanh toán.
Luồng xử lý	1. Khách hàng truy cập vào giỏ hàng hoặc chi tiết sản phẩm. 2. Nhấn nút “Thanh toán”. 3. Nhập thông tin giao hàng. 4. Tiến hành thanh toán 5. Hệ thống hiển thị thông báo “Chờ xác nhận đơn hàng”.

Dòng sự kiện lỗi	4. Nếu giao dịch bị lỗi hoặc khách hàng hủy giao dịch thì sẽ trở về trang trước đó.
------------------	---

Bảng 6. Chi tiết use case thanh toán

1.5.6. Thay đổi thông tin tài khoản

Tên Use case	Thay đổi thông tin
Mô tả	Cung cấp chức năng cho phép khách hàng thay đổi thông tin tài khoản của mình
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn nút “Cập nhật thông tin” để thay đổi thông tin tài khoản hoặc “Cập nhật ảnh đại diện” để thay đổi ảnh đại diện.
Luồng xử lý	<ol style="list-style-type: none"> 1. Khách hàng truy cập vào thông tin cá nhân. 2. Khách hàng nhấn chọn “Cập nhật thông tin” hoặc cập nhật ảnh đại diện. 3. Nhập thông tin cần sửa. 4. Hệ thống xác nhận yêu cầu của khách hàng. 5. Hệ thống cập nhật thông tin khách hàng.
Dòng sự kiện lỗi	Nếu khách hàng hủy yêu cầu sẽ trở về trang thông tin cá nhân khách hàng ngược lại bước 4.

Bảng 7. Chi tiết use case thay đổi thông tin tài khoản

1.5.7. Thay đổi mật khẩu

Tên Use case	Thay đổi mật khẩu
Mô tả	Cung cấp chức năng cho phép khách hàng thay đổi mật khẩu tài khoản của mình
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn nút “Đổi mật khẩu” để thay đổi mật khẩu
Luồng xử lý	<ol style="list-style-type: none"> 1. Khách hàng truy cập vào thông tin cá nhân. 2. Khách hàng nhấn chọn “Đổi mật khẩu”. 3. Nhập mật khẩu cũ, nhập và xác nhận mật khẩu mới. 4. Hệ thống xác nhận yêu cầu của khách hàng.

	5. Hệ thống cập nhật mật khẩu khách hàng.
Dòng sự kiện lỗi	3. Khách hàng nhập sai mật khẩu cũ hệ thống sẽ hiển thị thông báo. 4. Nếu khách hàng hủy yêu cầu sẽ trở về trang thông tin cá nhân khách hàng ngược lại bước 5.

Bảng 8. Chi tiết use case thay đổi mật khẩu

1.5.8. Xem đơn hàng

Tên Use case	Xem đơn hàng
Mô tả	Cung cấp chức năng cho phép khách hàng xem thông tin các đơn hàng của mình
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn nút “Đơn hàng” để xem các đơn hàng
Luồng xử lý	1. Khách hàng truy cập “Tài khoản”. 2. Khách hàng nhấn chọn “Đơn hàng”. 3. Hệ thống hiển thị các đơn hàng của khách hàng.
Dòng sự kiện lỗi	

Bảng 9. Chi tiết use case xem đơn hàng

1.5.9. Hủy đơn hàng

Tên Use case	Hủy đơn hàng
Mô tả	Cung cấp chức năng cho phép khách hàng hủy các đơn hàng khi chủ cửa hàng chưa xác nhận đơn hàng.
Tác nhân chính	Khách hàng
Tiền điều kiện	Khách hàng đã đăng nhập
Cách kích hoạt	Nhấn nút “Đơn hàng” để xem các đơn hàng
Luồng xử lý	1. Khách hàng truy cập “Tài khoản”. 2. Khách hàng nhấn chọn “Đơn hàng”. 3. Chọn đơn hàng cần hủy (nếu chủ cửa hàng chưa xác nhận). 4. Nhấn nút “Hủy” để hủy đơn hàng. 5. Hệ thống xác nhận yêu cầu khách hàng. 6. Hệ thống cập nhật trạng thái đơn hàng của khách hàng.

Dòng sự kiện lỗi	5. Nếu khách hàng hủy yêu cầu hệ thống sẽ đưa khách hàng trở về “Đơn hàng” ngược lại bước 6.
------------------	--

Bảng 10. Chi tiết use case hủy đơn hàng

1.5.10. Đăng xuất

Tên Use case	Đăng xuất
Mô tả	Cung cấp chức năng cho phép người dùng đăng xuất khỏi tài khoản
Tác nhân chính	Khách hàng thành viên, người quản lý
Tiền điều kiện	Khách hàng đã đăng nhập vào hệ thống
Cách kích hoạt	Bấm nút “Đăng xuất” trên giao diện website
Luồng xử lý	<ol style="list-style-type: none"> 1. Khách hàng truy cập tài khoản 2. Bấm nút “Đăng xuất” 3. Hệ thống xác thực việc đăng xuất 4. Hệ thống chuyển về trang chủ
Dòng sự kiện lỗi	

Bảng 11. Chi tiết use case đăng xuất

1.5.11. Thêm sản phẩm

Tên Use case	Thêm sản phẩm
Mô tả	Cung cấp chức năng cho phép người quản lý thêm sản phẩm mới
Tác nhân chính	Người quản lý
Tiền điều kiện	Người quản lý đã đăng nhập vào hệ thống
Cách kích hoạt	Bấm nút “Thêm sản phẩm”
Luồng xử lý	<ol style="list-style-type: none"> 1. Người quản lý truy cập tab “Sản phẩm” 2. Bấm nút “Thêm sản phẩm” 3. Nhập thông tin sản phẩm. 4. Hệ thống xác nhận yêu cầu. 5. Hệ thống cập nhật sản phẩm.
Dòng sự kiện lỗi	4. Nếu người dùng hủy yêu cầu thì hệ thống sẽ trở tab trước đó ngược lại bước 5.

Bảng 12. Chi tiết use case thêm sản phẩm

1.5.12. Sửa thông tin sản phẩm

Tên Use case	Sửa thông tin sản phẩm
Mô tả	Cung cấp chức năng cho phép người quản lý sửa thông tin sản phẩm
Tác nhân chính	Người quản lý
Tiền điều kiện	Người quản lý đã đăng nhập vào hệ thống
Cách kích hoạt	Bấm nút “Sửa”
Luồng xử lý	<ol style="list-style-type: none"> 1. Người quản lý truy cập tab “Sản phẩm” 2. Chọn sản phẩm cần sửa thông tin 3. Nhập thông tin cần thay đổi. 4. Hệ thống xác nhận yêu cầu. 5. Hệ thống cập nhật sản phẩm.
Dòng sự kiện lỗi	4. Nếu người dùng hủy yêu cầu thì hệ thống sẽ trở tab trước đó ngược lại bước 5.

Bảng 13. Chi tiết use case sửa thông tin sản phẩm

1.5.13. Xóa sản phẩm

Tên Use case	Xóa sản phẩm
Mô tả	Cung cấp chức năng cho phép người quản lý xóa sản phẩm
Tác nhân chính	Người quản lý
Tiền điều kiện	Người quản lý đã đăng nhập vào hệ thống
Cách kích hoạt	Bấm nút “Xóa”
Luồng xử lý	<ol style="list-style-type: none"> 1. Người quản lý truy cập tab “Sản phẩm” 2. Chọn sản phẩm cần xóa 3. Hệ thống xác nhận yêu cầu. 4. Hệ thống cập nhật sản phẩm.
Dòng sự kiện lỗi	3. Nếu người dùng hủy yêu cầu thì hệ thống sẽ trở tab trước đó ngược lại bước 4.

Bảng 14. Chi tiết use case xóa sản phẩm

1.5.14. Các chức năng khác tương tự như trên.

Chương 2: Thiết kế và cài đặt giải pháp

2.1. Cơ sở lý thuyết

2.1.1. Giới thiệu về API và RESTful API

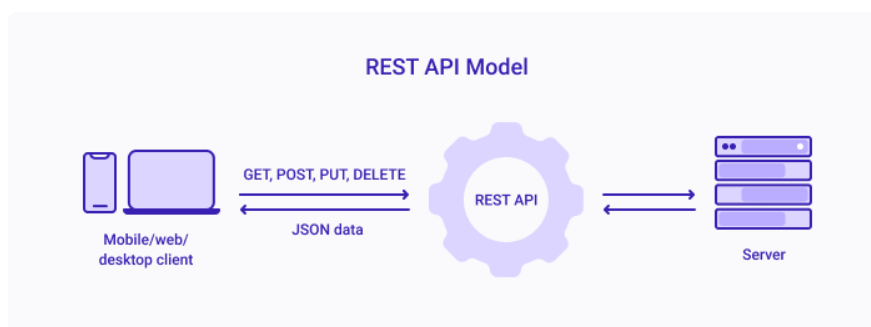
API (Application programming interface) là những quy tắc tương tác để các ứng dụng hoặc thành phần của ứng dụng có thể giao tiếp được với nhau. Mục đích của API là để đơn giản hóa việc lập trình bằng cách trừu tượng hóa đi những cài đặt phức tạp và chỉ phơi bày ra các cách thức để thực hiện hành động mà lập trình viên cần. Trong ngữ cảnh lập trình web, API là cách thức để ứng dụng trao đổi dữ liệu với một dịch vụ trực tuyến. API sẽ cung cấp các chức năng dịch vụ thông qua đường dẫn URL (Uniform Resource Locator) với định nghĩa thông số kỹ thuật như gửi request bằng HTTP (Hypertext Transfer Protocol) và trả về response message theo định dạng XML (Extensible Markup Language) hoặc JSON (JavaScript Object Notation).

REST (Representation State Transfer) là một kiến trúc phần mềm bao gồm các quy tắc để tạo ra dịch vụ web (webservice). Một webservice tuân thủ theo kiến trúc REST thì được gọi là RESTful webservice. Webservice này sử dụng giao thức HTTP để triển khai kiến trúc web. Như vậy, RESTful API chính là kiến trúc thiết kế API tuân thủ theo kiến trúc REST thông qua các phương thức của HTTP (GET, POST, PUT, DELETE,...). Tương ứng với mỗi phương thức HTTP sẽ thực hiện những tác vụ tương ứng:

Phương thức HTTP	Tác vụ
GET	Lấy dữ liệu
POST	Tạo mới dữ liệu
PUT	Cập nhật dữ liệu
DELETE	Xóa dữ liệu

Bảng 15. Các tác vụ cơ bản của REST dựa trên phương thức HTTP

Các tác vụ đọc, tạo, cập nhật, xóa được gọi là CRUD service (Create, Read, Update, Delete). Mỗi tác vụ trên phải được gọi thông qua địa chỉ URI (Uniform Resource Identifier) kèm theo phương thức và payload (có thể có hoặc không, thường là định dạng XML hoặc JSON).



Hình 3. Mô hình hoạt động cơ bản của REST

RESTful API sử dụng giao thức stateless (là một giao thức truyền thông không sử dụng session) và theo tiêu chuẩn nên hệ thống sẽ nhanh, đáng tin cậy và có thể mở rộng dễ dàng. Thông thường, RESTful API sẽ xác thực người dùng khi gửi yêu cầu đối với những tác vụ nguy hiểm như cập nhật hoặc xóa dữ liệu hoặc chỉ cho phép đối với người quản trị.

2.1.2. ReactJS



Hình 4. logo thư viện reactjs

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

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

2.1.2.1. JavaScript XML

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

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

2.1.2.2. Components

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

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

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

Hình 5. function components hợp lệ

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

Hoặc theo ES6 arrow function:

```
const Welcome = () =>{
  return ( <h1>Tôi là một functional component!</h1> );
};
```

Hình 6. function components sử dụng arrow function

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

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

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

Class components còn có:

- Phương thức khởi tạo, có hàm về vòng đời component, hàm render().
- State (dữ liệu ứng dụng).

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Hình 7. Ví dụ class components

- Tóm lại, một class components là:

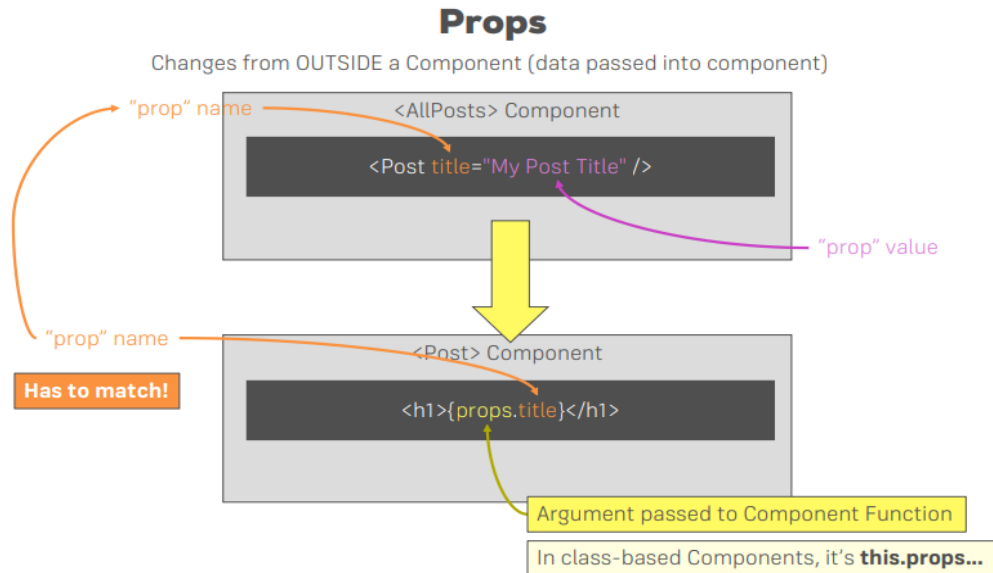
Là một class kế thừa từ React.Component.

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

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

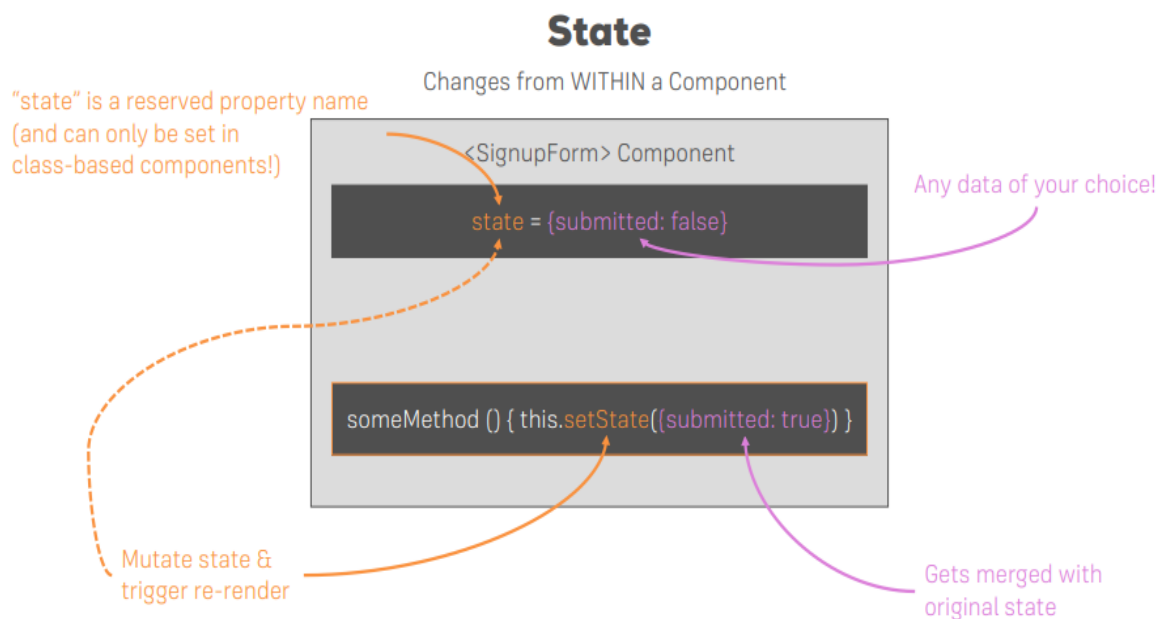
2.1.2.3. Props và State

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

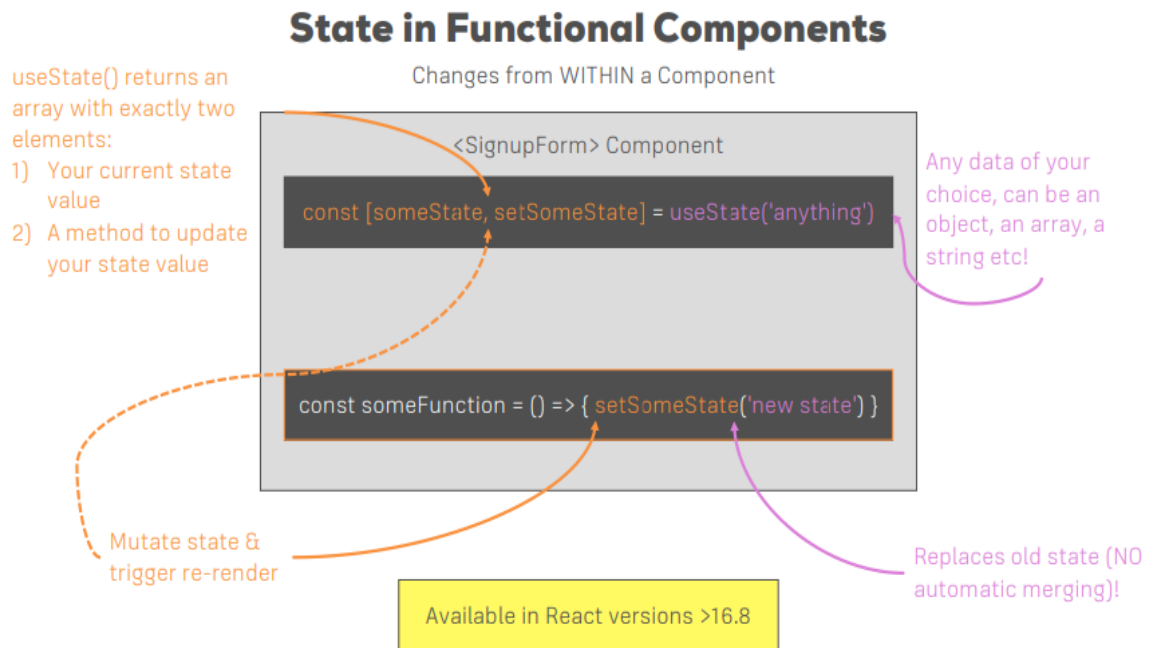


Hình 8. Props

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



Hình 9. State trong class components



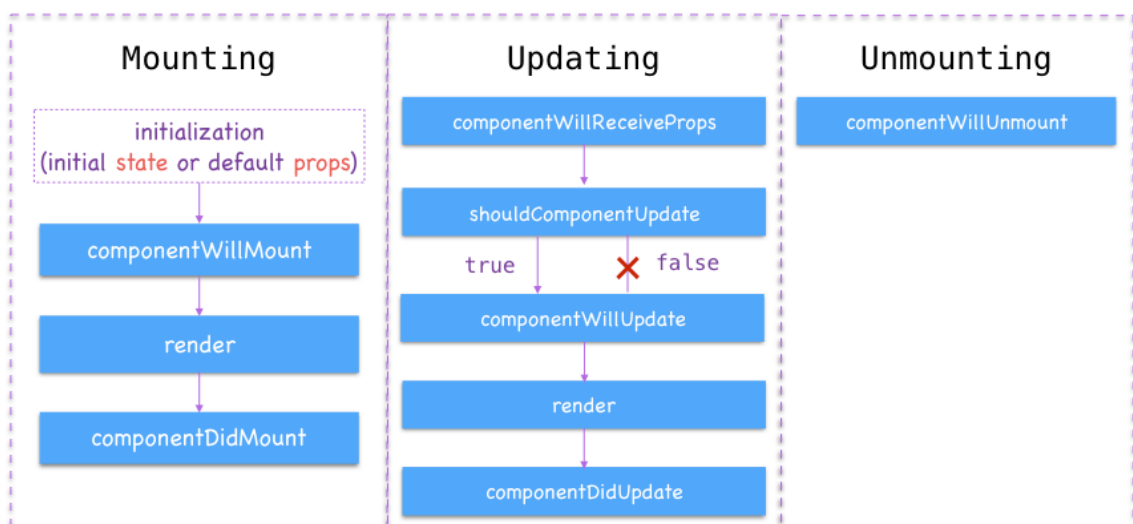
Hình 10. State trong function components

2.1.2.4. LifeCycle

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

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

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



Hình 11. LifeCycle trong ReactJS

2.1.2.5. Hook

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

2.1.3. Thư viện Redux



Hình 12. logo thư viện redux

Redux là một thư viện JavaScript mã nguồn mở (MIT License) dùng để quản lý state của ứng dụng, được sử dụng phổ biến với React hoặc Angular để xây dựng giao diện người dùng. Redux được tạo bởi Dan Abramov và Andrew Clark vào năm 2015.

State trong redux sẽ có các tính chất sau:

- **Từ một nguồn duy nhất:** state của ứng dụng được là một state toàn cục được lưu trữ thành một object tree.
- **State chỉ có thể đọc:** state không thể thay đổi trực tiếp mà chỉ có thể phát động bằng một action (object) mô tả những gì đã xảy ra.
- **State được thay đổi bởi những reducer (JS function):** khi muốn thay đổi state sẽ phát động ra một action, reducer sẽ tiếp nhận và xử lý thay đổi state tương ứng. Vì reducer chỉ là những hàm js bình thường, có thể truyền vào dữ liệu, gọi theo thứ tự hoặc tái sử dụng các reducer.

Redux được thiết kế cho việc quản lý tập trung, chia sẻ state giữa các component trở nên đơn giản, nhất quán. Để sử dụng redux với react cần phải nắm một số khái niệm cơ bản sau

- **Action:** là những thông tin payload được dùng để gửi dữ liệu đến store. Hiểu đơn giản, action chính là một event được gửi lên store kèm theo dữ liệu và loại action sẽ được thực hiện.
- **Reducer:** là những hàm thuần túy thực hiện việc nhận action được gửi lên và lấy state hiện tại của ứng dụng, xử lý và trả về một state mới. Global state chính là một object tree lưu trữ toàn bộ state của ứng dụng, action gửi lên sẽ mô tả những gì xảy ra. Khi đó, reducer sẽ tiếp nhận action và lấy một nhánh state nhỏ tương ứng từ global state, sau đó tạo ra một nhánh state mới dựa trên action và cập nhật vào global state.

– **Store:** là một object kết nối action và reducer lại với nhau, chịu trách nhiệm lưu giữ state của ứng dụng, cho phép truy xuất và cập nhật state thông qua dispatch (là một dịch vụ của store cho phép gửi action thông qua nó).

2.1.4. Thư viện Axios

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

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

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

2.1.5. Thư viện React UI – Material UI

Material UI là một thư viện các React Component đã được tích hợp thêm cả Google's Material Design. Các component trong Material UI là những React component, hỗ trợ hầu hết các thành phần cốt yếu để tạo dựng nên một trang web hoàn chỉnh. Khi sử dụng Material UI thì hầu như không cần phải cài thêm một thư viện UI bổ sung nào nữa thì cũng đủ để đáp ứng nhu cầu của dự án đưa ra.

Dưới đây là danh sách các component mà Material UI hỗ trợ:

Layout: Box, Container, Grid, Grid List, Hidden.

Input: Button, Button Group, Checkbox, Floating Action Button, Date/Time, Radio, Select, Slider, Switch, TextField, Transfer List.

Navigation: Bottom Navigation, Breadcrumbs, Drawer, Link, Menu, Stepper, Tabs.

Surfaces: App bar, Paper, Card, Accordion.

Feedback: Progress, Dialog, Snackbar, Backdrop.

Data Display: Avatar, Badge, Chip, Divider, Icons, Material Icons, List, Table, Tooltip, Typography.

Utils: Click Away Listener, CSS Baseline, Modal, No SSR, Popover, Popper, Portal, Textarea Autosize, Transition, useMediaQuery

Lab: Alert, Autocomplete, Data Grid, Pagination, Rating, Skeleton, Speed Dial, Timeline, Toggle Button, Tree View.

2.1.5. MySQL



Hình 13. logo của MySQL

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở miễn phí, được những doanh nghiệp lớn nhất thế giới như Facebook, Google, Adobe sử dụng để tiết kiệm thời gian và chi phí cho việc quản lý dữ liệu hiệu quả. MySQL được sáng lập bởi Michael "Monty" Widenius và David Axmark năm 1995. Hiện nay được phát triển bởi tập đoàn Oracle.

MySQL hỗ trợ trên nhiều hệ điều hành: Windows, Linux, MacOS,...

MySQL cũng được công nhận bởi DB-Engines là hệ quản trị cơ sở dữ liệu của năm 2019.

2.1.6. NodeJS



Hình 14. logo thư viện nodejs

NodeJS là một nền tảng được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript, giúp xây dựng các ứng dụng web một cách đơn giản và dễ dàng mở rộng.

NodeJS được phát triển bởi Ryan Dahl vào năm 2009 và có thể chạy trên nhiều hệ điều hành khác nhau: OS X, Microsoft Windows, Linux.

2.1.6.1. Blocking I/O và Nonblocking I/O

I/O là quá trình giao tiếp (lấy dữ liệu vào, trả dữ liệu ra) giữa một hệ thống thông tin và môi trường bên ngoài. Với CPU, thậm chí mọi giao tiếp dữ liệu với bên ngoài cấu trúc chip như việc nhập/ xuất dữ liệu với memory (RAM) cũng là tác vụ I/O.

Blocking I/O : Yêu cầu thực thi một IO operation, sau khi hoàn thành thì trả kết quả lại. Process/Thread gọi bị block cho đến khi có kết quả trả về hoặc xảy ra ngoại lệ.

Nonblocking I/O: Yêu cầu thực thi IO operation và trả về ngay lập tức (timeout = 0). Nếu operation chưa sẵn sàng để thực hiện thì thử lại sau. Tương đương với kiểm tra IO operation có sẵn sàng ngay hay không, nếu có thì thực hiện và trả về, nếu không thì thông báo thử lại sau.

2.1.6.2. Synchronious và Asynchronious

Synchronous: Các sự kiện diễn ra theo thứ tự. Một sự kiện chỉ được bắt đầu khi sự kiện trước kết thúc.

Asynchronous: Không theo thứ tự, các hành động có thể xảy ra đồng thời hoặc chỉ ít, mặc dù các hành động bắt đầu theo thứ tự nhưng kết thúc thì không. Một hành động có thể bắt đầu (và thậm chí kết thúc) trước khi hành động trước đó hoàn thành.

Libuv – một thư viện multi-platform hỗ trợ asynchronous I/O giúp Nodejs mang lại cơ chế Asynchronous Event-Driven Non-Blocking I/O

2.1.6.3. Callback

Callback là một hàm sẽ được thực hiện sau khi một hàm khác đã thực hiện xong.

Trong JavaScript, hàm là đối tượng. Do đó, các hàm có thể lấy các hàm làm đối số và có thể được trả về bởi các hàm khác. Các hàm thực hiện điều này được gọi là **higher - order function** (Hàm bậc cao hơn). Bất kỳ hàm nào được truyền dưới dạng đối số được gọi là **hàm callback**.

Javascript là một ngôn ngữ lập trình hướng sự kiện và bất đồng bộ nên callback function đóng vai trò rất quan trọng.

ES6 định nghĩa 3 state cho một lời gọi hàm không đồng bộ:

- **Pending:** hàm đang được thực hiện và chưa trả về kết quả. Trong lúc này, nếu cố tình console.log biến kết quả sẽ nhận được output < pending >
- **Fulfilled:** hàm đã thực hiện xong – thành công và trả về kết quả.
- **Rejected:** hàm đã thực hiện xong – không thành công. Thường thì sẽ bắt exception tại bước này.

Máy chủ NodeJS có thể nhận rất nhiều các yêu cầu (request) từ rất nhiều người dùng. Vì vậy để nâng cao khả năng phục vụ, tất cả các API của NodeJS được thiết kế hỗ trợ Callback.

2.6.1.4. Event Loop

NodeJS là một ứng dụng đơn luồng (Single Thread), nó hoạt động phía trên một nền tảng được viết bởi C++, nền tảng này sử dụng đa luồng (Multi-Thread) để thực hiện đồng thời các nhiệm vụ.

Mỗi yêu cầu (request) từ phía người dùng được NodeJS coi là một sự kiện (event), chúng được đặt vào một Event Queue (Hàng đợi sự kiện). NodeJS sử dụng quy tắc FIFO (First In First Out), điều này có nghĩa là những yêu cầu đến trước sẽ được xử lý trước.

Event Loop là một vòng lặp vô tận, nó sẽ chuyển các yêu cầu sang Thread Pool (Bể chứa các luồng), đồng thời mỗi yêu cầu sẽ được đăng ký một hàm Callback. Khi một yêu cầu được xử lý xong, hàm Callback tương ứng sẽ được gọi thực thi.

Thread Pool là một chương trình viết bằng ngôn ngữ C++, nó hỗ trợ đa luồng (Multi Threads), chính vì vậy tại đây các yêu cầu sẽ được xử lý trên các luồng khác nhau. NodeJS cũng hỗ trợ đa tiến trình (Multi Processes), điều này có nghĩa là chúng có thể được thực thi trên các lõi (Core) khác nhau.

Khi một yêu cầu được xử lý xong, NodeJS sẽ gọi hàm Callback (Đã được đăng ký cho yêu cầu này) để thực thi nó.

Nếu mỗi kết nối tới Server đều mở ra một luồng (Thread) sẽ rất tốn bộ nhớ. Điều này đã được chứng thực khi so sánh Apache và Nginx (Hai Web Server triển khai các ứng dụng PHP). Apache đã tiêu tốn bộ nhớ hơn rất nhiều so với Nginx.

NodeJS giống với Nginx là chúng chỉ sử dụng một luồng đơn (Single thread) để đón tiếp các kết nối từ phía người dùng, và coi mỗi yêu cầu của người dùng là một sự kiện.

Các hoạt động I/O rất tốn tài nguyên của hệ thống, vì vậy NodeJS quản lý chặt chẽ việc sử dụng các hoạt động I/O. Vì vậy chỉ cần sử dụng Callback khi bạn thực thi các nhiệm vụ liên quan tới I/O.

2.6.1.5. Module

Node.js sử dụng kiến trúc Module để đơn giản hóa việc tạo ra các ứng dụng phức tạp. Module là giống như các thư viện trong C, C#, Java, ... Mỗi module chứa một tập các hàm chức năng có liên quan đến một "đối tượng" của Module.

Các module được giữ tách biệt riêng với nhau, tách riêng với code base, khi nào cần sử dụng những cái nào thì gọi chúng ra và kết hợp lại với nhau tùy logic xử lý của bạn.

2.1.6.6. NPM - Node Package Manager

NPM viết tắt của Node Package Manager là một công cụ tạo và quản lý các thư viện lập trình Javascript cho NodeJS. Trong cộng đồng Javascript, các lập trình viên chia sẻ hàng trăm nghìn các thư viện với các đoạn code đã thực hiện sẵn một chức năng nào đó. Nó giúp cho các dự án mới tránh phải viết lại các thành phần cơ bản, các thư viện lập trình hay thậm chí cả các framework.

Các loại package

Dựa theo chức năng mà ta chia package ra làm 2 loại, đó là **Simple dependencies** và **Development dependencies**.

Simple dependencies là những package bắt buộc phải có trong quá trình chạy sản phẩm. Khi cài đặt Simple dependencies, Npm sẽ tự động cài đặt tất cả các dependencies cần thiết.

Development dependencies là những package bắt buộc khi phát triển cũng như phát hành sản phẩm. Khi cài đặt Development dependencies, Npm sẽ chỉ cài đặt các dependencies cần thiết.

2.1.7. ExpressJS Framwork



Hình 15. logo expressjs framework

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

Định tuyến (Routing)

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

Cấu trúc định tuyến cơ bản:

`app.METHOD(Path, Handler...)`

Trong đó:

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

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

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

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

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

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

```
app.get('/api/color',[authJwt.verifyToken,authJwt.isAdmin ],color.findAll)
app.post('/api/color',[authJwt.verifyToken,authJwt.isAdmin ],color.create)
app.delete('/api/color/:colorId',[authJwt.verifyToken,authJwt.isAdmin ],color.delete)
app.put('/api/color/:colorId',[authJwt.verifyToken,authJwt.isAdmin ],color.update)
```

Hình 16. Ví dụ cấu trúc route

Route parameters

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

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

Response methods

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

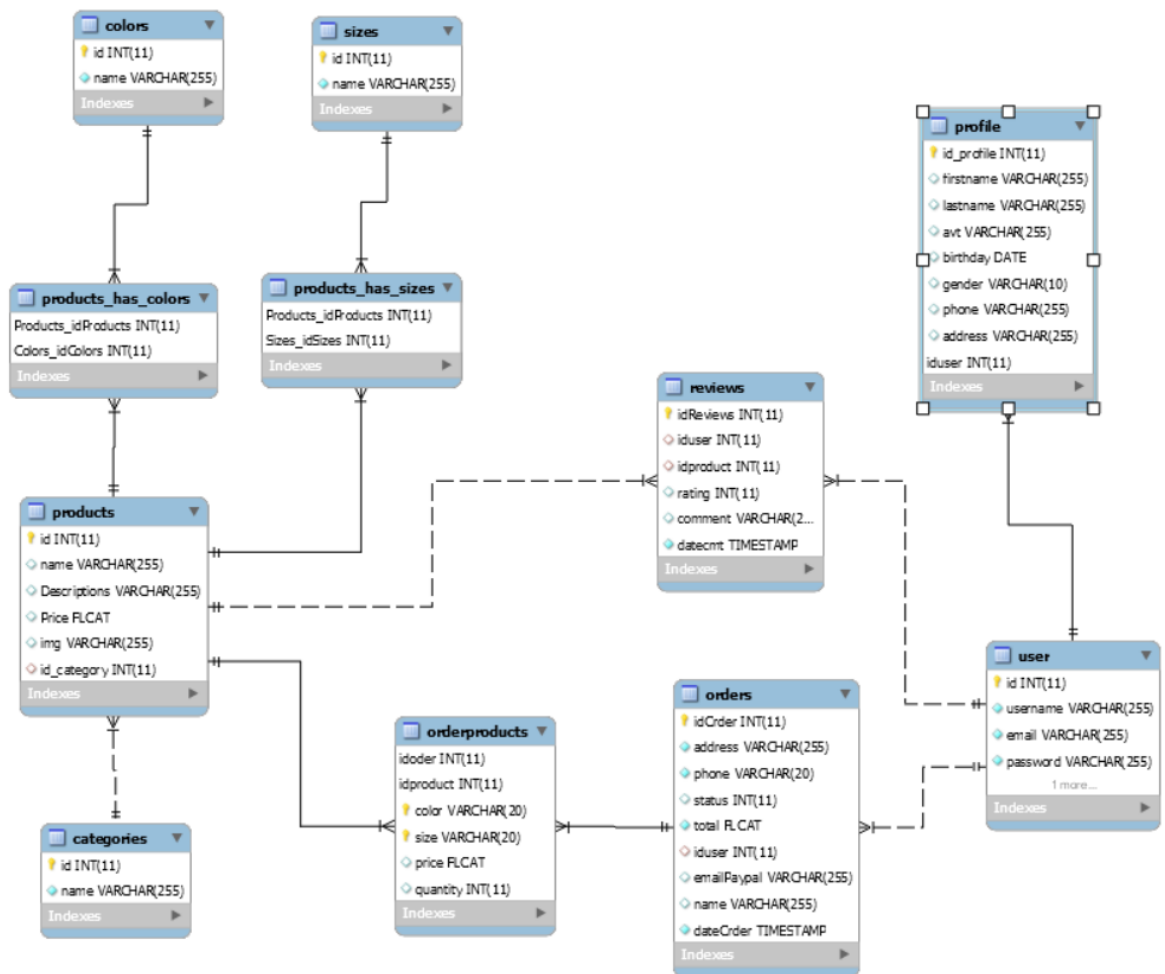
Tên phương thức	Ý nghĩa
<code>res.json()</code>	Trả về một dữ liệu dạng JSON
<code>res.redirect()</code>	Chuyển hướng đến một đường dẫn nào đó
<code>res.render()</code>	Trả về một view template
<code>res.send()</code>	Gửi dữ liệu dạng text

Bảng 16. một số phương thức (response methods) hỗ trợ hay dùng nhất

2.2. Thiết kế hệ thống

2.2.1. RESTful API

Thiết kế cơ sở dữ liệu



Hình 17. Mô hình cơ sở dữ liệu

Danh sách các bảng

STT	Tên bảng	Diễn giải
1	user	Tài khoản người dùng
2	profiles	Thông tin người dùng
3	orders	Đơn hàng
4	orderproducts	Chi tiết đơn hàng
5	products	Sản phẩm
6	sizes	Size sản phẩm
7	products_has_sizes	Chi tiết size sản phẩm
8	colors	Màu sản phẩm

9	products_has_colors	Chi tiết màu sản phẩm
10	categories	Loại sản phẩm
11	reviews	Bình luận, đánh giá sản phẩm

Bảng 17. Danh sách các bảng trong cơ sở dữ liệu

Mô hình bảng dữ liệu

Bảng Profiles

Profiles					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idProfiles	INT	x		ID thông tin người dùng
2	First_name	varchar(255)			Tên người dùng
3	Last_name	varchar(255)			Họ người dùng
4	Avatar	varchar(255)			Ảnh đại diện
5	Birthday	Date			Ngày sinh
6	Gender	INT			Giới tính
7	Number_phone	INT			Số điện thoại
8	Address	varchar(255)			Địa chỉ
9	iduser	INT			ID tài khoản người dùng

Bảng 18. Profiles

Bảng user

user					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	id	INT	x		ID của người dùng
2	email	varchar(255)			Email của người dùng
3	password	varchar(255)			Mật khẩu người dùng

4	username	varchar(255)			Tên đăng nhập người dùng
---	----------	--------------	--	--	--------------------------

Bảng 19. user

Bảng Orders

Orders					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idOrders	INT	x		ID của đơn hàng
2	Status	INT			Trạng thái đơn hàng
3	idUser	INT	x	x	ID của người dùng
4	Address	varchar(255)			Địa chỉ giao hàng
5	Phone	varchar(20)			Số điện thoại khách hàng để giao hàng
6	Total	Float			Tổng tiền đơn hàng
7	Name	varchar(255)			Tên khách hàng để giao hàng
8	emailPaypal	varchar(255)			Tài khoản email thanh toán paypal
9	dateOrder	Timetamp			Ngày thanh toán

Bảng 20. Orders

Bảng Categories

Categories					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idCategories	INT	x		ID của loại sản phẩm
2	name	varchar(255)			Tên loại sản phẩm

Bảng 21. Categories

Bảng Products

Products					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idProducts	INT	X		ID sản phẩm
2	Description	varchar(255)			Mô tả
3	Price	Float			Giá
4	Gender	INT			Sản phẩm dành cho giới tính
5	Categories_idCategories	INT	x	x	ID loại sản phẩm
6	Img	varchar(255)			Hình ảnh sản phẩm

Bảng 22. Products

Bảng orderproducts

orderproducts					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idOder	INT	X	X	ID đơn hàng
2	Quantity	int			Số lượng
3	dProduct	INT	X	X	ID sản phẩm
4	color	varchar(20)	X	X	Màu sản phẩm
5	price	float			Giá sản phẩm
6	size	varchar(20)			Size sản phẩm

Bảng 23. orderproducts

Bảng Sizes

Sizes					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idSizes	INT	x		ID size
2	name	varchar(255)			Tên size

Bảng 24. Sizes

Bảng Colors

Colors					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idColors	INT	x		ID màu sản phẩm
2	name	varchar(255)			Tên màu sản phẩm

Bảng 25. Colors

Bảng Products_has_Sizes

Products_has_Sizes					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	Products_idProducts	INT	x		ID sản phẩm
2	Sizes_idSizes	INT	x		ID size sản phẩm

Bảng 26. Products_has_Sizes

Bảng Products_has_Colors

Products_has_Colors					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	Products_idProducts	INT	x		ID sản phẩm
2	Colors_idColors	INT	x		ID màu sản phẩm

Bảng 27. Products_has_Colors

Bảng Reviews

Reviews					
STT	Tên trường	Kiểu	Khóa chính	Khóa ngoại	Mô tả
1	idReviews	INT	X	X	ID bình luận
2	iduser	INT		X	ID người dùng
3	Idproduct	INT		X	ID sản phẩm
4	rating	INT			Đánh giá sản phẩm
5	comment	varchar(255)			Bình luận sản phẩm

6	datecmt	Timetamp			Ngày đánh giá sản phẩm
---	---------	----------	--	--	------------------------

Bảng 28. Reviews

Định dạng URI

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

Trong đó:

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

Dưới đây là danh sách mô tả các REST endpoint:

STT	URI	x-access-token	HTTP method	Mô tả
1	/user/signin/		POST	Đăng nhập tài khoản người dùng
2	/user/signup/		POST	Đăng ký tài khoản
3	/user/profile/	Yes	GET	Lấy thông tin cá nhân
4	/user/profile/	Yes	PUT	Cập nhật thông tin cá nhân
5	/user/getall	Yes	GET	Lấy tất cả thông tin người dùng
6	/user/upload	Yes	POST	Cập nhật ảnh đại diện người dùng
7	/user/1	Yes	DELETE	Xóa người dùng có ID là 1
8	/product/		GET	Lấy danh sách tất cả sản phẩm
9	/product/1		GET	Lấy thông tin sản phẩm có id là 1
10	/product/	Yes	POST	Tạo mới một sản phẩm
11	/product/img/1	Yes	PUT	Cập nhật hình sản phẩm có ID là 1

12	/product/1	Yes	PUT	Cập nhật thông tin sản phẩm có id là 1
13	/product/1	Yes	DELETE	Xóa sản phẩm có id là 1
14	/size/	Yes	GET	Lấy danh sách tất cả size
15	/size/1	Yes	GET	Lấy size có id là 1
16	/size/	Yes	POST	Tạo mới một size
17	/size/1	Yes	PUT	Cập nhật size có id là 1
18	/size/1	Yes	DELETE	Xóa sản size có id là 1
19	/category/		GET	Lấy danh sách tất cả loại sản phẩm
20	/category/1	Yes	GET	Lấy loại sản phẩm có id là 1
21	/category/	Yes	POST	Tạo mới một loại sản phẩm
22	/category/1	Yes	PUT	Cập nhật loại sản phẩm có id là 1
23	/category/1	Yes	DELETE	Xóa sản loại sản phẩm có id là 1
14	/color/	Yes	GET	Lấy danh sách tất cả màu sản phẩm
15	/color/1		GET	Lấy màu sản phẩm có id là 1
16	/color/	Yes	POST	Tạo mới một màu sản phẩm
17	/color/1	Yes	PUT	Cập nhật size có id là 1
18	/color/1	Yes	DELETE	Xóa màu sản phẩm có id là 1
19	/order	Yes	POST	Tạo mới một hóa đơn

20	/order	Yes	GET	Lấy thông tin hóa đơn của người dùng
21	/order/getall	Yes	GET	Lấy thông tin hóa đơn của tất người dùng
22	/order/1	Yes	PUT	Yêu cầu hủy hóa đơn có id là 1 của người dùng
23	/order/conformcandle/1	Yes	DELETE	Hủy hóa đơn có id là 1 của người dùng
24	/order/conform/1	Yes	PUT	Xác nhận hóa đơn có id là 1 của người dùng
25	/review	Yes	POST	Tạo mới một đánh giá sản phẩm
26	/review/id	Yes	GET	Lấy thông tin đánh giá sản phẩm có id là 1
27	/review/	Yes	GET	Lấy thông tin đánh giá của tất cả sản phẩm
28	/review/1	Yes	DELETE	Xóa thông tin đánh giá sản phẩm có id là 1

Bảng 29. danh sách mô tả các REST endpoint

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

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

- URI: [host-address:port]/api/
- HTTP method: POST
- Content Type: JSON
- Headers: { “x-access-token”: “token” }

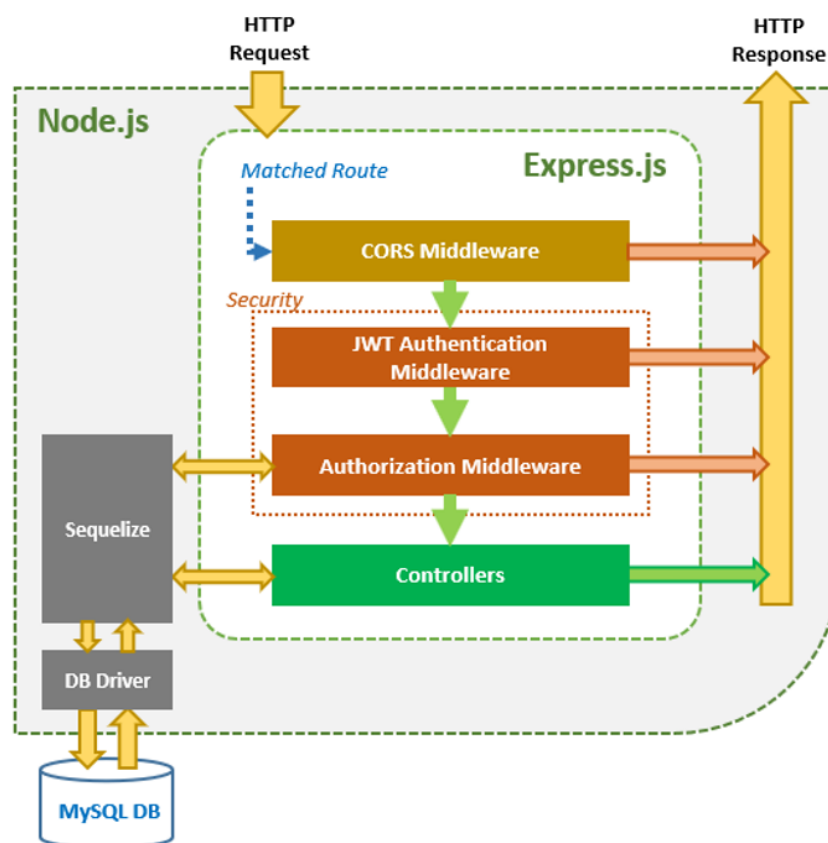
Dưới đây là danh sách một số status code thông dụng được trả về từ response:

STT	Code	TextCode	Mô tả
1	200	OK	Thành công
2	201	Created	Đã tạo ra một tài nguyên thành công

3	204	No Content	Thành công nhưng response không chứa nội dung
4	400	Bad Request	Lỗi server không hiểu yêu cầu
5	401	Bad Request	Lỗi yêu cầu không được xác thực hợp lệ
6	403	Forbidden	Lỗi yêu cầu không được phép
7	404	Not Found	Lỗi yêu cầu không được tìm thấy
8	405	Method Not Allow	Lỗi server từ chối tiếp nhận loại phương thức HTTP cụ thể
9	500	Server Internal Error	Lỗi không mong muốn trên server nhằm từ chối việc thực hiện yêu cầu

Bảng 30. danh sách một số status code thông dụng được trả về từ response

Kiến trúc các thành phần của REST API server



Hình 18. Kiến trúc các thành phần của REST API

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

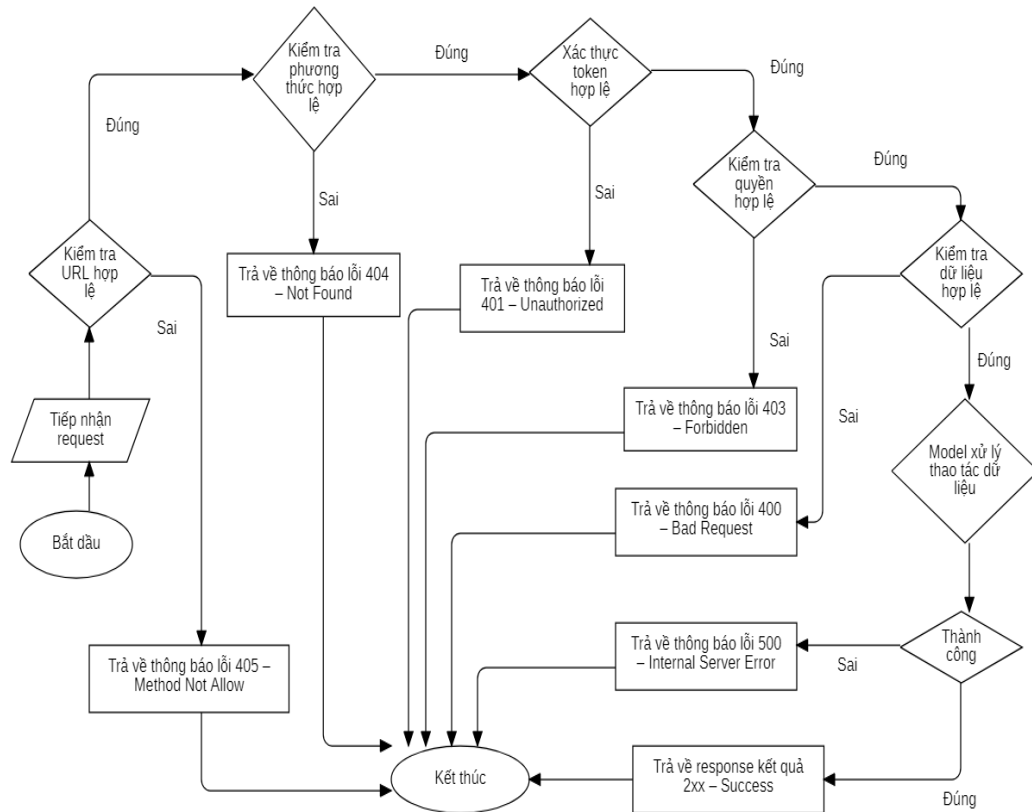
Với các bộ định tuyến(routes) của Express, **HTTP request** sẽ khớp với bộ định tuyến thông qua **CORS Middleware** trước khi đến lớp bảo mật(security)

Lớp bảo mật bao gồm:

- JWT Authentication Middleware: verify Signup, verify token.

- Authorization Middleware: check IsAdmin .

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

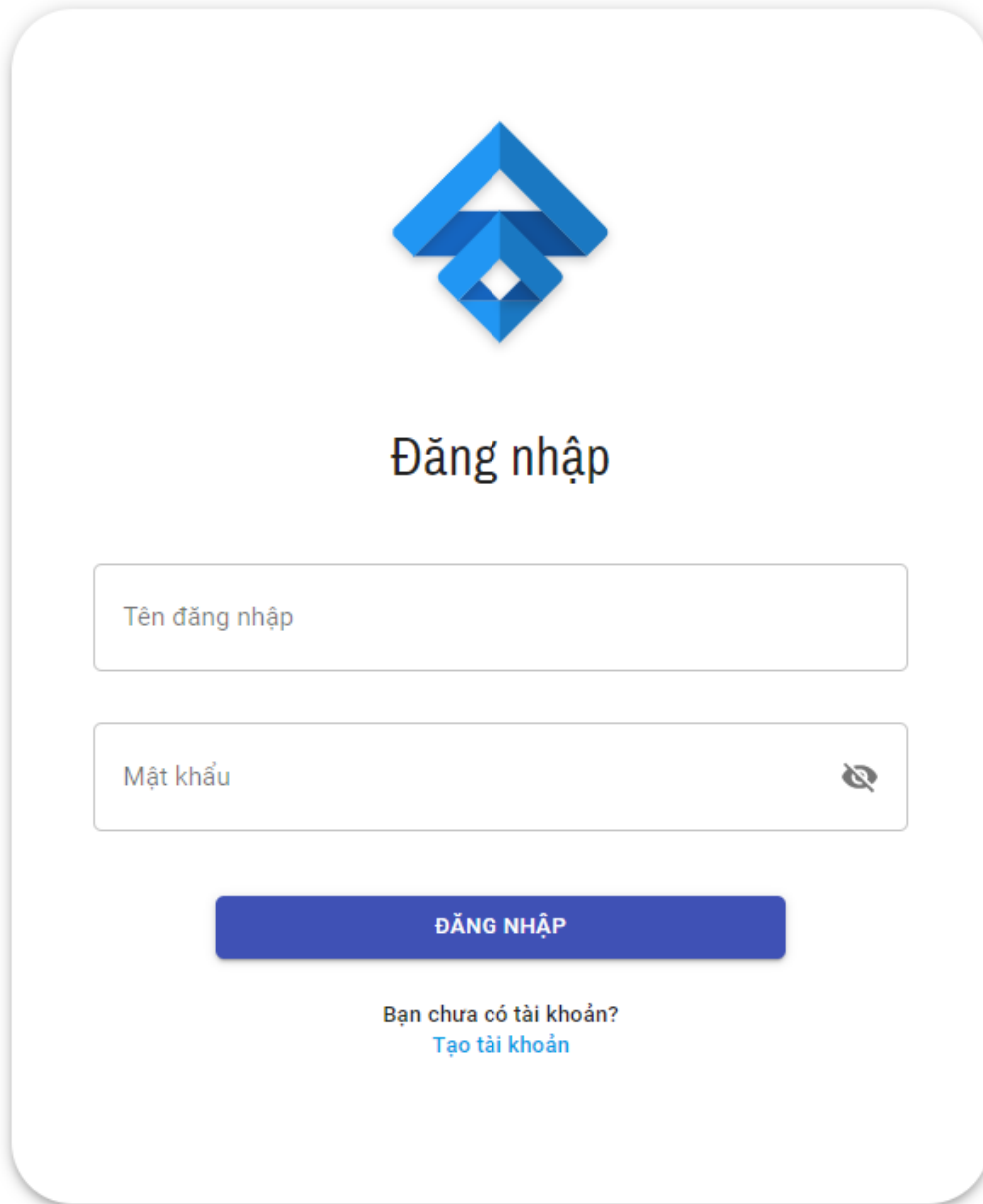


Hình 19. Lưu đồ xử lý yêu cầu của REST API server

2.2.2. Trang quản lý

Giao diện web

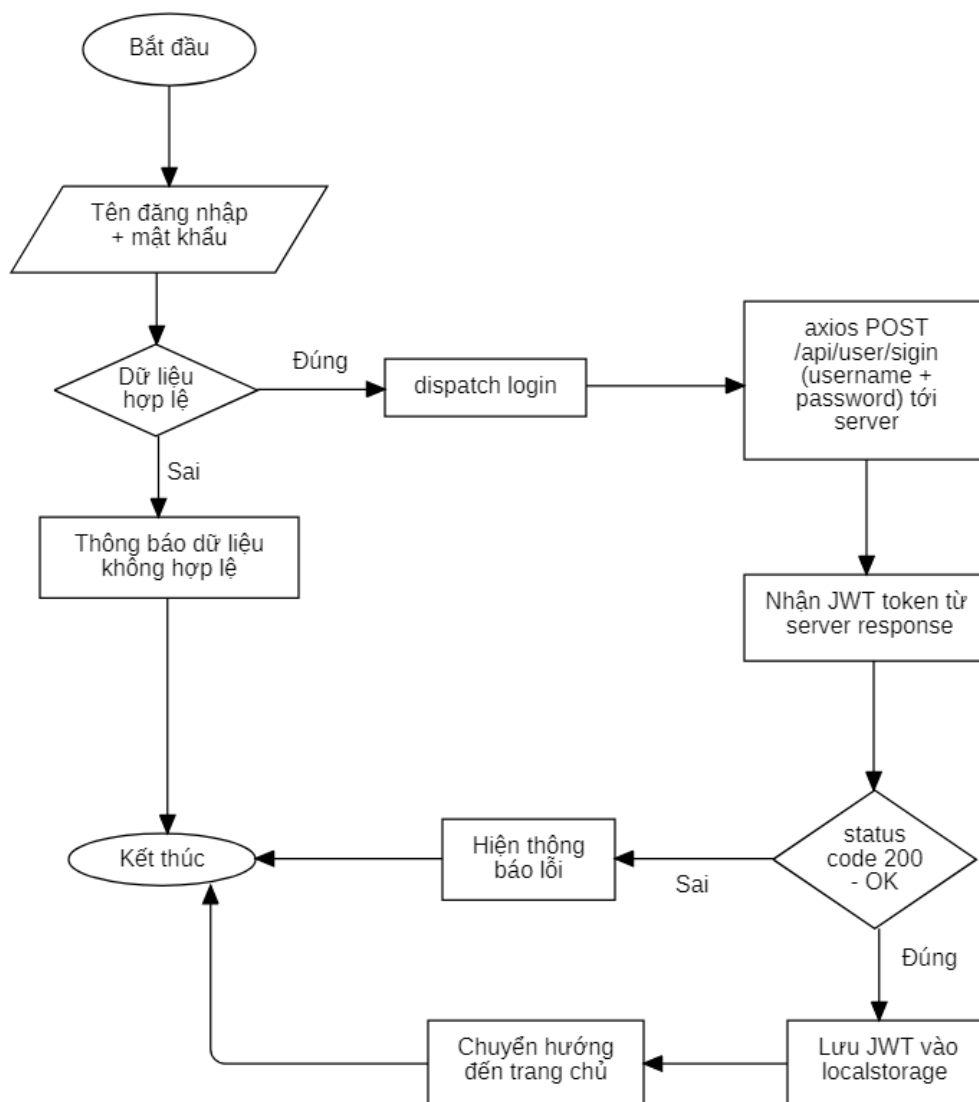
Màn hình hình đăng nhập



The image shows a login form interface. At the top center is a blue geometric logo consisting of three interlocking shapes. Below the logo is the title "Đăng nhập" in a large, bold, black font. Underneath the title are two input fields: the first is labeled "Tên đăng nhập" (Username) and the second is labeled "Mật khẩu" (Password). The password field has a small eye icon on the right side to toggle visibility. Below these fields is a blue button with the text "ĐĂNG NHẬP" in white, uppercase letters. At the bottom of the form, there is a link that says "Bạn chưa có tài khoản?" (Don't you have an account?) followed by the text "Tạo tài khoản" (Create account) in blue, which is a clickable link.

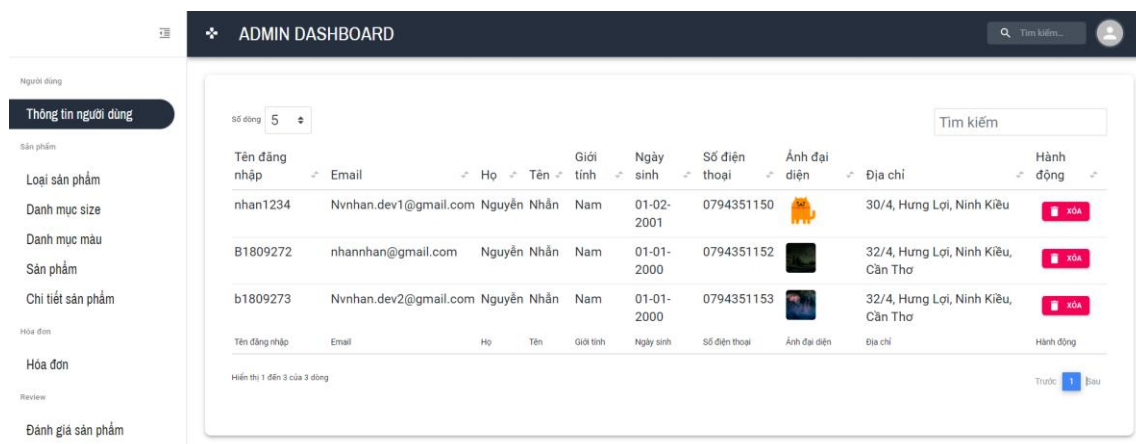
Hình 20. Giao diện màn hình đăng nhập

Màn hình đăng nhập gồm các trường tên đăng nhập, mật khẩu, tùy chọn hiển thị mật khẩu, chữ “Tạo tài khoản” chuyển hướng đến trang đăng ký và nút đăng nhập.



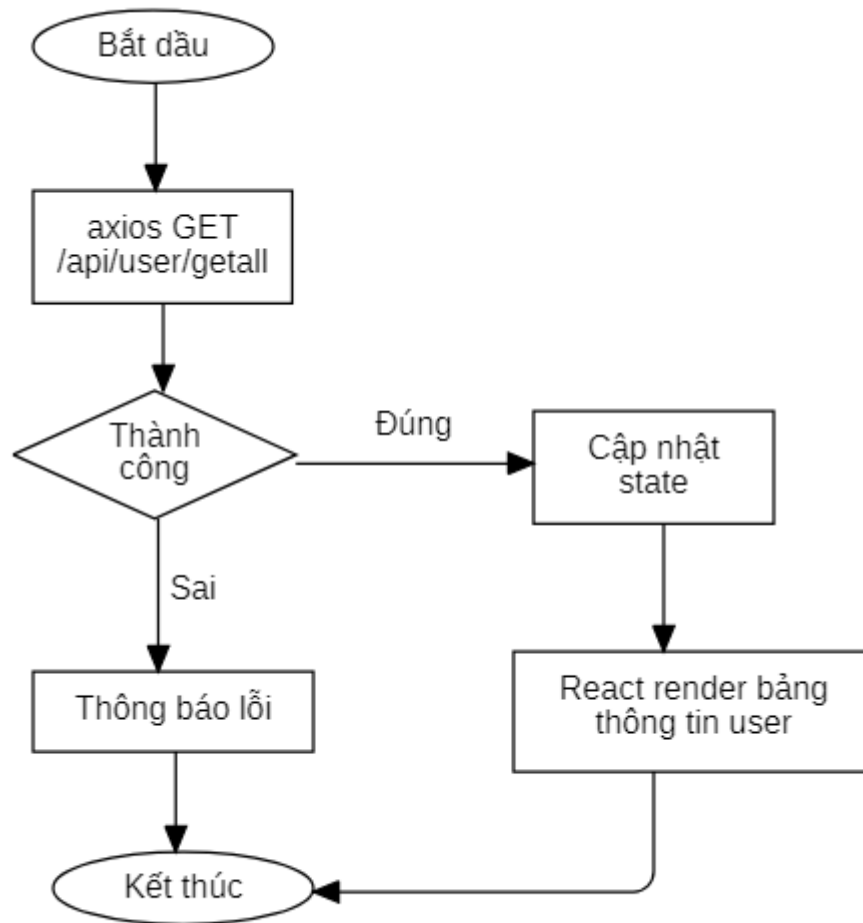
Hình 21. Lưu đồ xử lý thực hiện yêu cầu đăng nhập

Quản lý thông tin người dùng



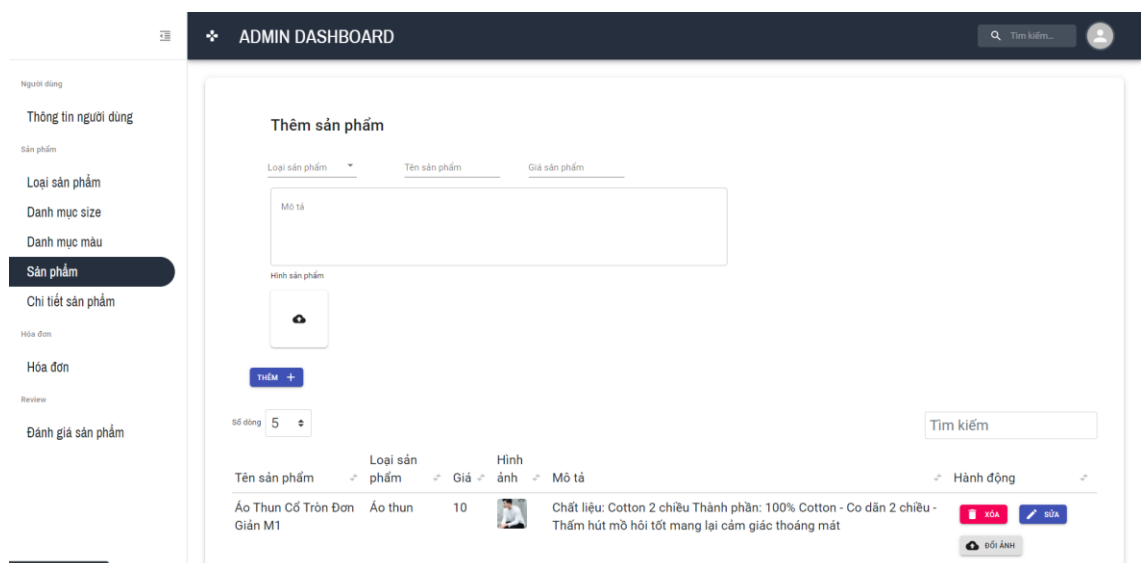
Hình 22. Giao diện quản lý thông tin người dùng

Quản lý thông tin người dùng sử dụng một bảng để hiển thị tất cả thông tin người dùng, nút xóa, các tùy chọn như tìm kiếm, sắp xếp, lọc và cụm các nút dùng để phân trang và hiển thị số dòng.



Hình 23. Lưu đồ xử lý liệt kê danh sách người dùng

Quản lý sản phẩm



Hình 24. Giao diện quản lý sản phẩm

Tương tự như quản lý thông tin người dùng, quản lý sản phẩm cũng sử dụng một bảng để hiển thị danh sách sản phẩm. Ngoài ra còn có một số nút tùy chọn như tạo mới sản phẩm, xóa, cập nhật ảnh và sửa thông tin sản phẩm.


Tạo mới một sản phẩm

Thêm sản phẩm

Loại sản phẩm ▼ Tên sản phẩm _____ Giá sản phẩm _____

Mô tả _____

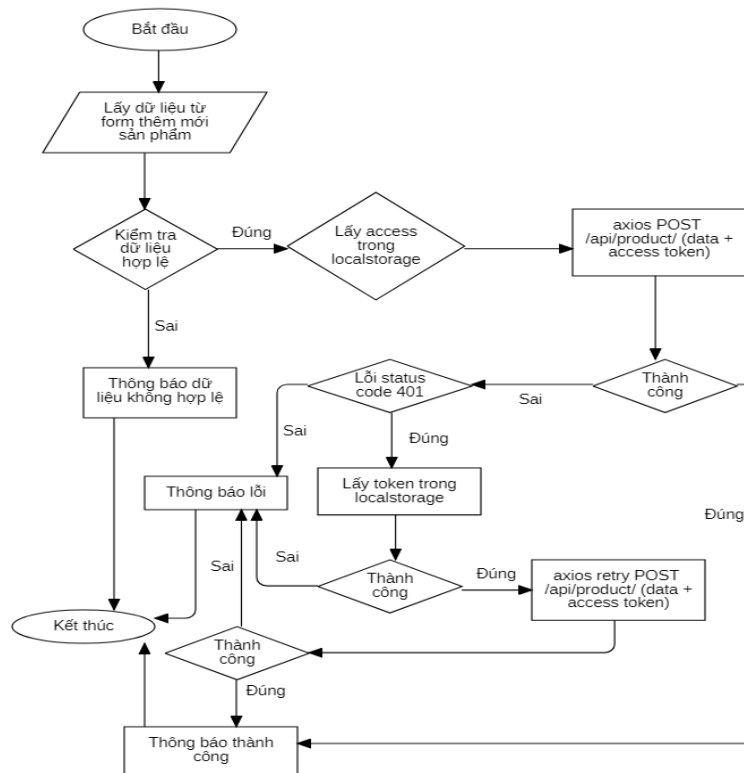
Hình sản phẩm



THÊM +

Hình 25. Giao diện tạo mới một sản phẩm

Tính năng này cho phép người quản lý nhập thông tin cần thiết để tạo mới một sản phẩm như: loại sản phẩm, tên sản phẩm, giá sản phẩm, mô tả và hình ảnh sản phẩm.



Hình 26. Lưu đồ xử lý tạo mới sản phẩm

Chỉnh sửa thông tin sản phẩm

Hình 27. Giao diện sửa thông tin sản phẩm

Chỉnh sửa thông tin sản phẩm sử dụng dialog hiện lên cho phép người quản lý chỉnh sửa thông tin và yêu cầu cập nhật. Khi người quản lý nhấn nút sửa trên dòng dữ liệu trong bảng sản phẩm dữ liệu sản phẩm này sẽ tự động điền vào các trường tương ứng trên dialog và cho phép người quản lý chỉnh sửa lại.

Cập nhật ảnh cho sản phẩm

Hình 28. Giao diện cập nhật ảnh sản phẩm

Cập nhật ảnh sản phẩm sử dụng dialog hiện lên cho phép người quản lý chọn và cập nhật ảnh sản phẩm.

Xóa sản phẩm

Bạn muốn xóa

HỦY

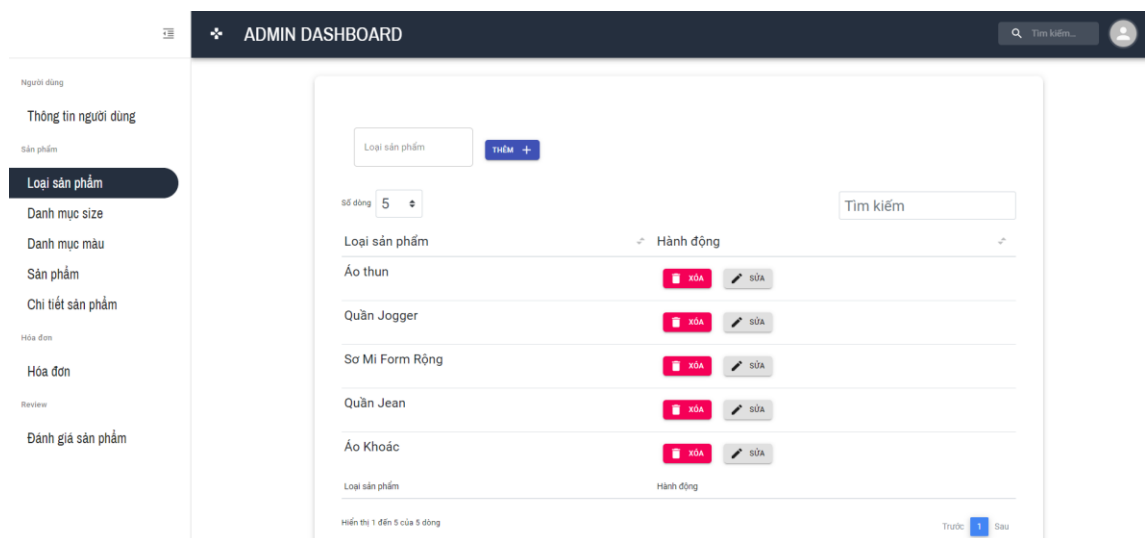
ĐỒNG Ý

Hình 29. Giao diện xóa sản phẩm

Xóa sản phẩm sử dụng dialog hiện lên cho phép người quản lý xác nhận xóa sản phẩm.

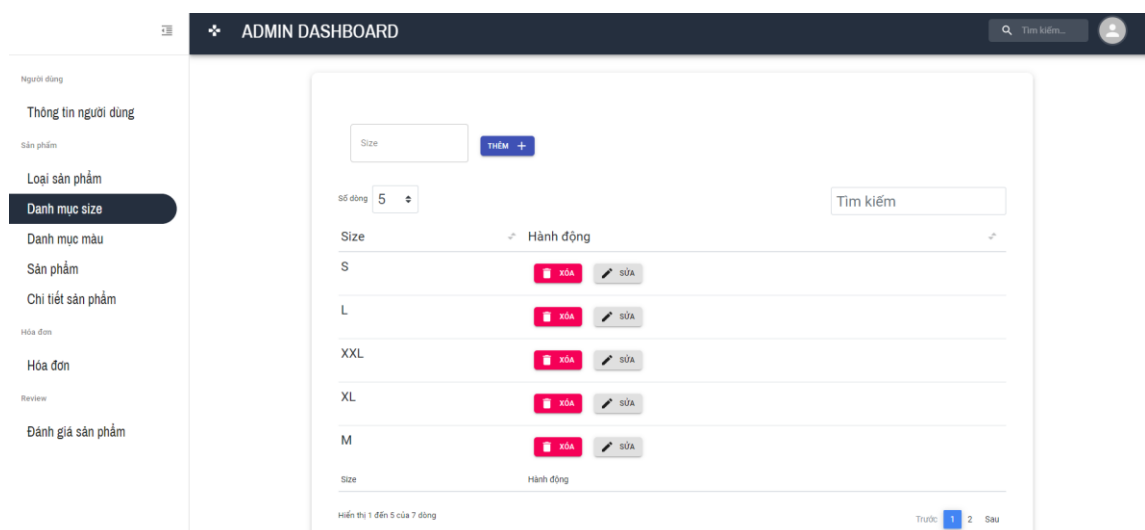
Các trang quản lý khác đều có các chức năng tương tự

Quản lý loại sản phẩm



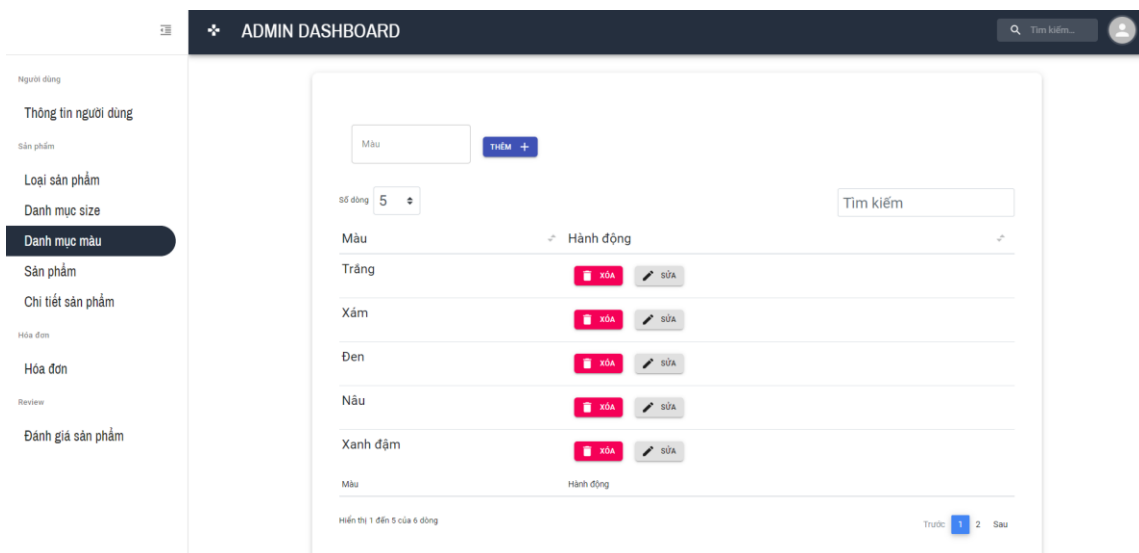
Hình 30. Giao diện trang quản lý loại sản phẩm

Quản lý danh mục size



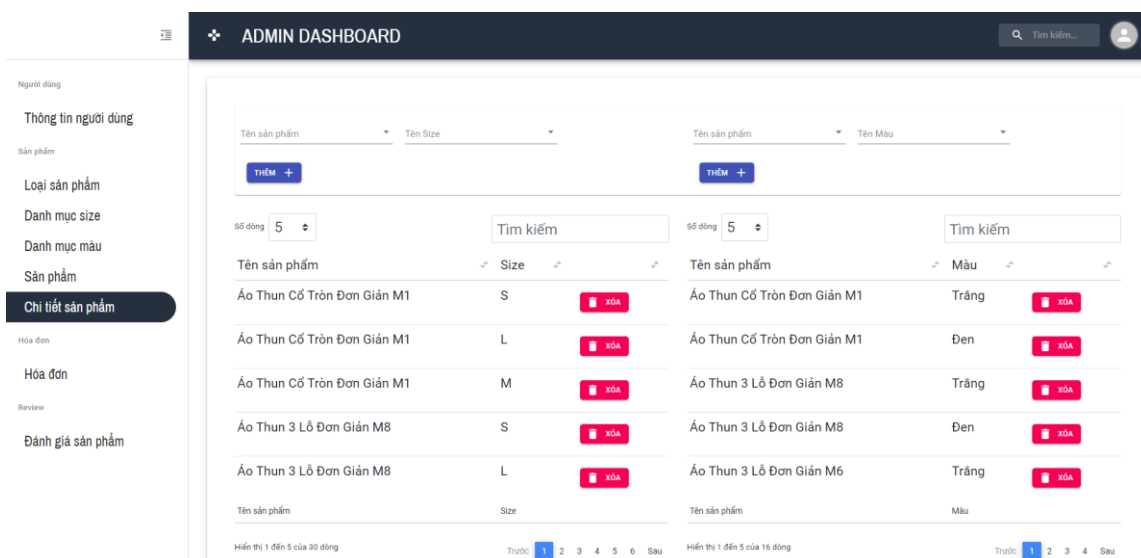
Hình 31. Giao diện trang quản lý danh mục size

Quản lý danh mục màu sắc



Hình 32. Giao diện quản lý danh mục màu sắc

Quản lý chi tiết sản phẩm



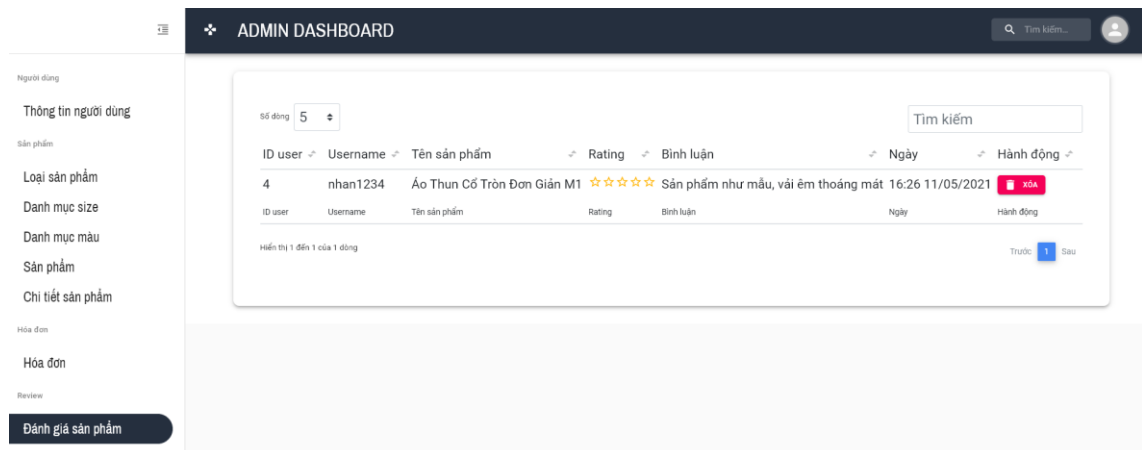
Hình 33. Giao diện quản lý chi tiết sản phẩm

Quản lý hóa đơn



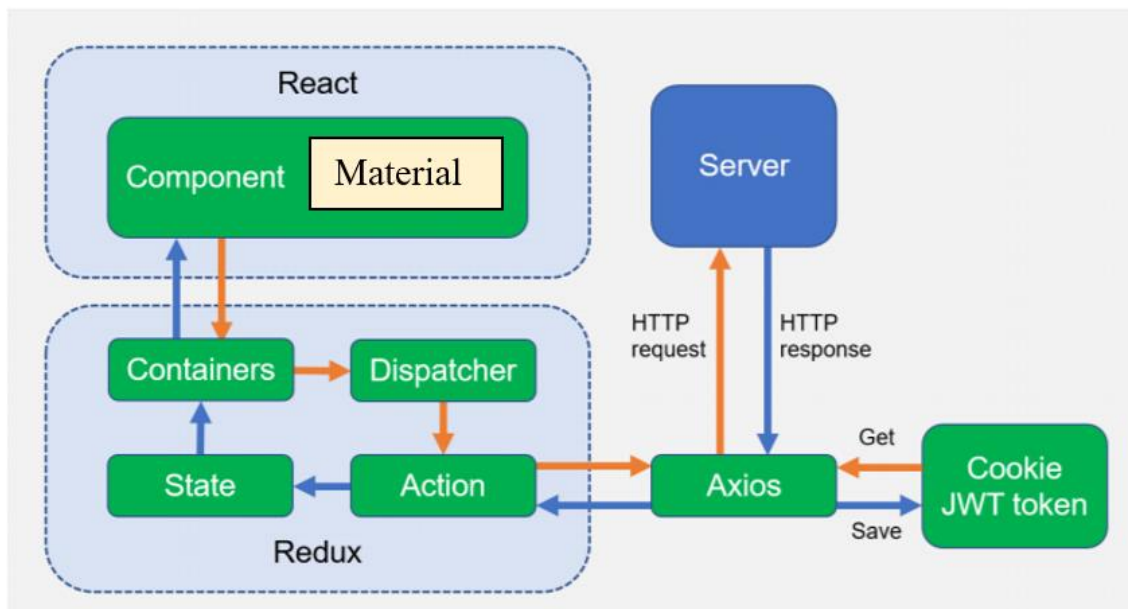
Hình 34. Giao diện trang quản lý hóa đơn

Quản lý bình luận, đánh giá sản phẩm.



Hình 35. Giao diện trang quản lý đánh giá sản phẩm

Kiến trúc các thành phần của web client




Hình 36. Kiến trúc các thành phần của web client

Web client sẽ kết hợp react và redux để quản lý state dễ dàng hơn. Sử dụng Material để tạo dựng giao diện người dùng và axios để trao đổi với REST API server. Ngoài ra, sử dụng cookie lưu trữ token để ghi nhớ phiên đăng nhập và tạo mới access token khi hết hạn.

2.2.3. Trang người dùng

Trang đăng ký



Đăng ký tài khoản

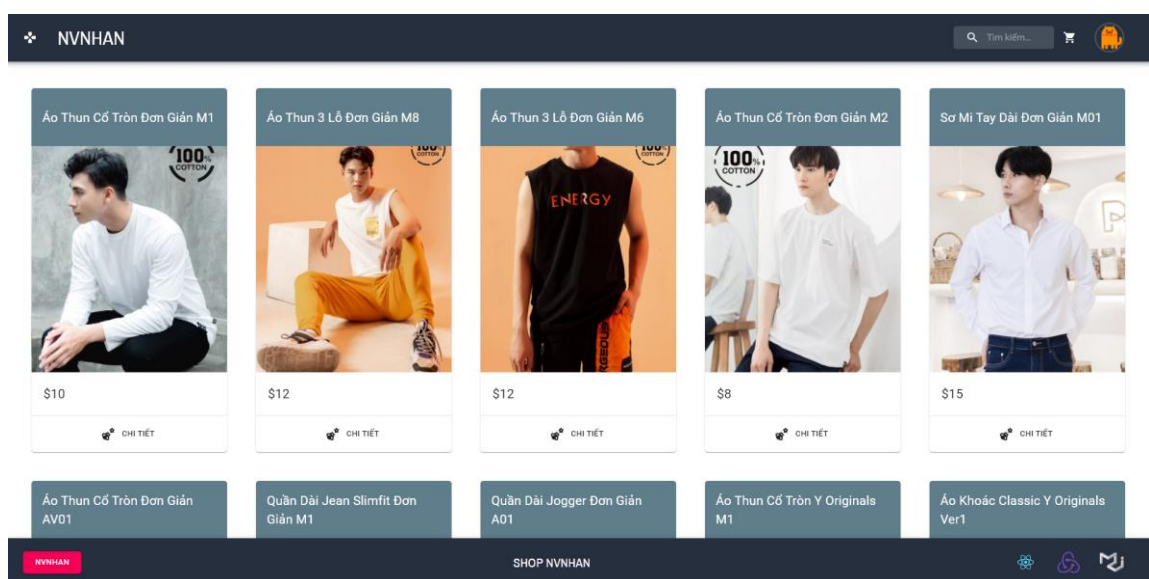
ĐĂNG KÝ

Bạn đã có tài khoản?
[Đăng nhập](#)

Hình 37. Giao diện trang đăng ký

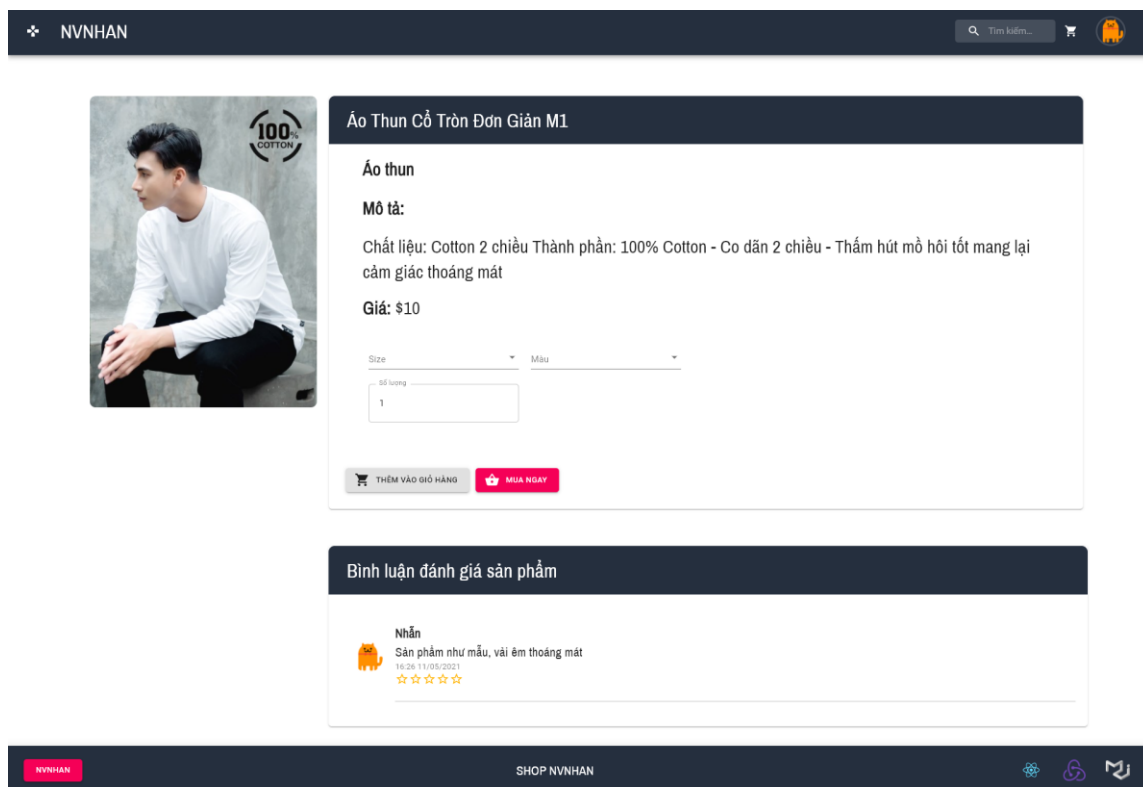
Màn hình đăng ký gồm các trường tên đăng nhập, email, mật khẩu, nhập lại mật khẩu tùy chọn hiển thị mật khẩu, chữ “Đăng nhập” chuyển hướng đến trang đăng nhập và nút đăng ký.

Trang chủ



Hình 38. Giao diện trang chủ

Trang chi tiết sản phẩm



Hình 39. Giao diện trang chi tiết sản phẩm

Trang chi tiết sản phẩm hiển thị thông tin chi tiết sản phẩm, các lựa chọn về màu, size, số lượng để mua sản phẩm, các bình luận đánh giá về sản phẩm. Ngoài ra còn có hai nút thêm vào giỏ hàng và mua ngay, nếu người dùng nhấn vào nút thêm vào giỏ hàng thì sản phẩm đó sẽ được thêm vào giỏ hàng, còn nếu người dùng nhấn vào nút mua ngay thì sẽ hiện thị dialog thanh toán sau đó sẽ tiến hành thanh toán sản phẩm đó bằng cổng thanh toán paypal.

Thanh toán

Tên

Số điện thoại

Địa chỉ

CẬP NHẬT THÔNG TIN

Áo Thun Cổ Tròn Đơn Giản M1

Phân loại: S, Đen

Số lượng: 1

Giá: \$10

Tổng thanh toán: \$10

PayPal Checkout
The safer, easier way to pay

HỦY

Hình 40. Dialog thanh toán sản phẩm

Hình 41. Giao diện trang thanh toán paypal

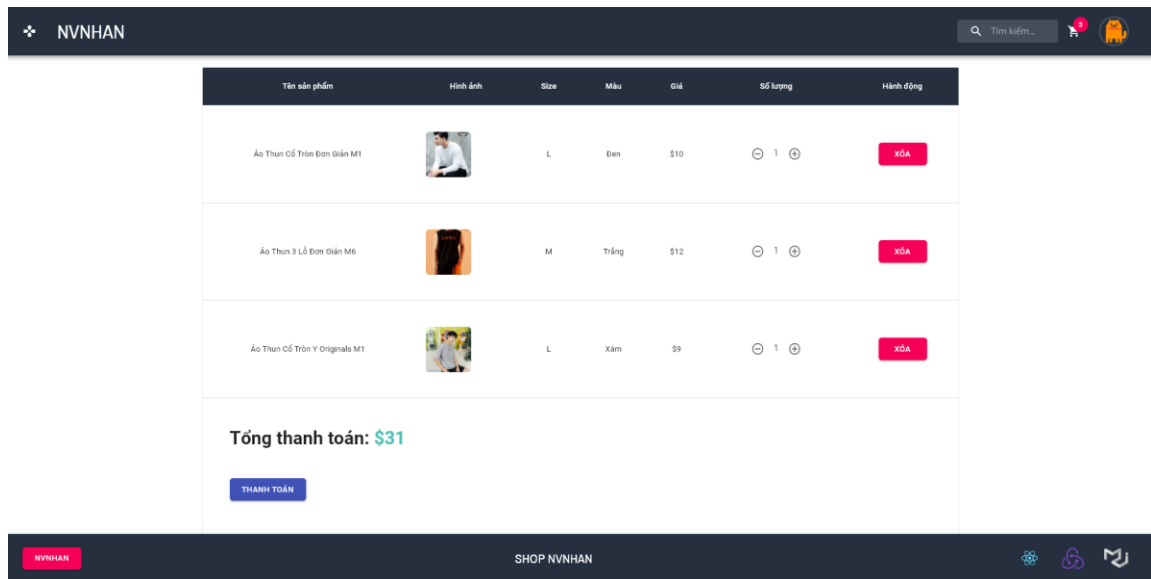
Trang đơn hàng

Đơn hàng	Ngày	Sản Phẩm	Giá	Tổng giá	Trạng thái	Hành động
424	09/05/2021 12:31	1. Áo Thun Cổ Tròn Đơn Giản M1 - Trắng - S	1. \$10	\$10	Đang chờ xác nhận	<button>hủy</button>
434	09/05/2021 12:34	1. Áo Thun Cổ Tròn Đơn Giản M1 - Trắng - S 2. Áo Thun 3 Lỗ Đơn Giản M8 - Trắng - S 3. Áo Thun 3 Lỗ Đơn Giản M6 - Đen - S	1. \$10 2. \$12 3. \$12	\$34	Đã xác nhận	

Hình 42. Giao diện trang đơn hàng

Khi thanh toán thành công đơn hàng sẽ được tạo ra, nếu người dùng muốn hủy đơn hàng phải hủy trước khi chủ cửa hàng xác nhận, đơn hàng hủy sẽ được hoàn tiền trong ba ngày (do chủ cửa hàng thực hiện).

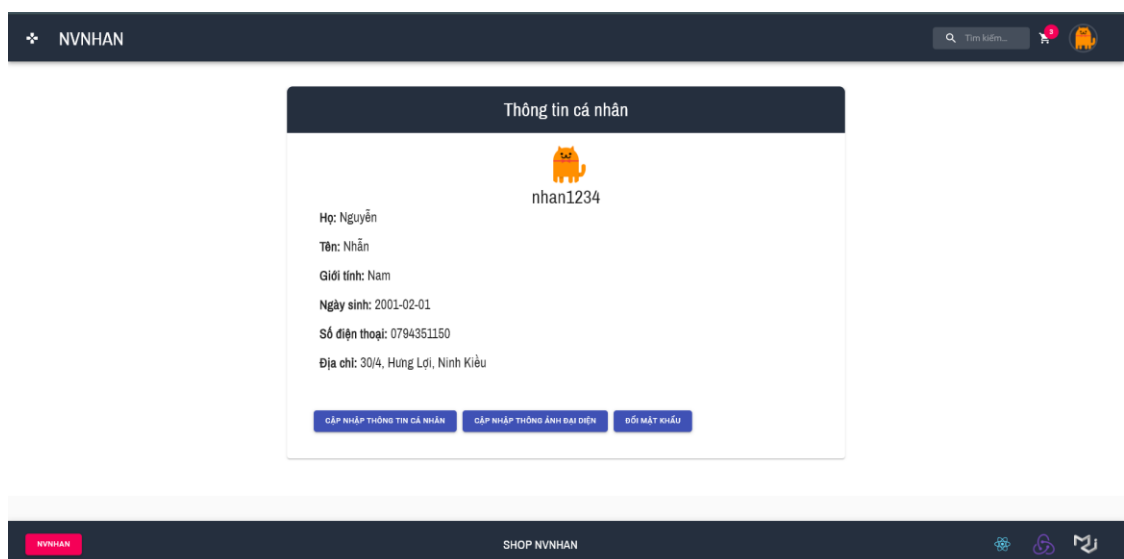
Trang giỏ hàng



Hình 43. Giao diện giỏ hàng

Trang giỏ hàng gồm một bảng liệt kê thông tin các sản phẩm gồm tên sản phẩm, hình ảnh, màu, size, số lượng, số lượng có thể tùy chỉnh và có thể xóa sản phẩm thông qua nút xóa.

Trang thông tin cá nhân



Hình 44. Giao diện trang thông tin cá nhân

Khi nhấn vào các nút cập nhật thông tin cá nhân, cập nhật ảnh đại diện, đổi mật khẩu thì các dialog tương ứng sẽ được hiện lên.

Cập nhật thông tin cá nhân

Họ

Nguyễn

Tên

Nhân

Giới tính:

☒ Nam ☐ Nữ

Ngày sinh

02/01/2001

Số điện thoại

0794351150

Địa chỉ

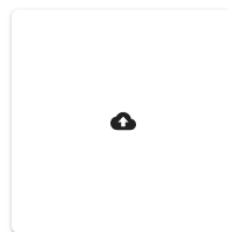
30/4, Hưng Lợi, Ninh Kiều

HỦY

ĐỒNG Ý

Hình 45. Giao diện dialog cập nhập thông tin cá nhân

Cập nhật ảnh đại diện



HỦY

ĐỒNG Ý

Hình 46. Giao diện dialog cập nhật ảnh đại diện

Đổi mật khẩu

Mật khẩu cũ

Mật khẩu mới

Nhập lại mật khẩu

HỦY

ĐỒNG Ý

Hình 47. Giao diện dialog đổi mật khẩu

Một số chức năng phụ:

- Đăng xuất: kết thúc phiên làm việc.
- Thông báo: hiện thông báo thành công hoặc thông báo lỗi.
- Tìm kiếm: tìm kiếm những dòng dữ liệu của một cột trong bảng.
- Sắp xếp: sắp xếp dữ liệu của một cột bảng.
- Phân trang: cụm phân trang dành cho bảng.
- Lọc: lọc ra các trường trùng của một cột trong bảng.

Chương 3: Đánh giá kiểm thử

3.1. Mục tiêu kiểm thử

Mục tiêu kiểm thử trọng tâm chính là kiểm tra xem trang web có hoạt động tốt với các tính năng được lập trình hay không, có phát sinh lỗi ngoài ý muốn hay không nhằm mục đích đưa ra hướng khắc phục hoặc cách khắc phục trong tương lai. Bên cạnh đó, mục tiêu thứ cấp kiểm thử chính là REST API server để xác định khả năng đáp ứng yêu cầu đưa ra hay không và xác định những lỗi phát sinh cần được khắc phục.

3.2. Kịch bản kiểm thử

Kịch bản 1: Trang quản lý

Tiến hành kiểm thử trang quản lý:

- Đăng nhập khi không nhập thông tin.
- Đăng nhập sai tên đăng nhập hoặc sai mật khẩu.
- Đăng nhập đúng tên đăng nhập, mật khẩu.
- Truy cập chức năng quản lý thông tin người dùng. Sau đó tìm kiếm, xóa một người dùng.
- Truy cập chức năng quản lý danh mục sản phẩm. Sau đó tìm kiếm, tạo, cập nhật, xóa một danh mục sản phẩm.
- Truy cập chức năng quản lý danh mục size. Sau đó tìm kiếm, tạo, cập nhật, xóa một danh mục size.
- Truy cập chức năng quản lý danh mục màu sắc. Sau đó tìm kiếm, tạo, cập nhật, xóa một danh mục màu sắc.
- Truy cập chức năng quản lý sản phẩm. Sau đó tìm kiếm, tạo, cập nhật, xóa một sản phẩm.
- Truy cập chức năng quản lý chi tiết sản phẩm. Sau đó tìm kiếm, tạo, xóa một chi tiết sản phẩm.
- Truy cập chức năng quản lý hóa đơn. Sau đó tìm kiếm, xác nhận, hủy một hóa đơn.
- Truy cập chức năng quản lý đánh giá sản phẩm. Sau đó tìm kiếm, xóa một đánh giá sản phẩm.
- Đăng xuất

Kịch bản 2: Trang người dùng

Tiến hành kiểm thử trang người dùng:

- Truy cập trang chủ.

- Đăng ký khi không nhập thông tin.
- Đăng ký khi nhập tên đăng nhập đã có trên hệ thống.
- Đăng ký khi nhập email đã có trên hệ thống.
- Đăng ký khi nhập đầy đủ thông tin, tên đăng nhập email, không tồn tại trên hệ thống.
- Xem chi tiết sản phẩm.
- Thanh toán sản phẩm khi chưa đăng nhập.
- Hủy thanh toán khi đang tiến hành thanh toán một sản phẩm.
- Thanh toán thành công một sản phẩm.
- Thêm sản phẩm vào giỏ hàng khi chưa chọn size, màu.
- Thêm sản phẩm vào giỏ hàng khi đã chọn size màu.
- Thêm sản phẩm vào giỏ hàng khi chưa đăng nhập.
- Xem chi tiết giỏ hàng.
- Tăng số lượng một sản phẩm trong giỏ hàng.
- Giảm số lượng một sản phẩm trong giỏ hàng.
- Thanh toán giỏ hàng khi chưa có sản phẩm.
- Thanh toán tất cả sản phẩm trong giỏ hàng.
- Hủy thanh toán khi thanh toán tất cả sản phẩm trong giỏ hàng.
- Xem chi tiết hóa đơn.
- Hủy một đơn hàng khi chủ cửa hàng chưa xác nhận.
- Xem chi tiết thông tin cá nhân.
- Thay đổi thông tin cá nhân.
- Cập nhật ảnh đại diện.
- Đổi mật khẩu khi nhập sai mật khẩu cũ.
- Đổi mật khẩu khi nhập đúng mật khẩu cũ.

Môi trường kiểm thử

- Thiết bị kiểm thử: LAPTOP-IV92SPDB
- Cấu hình thiết bị:
 - ✓ Chip AMD Ryzen 5 2500U
 - ✓ Hệ điều hành Windows 10

- ✓ RAM 12GB
- ✓ Trình duyệt Chrome Version 90.0.4430.212 (Official Build) (64-bit)

Kết nối mạng: Mạng LAN

3.3. Đánh giá kiểm thử

Kết quả kiểm thử kịch bản

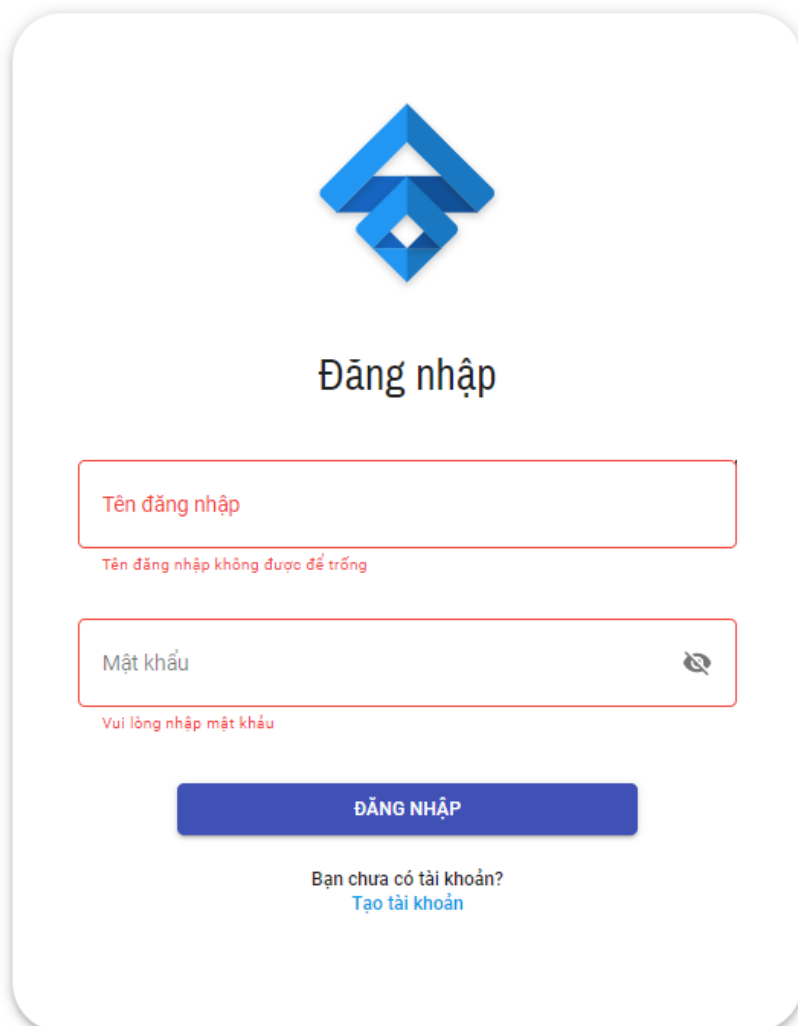
Kịch bản	Mức độ đạt yêu cầu
Kịch bản 1	Đạt
Kịch bản 2	Đạt

Bảng 31. Kết quả kiểm thử kịch bản

Kết quả kiểm thử

Một số hình ảnh kiểm thử trang người dùng

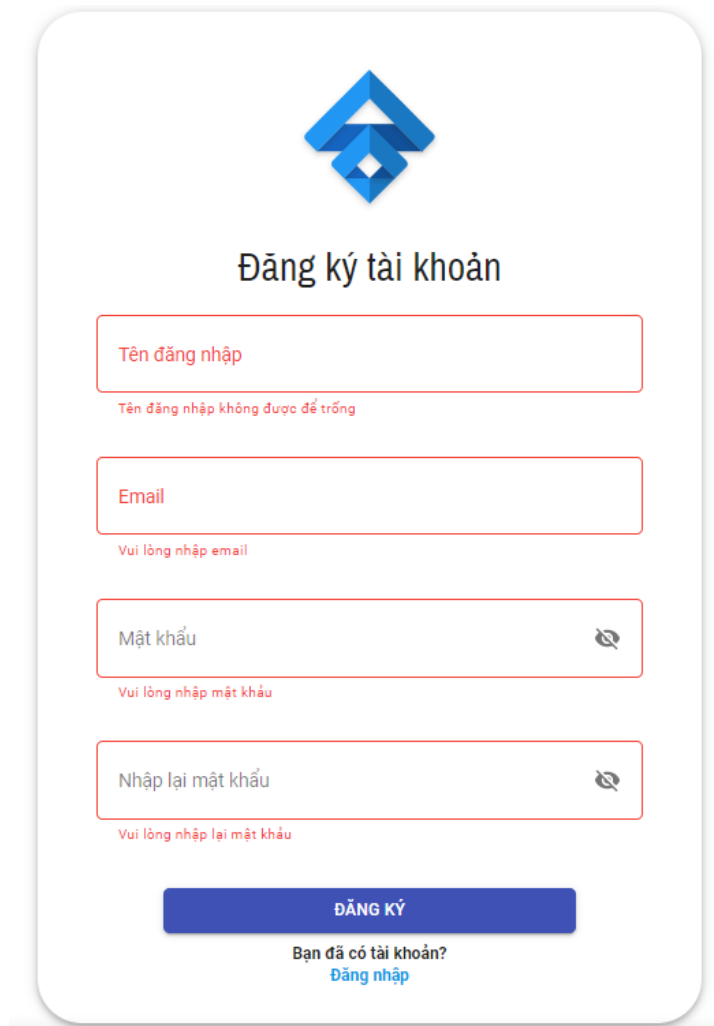
Giao diện trang đăng nhập khi chưa nhập thông tin



The screenshot shows a login page with a blue geometric logo at the top. Below the logo is the text "Đăng nhập". There are two input fields: "Tên đăng nhập" (Username) and "Mật khẩu" (Password). The "Tên đăng nhập" field has a red border and a red error message "Tên đăng nhập không được để trống" (Username cannot be empty). The "Mật khẩu" field has a red border and a red error message "Vui lòng nhập mật khẩu" (Please enter password). There is a blue button labeled "ĐĂNG NHẬP" (Login). At the bottom, there is a link "Bạn chưa có tài khoản? Tạo tài khoản" (Don't have an account? Create account).

Hình 48. Giao diện trang đăng nhập khi chưa nhập thông tin

Giao diện trang đăng ký khi chưa nhập thông tin



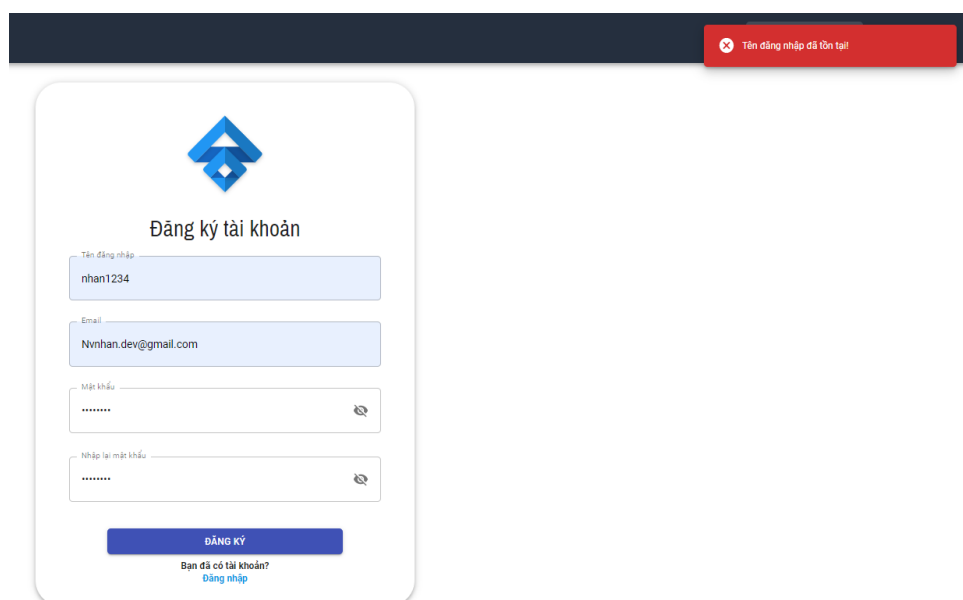
The registration form is displayed with the following fields and labels:

- Tên đăng nhập** (Username): A red border indicates it is required. Below the field, the text "Tên đăng nhập không được để trống" (Username cannot be empty) is shown in red.
- Email**: A red border indicates it is required. Below the field, the text "Vui lòng nhập email" (Please enter email) is shown in red.
- Mật khẩu** (Password): A red border indicates it is required. Below the field, the text "Vui lòng nhập mật khẩu" (Please enter password) is shown in red. An eye icon is present to toggle visibility.
- Nhập lại mật khẩu** (Repeat password): A red border indicates it is required. Below the field, the text "Vui lòng nhập lại mật khẩu" (Please repeat password) is shown in red. An eye icon is present to toggle visibility.

At the bottom, there is a blue button labeled **ĐĂNG KÝ** (Register). Below the button, the text "Bạn đã có tài khoản?" (Do you have an account?) is shown, followed by a blue link labeled **Đăng nhập** (Login).

Hình 49. Giao diện trang đăng ký khi chưa nhập thông tin

Giao diện trang đăng ký khi nhập tên đăng nhập đã tồn tại



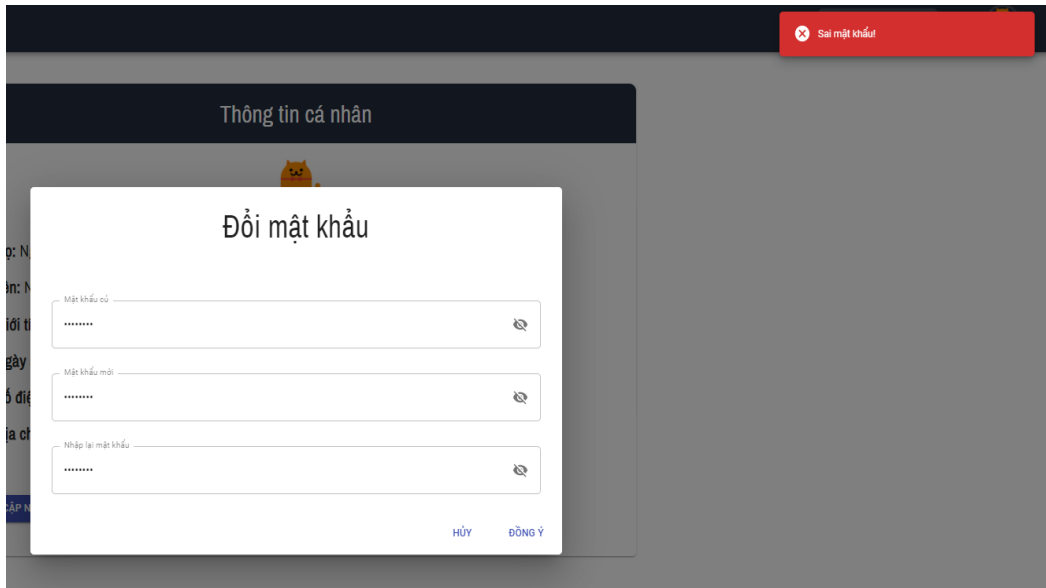
The registration form is displayed with the following fields and labels:

- Tên đăng nhập** (Username): The field contains the text "nhan1234". A red border indicates it is required. Below the field, the text "Tên đăng nhập đã tồn tại" (Username already exists) is shown in red.
- Email**: The field contains the text "Nvnhan.dev@gmail.com". A red border indicates it is required. Below the field, the text "Vui lòng nhập email" (Please enter email) is shown in red.
- Mật khẩu** (Password): The field contains the text "*****". A red border indicates it is required. Below the field, the text "Vui lòng nhập mật khẩu" (Please enter password) is shown in red. An eye icon is present to toggle visibility.
- Nhập lại mật khẩu** (Repeat password): The field contains the text "*****". A red border indicates it is required. Below the field, the text "Vui lòng nhập lại mật khẩu" (Please repeat password) is shown in red. An eye icon is present to toggle visibility.

At the bottom, there is a blue button labeled **ĐĂNG KÝ** (Register). Below the button, the text "Bạn đã có tài khoản?" (Do you have an account?) is shown, followed by a blue link labeled **Đăng nhập** (Login).

Hình 50. Giao diện trang đăng ký khi nhập tên đăng nhập đã tồn tại

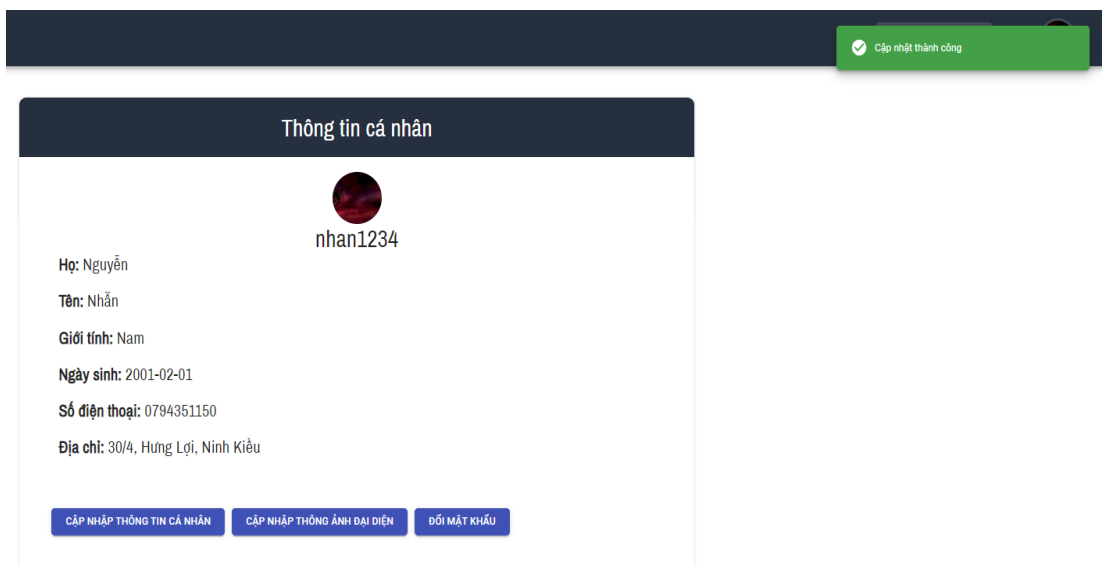
Giao diện trang đổi mật khi nhập mật khẩu cũ sai



The screenshot shows a web interface for changing a password. At the top, a red error banner displays "Sai mật khẩu!". Below this, a dark header contains the text "Thông tin cá nhân". The main content area features a white modal box titled "Đổi mật khẩu". Inside the modal, there are three input fields: "Mật khẩu cũ" (Old password), "Mật khẩu mới" (New password), and "Nhập lại mật khẩu" (Repeat new password). Each field has a toggle icon for visibility. At the bottom of the modal are two buttons: "HỦY" (Cancel) and "ĐỒNG Ý" (Agree).

Hình 51. Giao diện trang đổi mật khi nhập mật khẩu cũ sai

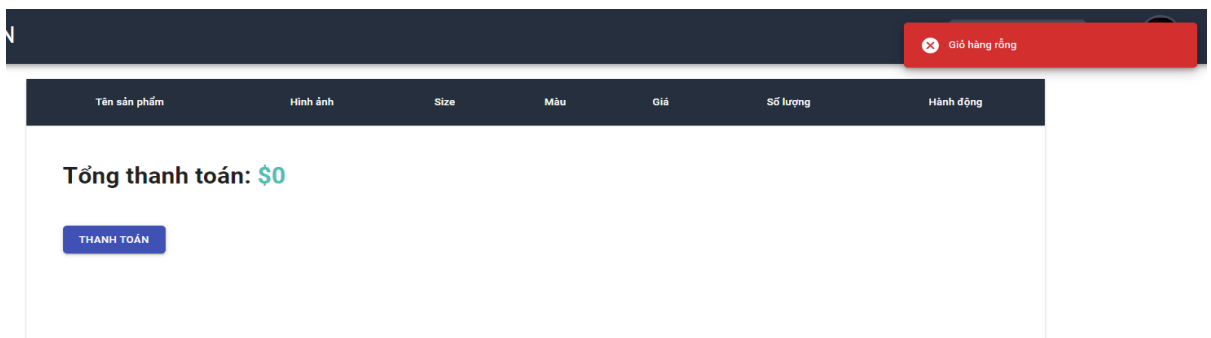
Giao diện khi cập nhật ảnh đại diện thành công



The screenshot displays a user profile page. A green success banner at the top right says "Cập nhật thành công". The profile section, titled "Thông tin cá nhân", shows a circular profile picture of a red dragon and the username "nhan1234". Below the profile picture, the following information is listed: "Họ: Nguyễn", "Tên: Nhân", "Giới tính: Nam", "Ngày sinh: 2001-02-01", "Số điện thoại: 0794351150", and "Địa chỉ: 30/4, Hưng Lợi, Ninh Kiều". At the bottom of the profile section are three buttons: "CẬP NHẬP THÔNG TIN CÁ NHÂN", "CẬP NHẬP THÔNG ẢNH ĐẠI DIỆN", and "ĐỔI MẬT KHẨU".

Hình 52. Giao diện khi cập nhật ảnh đại diện thành công

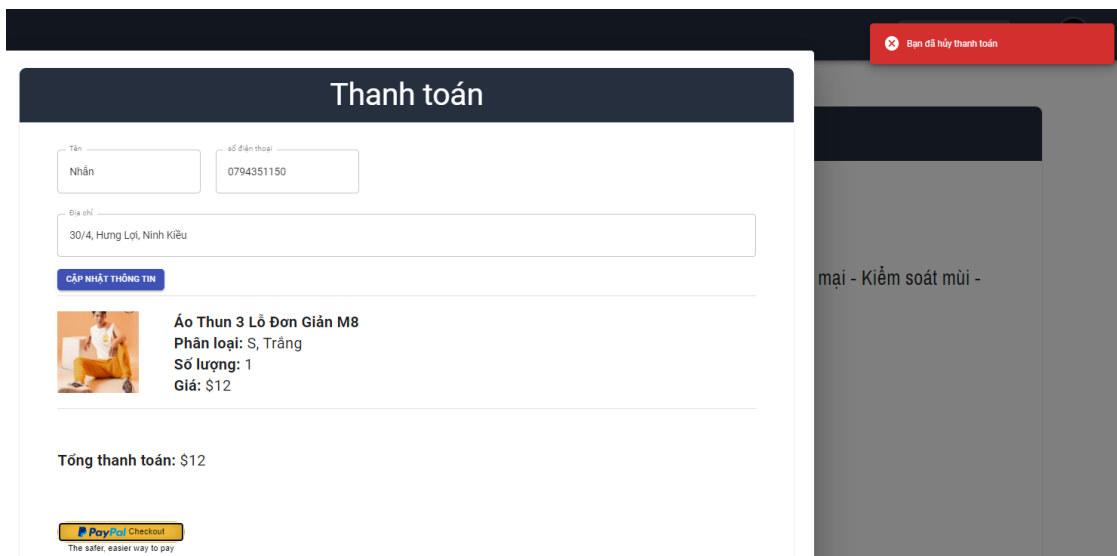
Giao diện thanh toán giỏ hàng khi giỏ hàng chưa có sản phẩm



The screenshot shows a shopping cart page. A red error banner at the top right displays "Giỏ hàng rỗng". Below the banner is a table with columns: "Tên sản phẩm", "Hình ảnh", "Size", "Màu", "Giá", "Số lượng", and "Hành động". The table is currently empty. Below the table, the text "Tổng thanh toán: \$0" is shown in green. At the bottom of the page is a blue button labeled "THANH TOÁN".

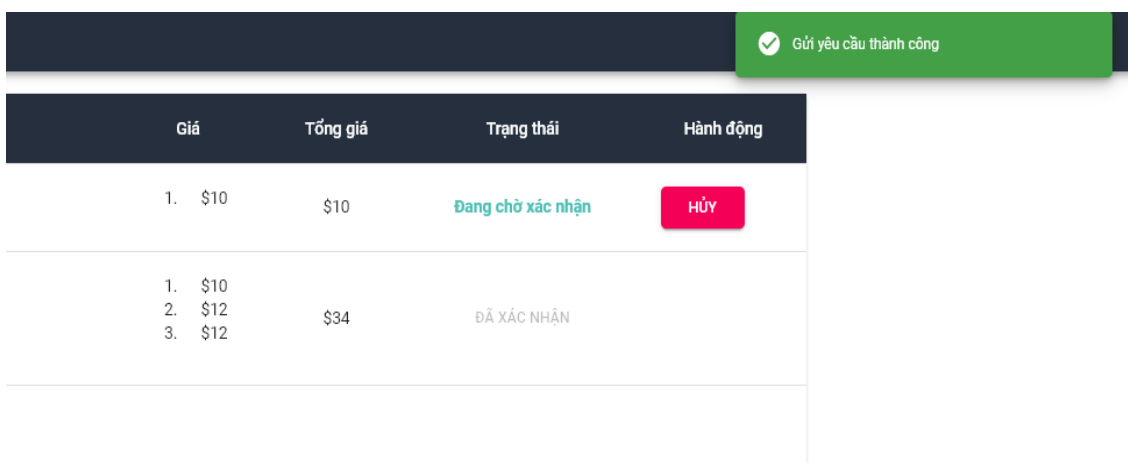
Hình 53. Giao diện thanh toán giỏ hàng khi giỏ hàng chưa có sản phẩm

Giao diện hủy thanh toán khi thanh toán sản phẩm



Hình 54. Giao diện hủy thanh toán khi thanh toán sản phẩm

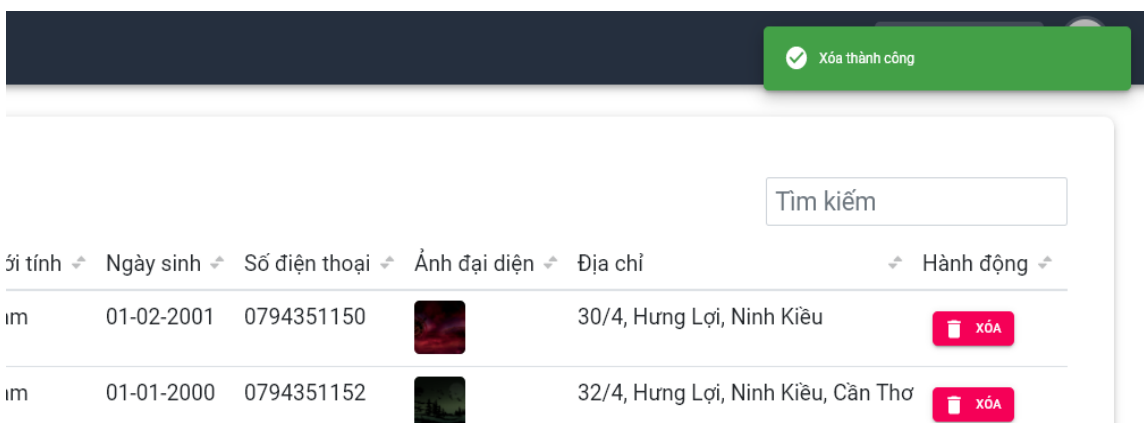
Giao diện hủy đơn hàng khi chủ cửa hàng chưa xác nhận



Hình 55. Giao diện hủy đơn hàng khi chủ cửa hàng chưa xác nhận

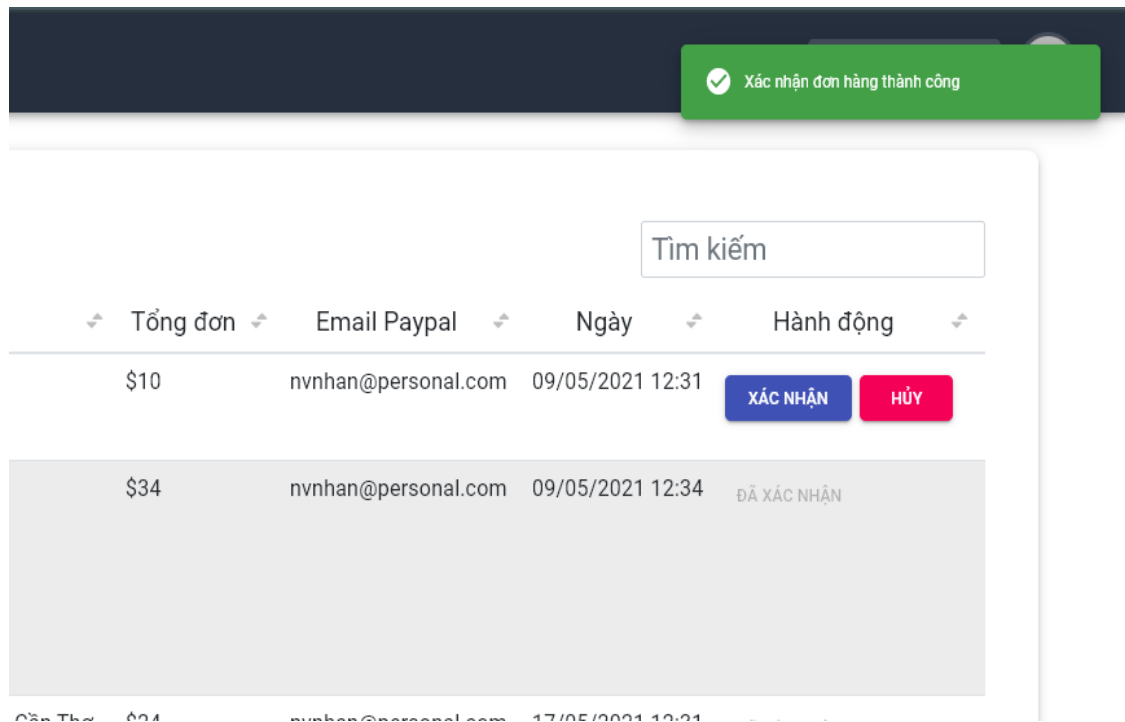
Một số hình ảnh kiểm thử trang quản lý

Giao diện khi xóa một người dùng



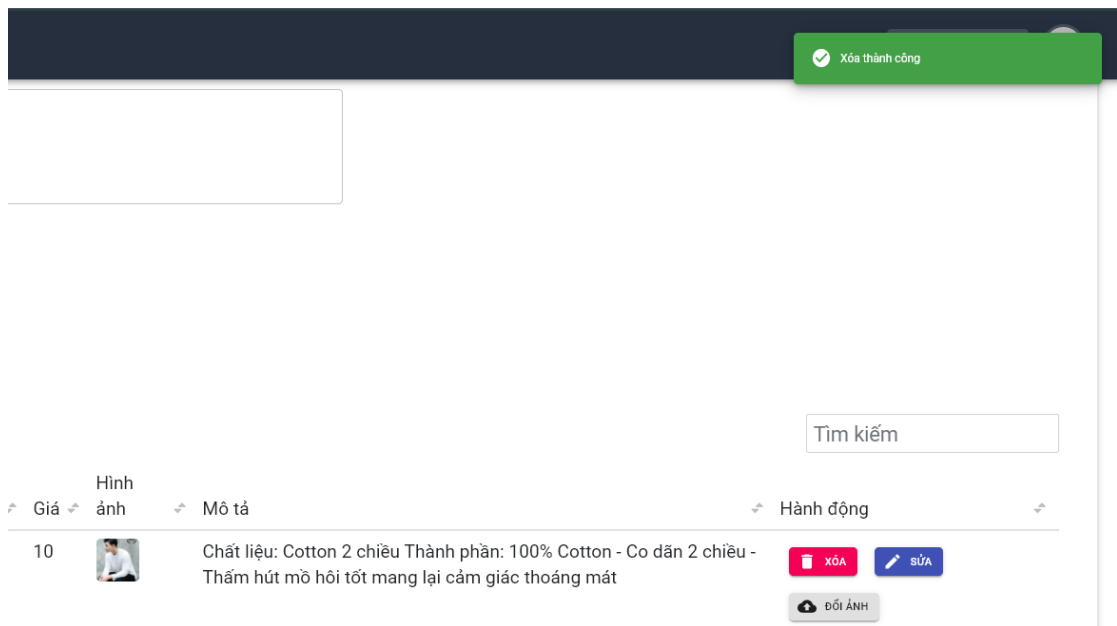
Hình 56. Giao diện khi xóa một người dùng

Giao diện khi xác nhận đơn hàng thành công



Hình 57. Giao diện khi xác nhận đơn hàng thành công

Giao diện khi xóa một sản phẩm thành công



Hình 58. Giao diện khi xóa một sản phẩm thành công

PHẦN 3: KẾT LUẬN

1. Kết quả đạt được

1.1. Kết quả

Sau thời gian nghiên cứu, tìm hiểu công nghệ, kiến thức chuyên môn để thực hiện đề tài thì giúp cho người thực hiện đề tài có cái nhìn tổng quan về quy trình phát triển phần mềm là như thế nào, làm sao để xây dựng được một hệ thống quản lý cửa hàng hoàn chỉnh từ các khâu phân tích, thiết kế đến hoàn thiện hệ thống.

Bên cạnh đó, còn học hỏi được rất nhiều những tiện ích khi sử dụng các công cụ StarUML, PowerDesigner, MySQL Workbench ... vào việc phát triển ứng dụng.

Xây dựng hoàn thiện hệ thống gồm 2 thành phần RESTful API server, Web client đạt mục tiêu đề ra ban đầu.

1.2. Hạn chế

Mất nhiều thời gian để học và làm quen với Framework mới, xây dựng giao diện đồng bộ.

Các thông tin của response cung cấp bởi REST API server còn hạn chế, khiến việc xử lý ở phía frontend phức tạp đôi chút.

2. Hướng phát triển

Phát triển thêm các chức năng cho nhóm người dùng sử dụng dễ dàng hơn.

Tăng cường hiệu năng, bảo mật cho ứng dụng khi đưa vào thực tế.

Phát triển chạy trên đa nền tảng (ứng dụng dành riêng cho nền tảng di động).

Hệ thống đưa ra thông báo cho khách hàng biết tình trạng đơn hàng sử dụng thời gian thực.

Hệ thống có thể đăng nhập bằng các mạng xã hội phổ biến khác.

Phát triển chatbot cho hệ thống.

Tài liệu tham khảo

- [1] ReactJS. [Online]. Có sẵn tại: <https://reactjs.org/docs/react-api.html> [truy cập ngày 04/02/2021].
- [2] NodeJS. [Online]. Có sẵn tại: <https://nodejs.org/dist/latest-v16.x/docs/api/> [truy cập ngày 04/02/2021].
- [3] Thư viện material ui. [Online]. Có sẵn tại: <https://material-ui.com/> [truy cập ngày 14/02/2021].
- [4] Thư viện react-router-dom. [Online]. Có sẵn tại: <https://reactrouter.com/web/example/basic> [truy cập ngày 14/02/2021].
- [5] MySQL. [Online]. Có sẵn tại: <https://dev.mysql.com/doc/> [truy cập ngày 14/02/2021].
- [6] Thư viện react-hook-form. [Online]. Có sẵn tại: <https://react-hook-form.com/v5/api> [truy cập ngày 28/02/2021].
- [7] Thư viện Axios. [Online]. Có sẵn tại: <https://www.npmjs.com/package/axios> [truy cập ngày 28/02/2021].
- [8] Thư viện redux-toolkit. [Online]. Có sẵn tại: <https://www.npmjs.com/package/axios> [truy cập ngày 28/02/2021].
- [9] ExpressJS. [Online]. Có sẵn tại: <https://expressjs.com/en/4x/api.html> [truy cập ngày 28/02/2021].
- [10] Jsonwebtoken. [Online]. Có sẵn tại: <https://expressjs.com/en/4x/api.html> [truy cập ngày 03/03/2021].
- [11] Multer. [Online]. Có sẵn tại: <https://www.npmjs.com/package/multer> [truy cập ngày 03/03/2021].
- [12] react-paypal-express-checkout. [Online]. Có sẵn tại: <https://www.npmjs.com/package/react-paypal-express-checkout> [truy cập ngày 03/05/2021].
- [13] Paypal. [Online]. Có sẵn tại: <https://developer.paypal.com/docs/> [truy cập ngày 03/05/2021].