

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN  
KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN



**BÁO CÁO ĐỒ ÁN CUỐI KÌ**  
**MÔN: KỸ THUẬT PHÁT TRIỂN HỆ THỐNG WEB**  
**(IE213.P21)**

**ĐỀ TÀI:**  
**XÂY DỰNG WEB BÁN CẦU LÔNG SMASH SHOP**

**Giảng viên hướng dẫn:** ThS. Võ Tấn Khoa

**Sinh viên thực hiện:**

Đoàn Nguyễn Lâm	22520736
Cao Thiên An	22520008
Bùi Thanh Phong	22521082

**Tp. Hồ Chí Minh, 02/2025**

## NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

....., ngày.....tháng.....năm 2025

**Người nhận xét**  
(Ký tên và ghi rõ họ tên)

## MỤC LỤC

MỤC LỤC .....	3
MỤC LỤC ẢNH.....	5
MỤC LỤC BẢNG .....	7
BẢNG PHÂN CÔNG .....	8
LỜI MỞ ĐẦU.....	11
CHƯƠNG 1. Tổng quan .....	12
1.1. Lí do chọn đề tài .....	12
1.2. Mô tả.....	12
1.3. Phạm vi đề tài.....	12
1.4. Công cụ và công nghệ sử dụng.....	13
1.4.1. Công cụ .....	13
1.4.2. Công nghệ sử dụng .....	14
CHƯƠNG 2. Đặc tả và phân tích hệ thống.....	15
2.1. Đặc tả .....	15
2.1.1. Yêu cầu chức năng.....	15
2.1.2. Yêu cầu phi chức năng.....	15
2.2. Phân tích hệ thống.....	16
2.2.1. Sơ đồ Usecase tổng quát .....	16
2.2.2. Danh sách các Actor.....	17
2.2.3. Danh sách các Use Case chính.....	17
CHƯƠNG 3. Thiết kế hệ thống .....	24
3.1. Kiến trúc hệ thống.....	24
3.1.1. Frontend .....	24
3.1.2. Backend.....	24
3.2. Thiết kế cơ sở dữ liệu.....	24
3.3. Thiết kế giao diện.....	27
3.3.1. Sơ đồ liên kết các màn hình.....	27
3.3.2. Danh sách các màn hình .....	27
CHƯƠNG 4. Hiện thực.....	40

4.1.	Backend.....	40
4.1.1.	Kiến trúc và công nghệ sử dụng .....	40
4.1.2.	Thiết kế dữ liệu – Các entity chính trong MongoDB .....	41
4.1.3.	Cơ chế xác thực JWT .....	41
4.1.4.	Tích hợp thanh toán trực tuyến qua cổng VNPAY .....	42
4.1.5.	Các module và chức năng chính .....	44
4.1.6.	Định nghĩa REST API điển hình.....	45
4.2.	Frontend .....	46
4.2.1.	Kiến trúc và công nghệ sử dụng .....	46
4.2.2.	Quản lý trạng thái với Redux.....	47
CHƯƠNG 5.	Triển khai và kiểm thử .....	48
5.1.	Triển khai .....	48
5.1.1.	Triển khai Backend .....	48
5.1.2.	Triển khai Frontend.....	51
5.1.3.	Cấu hình CI/CD .....	54
5.1.4.	Cấu hình Dokploy để tự động cập nhật Docker image từ Docker Hub .....	54
5.2.	Kiểm thử .....	55
CHƯƠNG 6.	Kết luận.....	57
6.1.	Ưu điểm .....	57
6.2.	Nhược điểm.....	57
6.3.	Hướng phát triển .....	57

## MỤC LỤC ẢNH

Hình 1: Sơ đồ use case tổng quát.....	17
Hình 2: Use case diagram UC1 – Đăng nhập và xác thực JWT .....	18
Hình 3: Sequence Diagram UC1 – Đăng nhập và xác thực JWT .....	18
Hình 4: Use Case Diagram UC2 – Thanh toán online.....	19
Hình 5: Sequence Diagram UC2 – Thanh toán online.....	20
Hình 6: Use Case Diagram UC3 – Mua hàng.....	21
Hình 7: Sequence Diagram UC3 – Mua hàng .....	22
Hình 8: Use case Diagram UC4 – Xử lý hết hạn Token .....	22
Hình 9: Sequence Diagram UC4 – Xử lý hết hạn Token.....	23
Hình 10: Sơ đồ tổng quát kiến trúc hệ thống .....	24
Hình 11: Class diagram.....	25
Hình 12: Sơ đồ liên kết các màn hình trang người dùng .....	27
Hình 13: Sơ đồ liên kết các màn hình .....	27
Hình 14: Màn hình đăng ký của người dùng .....	29
Hình 15: Màn hình đăng nhập của người dùng .....	29
Hình 16: Màn hình trang chủ - Header, slider và danh mục .....	30
Hình 17: Màn hình trang chủ - sản phẩm nổi bật .....	30
Hình 18: Màn hình trang chủ - Footer .....	31
Hình 19: Màn hình sản phẩm.....	31
Hình 20: Màn hình chi tiết sản phẩm.....	32
Hình 21: Màn hình giỏ hàng .....	32
Hình 22: Màn hình đặt hàng .....	33
Hình 23: Màn hình thông tin cá nhân .....	33
Hình 24: Màn hình lịch sử đơn hàng .....	34
Hình 25: Màn hình đăng nhập Admin.....	34
Hình 26: Màn hình trang thống kê – trên.....	35
Hình 27: Màn hình trang thống kê – dưới .....	35
Hình 28: Màn hình xem danh sách sản phẩm .....	36
Hình 29: Màn hình chi tiết sản phẩm dành cho Admin .....	36

Hình 30: Màn hình thêm sản phẩm - trên .....	37
Hình 31: Màn hình thêm sản phẩm – dưới .....	37
Hình 32: Màn hình sửa sản phẩm .....	38
Hình 33: Màn hình quản lý đơn hàng .....	38
Hình 34: Màn hình trang quản lý chi tiết đơn hàng .....	39
Hình 35: Sequence Diagram mô tả quy trình thanh toán trực tuyến .....	44
Hình 36: Nội dung .dockerignore .....	48
Hình 37: Tạo Project trên Dokploy .....	49
Hình 38: Tạo service mới trên dokploy .....	49
Hình 39: Cấu hình Provider cho Dokploy .....	50
Hình 40: Cấu hình biến môi trường cho Dokploy .....	50
Hình 41: Thêm domain cho dokploy .....	51
Hình 42: Deploy backend .....	51
Hình 43: Cấu hình deploy frontend lên vercel .....	53
Hình 44: Webhook URL của Dokploy .....	55
Hình 45: Thêm webhook vào Docker Hub .....	55

## MỤC LỤC BẢNG

Bảng 1: Danh sách các màn hình.....	28
Bảng 2: Các entity chính trong MongoDB .....	41
Bảng 3: Định nghĩa REST API điển hình .....	46
Bảng 4: Danh sách các ca kiểm thử chính .....	56

## BẢNG PHÂN CÔNG

MSSV	Họ và tên	Phân công công việc	Đóng góp	Đánh giá hoàn thành
22520736	Đoàn Nguyễn Lâm	<p><b>Về ý tưởng:</b></p> <ul style="list-style-type: none"> <li>- Tham gia thảo luận chọn đề tài</li> </ul> <p><b>Về viết báo cáo:</b></p> <ul style="list-style-type: none"> <li>- Chương 1: Mục 1.4</li> <li>- Chương 3: Mục 3.1</li> <li>- Chương 5: Mục 5.1</li> <li>- Chương 6</li> </ul> <p><b>Về thiết kế:</b></p> <ul style="list-style-type: none"> <li>- Đóng góp ý kiến chỉnh sửa các thành phần giao diện</li> </ul> <p><b>Về code:</b></p> <ul style="list-style-type: none"> <li>- Viết các api quản lí sản phẩm như: thêm/xóa/chỉnh sửa/thông tin sản phẩm</li> <li>- Viết api liên quan đến user: xem lịch sử mua hàng</li> <li>- Thực hiện phân trang, lọc, sắp xếp sản phẩm</li> <li>- Deploy website</li> <li>- Thực hiện CI/CD</li> </ul> <p><b>Đóng góp khác:</b></p> <ul style="list-style-type: none"> <li>- Phân chia công việc và theo dõi, hỗ trợ nhóm</li> <li>- Quay video demo</li> <li>- Tham gia quay video báo cáo</li> </ul>	33%	<p>Tích cực đóng góp xây dựng đồ án.</p> <p>Hoàn thành tốt công việc được giao.</p>



2251082	Bùi Thanh Phong	<p><b>Về ý tưởng:</b></p> <ul style="list-style-type: none"> <li>- Tham gia thảo luận chọn đề tài</li> </ul> <p><b>Về viết báo cáo:</b></p> <ul style="list-style-type: none"> <li>- Chương 2</li> <li>- Chương 3: Mục 3.2</li> <li>- Chương 4: Mục 4.1</li> <li>- Chương 5: Mục 5.2</li> </ul> <p><b>Về thiết kế:</b></p> <ul style="list-style-type: none"> <li>- Thiết kế giao diện trang user.</li> </ul> <p><b>Về code:</b></p> <ul style="list-style-type: none"> <li>- Thực hiện Schema dữ liệu</li> <li>- Thực hiện các tính năng xác thực như đăng nhập, đăng kí, đăng xuất.</li> <li>- Thực hiện chức năng thêm sản phẩm vào giỏ hàng, thanh toán</li> <li>- Nối Api trang đăng nhập, đăng xuất, đăng kí tài khoản</li> </ul> <p><b>Đóng góp khác:</b></p> <ul style="list-style-type: none"> <li>- Thực hiện kiểm thử bằng Mocha</li> </ul>	33%	<p>Hoàn thành công việc được giao</p> <p>Tích cực đóng góp xây dựng đồ án</p>
22520008	Cao Thiên An	<p><b>Về ý tưởng:</b></p> <ul style="list-style-type: none"> <li>- Tham gia thảo luận chọn đề tài</li> </ul> <p><b>Về viết báo cáo:</b></p> <ul style="list-style-type: none"> <li>- Chương 1: Mục 1.1, 1.2, 1.3</li> <li>- Chương 3: Mục 3.3</li> <li>- Chương 4: Mục 4.2</li> </ul>	34%	<p>Tích cực tham gia thảo luận, nhiệt tình hỗ trợ các thành viên.</p> <p>Hoàn thành tốt nhiệm vụ được giao.</p>

		<p><b>Về thiết kế:</b></p> <ul style="list-style-type: none"><li>- Thiết kế giao diện trang admin.</li></ul> <p><b>Về code:</b></p> <ul style="list-style-type: none"><li>- Code frontend trang admin và user.</li><li>- Nối API trang frontend và backend</li><li>- Hỗ trợ sửa lỗi liên quan đến đăng nhập</li><li>- Thực hiện responsive frontend</li></ul> <p><b>Đóng góp khác:</b></p> <ul style="list-style-type: none"><li>- Làm slide báo cáo</li><li>- Tham gia quay video báo cáo</li><li>- Chỉnh sửa video demo và video báo cáo</li></ul>		
--	--	--	--	--

## LỜI MỞ ĐẦU

Trong thời đại công nghệ số phát triển mạnh mẽ, thương mại điện tử ngày càng đóng vai trò quan trọng trong việc kết nối sản phẩm với người tiêu dùng. Với niềm đam mê bộ môn cầu lông – một môn thể thao ngày càng phổ biến tại Việt Nam – nhóm chúng em đã lựa chọn đề tài "Xây dựng Web bán cầu lông Smash Shop" như một cách để kết hợp giữa sở thích cá nhân và kiến thức chuyên môn về phát triển hệ thống web.

Thông qua đề tài này, nhóm có cơ hội vận dụng các kiến thức đã học về lập trình web, đặc biệt là công nghệ MERN Stack (MongoDB, Express.js, React.js, Node.js), để xây dựng một hệ thống bán hàng trực tuyến thực tế, hướng đến trải nghiệm người dùng thân thiện, tiện lợi, và dễ sử dụng. Dự án không chỉ giúp củng cố kỹ năng kỹ thuật, mà còn rèn luyện khả năng làm việc nhóm, tư duy hệ thống và giải quyết vấn đề.

Mặc dù trong quá trình thực hiện không tránh khỏi những khó khăn và hạn chế, nhóm đã nỗ lực hoàn thành đề tài theo đúng kế hoạch. Chúng em xin chân thành cảm ơn thầy Võ Tấn Khoa, giảng viên phụ trách môn học, cùng nhà trường đã tạo điều kiện để nhóm có cơ hội thực hiện và trình bày đồ án này.

Nhóm cũng xin gửi lời cảm ơn đến tất cả các thành viên vì tinh thần làm việc nghiêm túc, hỗ trợ lẫn nhau và cùng nhau hoàn thành tốt nhiệm vụ được giao.

Tuy đồ án vẫn còn những điểm chưa hoàn thiện do giới hạn về thời gian và kinh nghiệm, nhưng chúng em tin rằng đây là nền tảng quan trọng để tiếp tục học hỏi và phát triển hơn nữa trong tương lai.

# CHƯƠNG 1. Tổng quan

## 1.1. Lí do chọn đề tài

Trong bối cảnh thương mại điện tử đang phát triển mạnh mẽ, việc xây dựng các hệ thống bán hàng trực tuyến trở nên ngày càng phổ biến và cần thiết. Ngành cầu lông tại Việt Nam cũng đang ngày một phát triển, thu hút nhiều người chơi từ phong trào đến chuyên nghiệp. Tuy nhiên, số lượng website chuyên biệt về sản phẩm cầu lông còn khá hạn chế và chưa tối ưu trải nghiệm người dùng.

Việc lựa chọn đề tài “Web bán cầu lông Smash Shop” nhằm mục tiêu:

- Áp dụng kiến thức về lập trình web với MERN Stack (MongoDB, Express.js, React.js, Node.js) để xây dựng một hệ thống thực tế.
- Tạo ra một nền tảng giúp người dùng dễ dàng tìm kiếm, xem và đặt mua các sản phẩm cầu lông như vợt, giày, phụ kiện,...
- Nâng cao kỹ năng thiết kế hệ thống, xử lý dữ liệu, xây dựng giao diện và quản lý người dùng (User & Admin).

## 1.2. Mô tả

Hệ thống Smash Shop là một website thương mại điện tử đơn giản, chuyên cung cấp các sản phẩm phục vụ cho người chơi cầu lông. Website được phát triển theo mô hình client-server với công nghệ MERN Stack.

Hệ thống bao gồm 2 phần chính:

- Trang người dùng (User): Cho phép người dùng xem danh sách sản phẩm, xem chi tiết sản phẩm, thêm vào giỏ hàng, và tiến hành đặt mua.
- Trang quản trị (Admin): Cho phép quản lý sản phẩm (thêm/sửa/xóa), theo dõi đơn hàng, và cập nhật thông tin hàng hóa.

Các chức năng cơ bản đã được hoàn thiện bao gồm:

- Hiện thị danh sách sản phẩm.
- Giỏ hàng và tính tổng đơn hàng.
- Đặt mua sản phẩm.
- Hệ thống quản trị đơn giản dành cho admin.

## 1.3. Phạm vi đề tài

Phạm vi thực hiện của đề tài trong khuôn khổ môn học bao gồm:

- Xây dựng giao diện người dùng thân thiện bằng React.js.
- Thiết kế và triển khai backend API sử dụng Node.js và Express.js.
- Lưu trữ và truy xuất dữ liệu từ MongoDB.
- Phân quyền giữa người dùng thường và admin.
- Thực hiện các chức năng cơ bản: xem sản phẩm, thêm vào giỏ hàng, đặt hàng, quản lý sản phẩm.
- Thanh toán trực tuyến qua cổng thanh toán.
- Responsive design trên mobile (chỉ tối ưu cơ bản).

Không bao gồm trong phạm vi hiện tại:

- Tích hợp hệ thống vận chuyển thực tế.
- Bảo mật nâng cao (mã hóa JWT, xác thực 2 bước,...).

## 1.4. Công cụ và công nghệ sử dụng

### 1.4.1. Công cụ

- **Notion**: là một nền tảng quản lý công việc và ghi chú mạnh mẽ, hỗ trợ tổ chức và theo dõi tiến độ dự án. Nhờ có notion đó, các thành viên trong nhóm có nắm được các công việc cần thực hiện và tiến độ của dự án để thực hiện dự án một cách tốt nhất và đúng thời hạn.
- **Github**: nền tảng lưu trữ và quản lý dự án thông qua các Repository, code được các thành viên sử dụng chung, cập nhật liên tục cùng với việc dễ dàng xem lại lịch sử chỉnh sửa của dự án.
- **Cloudinary**: nền tảng quản lý phương tiện (media management) cho phép tải lên, lưu trữ, tối ưu hóa và phân phối hình ảnh, video trên website hoặc ứng dụng.
- **Dokploy**: là nền tảng hỗ trợ triển khai ứng dụng web một cách nhanh chóng và dễ dàng lên các máy chủ hoặc dịch vụ cloud.
- **Docker hub**: là kho lưu trữ container trực tuyến cho phép người dùng tải lên, chia sẻ và quản lý các Docker image, hỗ trợ nhóm phát triển sử dụng chung môi trường làm việc, từ đó đảm bảo tính nhất quán trong quá trình phát triển và triển khai ứng dụng.

#### 1.4.2. Công nghệ sử dụng

- **MongoDB:** là hệ quản trị cơ sở dữ liệu NoSQL, lưu trữ dữ liệu dưới dạng tài liệu (document) theo cấu trúc BSON tương tự JSON, giúp dễ dàng mở rộng, linh hoạt trong việc lưu trữ và truy xuất dữ liệu, đặc biệt phù hợp với các ứng dụng web có cấu trúc dữ liệu không cố định.
- **Express:** là một framework nhẹ và linh hoạt dành cho Node.js, hỗ trợ xây dựng các ứng dụng web và API một cách nhanh chóng, với hệ thống routing mạnh mẽ và khả năng mở rộng cao, thường được sử dụng trong các dự án web backend hiện đại.
- **React:** là thư viện JavaScript được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng một cách linh hoạt và hiệu quả, cho phép tạo ra các thành phần (component) tái sử dụng
- **Node.js:** là một môi trường chạy JavaScript bên ngoài trình duyệt, cho phép xây dựng các ứng dụng server-side nhanh chóng và hiệu quả. Với khả năng xử lý I/O không đồng bộ và mô hình sự kiện, Node.js giúp các ứng dụng web mở rộng và xử lý nhiều yêu cầu đồng thời mà không làm giảm hiệu suất.

## CHƯƠNG 2. Đặc tả và phân tích hệ thống

### 2.1. Đặc tả

#### 2.1.1. Yêu cầu chức năng

##### 1) Quản lý người dùng

- a) **Đăng ký:** Người dùng mới có thể đăng ký tài khoản bằng email và mật khẩu. Email phải hợp lệ và chưa được sử dụng.
- b) **Đăng nhập/Xác thực:** Hỗ trợ đăng nhập bằng JWT. Khi đăng nhập thành công, server cấp accessToken (hạn 15 phút) và refreshToken (hạn 7 ngày).
- c) **Gia hạn token tự động:** Khi accessToken hết hạn, client gửi refreshToken để lấy lại accessToken mới mà không cần đăng nhập lại.

##### 2) Quản lý danh mục sản phẩm

- a) Hiển thị **Danh sách sản phẩm** kèm hình ảnh, tên, giá, số lượng tồn kho.
- b) Hỗ trợ **tìm kiếm** (theo tên sản phẩm) và **lọc** (theo danh mục, thương hiệu, mức giá).
- c) Hiển thị **Chi tiết sản phẩm:** mô tả, thông số kỹ thuật, hình ảnh, đánh giá.

##### 3) Quản lý giỏ hàng

- a) **Thêm sản phẩm:** Chọn số lượng và thêm vào giỏ.
- b) **Chỉnh sửa:** Tăng/giảm số lượng hoặc xóa sản phẩm khỏi giỏ.
- c) **Hiển thị:** Tổng số lượng loại sản phẩm, tổng tiền, mã giảm giá.

##### 4) Quy trình đặt hàng & thanh toán

- a) **Tạo đơn hàng:** Người dùng xác nhận giỏ, nhập địa chỉ giao hàng, chọn phương thức thanh toán.
- b) **Thanh toán COD:** Chọn phương thức COD, hoàn tất đặt hàng, admin xác nhận và giao hàng.

##### 5) Thanh toán online (VNPAY):

- a) Client gọi API tạo URL thanh toán.
- b) Redirect đến VNPAY, người dùng hoàn tất thanh toán.
- c) VNPAY callback về hệ thống tại `/api/v1/vnpay/vnpay_return` để xác minh chữ ký.
- d) Redirect về frontend hiển thị kết quả.

##### 6) Quản trị (Admin)

- a) **Quản lý sản phẩm:** Thêm, sửa, xóa sản phẩm; quản lý hình ảnh, tồn kho.
- b) **Quản lý đơn hàng:** Xem chi tiết, thay đổi trạng thái (Chờ xử lý, Đang giao, Hoàn thành, Hủy).
- c) **Thống kê:** Doanh thu, số đơn theo ngày/tháng, sản phẩm bán chạy.

#### 2.1.2. Yêu cầu phi chức năng

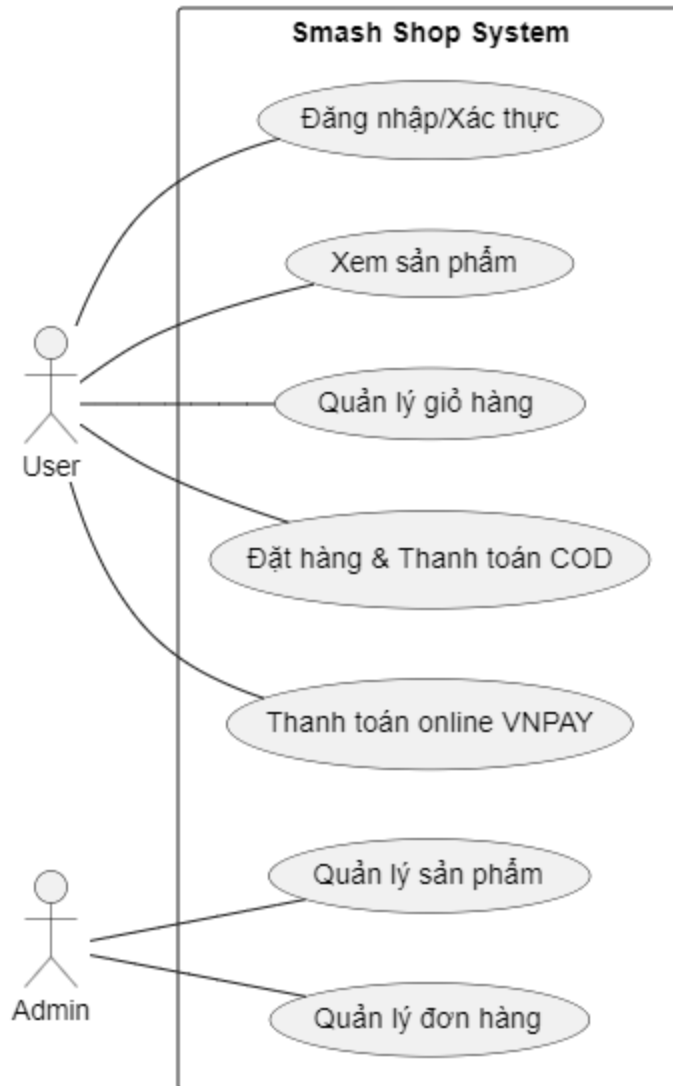
##### 1) Bảo mật:

- a) Sử dụng JWT cho xác thực, HTTPS cho kết nối client-server.
  - b) Refresh token chỉ gửi qua cookie httpOnly.
  - c) CSRF protection cho các endpoint quan trọng.
- 2) **Hiệu năng & mở rộng:**
- a) Sử dụng Redux Toolkit để quản lý state, tối ưu re-render.
  - b) Backend tối ưu truy vấn MongoDB, sử dụng indexing cho search/filter.
  - c) Hỗ trợ đồng thời 1.000 kết nối bằng việc cache, pagination và lazy loading.
- 3) **Độ tin cậy & chịu lỗi:**
- a) Retry logic cho thanh toán online nếu mất kết nối.
  - b) Logging lỗi, audit user action.
  - c) Giám sát health check endpoint /health.
- 4) **Trải nghiệm người dùng:**
- a) Responsive UI, tải trang chính < 200ms.
  - b) Thông báo (toast) cho mọi thao tác: thêm giỏ, thanh toán, lỗi.

## 2.2. Phân tích hệ thống

### 2.2.1. Sơ đồ Usecase tổng quát





*Hình 1: Sơ đồ use case tổng quát*

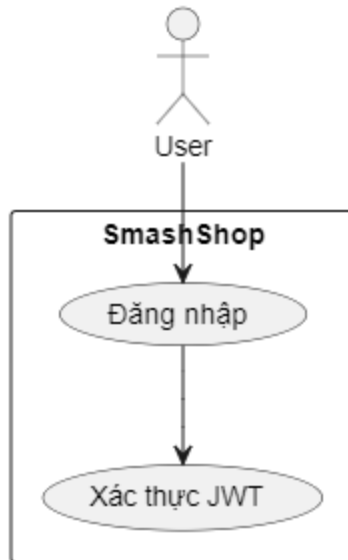
## 2.2.2. Danh sách các Actor

- **User:** Khách hàng truy cập web, thực hiện mua hàng, thanh toán.
- **Admin:** Quản trị viên hệ thống, quản lý sản phẩm và đơn hàng.

## 2.2.3. Danh sách các Use Case chính

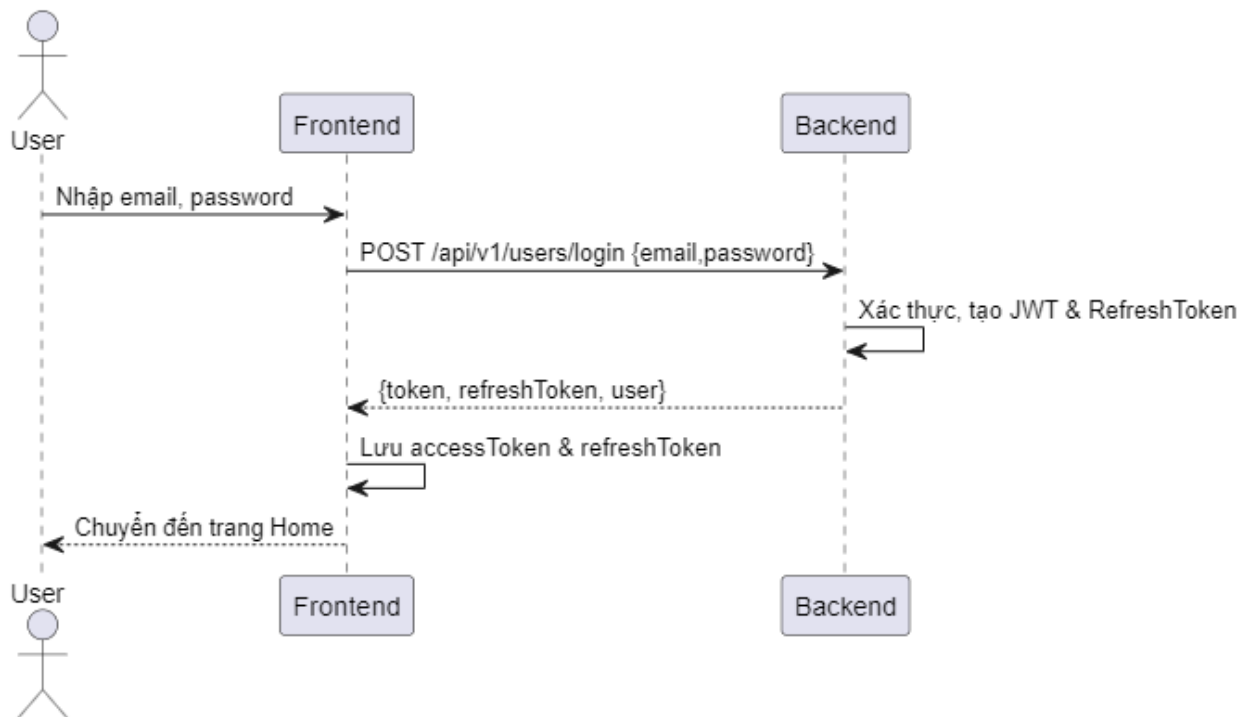
### 2.2.3.1. UC1 – Đăng nhập & Xác thực JWT

#### a) Use Case Diagram



Hình 2: Use case diagram UC1 – Đăng nhập và xác thực JWT

b) Sequence Diagram



Hình 3: Sequence Diagram UC1 – Đăng nhập và xác thực JWT

c) Mô tả chi tiết

Tiền điều kiện: Người dùng đã đăng ký tài khoản.

Luồng chính:

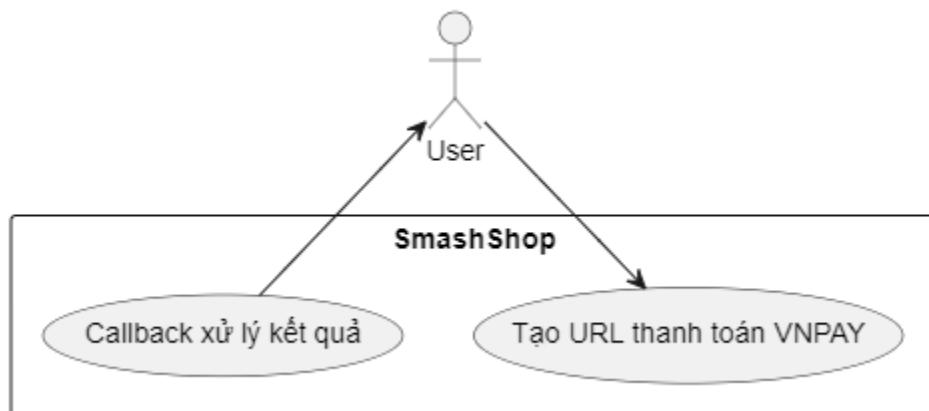
- 1) User nhập thông tin email và password trên form.
- 2) Frontend gửi yêu cầu POST đến `/api/v1/users/login`.
- 3) Backend xác thực thông tin, tạo `accessToken` (JWT) và `refreshToken`.
- 4) Backend trả về response chứa token và thông tin user.
- 5) Frontend lưu `accessToken` vào `LocalStorage`, `refreshToken` vào cookie `httpOnly`.
- 6) Frontend điều hướng User về trang chủ với trạng thái đã đăng nhập.

Luồng phụ:

- 1a) Sai thông tin: Backend trả lỗi 401, Frontend hiển thị thông báo lỗi.

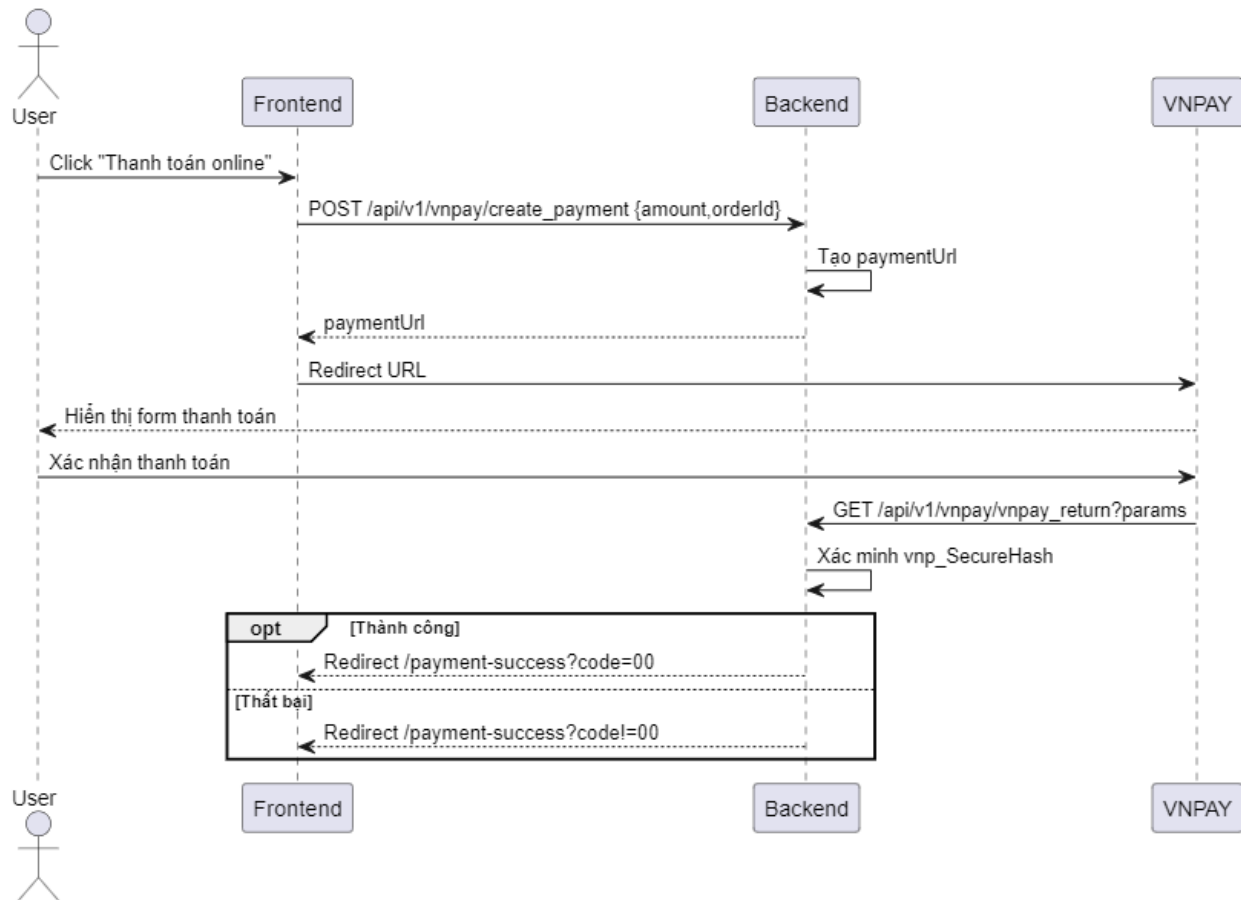
#### 2.2.3.2. UC2 – Thanh toán online (VNPAY)

a) Use Case Diagram



Hình 4: Use Case Diagram UC2 – Thanh toán online

b) Sequence Diagram



Hình 5: Sequence Diagram UC2 – Thanh toán online

### c) Mô tả chi tiết

Tiền điều kiện: User đã đăng nhập, giỏ hàng không rỗng.

Luồng chính:

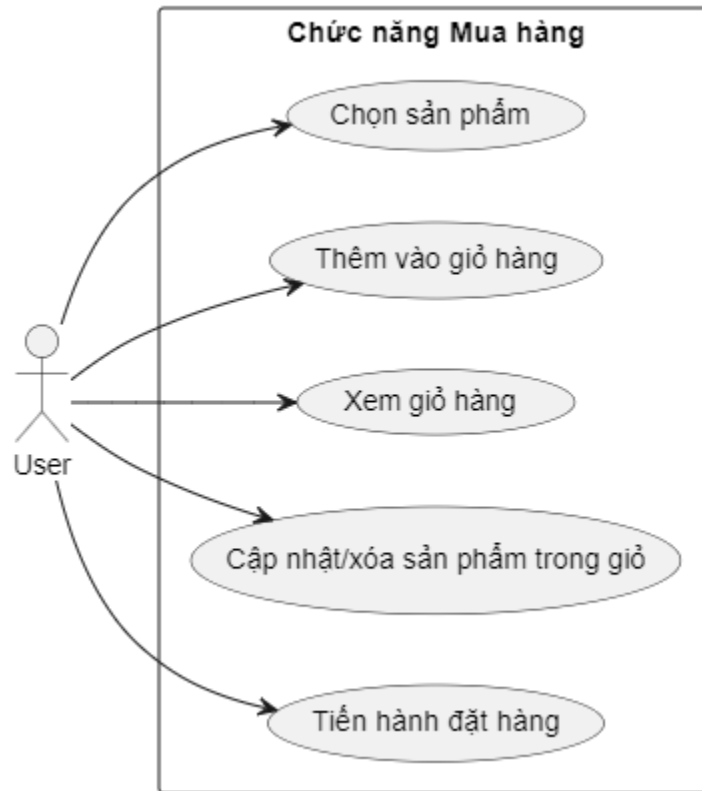
- 1) User chọn "Thanh toán online" tại trang cart.
- 2) Frontend gọi API tạo URL thanh toán (`/create_payment`).
- 3) Backend xây dựng `paymentUrl` và gửi về.
- 4) Frontend redirect User sang VNPAY.
- 5) User nhập thông tin thanh toán trên VNPAY và xác nhận.
- 6) VNPAY gọi callback về `/vnpay_return` cùng tham số trả về.
- 7) Backend kiểm tra chữ ký, xác định kết quả.
- 8) Backend redirect User về `frontend/payment-success` với mã kết quả.

Luồng phụ:

2a) Hủy thanh toán: VNPAY trả mã != '00', backend redirect về thất bại.

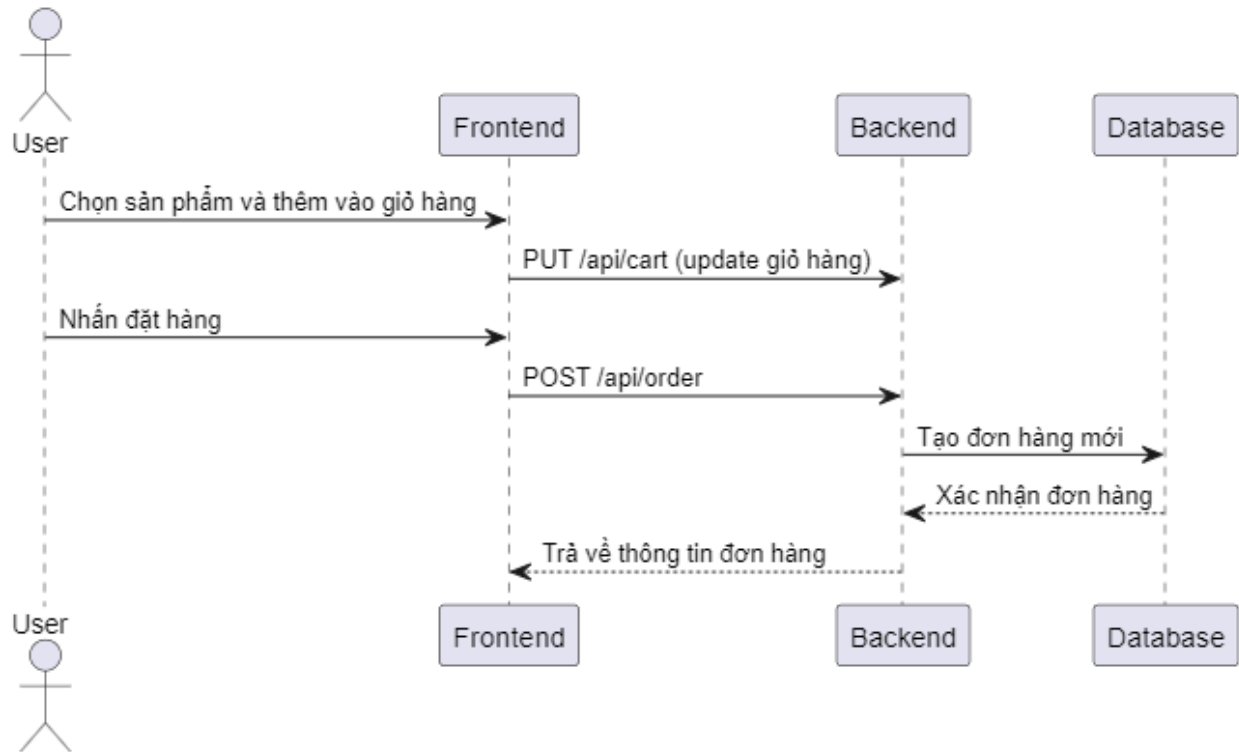
### 2.2.3.3. UC3 – Mua hàng

a) Use Case Diagram



Hình 6: Use Case Diagram UC3 – Mua hàng

b) Sequence Diagram



Hình 7: Sequence Diagram UC3 – Mua hàng

c) Mô tả chi tiết

Luồng chính:

- 1) Người dùng chọn sản phẩm và thêm vào giỏ hàng.
- 2) Người dùng tiến hành đặt hàng.
- 3) Hệ thống tạo đơn hàng mới trong database và phản hồi về thông tin đơn hàng đã đặt.

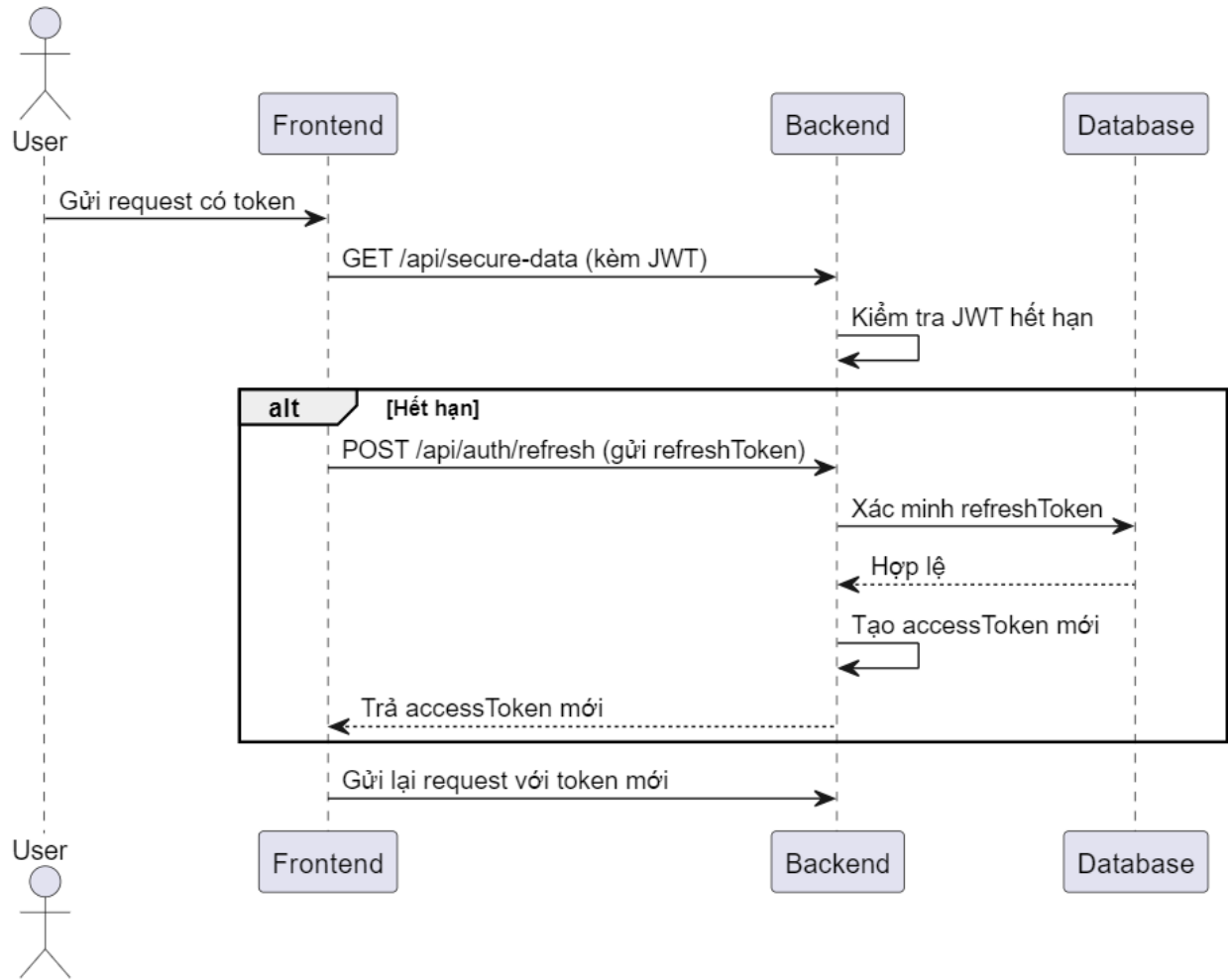
2.2.3.4. Use Case 4: Xử lý hết hạn Token (Refresh Token)

a) Usecase Diagram



Hình 8: Use case Diagram UC4 – Xử lý hết hạn Token

b) Sequence Diagram



Hình 9: Sequence Diagram UC4 – Xử lý hết hạn Token

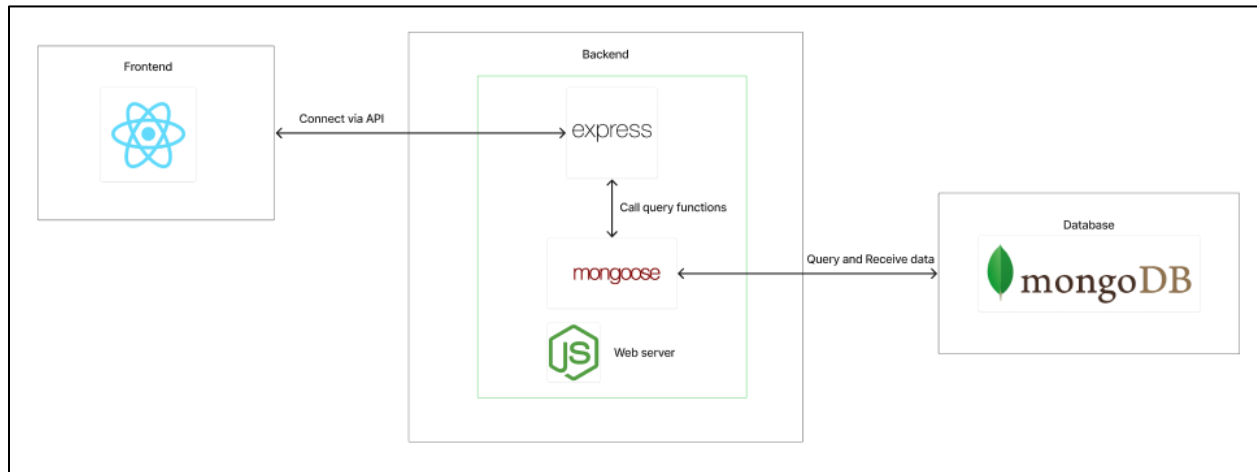
c) Mô tả chi tiết

Luồng chính:

- 1) Người dùng gửi request có JWT.
- 2) Backend phát hiện JWT hết hạn → frontend gửi refresh token.
- 3) Backend xác minh và cấp JWT mới → frontend tiếp tục request ban đầu.

## CHƯƠNG 3. Thiết kế hệ thống

### 3.1. Kiến trúc hệ thống



Hình 10: Sơ đồ tổng quát kiến trúc hệ thống

Kiến trúc hệ thống được chia thành hai phần: Frontend và Backend, kết nối qua API, sử dụng MongoDB làm hệ quản trị cơ sở dữ liệu

#### 3.1.1. Frontend

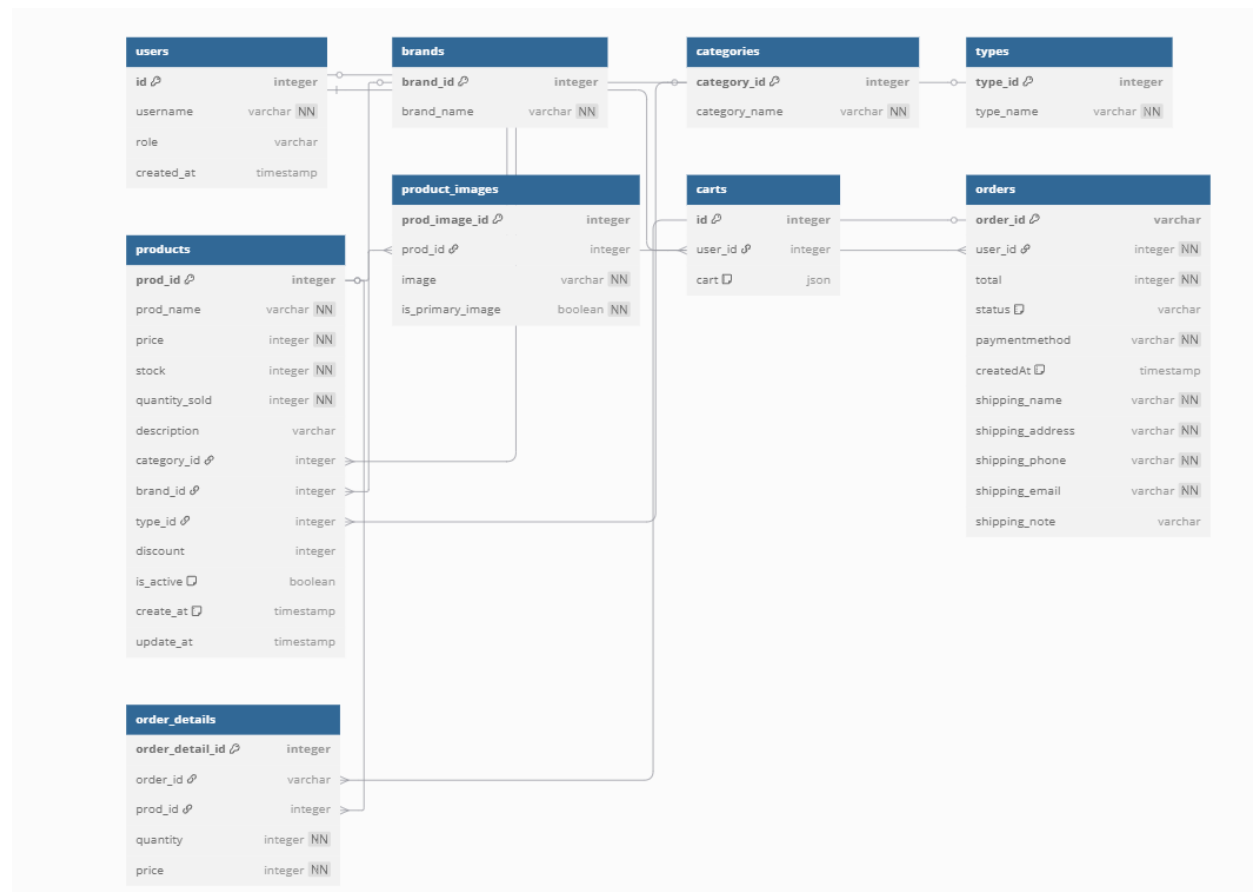
Giao diện được phát triển bằng ReactJs, kết hợp các thư viện như React Router Dom, Font Awesome, Recharts, Day.js, React Markdown,... để hỗ trợ thiết kế. Redux được dùng để quản lý trạng thái. Thư viện Axios được tích hợp để thực hiện các yêu cầu HTTP (GET, POST, PUT, DELETE) tương tác với Backend. Khi người dùng thao tác trên giao diện, React sẽ gửi các request (thông qua REST API) đến server xử lý.

#### 3.1.2. Backend

- Node.js đóng vai trò là web server, xử lý các request từ Frontend.
- Express là framework chính được sử dụng để xây dựng các API. Nó hỗ trợ định tuyến, xử lý middleware và các phương thức HTTP, giúp việc phát triển API trở nên linh hoạt và hiệu quả.
- Mongoose là thư viện giúp kết nối và thao tác với MongoDB bằng cú pháp JavaScript đơn giản. Express sẽ gọi các hàm truy vấn từ Mongoose để thực hiện việc đọc/ghi dữ liệu từ cơ sở dữ liệu.

### 3.2. Thiết kế cơ sở dữ liệu





Hình 11: Class diagram

## 1. users – Thông tin người dùng

- id: Khóa chính.
- username: Tên đăng nhập.
- role: Vai trò người dùng (admin, customer, v.v.).
- created\_at: Ngày tạo tài khoản.

## 2. products – Thông tin sản phẩm

- prod\_id: Khóa chính.
- prod\_name, price, stock, quantity\_sold, description: Thông tin sản phẩm.
- category\_id, brand\_id, type\_id: Khóa ngoại đến bảng categories, brands, types.
- discount: Mức giảm giá.
- is\_active: Trạng thái còn bán hay không.

- `create_at`, `update_at`: Thời gian tạo và cập nhật sản phẩm.

### **3. categories – Danh mục sản phẩm**

- `category_id`: Khóa chính.
- `category_name`: Tên danh mục.

### **4. brands – Thương hiệu**

- `brand_id`: Khóa chính.
- `brand_name`: Tên thương hiệu.

### **5. types – Loại sản phẩm**

- `type_id`: Khóa chính.
- `type_name`: Tên loại sản phẩm.

### **6. product\_images – Hình ảnh sản phẩm**

- `prod_image_id`: Khóa chính.
- `prod_id`: Khóa ngoại đến products.
- `image`: Đường dẫn ảnh.
- `is_primary_image`: Ảnh chính hay không.

### **7. carts – Giỏ hàng**

- `id`: Khóa chính.
- `user_id`: Khóa ngoại đến users.
- `cart`: Trường json lưu nội dung giỏ hàng (gồm sản phẩm, số lượng...).

### **8. orders – Đơn hàng**

- `order_id`: Khóa chính (kiểu varchar).
- `user_id`: Khóa ngoại đến users.
- `total`, `status`, `paymentmethod`, `createdAt`: Thông tin đơn hàng.
- `shipping_*`: Thông tin giao hàng (tên, địa chỉ, số điện thoại, email, ghi chú).

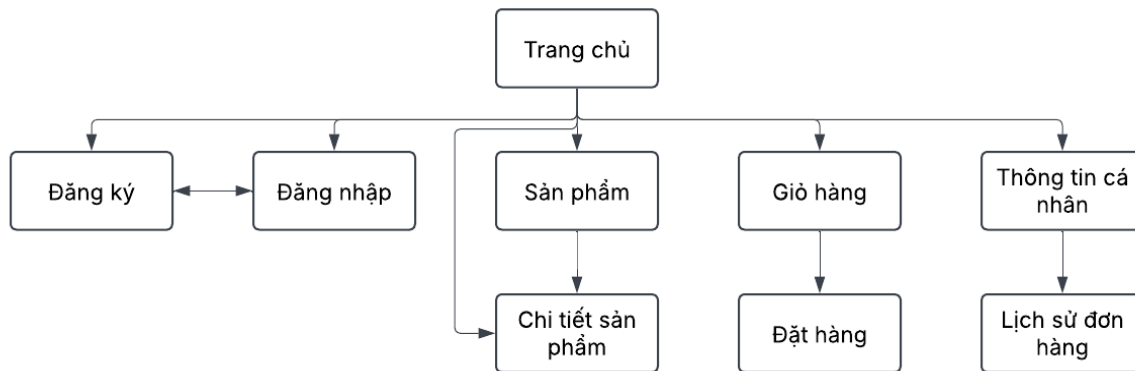
### **9. order\_details – Chi tiết đơn hàng**

- `order_detail_id`: Khóa chính.

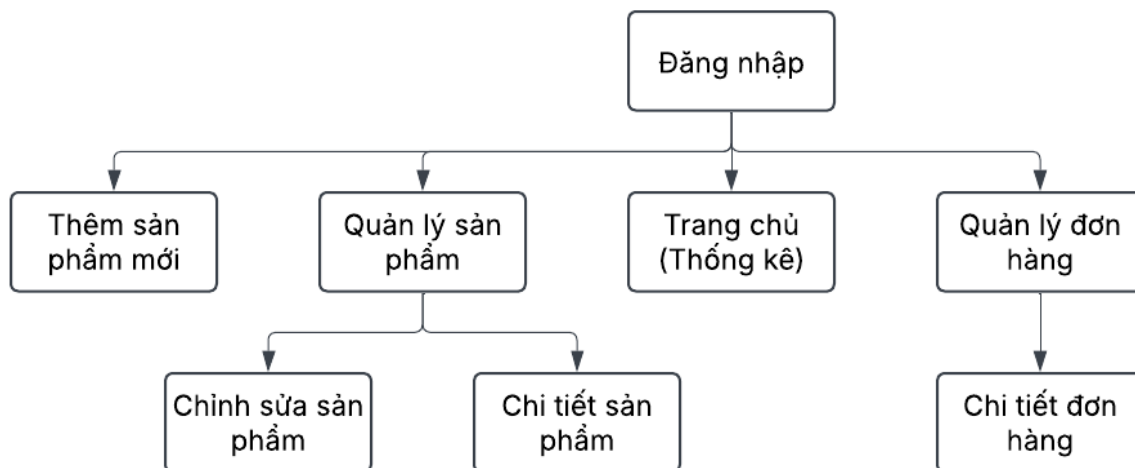
- order\_id: Khóa ngoại đến orders.
- prod\_id: Khóa ngoại đến products.
- quantity, price: Số lượng và giá tại thời điểm mua.

### 3.3. Thiết kế giao diện

#### 3.3.1. Sơ đồ liên kết các màn hình



Hình 12: Sơ đồ liên kết các màn hình trang người dùng



Hình 13: Sơ đồ liên kết các màn hình

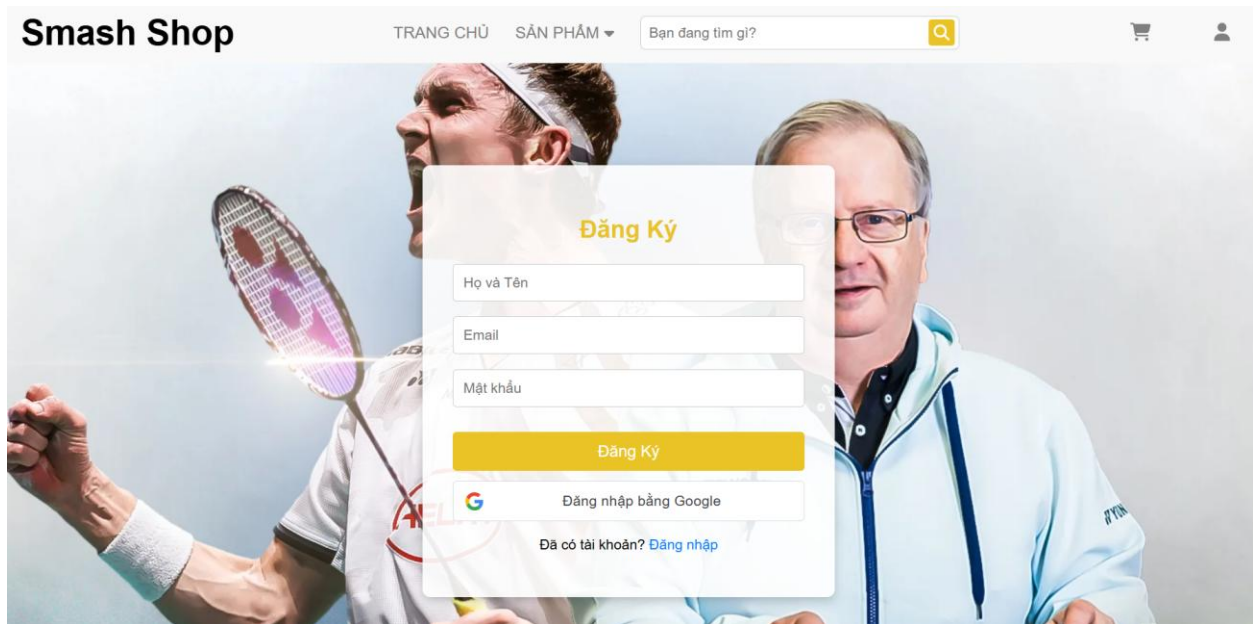
#### 3.3.2. Danh sách các màn hình

STT	Màn hình	Mô tả
1	Đăng ký	Cho phép người dùng tạo tài khoản mới bằng cách nhập email, mật khẩu,...

2	Đăng nhập	Giao diện để người dùng nhập thông tin và truy cập vào hệ thống.
3	Trang chủ	Hiện thị sản phẩm nổi bật và điều hướng đến các phần chính của trang.
4	Sản phẩm	Danh sách toàn bộ sản phẩm hiện có trên hệ thống.
5	Chi tiết sản phẩm	Thông tin chi tiết từng sản phẩm, bao gồm hình ảnh, giá, mô tả,...
6	Giỏ hàng	Hiện thị các sản phẩm người dùng đã chọn để mua.
7	Đặt hàng	Giao diện xác nhận đơn hàng và nhập thông tin giao hàng.
8	Thông tin cá nhân	Cho phép người dùng cập nhật thông tin cá nhân.
9	Lịch sử đơn hàng	Hiện thị các đơn hàng đã đặt, trạng thái.
10	Đăng nhập Admin	Giao diện đăng nhập dành riêng cho quản trị viên.
11	Thống kê	Tổng hợp doanh thu, số lượng đơn hàng,... cho admin.
12	Danh sách sản phẩm	Admin quản lý tất cả sản phẩm hiện có: xem, thêm, sửa, xóa.
13	Chi tiết sản phẩm	Hiện thị đầy đủ thông tin 1 sản phẩm.
14	Thêm sản phẩm mới	Admin nhập thông tin để tạo sản phẩm mới.
15	Chỉnh sửa sản phẩm	Admin cập nhật thông tin sản phẩm đã có.
16	Danh sách đơn hàng	Admin xem toàn bộ đơn hàng của khách đã đặt.
17	Chi tiết đơn hàng	Xem chi tiết từng đơn hàng, sản phẩm, trạng thái giao hàng,...

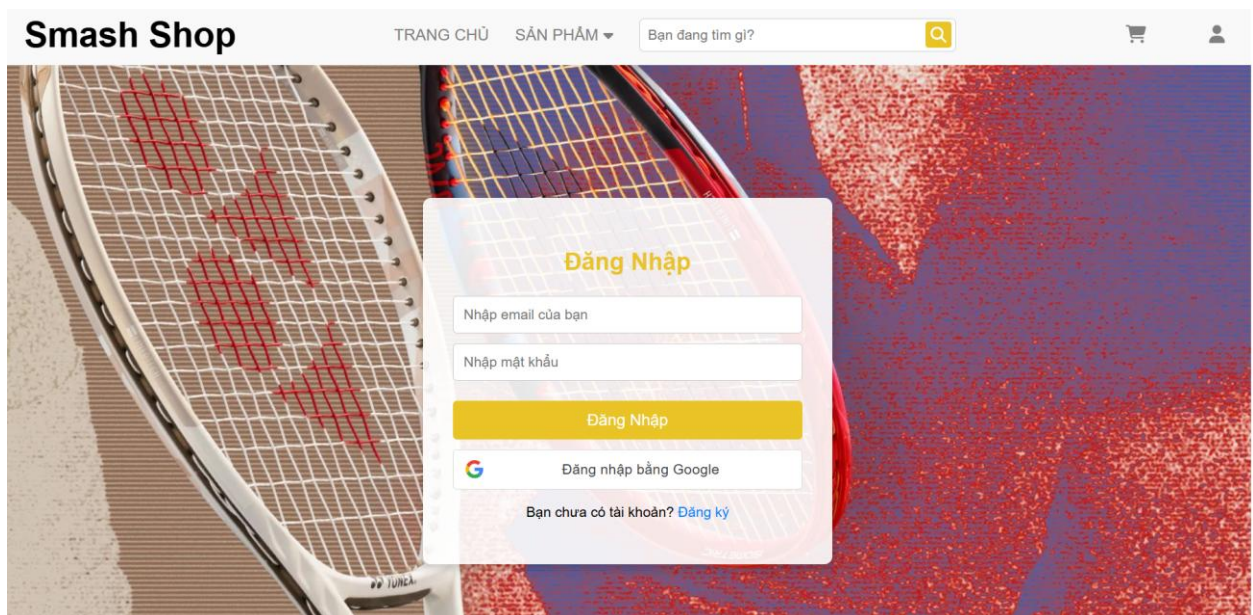
*Bảng 1: Danh sách các màn hình*

### 3.3.2.1. Màn hình đăng kí



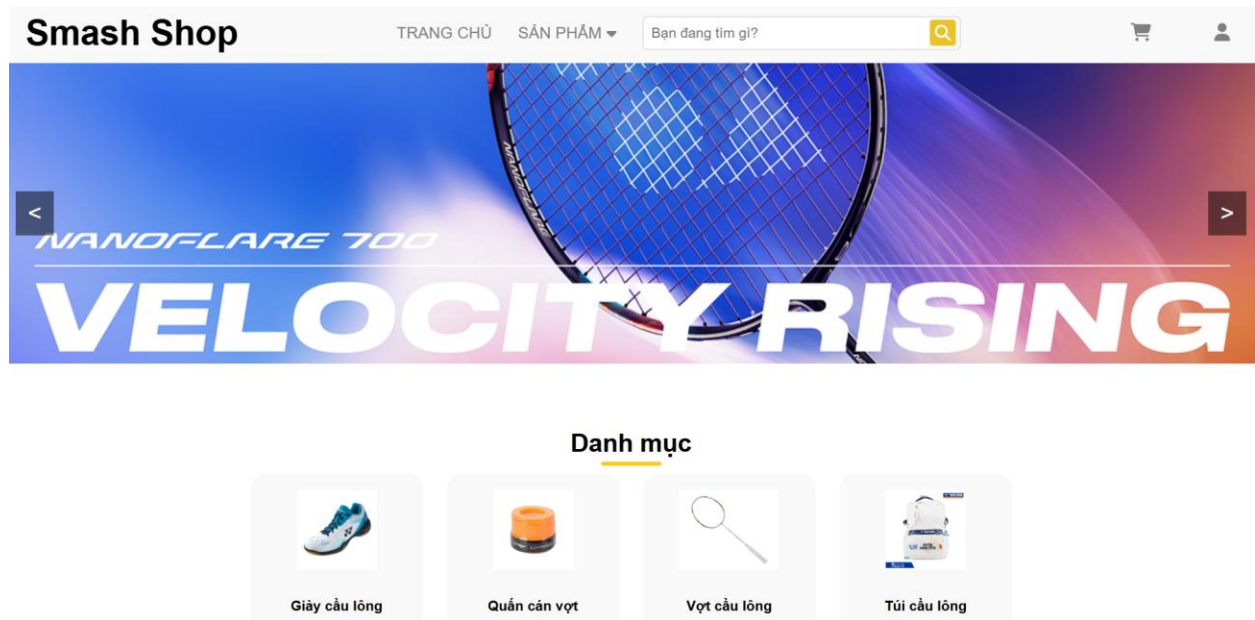
Hình 14: Màn hình đăng ký của người dùng

### 3.3.2.2. Màn hình đăng nhập

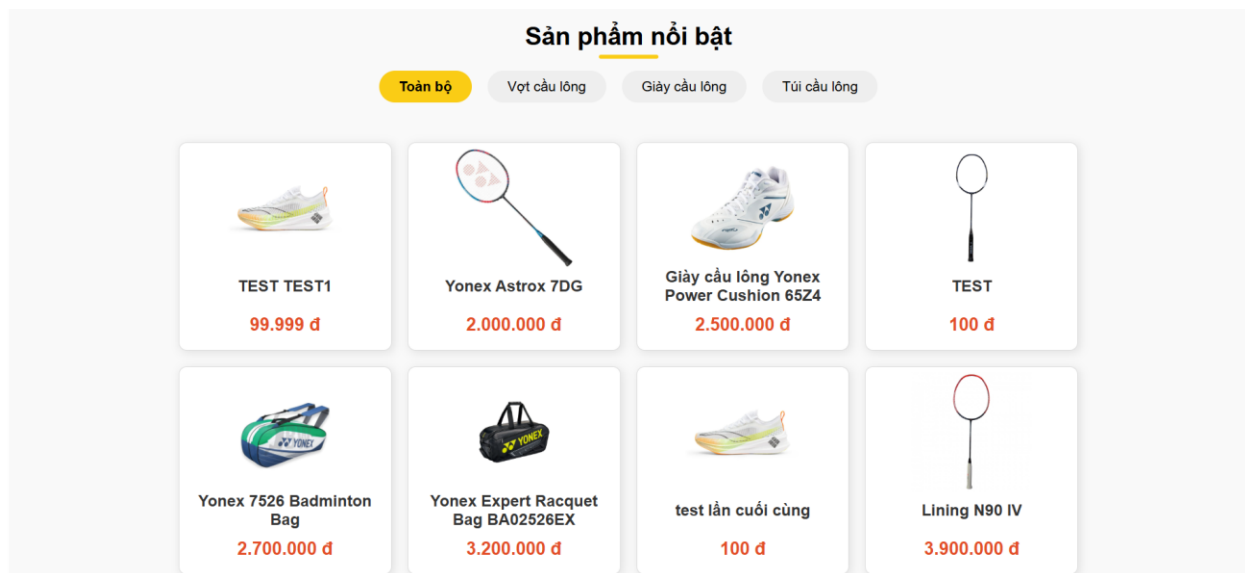


Hình 15: Màn hình đăng nhập của người dùng

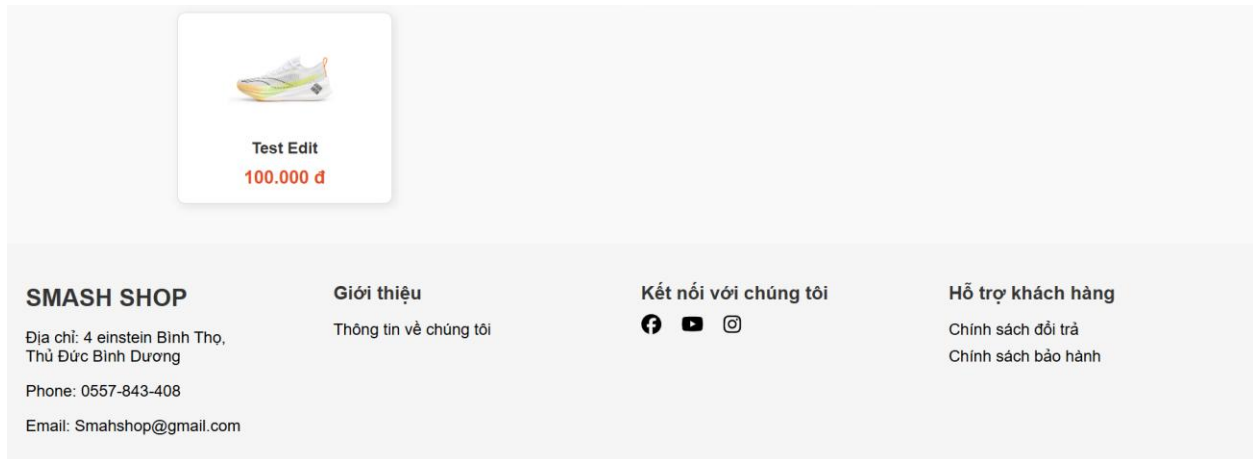
### 3.3.2.3. Màn hình trang chủ



Hình 16: Màn hình trang chủ - Header, slider và danh mục

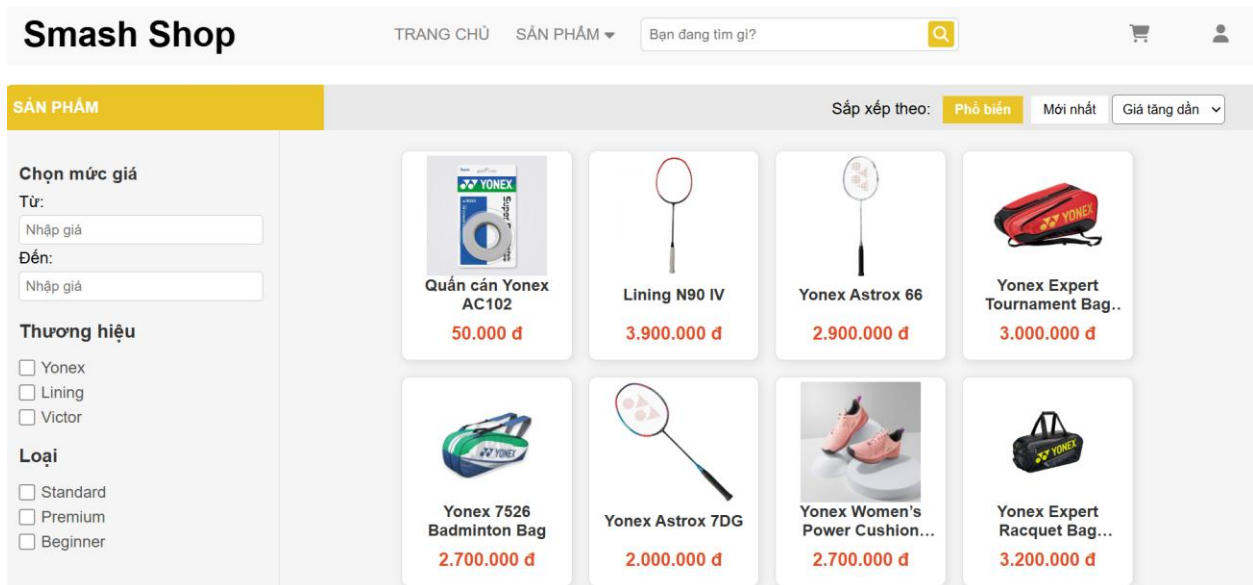


Hình 17: Màn hình trang chủ - sản phẩm nổi bật



Hình 18: Màn hình trang chủ - Footer

### 3.3.2.4. Màn hình sản phẩm




Hình 19: Màn hình sản phẩm

### 3.3.2.5. Màn hình chi tiết sản phẩm

Smash Shop
TRANG CHỦ SẢN PHẨM
Bạn đang tìm gì?

TRANG CHỦ > Túi cầu lông > Yonex 7526 Badminton Bag



### Yonex 7526 Badminton Bag

2.700.000 đ

Số lượng:

Số lượng trong kho: 14

Thương hiệu: Yonex

Danh mục: Túi cầu lông



THÔNG TIN CHI TIẾT

Túi cầu lông Yonex 7526 tiện dụng với thiết kế thời trang, nhiều ngăn lưu trữ cho vợt và phụ kiện.

Hình 20: Màn hình chi tiết sản phẩm

## 3.3.2.6. Màn hình giỏ hàng

TRANG CHỦ > GIỎ HÀNG

Sản phẩm	Đơn giá	Số lượng	Thành tiền	
 Yonex Astrox 7DG	2.000.000 đ	<input type="text" value="1"/> <input type="button" value="+"/> <input type="button" value="-"/>	2.000.000 đ	<input type="button" value="Xóa"/>
 Quần cân Yonex AC102	50.000 đ	<input type="text" value="2"/> <input type="button" value="+"/> <input type="button" value="-"/>	100.000 đ	<input type="button" value="Xóa"/>
			<b>Tổng tiền: 2.100.000 đ</b>	

Mã giảm giá

Số lượng: 3  
Thành tiền: 2.100.000 đ

Hình 21: Màn hình giỏ hàng

## 3.3.2.7. Màn hình đặt hàng



**TRANG CHỦ > GIỎ HÀNG**

**Thông tin đặt hàng**

**Name**

**Address**

**Phone**

**Email**

**Ghi chú**

**Phương thức thanh toán:**

☒ Thanh toán khi nhận hàng
 ☐ Thanh toán qua VNPay

**Đơn hàng**

Yonex Astrox 7DG ×1	2.000.000 đ
Quần cân Yonex AC102 ×2	50.000 đ
<b>Tổng đơn</b>	<b>2.100.000 đ</b>

**Đặt hàng**

Hình 22: Màn hình đặt hàng

### 3.3.2.8. Màn hình thông tin cá nhân

**Smash Shop**

**TRANG CHỦ**
**SẢN PHẨM**

**TRANG CHỦ > TRANG CÁ NHÂN**

**Thông tin tài khoản**

Đơn hàng đã đặt
 Đổi mật khẩu
 Đăng xuất

**Thông tin tài khoản**

**Họ và tên:** Thiên An

**Email:** thienan2@gmail.com

**Số điện thoại:**

**Chỉnh sửa**

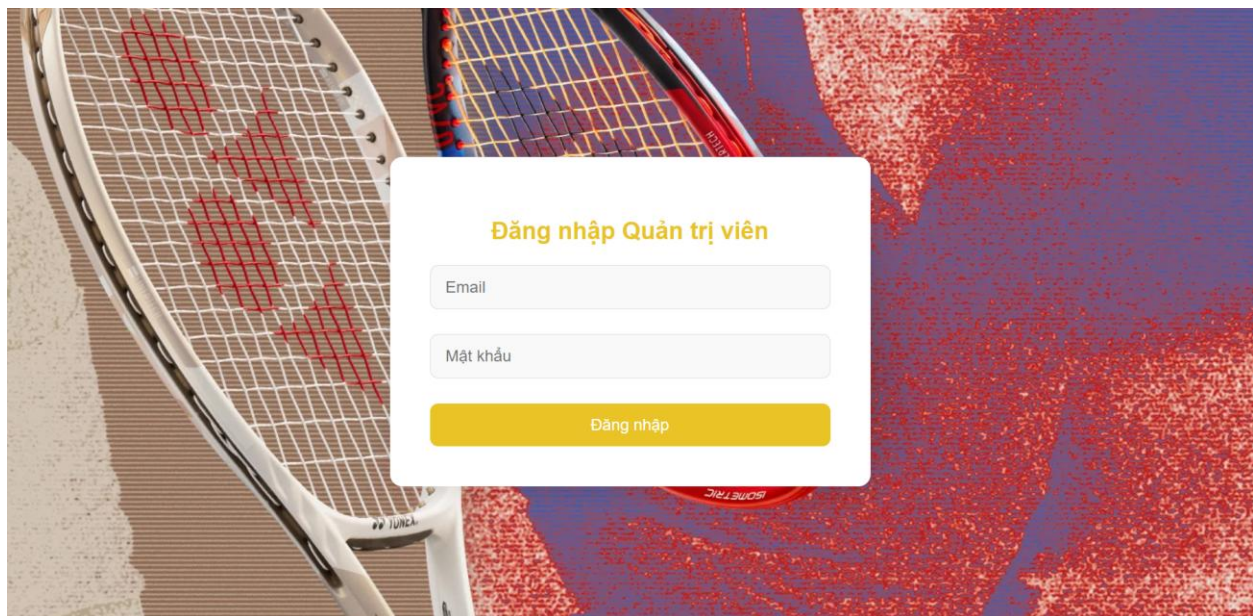
Hình 23: Màn hình thông tin cá nhân

### 3.3.2.9. Màn hình lịch sử đơn hàng

TRANG CHỦ > TRANG CÁ NHÂN											
<div><div><div><div><div></div><div>Thông tin tài khoản</div></div></div><div><div><div></div><div>Đơn hàng đã đặt</div></div></div><div><div><div></div><div>Đổi mật khẩu</div></div></div><div><div><div></div><div>Đăng xuất</div></div></div></div></div>	<div>Đơn hàng đã đặt</div> <table><tr><th>Mã đơn</th><th>Trạng thái</th><th>Ngày</th><th>Tổng tiền</th></tr><tr><td>6816b477de0f7c8db86441ef</td><td>Succeeded</td><td>4/5/2025</td><td>50.000đ</td></tr></table>			Mã đơn	Trạng thái	Ngày	Tổng tiền	6816b477de0f7c8db86441ef	Succeeded	4/5/2025	50.000đ
Mã đơn	Trạng thái	Ngày	Tổng tiền								
6816b477de0f7c8db86441ef	Succeeded	4/5/2025	50.000đ								

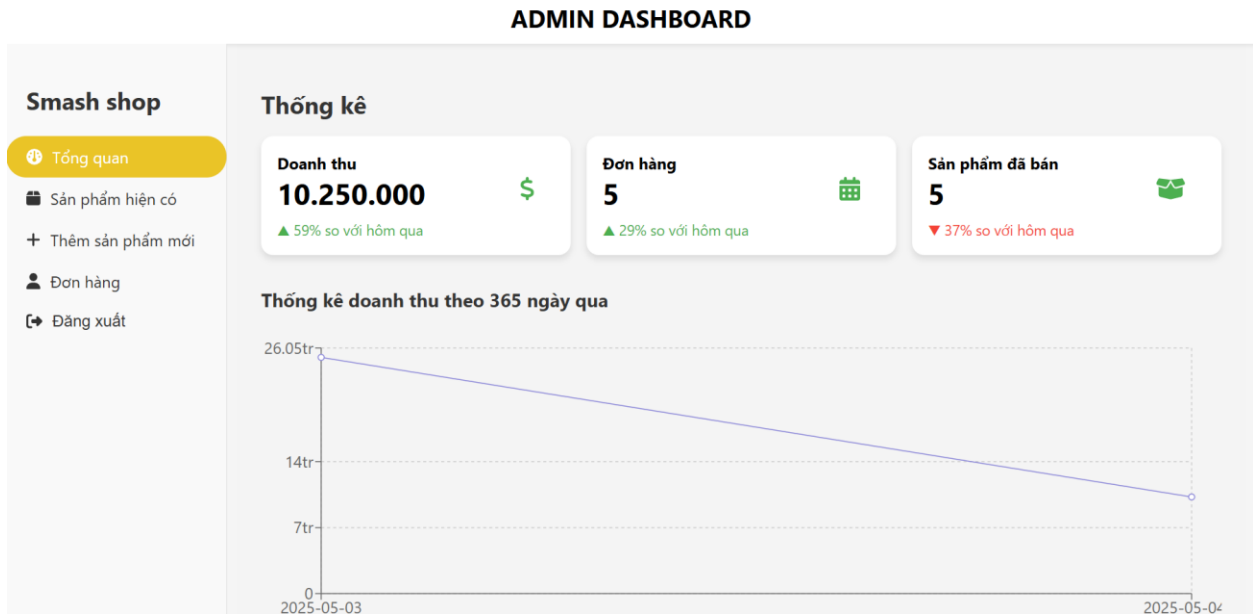
Hình 24: Màn hình lịch sử đơn hàng

### 3.3.2.10. Màn hình đăng nhập Admin

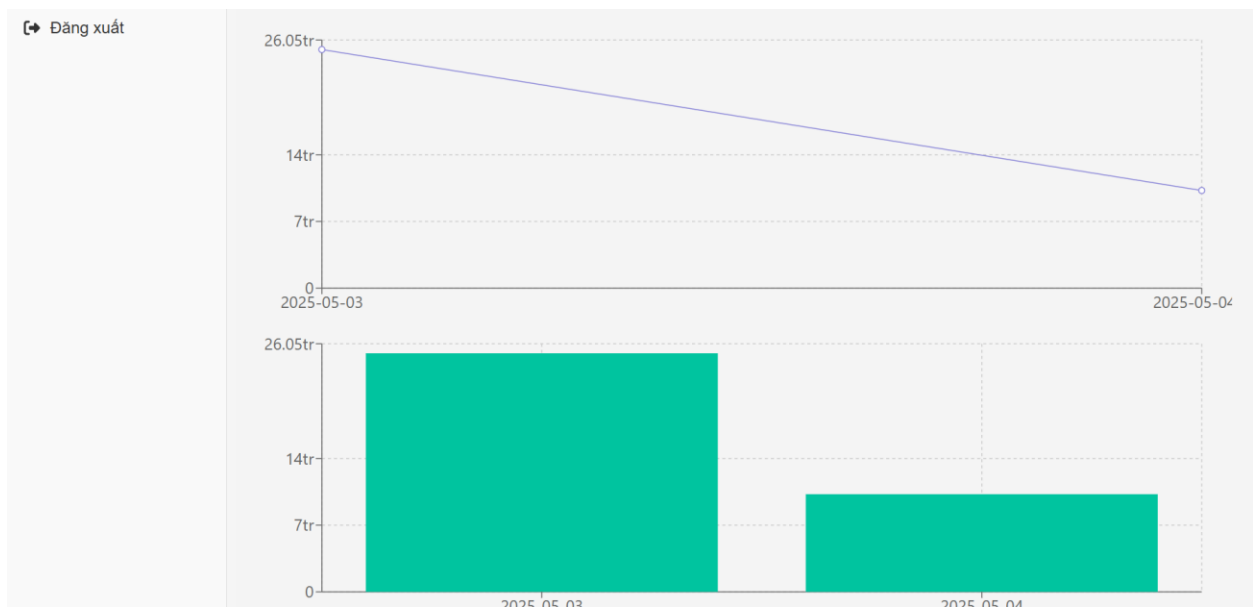


Hình 25: Màn hình đăng nhập Admin

### 3.3.2.11. Màn hình thống kê















Hình 26: Màn hình trang thống kê – trên




Hình 27: Màn hình trang thống kê – dưới

### 3.3.2.12. Màn hình quản lý sản phẩm

<b>Smash shop</b> <ul style="list-style-type: none"> <li>Tổng quan</li> <li><b>Sản phẩm hiện có</b></li> <li>Thêm sản phẩm mới</li> <li>Đơn hàng</li> <li>Đăng xuất</li> </ul>		<b>Sản phẩm hiện có</b>			
Ảnh	Tên sản phẩm	Danh mục	Thương hiệu	Giá	
	TEST TEST1	Giày cầu lông	Yonex	99.999đ	 
	Yonex Astrox 7DG	Vợt cầu lông	Yonex	2.000.000đ	 
	Giày cầu lông Yonex Power Cushion 65Z4	Giày cầu lông	Yonex	2.500.000đ	 
	TEST	Vợt cầu lông	Yonex	100đ	 

Hình 28: Màn hình xem danh sách sản phẩm

## 3.3.2.13. Màn hình chi tiết sản phẩm (Admin)

<b>Smash shop</b> <ul style="list-style-type: none"> <li>Tổng quan</li> <li><b>Sản phẩm hiện có</b></li> <li>Thêm sản phẩm mới</li> <li>Đơn hàng</li> <li>Đăng xuất</li> </ul>		<b>Thông tin sản phẩm</b>	
		<b>ID:</b> 11	
		<b>Tên sản phẩm:</b> Yonex Astrox 7DG	
		<b>Giá:</b> 2.000.000 VND	
		<b>Số lượng trong kho:</b> 11	
		<b>Đã bán:</b> 14	
		<b>Giảm giá:</b> 8%	
		<b>Loại:</b> Beginner	
		<b>Danh mục:</b> Vợt cầu lông	
		<b>Thương hiệu:</b> Yonex	
		<b>Mô tả:</b> Vợt cầu lông Yonex Astrox 7DG với thiết kế hơi nặng đầu, thân trung bình, phù hợp cho lối chơi công thủ toàn diện. Công nghệ Durable Grade (DG) tăng độ	

Hình 29: Màn hình chi tiết sản phẩm dành cho Admin

## 3.3.2.14. Màn hình thêm sản phẩm

**ADMIN DASHBOARD**

**Smash shop**

- Tổng quan
- Sản phẩm hiện có
- + Thêm sản phẩm mới**
- Đơn hàng
- Đăng xuất

### Thêm sản phẩm mới

Tên sản phẩm

Giá gốc

Số lượng trong kho

Số lượng đã bán

*Hình 30: Màn hình thêm sản phẩm - trên*

Thương hiệu

Loại sản phẩm

Giảm giá (%)

Tải lên hình ảnh sản phẩm

Không có tệp nào được chọn

*Hình 31: Màn hình thêm sản phẩm – dưới*

#### 3.3.2.15. Màn hình sửa sản phẩm

**Smash shop**

- Tổng quan
- Sản phẩm hiện có
- Thêm sản phẩm mới
- Đơn hàng
- Đăng xuất

### Chỉnh sửa sản phẩm

**Tên sản phẩm**

**Giá gốc**

**Số lượng trong kho**

**Số lượng đã bán**

**Mô tả**

Vợt cầu lông Yonex Astrox 7DG với thiết kế hơi nặng đầu, thân trung bình, phù hợp cho lối chơi công thủ toàn diện. Công nghệ Durable Grade (DG) tăng độ bền và hiệu suất.

Hình 32: Màn hình sửa sản phẩm

## 3.3.2.16. Màn hình quản lý đơn hàng

ADMIN DASHBOARD					
<b>Smash shop</b> <ul style="list-style-type: none"> <li>Tổng quan</li> <li>Sản phẩm hiện có</li> <li>Thêm sản phẩm mới</li> <li><b>Đơn hàng</b></li> <li>Đăng xuất</li> </ul>	<b>Danh sách đơn hàng</b>				
	<b>ID đơn hàng</b>	<b>Giá trị đơn</b>	<b>Khách hàng</b>	<b>Ngày tạo</b>	<b>Trạng thái</b>
	c684bcfd-cc9d-4f5c-b395-8d2915c9e398	9.050.000đ	nguylam	3/5/2025	Succeeded
	b9a44f9a-9489-4249-8295-a67d526f0467	10.900.000đ	nguylam	3/5/2025	Processing
	6ec5c412-6292-4324-bb31-0e8a1aa6c2cd	2.000.000đ	Phong Bui	3/5/2025	Pending
	c1fd4777-6fa1-4860-8c68-c8ebf94d6dcb	2.700.000đ	Phong Bui	3/5/2025	Pending
	08112d3d-8fff-48f1-a7b4-cf20bdaae74f	2.000.000đ	Phong Bui	3/5/2025	Pending
	fca2b656-c0cf-445c-9729-20a0be11f18e	2.000.000đ	Phong Bui	3/5/2025	Succeeded
	ee1c205d-fe03-416c-96c2-092c78b91984	2.700.000đ	Phong Bui	3/5/2025	Succeeded
	855cb68b-ea11-4dbb-a516-1f9e6a5f2dfb	2.000.000đ	Phong Bui	3/5/2025	Succeeded

Hình 33: Màn hình quản lý đơn hàng

## 3.3.2.17. Màn hình chi tiết đơn hàng

Smash shop

Tổng quan


Sản phẩm hiện có

Thêm sản phẩm mới

Đơn hàng

Đăng xuất

Chi tiết đơn hàng

Ảnh	Tên	Đơn giá	Số lượng	Tổng
	Yonex Astrox 7DG	2.000.000 đ	1	2.000.000 đ

Tạm tính: 2.000.000 đ

Phí vận chuyển: 0 đ

Tổng cộng: 2.000.000 đ

Tóm tắt

Mã đơn hàng: 6ec5c412-6292-4324-bb31-0e8a1aa6c2cd

Ngày đặt hàng: 3/5/2025

Họ và tên: Phong Bui

Email: phongbui22012004@gmail.com

Địa chỉ

Chợ Tân Kiều

Trạng thái đơn hàng

Pending

Lưu

Hình 34: Màn hình trang quản lý chi tiết đơn hàng

## CHƯƠNG 4. Hiện thực

### 4.1. Backend

#### 4.1.1. Kiến trúc và công nghệ sử dụng

- Mô hình MVC (Model–View–Controller):
  - Model: Định nghĩa schema và thao tác với MongoDB thông qua Mongoose.
  - Controller: Xử lý logic nghiệp vụ, gọi DAO/Service, và trả về JSON cho client.
  - View: Không áp dụng riêng trên backend; toàn bộ dữ liệu trả về dạng JSON cho frontend React.js.
- Node.js & Express.js
  - Node.js cung cấp môi trường runtime bất đồng bộ, phù hợp cho các tác vụ I/O-bound.
  - Express.js đảm nhiệm routing, middleware, và xử lý request/response một cách linh hoạt.
- MongoDB & Mongoose
  - MongoDB – cơ sở dữ liệu NoSQL giúp lưu trữ linh hoạt, dễ mở rộng.
  - Mongoose – ODM hỗ trợ validate, hooks, và các thao tác CRUD.
- Các thư viện khác
  - dotenv: Quản lý biến môi trường (DB\_URI, JWT\_SECRET, VNPAY\_SECRET...).
  - cors: Cho phép frontend (React.js) gọi API từ domain khác.
  - bcrypt: Mã hóa và so khớp mật khẩu.
  - jsonwebtoken: Tạo và giải mã JWT.
  - qs, crypto: Hỗ trợ ký và kiểm tra callback của VNPAY.



## 4.1.2. Thiết kế dữ liệu – Các entity chính trong MongoDB

Entity	Các trường chính	Mô tả ngắn
<b>Admin</b>	_id, email, password, role	Quản trị hệ thống
<b>User</b>	_id, name, email, password, role, avatar	Người dùng thường & phân quyền admin
<b>Product</b>	_id, name, description, price, stock, brand, type, category	Sản phẩm cầu lông
<b>ProductImage</b>	_id, productId, url, isDefault	Ảnh sản phẩm
<b>Brand</b>	_id, name	Thương hiệu
<b>Type</b>	_id, name	Loại sản phẩm (ví dụ: vợt, giày...)
<b>Category</b>	_id, name	Phân mục (nam, nữ, trẻ em...)
<b>Order</b>	_id, userId, totalPrice, status, createdAt	Đơn hàng
<b>OrderDetail</b>	_id, orderId, productId, quantity, price	Chi tiết từng sản phẩm trong đơn hàng
<b>Wishlist</b>	_id, userId, products: [productId]	Danh sách yêu thích
<b>Review</b>	_id, userId, productId, rating, comment	Đánh giá sản phẩm

*Bảng 2: Các entity chính trong MongoDB*

## 4.1.3. Cơ chế xác thực JWT

- **Đăng nhập (/api/auth/login)**

1. Client gửi email & password.
2. Nếu hợp lệ, server cấp:
  - **Access Token** (TTL = 15 phút)
  - **Refresh Token** (TTL = 7 ngày)
3. Refresh token được lưu cả trong cookie và trong collection RefreshToken trên MongoDB.

- **Middleware bảo vệ route (authMiddleware)**

- Kiểm tra header Authorization: Bearer <accessToken>, giải mã bằng jwt.verify().
- Nếu token hết hạn, trả HTTP 401 để client gọi /api/auth/refreshtoken.

- **Gia hạn token (/api/auth/refreshtoken)**

- Client gửi refreshToken (từ cookie).
- Server xác thực tồn tại và chưa blacklist, rồi cấp lại accessToken mới (và có thể refreshToken mới).
- **Đăng xuất (/api/auth/logout)**
  - Xóa refreshToken khỏi DB và cookie để vô hiệu hóa phiên.

#### 4.1.4. Tích hợp thanh toán trực tuyến qua cổng VNPAY

##### Mục tiêu

Hệ thống **Smash Shop** hỗ trợ thanh toán trực tuyến thông qua cổng VNPAY nhằm mang đến trải nghiệm mua sắm tiện lợi và an toàn cho người dùng. Việc tích hợp này cho phép khách hàng lựa chọn các phương thức thanh toán phổ biến tại Việt Nam, bao gồm:

- Thẻ ATM nội địa (qua Internet Banking).
- Thẻ tín dụng/quốc tế (Visa, MasterCard).
- Ví điện tử và ứng dụng ngân hàng hỗ trợ VNPAY-QR. [sandbox.vnpayment.vn](https://sandbox.vnpayment.vn)

##### Quy trình thanh toán

Quy trình thanh toán trực tuyến qua VNPAY trong hệ thống bao gồm các bước sau:

##### 1. Tạo đơn hàng và URL thanh toán:

- Người dùng xác nhận giỏ hàng và lựa chọn thanh toán trực tuyến.
- Hệ thống backend tạo đơn hàng và sinh URL thanh toán theo định dạng của VNPAY, bao gồm các tham số như:
  - vnp\_Amount: Số tiền thanh toán.
  - vnp\_TxnRef: Mã tham chiếu đơn hàng.
  - vnp\_OrderInfo: Thông tin đơn hàng.
  - vnp\_ReturnUrl: URL để VNPAY redirect sau khi thanh toán.
  - vnp\_SecureHash: Chuỗi hash để xác thực dữ liệu.
- Hệ thống sử dụng thuật toán HMAC SHA512 để tạo vnp\_SecureHash nhằm đảm bảo tính toàn vẹn của dữ liệu.

##### 2. Chuyển hướng đến cổng VNPAY:

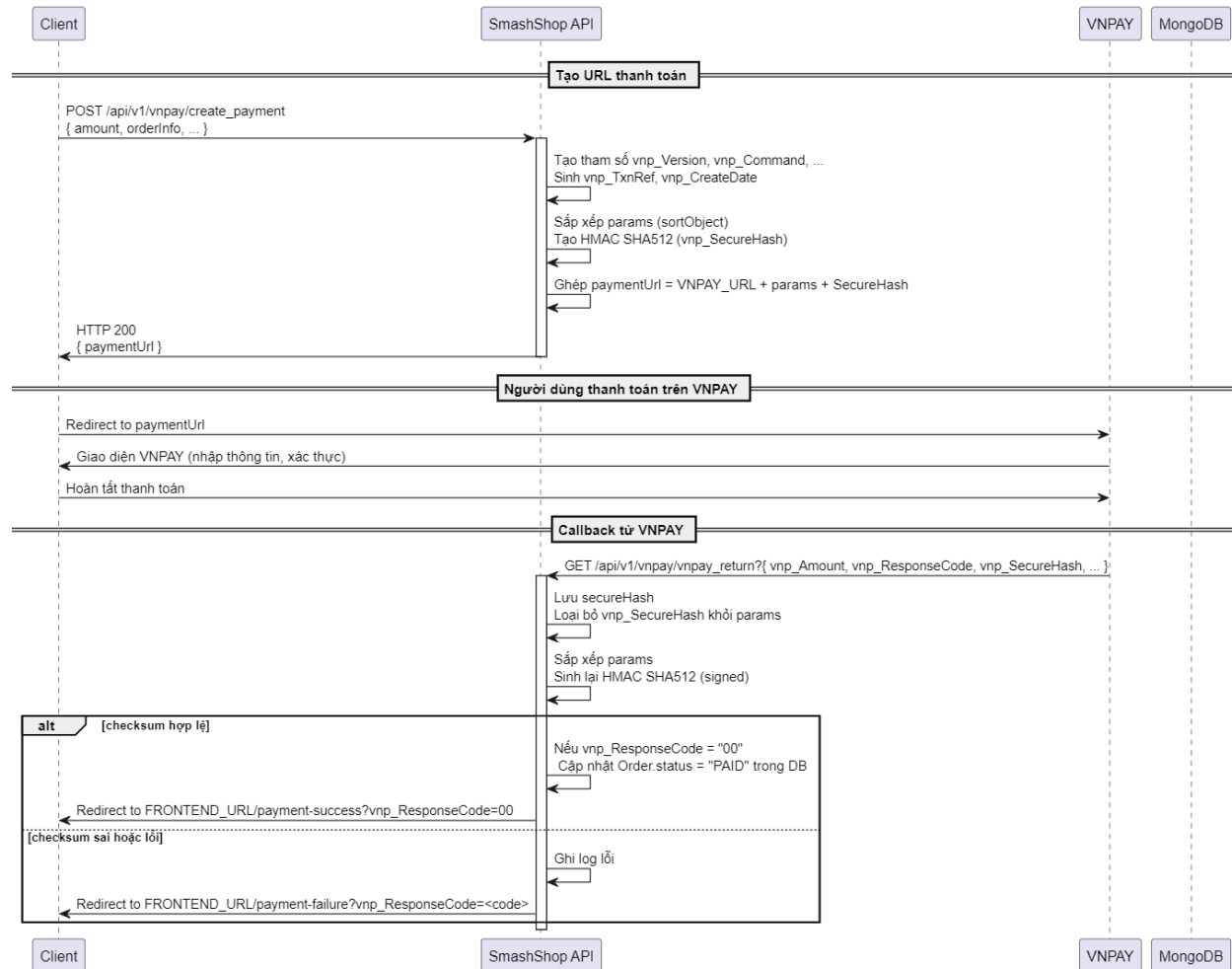
- Frontend nhận paymentUrl từ backend và chuyển hướng người dùng đến giao diện thanh toán của VNPAY.
- Người dùng thực hiện các bước thanh toán trên cổng VNPAY.

### 3. Xử lý phản hồi từ VNPAY:

- Sau khi thanh toán, VNPAY redirect người dùng về vnp\_ReturnUrl kèm theo các tham số phản hồi.
- Backend xử lý các bước sau:
  - Xác thực vnp\_SecureHash để đảm bảo dữ liệu không bị thay đổi.
  - Kiểm tra vnp\_ResponseCode để xác định trạng thái thanh toán:
    - 00: Thanh toán thành công.
    - Khác 00: Thanh toán thất bại hoặc bị hủy.
  - Cập nhật trạng thái đơn hàng trong cơ sở dữ liệu tương ứng với kết quả thanh toán.
  - Redirect người dùng về frontend với thông báo kết quả thanh toán.

### Sơ đồ trình tự thanh toán

Dưới đây là sơ đồ trình tự (sequence diagram) mô tả quy trình thanh toán trực tuyến qua VNPAY:



Hình 35: Sequence Diagram mô tả quy trình thanh toán trực tuyến

#### 4.1.5. Các module và chức năng chính

##### 1. Quản lý người dùng (/api/v1/users, /api/auth)

- Đăng ký, đăng nhập, gia hạn token, đăng xuất, quên/mật khẩu, CRUD user (admin).

##### 2. Quản lý sản phẩm (/api/v1/products, /api/v1/productImages, /api/v1/brand, /api/v1/type, /api/v1/category)

- Lấy danh sách, chi tiết, tìm kiếm, lọc, thêm/sửa/xóa (admin).

##### 3. Giỏ hàng (/api/v1/cart) (middleware auth)

- Thêm, sửa số lượng, xóa, lấy toàn bộ giỏ.

##### 4. Đặt hàng & thanh toán (/api/v1/order, /api/v1/vnpay)

- Tạo đơn hàng, xem lịch sử, cập nhật trạng thái (admin), chi tiết đơn.
- Thanh toán COD hoặc VNPAY với callback xác minh checksum.

### 5. Dashboard admin (/api/v1/dashboard)

- Thống kê doanh thu, số đơn theo ngày/tháng, sản phẩm bán chạy.

#### 4.1.6. Định nghĩa REST API điển hình

Endpoint	Phương thức	Mô tả	Middleware
/api/auth/login	POST	Đăng nhập & cấp JWT	none
/api/auth/refreshtoken	POST	Gia hạn accessToken	none
/api/auth/logout	POST	Đăng xuất	authMiddleware
/api/v1/users/register	POST	Đăng ký user	none
/api/v1/users/profile	GET	Lấy thông tin user hiện tại	authMiddleware
/api/v1/users	GET	Lấy danh sách user (admin)	adminMiddleware
/api/v1/products	GET	Lấy danh sách sản phẩm	none
/api/v1/products/:id	GET	Lấy chi tiết sản phẩm	none
/api/v1/products	POST	Tạo mới sản phẩm (upload ảnh)	adminMiddleware
/api/v1/cart	POST/PATCH	Thêm/cập nhật số lượng trong giỏ	authMiddleware
/api/v1/cart	DELETE	Xóa sản phẩm khỏi giỏ	authMiddleware
/api/v1/cart	GET	Lấy giỏ hàng	authMiddleware
/api/v1/order	POST	Tạo đơn hàng	authMiddleware

/api/v1/order/order_history	GET	Lịch sử đơn hàng user	authMiddleware
/api/v1/order	GET	Lấy tất cả đơn (admin)	adminMiddleware
/api/v1/order	PUT	Cập nhật trạng thái đơn (admin)	adminMiddleware
/api/v1/order/detail/:id	GET	Chi tiết đơn hàng	authMiddleware
/api/v1/vnpay/create_payment	POST	Tạo URL thanh toán VNPAY	authMiddleware
/api/v1/vnpay/vnpay_return	GET	Callback & xác minh checksum VNPAY	none
/api/auth/login	POST	Đăng nhập & cấp JWT	none
/api/auth/refreshtoken	POST	Gia hạn accessToken	none

Bảng 3: Định nghĩa REST API điển hình

## 4.2. Frontend

### 4.2.1. Kiến trúc và công nghệ sử dụng

Phần giao diện người dùng (frontend) của hệ thống Smash Shop được phát triển bằng ReactJS, một thư viện JavaScript mạnh mẽ, chuyên dùng để xây dựng giao diện động theo hướng component hóa. Ứng dụng được tổ chức theo mô hình SPA (Single Page Application), giúp tối ưu hóa trải nghiệm người dùng bằng cách giảm thời gian tải lại trang và phản hồi nhanh.

Giao diện được thiết kế hoàn toàn bằng CSS thuần, nhằm đảm bảo tính đơn giản, dễ kiểm soát và tùy biến theo ý đồ thiết kế. Hệ thống sử dụng responsive layout để hỗ trợ hiển thị tốt trên nhiều loại thiết bị (desktop, tablet, mobile).

Cấu trúc project phía frontend được phân chia rõ ràng thành các thư mục:

- components/: chứa các thành phần giao diện tái sử dụng như header, footer, productCard,...
- pages/: tập trung các màn hình chức năng như trang chủ, sản phẩm, giỏ hàng,...
- app/: chứa cấu hình tổng thể như routing, redux store,...

- assets/: chứa ảnh và các tài nguyên tĩnh phục vụ hiển thị giao diện.

Ngoài ra, các màn hình người dùng và admin được tách biệt nhằm phục vụ trải nghiệm phù hợp theo từng vai trò.

#### 4.2.2. Quản lý trạng thái với Redux

Để đảm bảo khả năng quản lý trạng thái hiệu quả và dễ mở rộng trong hệ thống, dự án Smash Shop sử dụng Redux kết hợp Redux Toolkit và Redux Toolkit Query (RTK Query). Đây là giải pháp hiện đại, giảm thiểu boilerplate code và tối ưu trải nghiệm phát triển.

Toàn bộ logic quản lý trạng thái được tổ chức theo kiến trúc module hóa rõ ràng, bao gồm:

- Thư mục app/: chứa file store.js – nơi cấu hình Redux store. Store này kết hợp các reducer từ slice cũng như middleware từ RTK Query.
- Thư mục features/: bao gồm các slice đặc thù cho từng nghiệp vụ:
  - o auth/ → authSlice.js: quản lý thông tin đăng nhập, token, trạng thái người dùng.
  - o cart/ → cartSlice.js: xử lý giỏ hàng, cập nhật số lượng sản phẩm.
  - o order/ → orderSlice.js: lưu thông tin đơn hàng trong quá trình xử lý.
  - o product/ → productSlice.js: quản lý sản phẩm trong giao diện người dùng.
  - o user/ → userSlice.js: cập nhật thông tin cá nhân và quản lý tài khoản.
- Các thao tác gọi API được tổ chức trong thư mục apis/, chia theo từng domain riêng như cart.js, products.js, order.js,... Các file này sử dụng Axios (được cấu hình sẵn trong axios.js) để thực hiện các phương thức như GET, POST, PUT, DELETE.

Lợi ích đạt được:

- Tối ưu tốc độ phát triển với cấu trúc chuẩn hóa.
- Dễ kiểm soát luồng dữ liệu giữa các màn hình và component.
- Khả năng mở rộng và tái sử dụng cao.
- Giảm thiểu lỗi logic do tự động hóa phần lớn quy trình gọi API và quản lý cache.

## CHƯƠNG 5. Triển khai và kiểm thử

### 5.1. Triển khai

#### 5.1.1. Triển khai Backend

Backend được đóng gói thành một Docker image và được đẩy (push) lên Docker Hub. Sau đó, Dokploy sẽ được sử dụng để triển khai image này dưới dạng một container, cho phép ứng dụng backend hoạt động trực tuyến.

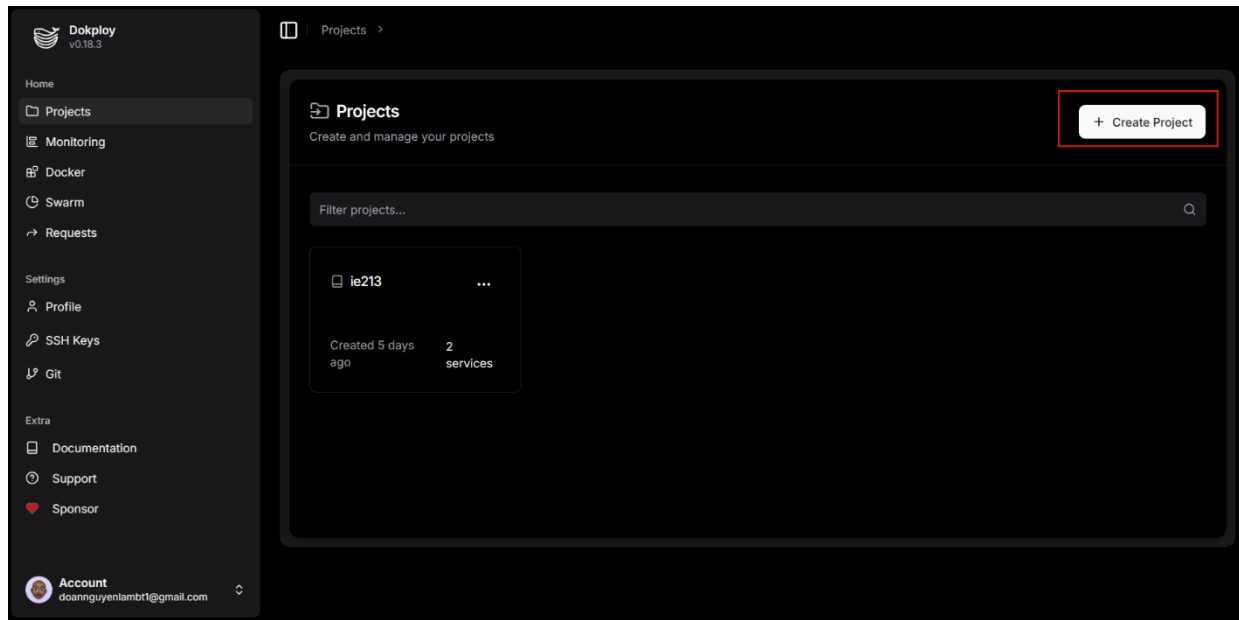
- Tạo docker image:
  - Trong thư mục backend, tạo file Dockerfile để định nghĩa cách build image
  - Tạo file .dockerignore để loại bỏ các tệp không cần thiết ra khỏi docker context

```
1  node_modules
2  .git
3  .env
```

*Hình 36: Nội dung .dockerignore*

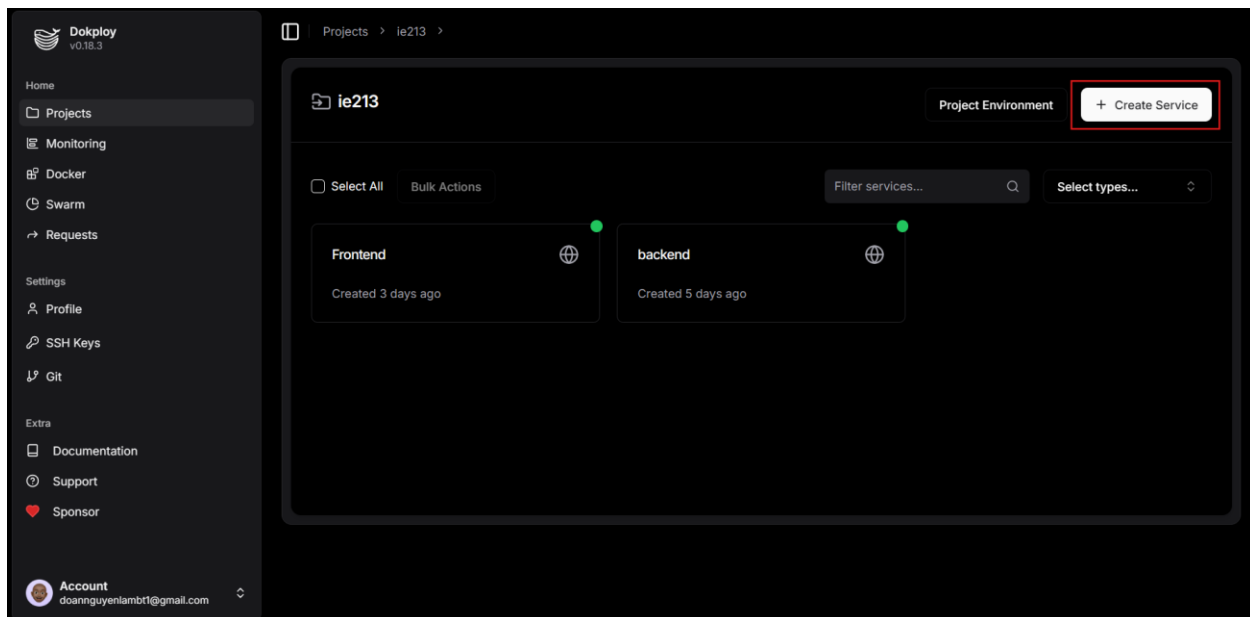
- Đăng nhập và push image lên Dockerhub:
  - Đăng nhập vào dockerhub bằng lệnh: docker login
  - Build docker image: docker build -t dockerhub\_username/ie213-backend .
  - Push docker image lên dockerhub: docker push dockerhub\_username/ie213-backend
- Deploy backend lên Dokploy
  - Đăng nhập vào Dokploy và tạo project mới.





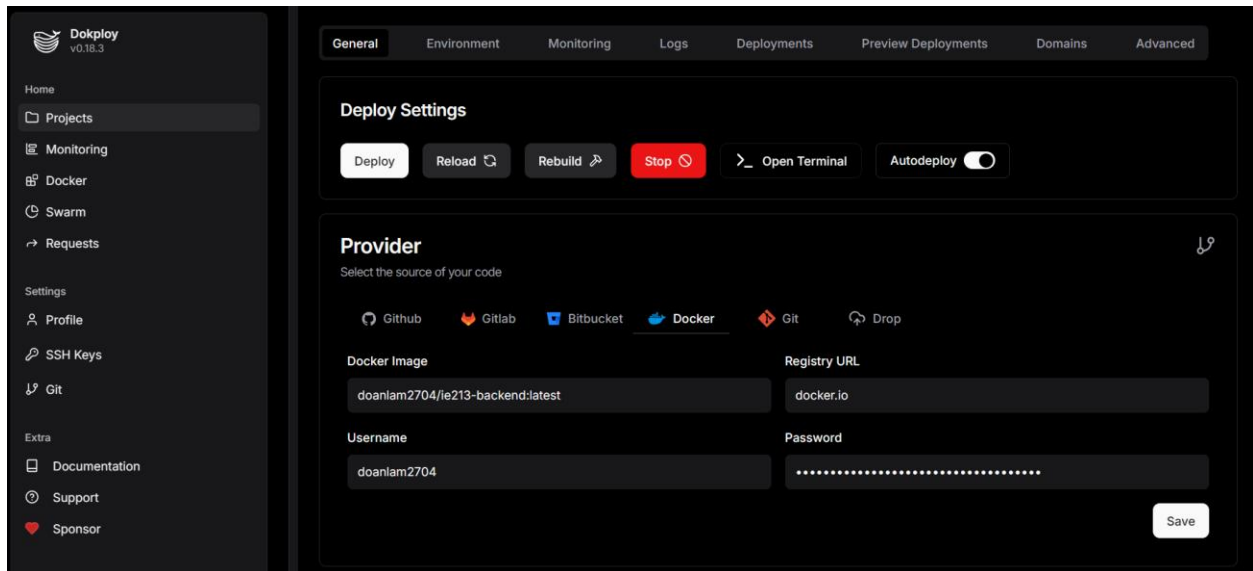
Hình 37: Tạo Project trên Dokploy

- Tạo service mới và nhập các thông tin cần thiết



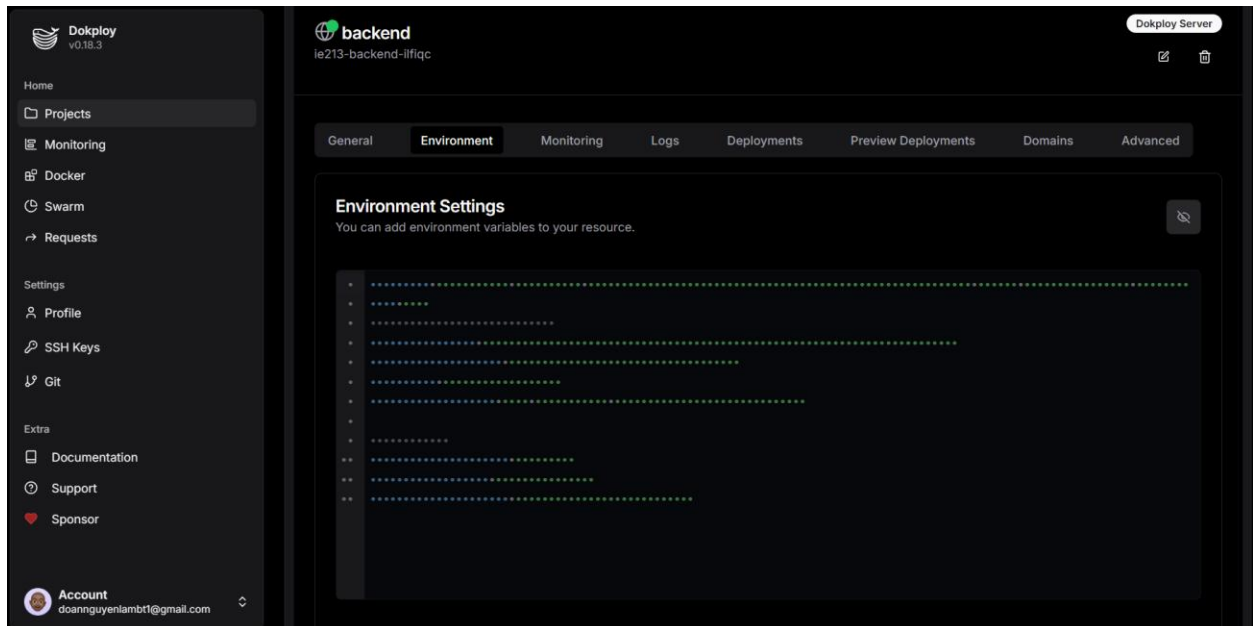
Hình 38: Tạo service mới trên dokploy

- Chọn service vừa tạo, chuyển sang tab General. Ở mục Provider, chọn Docker và nhập các thông tin cần thiết.



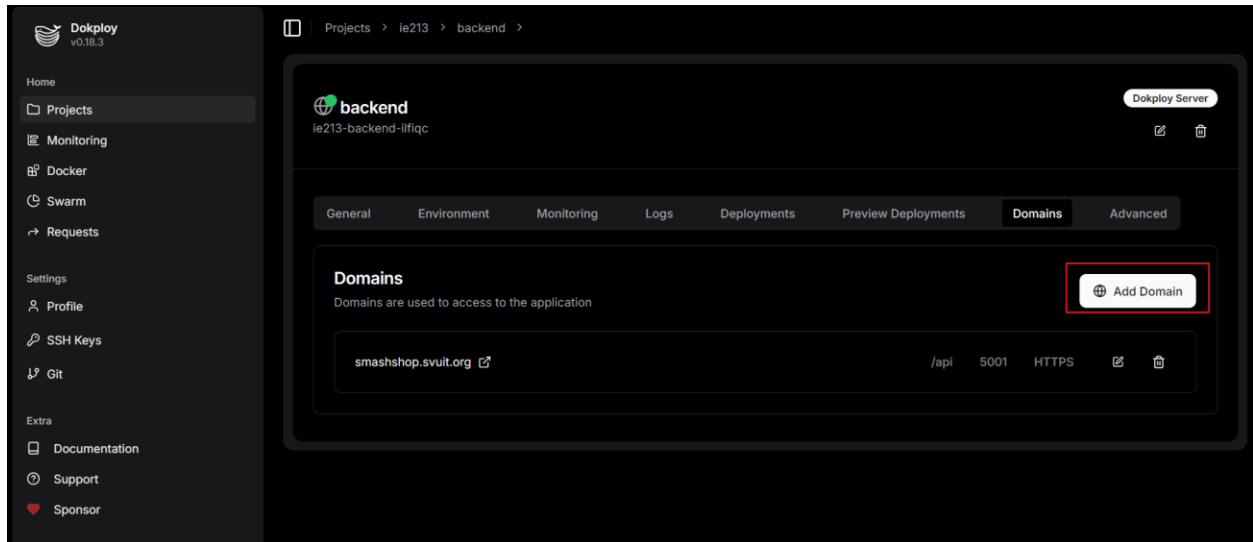
Hình 39: Cấu hình Provider cho Dokploy

- Chuyển sang tab Environment và cấu hình biến môi trường.



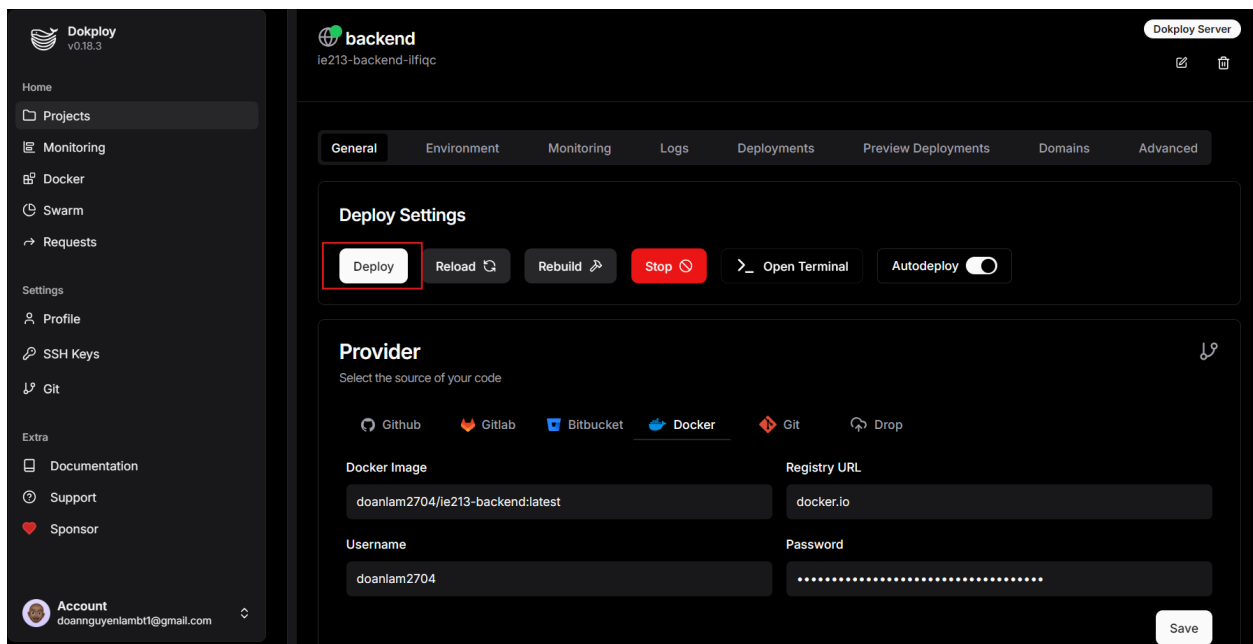
Hình 40: Cấu hình biến môi trường cho Dokploy

- Chuyển sang tab Domains và thêm domain



Hình 41: Thêm domain cho dokploy

- Chuyển sang tab General và nhấn “Deploy” để hoàn tất quá trình deploy backend



Hình 42: Deploy backend

### 5.1.2. Triển khai Frontend

Cách 1: Triển khai frontend lên Dokploy:

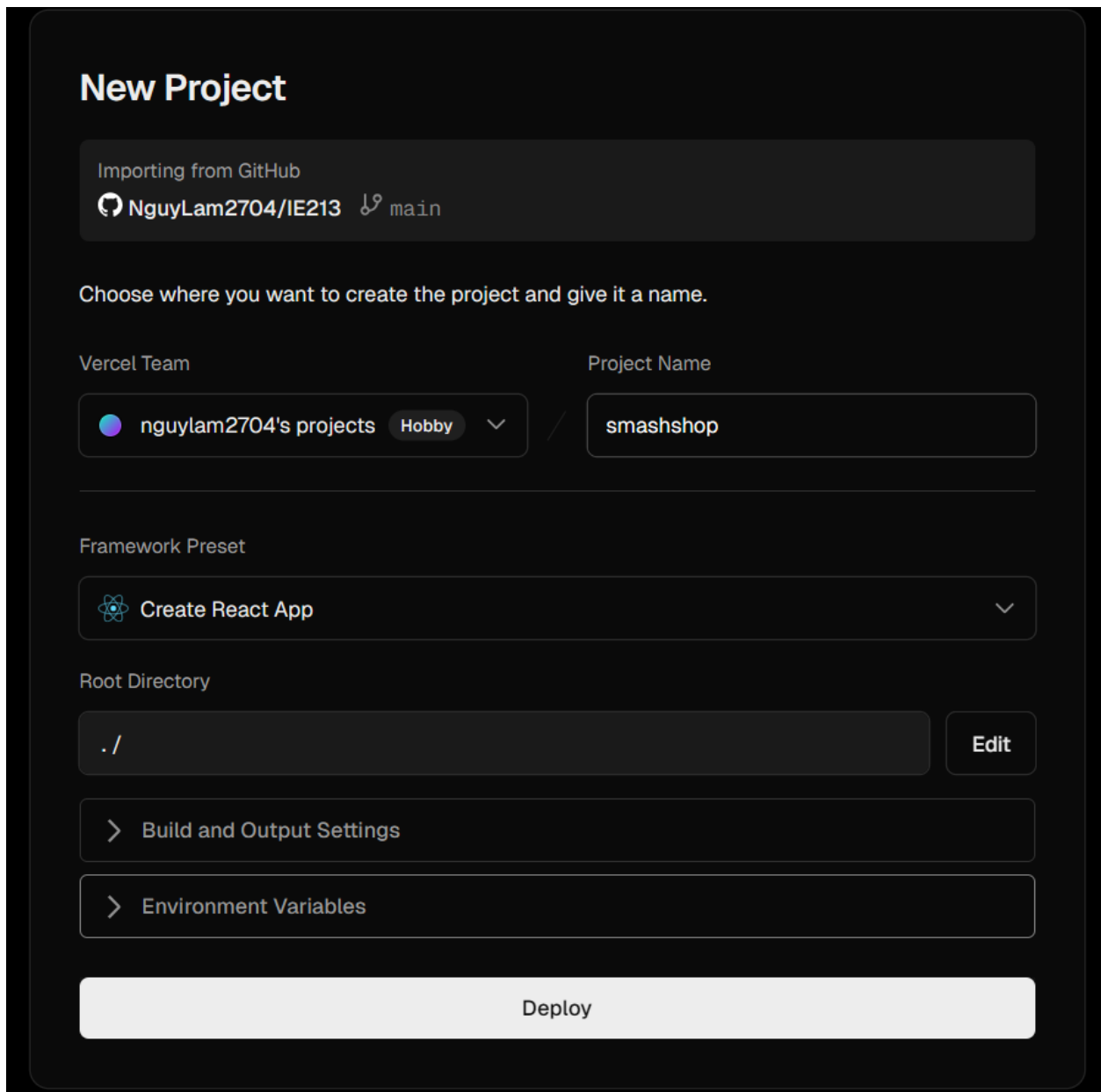
Quy trình triển khai Frontend tương tự như Backend. Trước tiên, ứng dụng frontend được build và đóng gói thành một Docker image, sau đó được đẩy lên Docker Hub. Cuối cùng, Dokploy được sử dụng để triển khai image này thành một container.

Các bước thực hiện như sau:

- Tạo docker image:
  - Trong thư mục frontend, tạo file Dockerfile để định nghĩa cách build image
    - Đối với Frontend sẽ dùng kỹ thuật multi-stage để giảm kích thước dữ liệu.
      - Stage 1: Sử dụng image Nodejs để build frontend
      - Stage 2: Serve React bằng Ngnix
  - Tạo file .dockerignore để loại bỏ các tệp không cần thiết ra khỏi docker context
- Đăng nhập và push image lên Dockerhub:
  - Đăng nhập vào dockerhub bằng lệnh: `docker login`
  - Build docker image: `docker build -t dockerhub_username/ie213-frontend .`
  - Push docker image lên dockerhub: `docker push dockerhub_username/ie213-frontend`
- Triển khai frontend lên Dokploy: Quá trình triển khai frontend lên Dokploy hoàn toàn tương tự với quá trình triển khai backend

Cách 2: Triển khai Frontend lên vercel:

- Đăng nhập vào Vercel và tạo 1 project mới
- Import repository của dự án
- Đặt tên cho project và cấu hình framework và root directory, và biến môi trường



**New Project**

Importing from GitHub  
NguyLam2704/IE213 main

Choose where you want to create the project and give it a name.

Vercel Team: nguylam2704's projects Hobby

Project Name: smashshop

Framework Preset: Create React App

Root Directory: ./ Edit

> Build and Output Settings

> Environment Variables

Deploy

Hình 43: Cấu hình deploy frontend lên vercel

- Nhấn deploy để triển khai frontend

Kết quả:

Đường dẫn tới frontend được triển khai bằng dokploy: <https://smashshop.svuit.org/>

Đường dẫn tới frontend được triển khai bằng vercel: <https://smashshop.vercel.app/>

### 5.1.3. Cấu hình CI/CD

Hệ thống CI/CD được thiết lập gồm 2 workflow riêng biệt cho frontend và backend. Cả hai workflow đều có nhiệm vụ tự động build Docker image và đẩy lên Docker Hub mỗi khi có thay đổi được push lên nhánh “main”.

- Backend:

- Bước 1: Tải mã nguồn của repository về máy ảo đang chạy Github Action.
- Bước 2: Đăng nhập vào Docker Hub bằng thông tin tài khoản được lưu trong GitHub Secrets.
- Bước 3: Build docker image từ thư mục ./backend với Dockerfile tương ứng
- Bước 4: Sau khi build, image sẽ được gắn tag là `dockerhub_username/ie213-backend:latest` và được push lên Docker Hub

- Frontend:

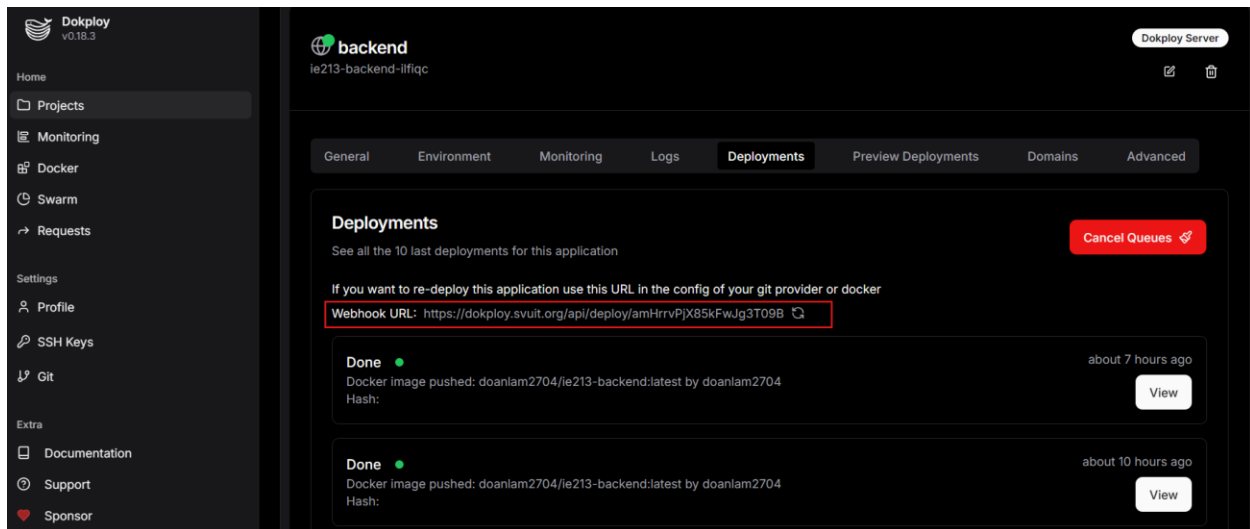
- Bước 1: Tải mã nguồn của repository về máy ảo đang chạy Github Action.
- Bước 2: Đăng nhập vào Docker Hub bằng thông tin tài khoản được lưu trong GitHub Secrets.
- Bước 3: Build docker image từ thư mục ./frontend bằng Dockerfile tương ứng
- Bước 4: Truyền biến môi trường `REACT_APP_BACKEND_URL` vào quá trình build
- Bước 5: Image sau khi build sẽ được push lên Docker Hub với tag `dockerhub_username/ie213-backend:latest`

### 5.1.4. Cấu hình Dokploy để tự động cập nhật Docker image từ Docker Hub

Dokploy được cấu hình để tự động kiểm tra và cập nhật phiên bản mới nhất của Docker image từ Docker Hub. Mỗi khi một image mới được push lên, Dokploy sẽ tự động triển khai lại container tương ứng, đảm bảo ứng dụng luôn chạy với phiên bản mới nhất mà không cần thao tác thủ công.

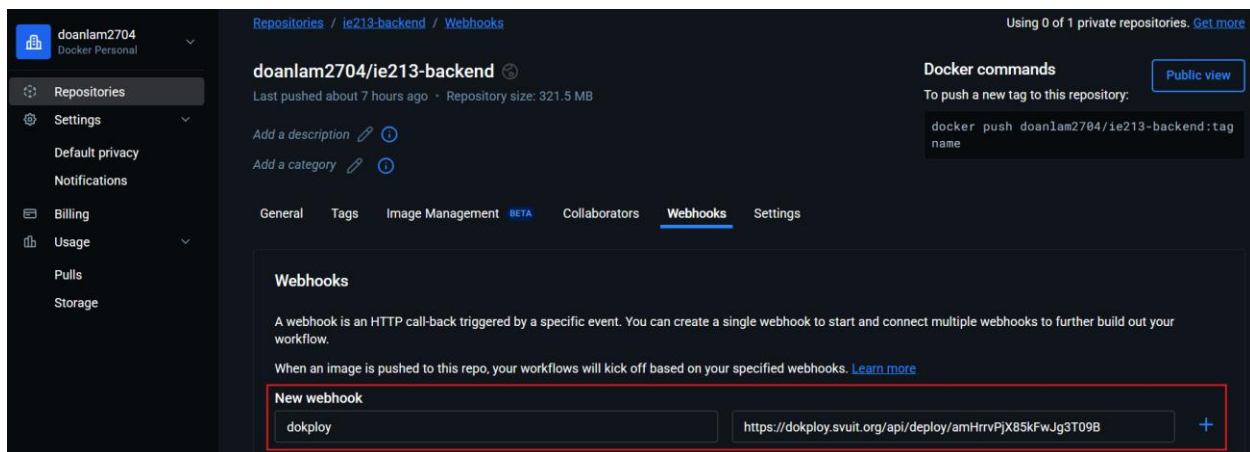
Các bước thực hiện:

- Bước 1: Chọn service muốn tự động cập nhật, chuyển sang tab Deployments và sao chép Webhook URL.



Hình 44: Webhook URL của Dokploy

- Bước 2: Truy cập vào Docker Hub và chọn image cần kiểm tra
- Bước 3: Chuyển sang tab Webhooks và tạo Webhook mới bằng Webhook URL đã copy



Hình 45: Thêm webhook vào Docker Hub

## 5.2. Kiểm thử

Trong quá trình phát triển hệ thống, nhóm đã thực hiện kiểm thử backend bằng framework **Mocha**, kết hợp với **Supertest** nhằm đảm bảo các API hoạt động đúng chức năng. Việc kiểm thử tập trung vào một số chức năng chính như đăng nhập, thanh toán và xử lý giỏ hàng.

### a) Môi trường kiểm thử

- **Ngôn ngữ:** JavaScript (Node.js)

- **Thư viện kiểm thử:** Mocha, Chai, Supertest
- **Phương pháp:** Kiểm thử chức năng (Functional Testing)
- **Kiểu kiểm thử:** Tự động (Automated Unit & Integration Test)

**b) Các ca kiểm thử chính**

STT	Chức năng	Mô tả kiểm thử	Kết quả mong đợi	Kết quả thực tế
1	Đăng nhập thành công	Gửi email và mật khẩu hợp lệ đến API /login	Nhận được token và thông tin người dùng	Đạt
2	Đăng nhập thất bại	Gửi thông tin đăng nhập sai	Nhận mã lỗi 401 và thông báo lỗi	Đạt
3	Tạo URL thanh toán VNPAY	Gửi đơn hàng với số tiền và mã đơn hàng đến /vnpay/create_payment	Nhận paymentUrl chứa tham số VNPAY	Đạt
4	Sai checksum VNPAY	Giả lập việc chỉnh sửa checksum VNPAY và gọi /vnpay_return	Nhận mã lỗi 400 và thông báo checksum sai	Đạt

*Bảng 4: Danh sách các ca kiểm thử chính*



## **CHƯƠNG 6. Kết luận**

### **6.1. Ưu điểm**

- Thiết kế responsive, tương thích tốt với nhiều nền tảng và kích thước màn hình khác nhau.
- Cung cấp giao diện quản trị dành cho admin, cho phép quản lý sản phẩm, đơn hàng và theo dõi hoạt động hệ thống.
- Tính năng lọc, tìm kiếm và sắp xếp sản phẩm linh hoạt, hỗ trợ nhiều tiêu chí kết hợp để người dùng dễ dàng tìm được sản phẩm mong muốn.
- Tích hợp cổng thanh toán VNPAY, hỗ trợ thanh toán trực.
- Ứng dụng mô hình Gemini AI để hỗ trợ admin tự động sinh mô tả sản phẩm nhanh chóng và hiệu quả, giúp quá trình thêm sản phẩm nhanh chóng hơn.
- Giao diện người dùng thân thiện, trực quan, dễ sử dụng với hầu hết các đối tượng người dùng.
- Yêu cầu đăng nhập trước khi mua hàng: người dùng cần đăng nhập để thêm sản phẩm vào giỏ và đặt hàng. Mật khẩu được mã hóa trong quá trình đăng nhập và lưu trữ, đảm bảo tính bảo mật thông tin cá nhân.
- Các thành viên có tinh thần đoàn kết giúp đỡ lẫn nhau khi gặp khó khăn trong việc thực hiện đồ án.

### **6.2. Nhược điểm**

- Một số thành phần giao diện chưa được tối ưu hóa hoàn toàn, dẫn đến trải nghiệm người dùng chưa thực sự mượt mà.
- Chưa hỗ trợ đăng nhập bằng các nền tảng bên thứ ba như Google hoặc Facebook.
- Một số đoạn mã xử lý phía backend chưa được hoàn thiện và tối ưu, cần cải tiến để nâng cao hiệu suất và độ ổn định.
- Chức năng đánh giá (review) sản phẩm vẫn chưa được triển khai, gây thiếu sót trong việc thu thập phản hồi từ người dùng.
- Do hạn chế về thời gian và kinh nghiệm nên một số tính năng đã đề ra vẫn chưa được thực hiện.

### **6.3. Hướng phát triển**

- Tiến hành điều chỉnh và cải thiện một số thành phần giao diện nhằm nâng cao trải nghiệm người dùng.
- Xây dựng thêm chức năng đánh giá sản phẩm và tích hợp chatbot hỗ trợ tư vấn khách hàng tự động.
- Hoàn thiện và tối ưu các đoạn mã xử lý để nâng cao hiệu năng và độ ổn định của hệ thống.
- Cải tiến giao diện phân tích dữ liệu trong trang quản trị (admin) để trực quan, dễ theo dõi và phân tích hơn.
- Tích hợp chức năng đăng nhập bằng các nền tảng bên thứ ba như Google và Facebook nhằm tăng tính tiện lợi cho người dùng.