

**PHENIKAA UNIVERSITY**  
**FACULTY OF COMPUTER SCIENCE**



**BÁO CÁO BÀI TẬP**  
**MÔN HỌC: ĐỒ ÁN LIÊN NGÀNH**

***Đề tài: Phát triển website bán giày trực tuyến dựa trên nền tảng Evershop***

**Thành viên nhóm**  
**Lê Quang Minh                      22010483**

**Hà Nội, 2025**

# MỤC LỤC

	1
1. GIỚI THIỆU CHUNG	3
1.1 Lý do chọn đề tài	3
1.2 Mục tiêu nghiên cứu	4
1.3 Phạm vi và giới hạn	4
1.4 Phương pháp nghiên cứu	4
2. TỔNG QUAN NỀN TẢNG EVERSHOP	4
2.1 Giới thiệu EverShop — khái niệm và đặc điểm	5
2.2 Công nghệ nền tảng: Node.js, PostgreSQL, React, GraphQL	5
2.3 Kiến trúc mô-đun và cấu trúc thư mục chính	6
2.4 So sánh với các nền tảng thương mại điện tử mã nguồn mở khác	7
3. KIẾN TRÚC HỆ THỐNG CHI TIẾT	9
Tổng quan ba tầng chính:	9
3.1 Thành phần Frontend: Storefront & Theme System	10
3.2 Thành phần Backend: API, GraphQL, dịch vụ & module mở rộng	11
3.3 Cơ sở dữ liệu và tầng lưu trữ (PostgreSQL)	12
3.4 Mô-hình mở rộng (Extensions, Plugins) và cách tương tác với lõi	13
4. CHỨC NĂNG SẴN CÓ CỦA EVERSHOP	13
4.1 Chức năng phía người dùng (Storefront)	13
4.2 Chức năng phía quản trị viên (Admin Panel)	16
4.3 Các chức năng hỗ trợ hệ thống: xác thực, phân quyền, bảo mật, quản lý dữ liệu	18
5. USE-CASE VÀ LƯỒNG HOẠT ĐỘNG HỆ THỐNG	19
5.1 Định nghĩa tác nhân (Actors) và vai trò	19
5.2 Luồng chính: trình duyệt sản phẩm, thêm giỏ hàng, thanh toán, quản lý đơn hàng, mở rộng module	19
5.3 Luồng hoạt động chi tiết ví dụ: tạo đơn hàng / mở rộng module mới	21
6. CÀI ĐẶT, CẤU HÌNH VÀ TRIỂN KHAI	22
6.1 Yêu cầu hệ thống & cấu hình ban đầu (theo hướng dẫn từ <a href="https://evershop.io">evershop.io</a> )	22
6.2 Hướng dẫn cài đặt nền tảng	23
6.3 Quy trình cấu hình store, theme, extensions	24
6.5 Giao diện minh hoạ	25
6.5.1 Giao diện người dùng (Storefront)	26
6.5.2 Giao diện quản trị (Admin Panel)	27
7. TÙY BIẾN, MỞ RỘNG VÀ PHÁT TRIỂN THÊM	28
7.1. Khái niệm về Extension trong EverShop	28
7.2. Quy trình tạo một Extension mới	29
7.3 Extension đã mở rộng 1: google_login	31
7.4 Extension đã mở rộng 2: product_review	33
7.5. Trang tùy chỉnh: About-Me Page	33
Mục tiêu	33
1 Tạo cấu trúc thư mục extension	33

2 Thêm cấu hình extension vào file config/default.json	34
3 Cấu hình route cho trang About-Me	35
8. KẾT QUẢ, ĐÁNH GIÁ VÀ KHUYẾN NGHỊ	37
8.1 Kết quả đạt được khi triển khai demo hoặc prototype	37
8.2 Đánh giá ưu điểm và hạn chế của nền tảng	38
8.3 Khuyến nghị và hướng phát triển tiếp theo	38
9. KẾT LUẬN	39
10. TÀI LIỆU THAM KHẢO	39

## 1. GIỚI THIỆU CHUNG

### 1.1 Lý do chọn đề tài

Trong bối cảnh kinh tế số hóa toàn cầu, các nền tảng thương mại điện tử mã nguồn mở đang trở thành lựa chọn hàng đầu cho các doanh nghiệp vừa và nhỏ cũng như các startup muốn xây dựng cửa hàng trực tuyến. EverShop là một nền tảng thương mại điện tử mã nguồn mở được phát triển dựa trên công nghệ Node.js, React, và GraphQL, cung cấp một kiến trúc mô-đun linh hoạt và dễ mở rộng. Việc lựa chọn EverShop làm chủ đề nghiên cứu nhằm giúp sinh viên, giảng viên và các nhà phát triển hiểu rõ cấu trúc, chức năng và quy trình triển khai của một nền tảng thương mại điện tử hiện đại. Điều này có ý nghĩa quan trọng trong việc nâng cao kỹ năng phát triển ứng dụng web và hiểu biết về các công nghệ backend-frontend tiên tiến.

### 1.2 Mục tiêu nghiên cứu

Báo cáo này nhằm các mục tiêu chính sau đây:

- Cung cấp cái nhìn toàn diện về nền tảng EverShop, bao gồm các đặc điểm chính, công nghệ nền tảng, và kiến trúc hệ thống.
- Giải thích chi tiết các thành phần khác nhau của EverShop từ phía người dùng (storefront) đến phía quản trị viên (admin panel), cũng như các chức năng hỗ trợ hệ thống.
- Hướng dẫn cách cài đặt, cấu hình và triển khai EverShop lên môi trường thực tế.
- Phân tích các use-case thực tế và luồng hoạt động của hệ thống.
- Đưa ra đánh giá khách quan về ưu điểm, hạn chế và các hướng phát triển tiếp theo của nền tảng.

### 1.3 Phạm vi và giới hạn

Phạm vi báo cáo tập trung vào các khía cạnh kỹ thuật của EverShop, bao gồm kiến trúc hệ thống, các module chính, API, cơ sở dữ liệu, và quy trình cài đặt. Báo cáo không bao gồm các khía cạnh kinh doanh như định giá, mô hình thu nhập hay chiến lược tiếp thị. Báo cáo được viết dựa trên thông tin chính thức từ trang web <https://evershop.io/> và các tài liệu kỹ thuật liên quan.

## 1.4 Phương pháp nghiên cứu

Báo cáo sử dụng phương pháp nghiên cứu tài liệu kỹ thuật, kết hợp với phân tích so sánh và mô tả chi tiết. Các thông tin được thu thập từ:

- Trang chính thức của EverShop tại <https://evershop.io/>
- Tài liệu phát triển (documentation) chính thức của EverShop
- Các hướng dẫn cài đặt và hướng dẫn sử dụng từ cộng đồng
- So sánh với các nền tảng thương mại điện tử mã nguồn mở khác
- Tài liệu kỹ thuật về các công nghệ liên quan (Node.js, PostgreSQL, React, GraphQL)

Phương pháp tiếp cận là từ macro (tổng quan nền tảng) đến micro (chi tiết từng thành phần), sau đó là các ứng dụng thực tế và khuyến nghị.

## 2. TỔNG QUAN NỀN TẢNG EVERSHOP

### 2.1 Giới thiệu EverShop — khái niệm và đặc điểm

EverShop là một nền tảng thương mại điện tử mã nguồn mở được xây dựng dựa trên công nghệ Node.js và PostgreSQL. Đây là một ứng dụng web hiện đại, kết hợp cả thành phần backend và frontend trong một kiến trúc thống nhất (monolithic architecture). Tên "EverShop" có thể được hiểu là một cửa hàng vĩnh viễn — một nền tảng có thể mở rộng vô hạn để đáp ứng nhu cầu kinh doanh.

Các đặc điểm nổi bật của EverShop bao gồm:

**Mã nguồn mở:** EverShop được phát hành theo giấy phép GPL-3.0, cho phép người dùng tự do sửa đổi, phân phối và sử dụng mã nguồn. Điều này tạo ra cộng đồng nhà phát triển tích cực đóng góp vào sự phát triển của nền tảng.

**Kiến trúc mô-đun:** Tất cả chức năng được tổ chức thành các module độc lập. Mỗi module đại diện cho một tính năng kinh doanh cụ thể, từ quản lý sản phẩm (catalog) đến xử lý đơn hàng (checkout) đến thanh toán (stripe). Các module này có thể được tắt, bật hoặc mở rộng một cách độc lập.

**Giao diện quản trị toàn diện:** EverShop cung cấp một admin panel được xây dựng bằng React, tương tự như Shopify. Giao diện này cho phép các nhà quản trị quản lý

sản phẩm, đơn hàng, khách hàng, cấu hình thanh toán, vận chuyển, và nhiều khía cạnh khác của cửa hàng.

**Storefront tùy biến:** Phía khách hàng (storefront) được xây dựng bằng React, hỗ trợ chủ đề (themes) tùy biến. Các nhà phát triển có thể tạo các chủ đề khác nhau mà không cần sửa đổi mã nguồn cốt lõi của nền tảng.

**API-first architecture:** EverShop được thiết kế với kiến trúc ưu tiên API, cung cấp cả REST API và GraphQL API. Điều này cho phép các ứng dụng bên thứ ba tương tác linh hoạt với hệ thống.

## 2.2 Công nghệ nền tảng: Node.js, PostgreSQL, React, GraphQL

Để hiểu rõ hơn về EverShop, cần phải nắm vững các công nghệ nền tảng mà nó được xây dựng trên đó.

**Node.js:** Đây là một môi trường chạy JavaScript phía máy chủ, cho phép các nhà phát triển sử dụng JavaScript để viết backend. Node.js nổi tiếng vì khả năng xử lý các request không đồng bộ (asynchronous) một cách hiệu quả, giúp tăng hiệu suất khi xử lý nhiều client kết nối đồng thời. Evershop yêu cầu Node.js phiên bản 14 hoặc cao hơn.

**PostgreSQL:** Đây là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở, độ tin cậy cao, khả năng mở rộng, và nhiều tính năng nâng cao. PostgreSQL hỗ trợ các kiểu dữ liệu phức tạp như JSON, arrays, và XML. Nó tuân thủ ACID (Atomicity, Consistency, Isolation, Durability), đảm bảo tính toàn vẹn dữ liệu. EverShop yêu cầu PostgreSQL phiên bản 13 hoặc cao hơn.

**React:** Đây là một thư viện JavaScript được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng. React sử dụng khái niệm "component" để chia nhỏ UI thành các phần tử có thể tái sử dụng. Các component này là tự quản lý (self-managing), có trạng thái (state) riêng và có thể kết hợp lại để tạo thành UI phức tạp. React sử dụng Virtual DOM, giúp cải thiện hiệu suất bằng cách chỉ cập nhật những phần thực sự thay đổi trên DOM thực.

**GraphQL:** Đây là ngôn ngữ truy vấn và runtime cho API được phát triển bởi Facebook. GraphQL cho phép client chỉ định chính xác những dữ liệu nó cần, giảm số lượng dữ liệu không cần thiết được truyền qua mạng. Thay vì có nhiều endpoint REST, GraphQL có một endpoint duy nhất nhưng hỗ trợ các truy vấn phức tạp. Ví dụ, client có thể yêu cầu thông tin sản phẩm, giá cả, và ảnh trong một truy vấn duy nhất.

**Express.js:** Đây là framework Node.js nhẹ, được sử dụng để xây dựng các API RESTful. Express cung cấp các middleware, routing, và các công cụ khác để xây dựng backend hiệu quả.

**Tailwind CSS:** Đây là framework CSS utility-first, cho phép các nhà phát triển xây dựng giao diện bằng cách kết hợp các lớp utility được định nghĩa trước. Tailwind giúp tạo giao diện nhất quán và có thể tùy biến dễ dàng mà không cần viết CSS tùy chỉnh.

**Webpack:** Đây là một bundler (công cụ đóng gói) module cho JavaScript, cho phép các nhà phát triển kết hợp các tệp JavaScript, CSS, hình ảnh, và các tài nguyên khác thành các bundle tối ưu hóa cho production.

## 2.3 Kiến trúc mô-đun và cấu trúc thư mục chính

EverShop được tổ chức thành các module độc lập, mỗi module chứa logic, dữ liệu, và giao diện cho một tính năng kinh doanh cụ thể. Cấu trúc của một module điển hình như sau:

```
modules/  
├── catalog/  
│   ├── src/  
│   │   ├── api/  
│   │   │   ├── createProduct/  
│   │   │   ├── updateProduct/  
│   │   │   └── ...  
│   │   ├── pages/  
│   │   │   ├── admin/  
│   │   │   │   ├── productEdit/  
│   │   │   │   ├── productNew/  
│   │   │   │   └── ...  
│   │   │   ├── frontStore/  
│   │   │   │   ├── productView/  
│   │   │   │   ├── categoryView/  
│   │   │   │   └── ...  
│   │   │   └── global/  
│   │   ├── graphql/  
│   │   │   └── types/  
│   │   │       ├── Product/  
│   │   │       ├── Category/  
│   │   │       └── ...  
│   │   ├── services/  
│   │   ├── migration/  
│   │   ├── bootstrap.ts  
│   │   └── package.json  
└── ...
```

Mỗi module có các thành phần sau:

- **api/**: Chứa các endpoint REST API cho module
- **pages/**: Chứa các component React cho admin và storefront
- **graphql/**: Định nghĩa các kiểu dữ liệu GraphQL và resolver
- **services/**: Chứa logic nghiệp vụ, validation, và các service khác
- **migration/**: Chứa các script để tạo hoặc cập nhật cơ sở dữ liệu

Các module chính trong EverShop bao gồm:

- **catalog**: Quản lý sản phẩm, danh mục, thuộc tính
- **checkout**: Xử lý quá trình thanh toán
- **order**: Quản lý đơn hàng
- **customer**: Quản lý khách hàng
- **stripe**: Tích hợp cổng thanh toán Stripe
- **cms**: Quản lý trang tĩnh và nội dung

## 2.4 So sánh với các nền tảng thương mại điện tử mã nguồn mở khác

Để hiểu rõ vị trí của EverShop, cần so sánh nó với các nền tảng thương mại điện tử mã nguồn mở khác như Magento, WooCommerce, và OpenCart.

Tiêu chí	EverShop	Magento	WooCommerce	OpenCart
Công nghệ	Node.js, React, PostgreSQL	PHP, MySQL/MariaDB	PHP, MySQL	PHP, MySQL
Độ dễ sử dụng	Trung bình - cần kiến thức về JavaScript	Phức tạp - cần kiến thức sâu	Dễ - tích hợp WordPress	Trung bình
Mô đun hóa	Rất cao - kiến trúc mô-đun mạnh	Cao	Trung bình	Trung bình
Khả năng mở rộng	Cao	Rất cao	Trung bình	Trung bình

<b>Hiệu suất</b>	Cao - Node.js xử lý async tốt	Cao - nhưng cần cấu hình cẩn thận	Trung bình - phụ thuộc vào plugins	Trung bình
<b>Curva học tập</b>	Trung bình	Cao	Thấp	Trung bình
<b>Cộng đồng</b>	Đang phát triển	Rất lớn	Rất lớn	Nhỏ
<b>Admin Panel</b>	React-based, hiện đại	PHP-based, cũ nhưng mạnh	WordPress admin	jQuery-based
<b>API</b>	REST + GraphQL	REST + SOAP	REST plugin-based	REST extension-based

EverShop nổi bật với kiến trúc mô-đun mạnh mẽ, công nghệ hiện đại (Node.js + React), và API modern (GraphQL). Tuy nhiên, cộng đồng chưa lớn như Magento hay WooCommerce, và khiến đó là một thách thức cho những người mới bắt đầu.

### 3. KIẾN TRÚC HỆ THỐNG CHI TIẾT

EverShop được xây dựng theo kiến trúc mô-đun ba tầng (Three-Layer Architecture), kết hợp với mô hình Headless Commerce hiện đại, cho phép tách biệt giữa giao diện người dùng (frontend) và xử lý dữ liệu (backend).

Tổng quan ba tầng chính:

Tầng	Thành phần chính	Chức năng
Frontend (Storefront & Admin UI)	React, TailwindCSS, Next UI	Giao diện tương tác với người dùng, hiển thị sản phẩm, giỏ hàng, thanh toán, và trang quản trị.
Backend (API Server)	Node.js, Express, GraphQL	Cung cấp các API (GraphQL & REST), xử lý logic nghiệp vụ, xác thực, thanh toán, và quản lý cơ sở dữ liệu.



Database Layer (Data Storage)	PostgreSQL	Lưu trữ toàn bộ dữ liệu về sản phẩm, khách hàng, đơn hàng, đánh giá, và cấu hình cửa hàng.
----------------------------------	------------	--

### Cấu trúc thư mục hệ thống :

```

MY-APP/
├── .evershop/                # Cấu hình hệ thống nội bộ của EverShop
├── config/                  # Thư mục chứa tệp cấu hình (default.json)
│
├── extensions/              # Các extension mở rộng do người dùng phát triển
│   └── aboutMe/              # Ví dụ: Extension trang giới thiệu cá nhân (/about-me)
│
├── media/                   # Lưu trữ hình ảnh sản phẩm, banner, logo
├── node_modules/            # Các thư viện npm được cài đặt
├── public/                  # Tài nguyên tĩnh (ảnh, biểu tượng, tập tin JS/CSS)
│
├── themes/                  # Giao diện (theme) của Storefront
│   └── sample/              # Theme mẫu mặc định
│
├── .env                     # File cấu hình môi trường (DB, API Key, Stripe,...)
└── package.json             # Khai báo phụ thuộc và script khởi chạy dự án

```

## 3.1 Thành phần Frontend: Storefront & Theme System

Frontend của EverShop bao gồm hai phần chính: storefront (cửa hàng) và admin panel. Cả hai đều được xây dựng bằng React, nhưng phục vụ các mục đích khác nhau.

**Storefront** là phía công cộng của cửa hàng, nơi khách hàng tương tác để duyệt sản phẩm, thêm vào giỏ hàng, và thực hiện thanh toán. Storefront được thiết kế để tối ưu hóa khả năng tùy biến thông qua một hệ thống chủ đề (theme system) linh hoạt.

Hệ thống chủ đề EverShop cho phép các nhà phát triển tạo ra các giao diện độc đáo mà không cần sửa đổi mã lõi của nền tảng. Mỗi chủ đề mà nhà phát triển muốn thay đổi sẽ được tổ chức thành các folder như sau:

```
themes/  
└─ themeName/  
    ├── public/           # Hình ảnh, font, tài nguyên tĩnh  
    ├── src/  
    │   ├── components/  # Component React tái sử dụng  
    │   └── pages/  
    │       ├── all/      # Component dùng cho tất cả trang  
    │       ├── homepage/  
    │       ├── productView/  
    │       └── checkout/  
    ├── package.json  
    └── tsconfig.json
```

Các trang trong chủ đề được tổ chức thành các component React. EverShop cung cấp khái niệm "master component" — component cấp cao nhất của một trang. Các component này được tự động tải, được ghi đè lên component cũ và kết hợp lại để tạo nên trang hoàn chỉnh.

**Ví dụ:** Trang xem sản phẩm (productView) có thể bao gồm các master component như:

- ProductImages.tsx - Hiển thị hình ảnh sản phẩm
- ProductInfo.tsx - Hiển thị thông tin sản phẩm
- ProductOptions.tsx - Hiển thị các tùy chọn sản phẩm

Các component này được override (ghi đè) bởi chủ đề để tùy biến giao diện mà không cần thay đổi logic cốt lõi.

EverShop sử dụng Tailwind CSS làm framework CSS chính, cho phép các nhà phát triển tạo giao diện nhanh chóng bằng các lớp utility được định nghĩa trước. Các nhà phát triển cũng có thể tạo các file SCSS tùy chỉnh nếu cần.

### 3.2 Thành phần Backend: API, GraphQL, dịch vụ & module mở rộng

Backend của EverShop được xây dựng trên Node.js với Express.js. Backend cung cấp hai loại API: REST API và GraphQL API.

**REST API** được sử dụng chủ yếu cho các thao tác tạo, cập nhật, xóa (CRUD operations). Các endpoint REST được tổ chức theo module, với các file route.json định nghĩa các chi tiết của endpoint (HTTP method, path, access control).

Ví dụ, endpoint tạo sản phẩm:

```
{
  "methods": ["POST"],
  "path": "/api/products",
  "access": "admin"
}
```

**GraphQL API** được sử dụng chủ yếu cho truy vấn dữ liệu. GraphQL cho phép client chỉ định chính xác những dữ liệu nó cần. Mỗi module định nghĩa các kiểu GraphQL riêng của nó:

```
catalog/
├─ graphql/
│   └─ types/
│       ├── Product/
│       │   ├── Product.graphql
│       │   └── Product.resolvers.ts
│       ├── Category/
│       ├── Attribute/
│       └── ...
```

Các file `.graphql` chứa định nghĩa kiểu, còn các file `.resolvers.ts` chứa logic để lấy dữ liệu từ cơ sở dữ liệu.

**Services** là các lớp logic nghiệp vụ độc lập, không liên quan trực tiếp đến API. Services được sử dụng bởi API endpoints để xử lý logic phức tạp. Ví dụ, `ProductValidator` service có thể kiểm tra xem thông tin sản phẩm có hợp lệ hay không.

**Middleware** là các hàm được thực thi trước hoặc sau API endpoint chính. EverShop sử dụng hệ thống middleware để xử lý các khía cạnh như xác thực, validation dữ liệu, và xử lý lỗi.

### 3.3 Cơ sở dữ liệu và tầng lưu trữ (PostgreSQL)

EverShop sử dụng PostgreSQL làm cơ sở dữ liệu chính. PostgreSQL là một RDBMS mạnh mẽ, hỗ trợ ACID compliance, giúp đảm bảo tính toàn vẹn dữ liệu.

Để kết nối với PostgreSQL, EverShop sử dụng các biến môi trường được lưu trong file `.env`:

```
DB_HOST=localhost
DB_PORT=5432
DB_USER=postgres
DB_PASSWORD=postgres
DB_NAME=evershop
```

```
DB_SSLMODE=disable
```

EverShop cung cấp một lớp truy cập cơ sở dữ liệu trừu tượng (database abstraction layer), giúp các nhà phát triển tương tác với cơ sở dữ liệu mà không cần viết SQL trực tiếp. Thay vào đó, EverShop sử dụng một query builder:

```
import { pool } from "@evershop/evershop/lib/postgres";
import { select } from "@evershop/postgres-query-builder";

const query = select()
  .from("product")
  .where("status", "=", 1);

const products = await query.load(pool);
```

Các bảng chính trong EverShop bao gồm:

- **product**: Lưu trữ thông tin sản phẩm
- **product\_description**: Lưu trữ mô tả sản phẩm
- **order**: Lưu trữ thông tin đơn hàng
- **order\_item**: Lưu trữ các sản phẩm trong đơn hàng
- **customer**: Lưu trữ thông tin khách hàng
- **category**: Lưu trữ danh mục sản phẩm

EverShop cung cấp hỗ trợ transaction, cho phép các nhà phát triển thực hiện các thao tác cơ sở dữ liệu phức tạp trong một transaction duy nhất, đảm bảo tất cả các thao tác thành công hoặc tất cả đều không thành công (atomicity).

### 3.4 Mô-hình mở rộng (Extensions, Plugins) và cách tương tác với lõi

EverShop được thiết kế để dễ mở rộng thông qua hệ thống extension (tiện ích mở rộng). Các extension có thể thêm các tính năng mới hoặc sửa đổi các tính năng hiện có mà không cần sửa đổi mã lõi.

Các extension được cấu hình trong file config/default.json:

```
{
  "system": {
    "extensions": [
      {
        "name": "google_login",
        "resolve": "node_modules/@evershop/google_login",
        "enabled": true,
        "priority": 10
      }
    ]
  }
}
```

```
]
}
}
```

Các extension hoạt động bằng cách kết nối vào các điểm mở rộng (extension points) được định nghĩa bởi lõi hoặc các module khác. Ví dụ, một extension có thể kết nối vào GraphQL type "Product" để thêm một field mới:

```
extend type Product {
  customField: String
}
```

Các extension được tổ chức giống như module, với các thư mục api, pages, graphql, services, và migration. Điều này tạo ra một cấu trúc nhất quán và dễ hiểu cho các nhà phát triển.

## 4. CHỨC NĂNG SẴN CÓ CỦA EVERSHOP

### 4.1 Chức năng phía người dùng (Storefront)

Storefront của EverShop cung cấp đầy đủ các chức năng tiêu chuẩn của một cửa hàng thương mại điện tử:

**Duyệt sản phẩm:** Khách hàng có thể duyệt danh sách các sản phẩm được sắp xếp theo danh mục. Mỗi sản phẩm hiển thị hình ảnh, tên, mô tả, giá, và tình trạng hàng.

**Trang chi tiết sản phẩm:** Bằng cách nhấp vào một sản phẩm, khách hàng xem trang chi tiết sản phẩm với thông tin toàn diện, bao gồm mô tả dài, hình ảnh bổ sung, tùy chọn sản phẩm (kích cỡ, màu sắc, v.v.), giá, và tình trạng hàng.

**Giỏ hàng:** Khách hàng có thể thêm sản phẩm vào giỏ hàng, xem tổng giá, chỉnh sửa số lượng, hoặc xóa sản phẩm khỏi giỏ hàng.

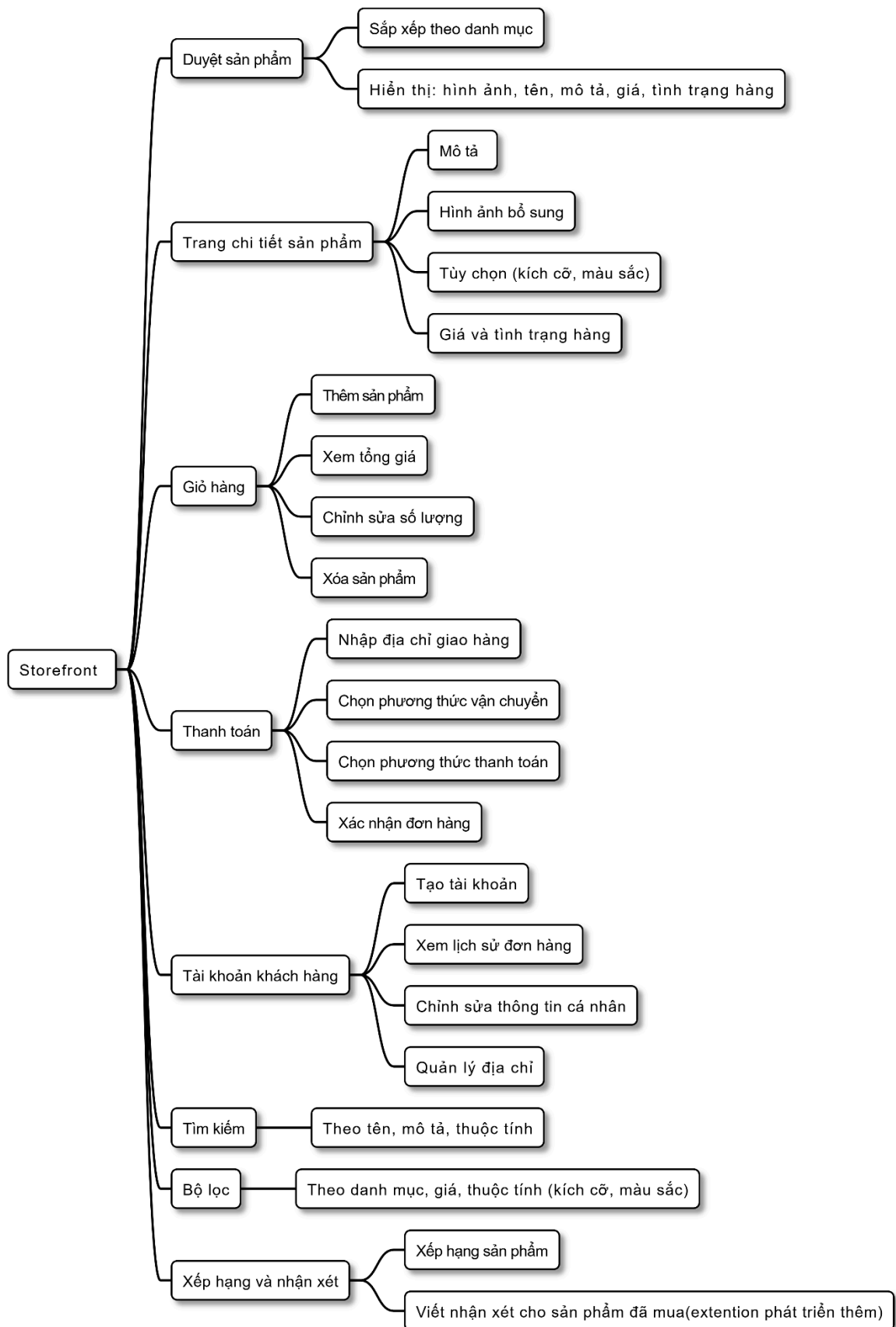
**Thanh toán:** Quá trình thanh toán được thiết kế đơn giản và an toàn. Khách hàng nhập thông tin địa chỉ giao hàng, chọn phương thức vận chuyển, chọn phương thức thanh toán, và xác nhận đơn hàng.

**Tài khoản khách hàng:** Khách hàng có thể tạo tài khoản, xem lịch sử đơn hàng, chỉnh sửa thông tin cá nhân, và quản lý địa chỉ.

**Tìm kiếm:** Khách hàng có thể tìm kiếm sản phẩm theo tên, mô tả, hoặc thuộc tính.

**Bộ lọc:** Khách hàng có thể lọc sản phẩm theo danh mục, giá, thuộc tính (kích cỡ, màu sắc, v.v.).

**Xếp hạng và nhận xét:** Khách hàng có thể xếp hạng và viết nhận xét cho các sản phẩm họ đã mua.



## 4.2 Chức năng phía quản trị viên (Admin Panel)

Admin panel của EverShop cung cấp các chức năng quản lý toàn diện:

**Quản lý sản phẩm:** Quản trị viên có thể tạo, chỉnh sửa, xóa sản phẩm. Với mỗi sản phẩm, quản trị viên có thể:

- Nhập thông tin cơ bản (tên, SKU, giá, trọng lượng)
- Thêm mô tả và mô tả ngắn
- Tải lên hình ảnh
- Chỉ định danh mục
- Tạo các biến thể sản phẩm (variants)
- Quản lý tồn kho

**Quản lý danh mục:** Quản trị viên có thể tạo, chỉnh sửa, xóa danh mục sản phẩm, tổ chức sản phẩm theo cấp bậc.

**Quản lý đơn hàng:** Quản trị viên có thể xem các đơn hàng, cập nhật trạng thái đơn hàng, tạo hoá đơn, quản lý vận chuyển.

**Quản lý khách hàng:** Quản trị viên có thể xem danh sách khách hàng, xem lịch sử mua hàng của từng khách hàng.

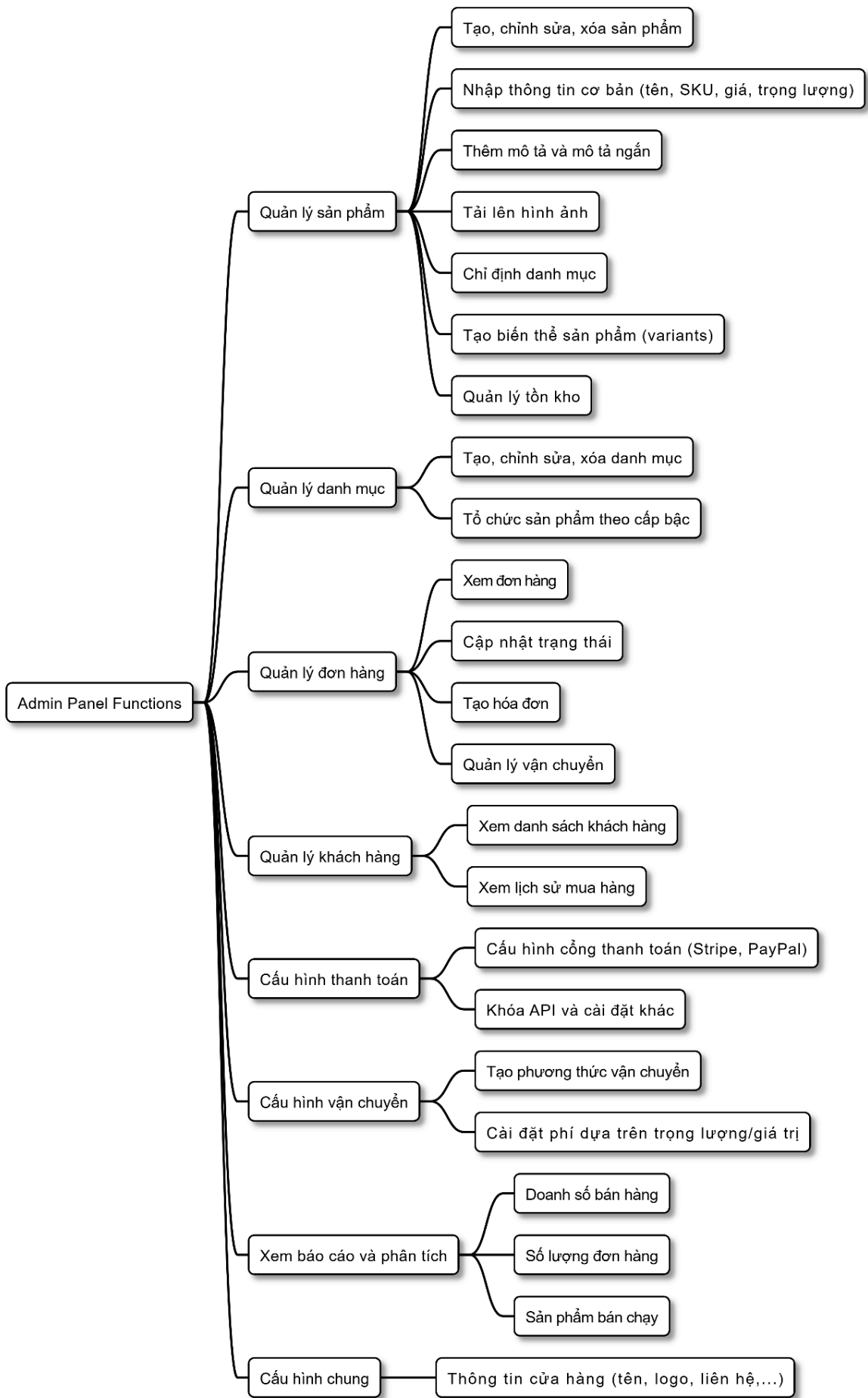
**Cấu hình thanh toán:** Quản trị viên có thể cấu hình các cổng thanh toán được hỗ trợ (Stripe, PayPal, v.v.), bao gồm khóa API và các cài đặt khác.

**Cấu hình vận chuyển:** Quản trị viên có thể tạo các phương thức vận chuyển, cài đặt phí vận chuyển dựa trên trọng lượng hoặc giá trị đơn hàng.

**Báo cáo và phân tích:** Admin panel cung cấp các báo cáo về doanh số bán hàng, số lượng đơn hàng, top sản phẩm bán chạy.

**Cấu hình chung:** Quản trị viên có thể cấu hình thông tin cửa hàng (tên, logo, thông tin liên hệ), múi giờ...





### 4.3 Các chức năng hỗ trợ hệ thống: xác thực, phân quyền, bảo mật, quản lý dữ liệu

**Xác thực (Authentication):** EverShop cung cấp xác thực dựa trên cookie. Khi người dùng admin đăng nhập, hệ thống tạo một session và lưu trữ session ID trong cookie. Các request tiếp theo phải bao gồm cookie này để xác minh danh tính của người dùng.

EverShop cũng hỗ trợ các phương thức xác thực thứ ba thông qua các extension(cần phát triển riêng), chẳng hạn như Google Login. Khách hàng có thể đăng nhập bằng tài khoản Google của họ mà không cần tạo tài khoản riêng.

**Phân quyền (Authorization):** EverShop cung cấp hệ thống phân quyền dựa trên vai trò (role-based access control). Mỗi người dùng admin có một vai trò (ví dụ: quản trị viên, quản lý sản phẩm), và mỗi vai trò có một tập hợp các quyền (permission) xác định các hành động mà người dùng có thể thực hiện.

Các quyền được kiểm tra ở cấp API, đảm bảo rằng chỉ những người dùng được phép mới có thể truy cập các endpoint cụ thể.

**Bảo mật:** EverShop áp dụng các biện pháp bảo mật tiêu chuẩn:

- **SQL Injection Prevention:** Query builder của EverShop tự động escape các giá trị, ngăn chặn tấn công SQL injection.
- **CSRF Protection:** EverShop sử dụng token CSRF để bảo vệ chống lại các tấn công Cross-Site Request Forgery.
- **Password Hashing:** Mật khẩu người dùng được hash (băm) bằng các thuật toán mạnh trước khi lưu trữ.
- **HTTPS Support:** EverShop hỗ trợ kết nối HTTPS an toàn.
- **SSL Mode:** PostgreSQL có thể được cấu hình để sử dụng kết nối SSL an toàn.

**Quản lý dữ liệu:**

- **Backup:** EverShop không cung cấp công cụ backup tích hợp, nhưng vì nó sử dụng PostgreSQL, các nhà phát triển có thể sử dụng các công cụ backup PostgreSQL tiêu chuẩn.
  - **Migration:** EverShop cung cấp hệ thống migration để quản lý các thay đổi cơ sở dữ liệu theo phiên bản.
  - **Data Validation:** EverShop xác thực dữ liệu ở cấp API trước khi lưu trữ vào cơ sở dữ liệu.
-

## 5. USE-CASE VÀ LƯỒNG HOẠT ĐỘNG HỆ THỐNG

### 5.1 Định nghĩa tác nhân (Actors) và vai trò

Trong hệ thống EverShop, có các tác nhân (actors) chính sau:

**Khách hàng (Customer):** Đây là người dùng cuối tương tác với storefront. Khách hàng duyệt sản phẩm, thêm vào giỏ hàng, thanh toán, và quản lý tài khoản của mình. Khách hàng có thể là khách hàng đã đăng ký hoặc khách hàng không đăng ký (guest).

**Quản trị viên (Administrator):** Đây là người quản lý toàn bộ hệ thống. Họ có tất cả các quyền, bao gồm quản lý người dùng, cấu hình cửa hàng, cấu hình thanh toán, cấu hình vận chuyển, và truy cập báo cáo.

**Nhà phát triển (Developer):** Mặc dù không tương tác trực tiếp với EverShop thông qua giao diện người dùng, nhà phát triển có thể mở rộng EverShop bằng cách tạo các extension hoặc chủ đề tùy chỉnh.

### 5.2 Luồng chính: trình duyệt sản phẩm, thêm giỏ hàng, thanh toán, quản lý đơn hàng, mở rộng module

#### Luồng 1: Khách hàng duyệt sản phẩm

Tác nhân: Khách hàng

Mục tiêu: Tìm sản phẩm quan tâm

Luồng hoạt động chính:

1. Khách hàng truy cập trang chủ của cửa hàng
2. Khách hàng nhấp vào một danh mục để xem các sản phẩm trong danh mục đó
3. EverShop gọi GraphQL query để lấy danh sách sản phẩm
4. Danh sách sản phẩm được hiển thị trên trang
5. Khách hàng có thể lọc sản phẩm theo giá hoặc thuộc tính
6. Khách hàng nhấp vào một sản phẩm để xem chi tiết

#### Luồng 2: Khách hàng thêm sản phẩm vào giỏ hàng

Tác nhân: Khách hàng

Mục tiêu: Thêm sản phẩm vào giỏ hàng để chuẩn bị thanh toán

Luồng hoạt động chính:

1. Khách hàng ở trang chi tiết sản phẩm
2. Khách hàng chọn các tùy chọn sản phẩm (kích cỡ, màu sắc, v.v.)
3. Khách hàng nhập số lượng
4. Khách hàng nhấn nút "Thêm vào giỏ hàng"
5. EverShop gọi API endpoint `/api/cart/items` để thêm sản phẩm
6. Backend xác thực dữ liệu, kiểm tra tồn kho
7. Sản phẩm được thêm vào giỏ hàng
8. Frontend cập nhật giao diện để hiển thị số lượng sản phẩm trong giỏ

### **Luồng 3: Khách hàng thanh toán**

Tác nhân: Khách hàng

Mục tiêu: Hoàn tất việc mua hàng

Luồng hoạt động chính:

1. Khách hàng ở trang giỏ hàng
2. Khách hàng nhấn nút "Thanh toán"
3. EverShop hiển thị trang thanh toán
4. Khách hàng nhập hoặc chọn địa chỉ giao hàng
5. Khách hàng chọn phương thức vận chuyển
6. Khách hàng chọn phương thức thanh toán (Stripe, PayPal, COD, v.v.)
7. Nếu chọn Stripe, khách hàng được chuyển hướng đến trang thanh toán Stripe
8. Sau khi thanh toán thành công, đơn hàng được tạo trong EverShop
9. Khách hàng nhận được email xác nhận

### **Luồng 4: Quản trị viên quản lý đơn hàng**

Tác nhân: Quản trị viên

Mục tiêu: Cập nhật trạng thái đơn hàng

Luồng hoạt động chính:

1. Quản trị viên đăng nhập vào admin panel
2. Quản trị viên điều hướng đến phần "Đơn hàng"
3. Quản trị viên xem danh sách các đơn hàng
4. Quản trị viên nhấp vào một đơn hàng để xem chi tiết
5. Quản trị viên xem thông tin đơn hàng: sản phẩm, khách hàng, địa chỉ giao hàng
6. Quản trị viên tạo hoá đơn nếu cần
7. Quản trị viên cập nhật trạng thái đơn hàng (ví dụ: từ "Chờ xử lý" sang "Đang giao")
8. Quản trị viên nhập mã theo dõi vận chuyển
9. Khách hàng được gửi email thông báo về cập nhật đơn hàng

## Luồng 5: Nhà phát triển mở rộng module

Tác nhân: Nhà phát triển

Mục tiêu: Thêm trường tùy chỉnh vào sản phẩm

Luồng chính:

1. Nhà phát triển tạo một extension mới

Extension chứa một GraphQL extension để thêm trường vào kiểu Product:

```
extend type Product {  
  customField: String  
}
```

- 2.
3. Extension chứa một migration để thêm cột vào bảng product
4. Nhà phát triển cấu hình extension trong config/default.json
5. Nhà phát triển chạy npm run build
6. Trường mới bây giờ có sẵn trong admin panel và storefront

## 5.3 Luồng hoạt động chi tiết ví dụ: tạo đơn hàng / mở rộng module mới

### Luồng tạo đơn hàng (Order Creation Flow)

Khách hàng → Storefront

↓

Khách hàng nhấp "Thanh toán"

↓

Storefront gọi GraphQL query để lấy giỏ hàng

↓

Backend trả về thông tin giỏ hàng

↓

Khách hàng nhập địa chỉ giao hàng

↓

Storefront gọi GraphQL query để lấy phương thức vận chuyển

↓

Backend tính phí vận chuyển dựa trên giỏ hàng và địa chỉ

↓

Khách hàng chọn phương thức thanh toán

↓

Khách hàng chọn Stripe

↓

Storefront chuyển hướng đến Stripe checkout

↓

Khách hàng thanh toán

↓

Stripe gửi webhook đến backend

↓

Backend xác minh webhook và tạo đơn hàng

↓

Backend gửi email xác nhận đơn hàng đến khách hàng

↓

Storefront chuyển hướng đến trang cảm ơn

Luồng này liên quan đến nhiều thành phần: Storefront (React), GraphQL API, REST API, cơ sở dữ liệu, cổng thanh toán Stripe, và hệ thống email.

## 6. CÀI ĐẶT, CẤU HÌNH VÀ TRIỂN KHAI

### 6.1 Yêu cầu hệ thống & cấu hình ban đầu (theo hướng dẫn từ [evershop.io](https://evershop.io))

Trước khi cài đặt EverShop, cần đảm bảo rằng hệ thống đáp ứng các yêu cầu sau:

#### Phần mềm tối thiểu:

- Node.js: Phiên bản 14 hoặc cao hơn (khuyến nghị phiên bản LTS mới nhất)
- npm (Node Package Manager): Phiên bản 7 hoặc cao hơn
- PostgreSQL: Phiên bản 13 hoặc cao hơn
- Docker (tùy chọn, để chạy trong container)

#### Phần cứng tối thiểu:

- CPU: 1 core (cho môi trường development)
- RAM: 2GB (cho môi trường development); 4GB+ (cho production)
- Lưu trữ: 5GB (tùy thuộc vào số lượng sản phẩm và ảnh)

#### Hệ điều hành được hỗ trợ:

- Linux (Ubuntu, CentOS, v.v.)
- macOS
- Windows (thông qua WSL2 hoặc native Node.js)

#### Cấu hình ban đầu:

Trước khi cài đặt, cần chuẩn bị:

1. **Database:** Tạo một cơ sở dữ liệu PostgreSQL mới cho EverShop. Ghi nhớ tên cơ sở dữ liệu, tên người dùng, mật khẩu, và máy chủ PostgreSQL.
2. **Domain:** Nếu cài đặt trên production, chuẩn bị một tên miền. Nếu cài đặt trên localhost, không cần.
3. **Email:** Chuẩn bị cấu hình SMTP để gửi email (xác nhận đơn hàng, đặt lại mật khẩu, v.v.).

4. **Cổng thanh toán:** Tạo tài khoản trên các cổng thanh toán như Stripe nếu muốn hỗ trợ thanh toán trực tuyến.

## 6.2 Hướng dẫn cài đặt nền tảng

EverShop cung cấp ba cách cài đặt: sử dụng create-evershop-app, sử dụng Docker, hoặc cài đặt thủ công.

### Phương pháp 1: Sử dụng create-evershop-app (Khuyến nghị cho người mới bắt đầu)

```
npx create-evershop-app my-evershop-store
cd my-evershop-store
npm run setup
npm run start
```

Lệnh `create-evershop-app` sẽ tạo một folder mới, cài đặt tất cả các phụ thuộc, và chuẩn bị cơ sở dữ liệu. Sau đó, EverShop sẽ chạy ở `http://localhost:3000`.

### Phương pháp 2: Sử dụng Docker (Khuyến nghị cho production)

```
curl -sSL
https://raw.githubusercontent.com/evershopcommerce/evershop/main/docker-compose.yml > docker-compose.yml
docker-compose up -d
```

Điều này sẽ tải file `docker-compose.yml`, khởi động các container cho EverShop và PostgreSQL. EverShop sẽ chạy ở `http://localhost:3000` và admin panel ở `http://localhost:3000/admin`.

Để tạo tài khoản admin:

```
docker-compose exec app npm run user:create -- --email
"admin@example.com" --password "password" --name "Admin"
```

### Phương pháp 3: Cài đặt thủ công

```
# Bước 1: Tạo một folder mới và khởi tạo npm project
mkdir my-evershop-store
cd my-evershop-store
npm init -y
```

```
# Bước 2: Cài đặt package EverShop
npm install @evershop/evershop
```

```
# Bước 3: Thêm các script vào package.json
```

```
# Chỉnh sửa package.json để thêm:
{
  "scripts": {
    "setup": "evershop install",
    "build": "evershop build",
    "start": "evershop start",
    "start:debug": "evershop start --debug",
    "user:create": "evershop user:create"
  }
}
```

```
# Bước 4: Chạy cài đặt
npm run setup
```

```
# Bước 5: Khởi động
npm run build
npm run start
```

Quá trình cài đặt sẽ:

1. Tải xuống EverShop và các phụ thuộc
2. Tạo cấu trúc thư mục
3. Tạo bảng cơ sở dữ liệu
4. Biên dịch (build) các asset

## 6.3 Quy trình cấu hình store, theme, extensions

### Cấu hình cửa hàng cơ bản

Sau khi cài đặt, cần cấu hình thông tin cửa hàng cơ bản. Đăng nhập vào admin panel tại <http://localhost:3000/admin> bằng tài khoản admin đã tạo.

Trong admin panel, điều hướng đến "Settings" (Cài đặt) để:

1. **Thông tin cửa hàng:** Nhập tên cửa hàng, logo, mô tả, thông tin liên hệ, địa chỉ
2. **Email:** Cấu hình địa chỉ email của cửa hàng

### Cấu hình cổng thanh toán

Để chấp nhận thanh toán từ khách hàng:

1. Tạo tài khoản Stripe tại <https://stripe.com/>
2. Lấy API key công khai (public key) và khóa bí mật (secret key)
3. Trong admin panel, điều hướng đến "SETTING" → "PAYMENT SETTING" (Cài đặt thanh toán)



#### 4. Bật Stripe và nhập khóa API

### Cấu hình vận chuyển

1. Trong admin panel, điều hướng đến "SETTING" → "SHIPPING SETTING" (Cài đặt vận chuyển)
2. Tạo các phương thức vận chuyển (miễn phí, được thanh toán, v.v.)
3. Cài đặt phí vận chuyển dựa trên trọng lượng, giá trị đơn hàng hoặc để mặc định

### Cấu hình thuế

1. Trong admin panel, điều hướng đến "SETTING" → "TAX SETTING" (Cài đặt thuế)
2. Thiết lập các mức thuế áp dụng cho sản phẩm và đơn hàng
3. Tạo các quy tắc thuế theo khu vực (quốc gia, tỉnh/thành, quận/huyện) hoặc theo loại sản phẩm
4. Cấu hình cách hiển thị thuế: gộp trong giá sản phẩm (Tax Inclusive) hoặc tách riêng (Tax Exclusive)

### Cài đặt extension

1. Tạo một extension mới hoặc cài đặt một extension từ npm, ví dụ đăng nhập bằng Google:

```
npm install @evershop/google_login
```

2. Cấu hình extension trong config/default.json:

```
{
  "system": {
    "extensions": [
      {
        "name": "google_login",
        "resolve": "node_modules/@evershop/google_login",
        "enabled": true,
        "priority": 10
      }
    ]
  }
}
```

3. Chạy npm run build( Chi tiết sẽ được viết ở phần 7)

## 6.5 Giao diện minh họa

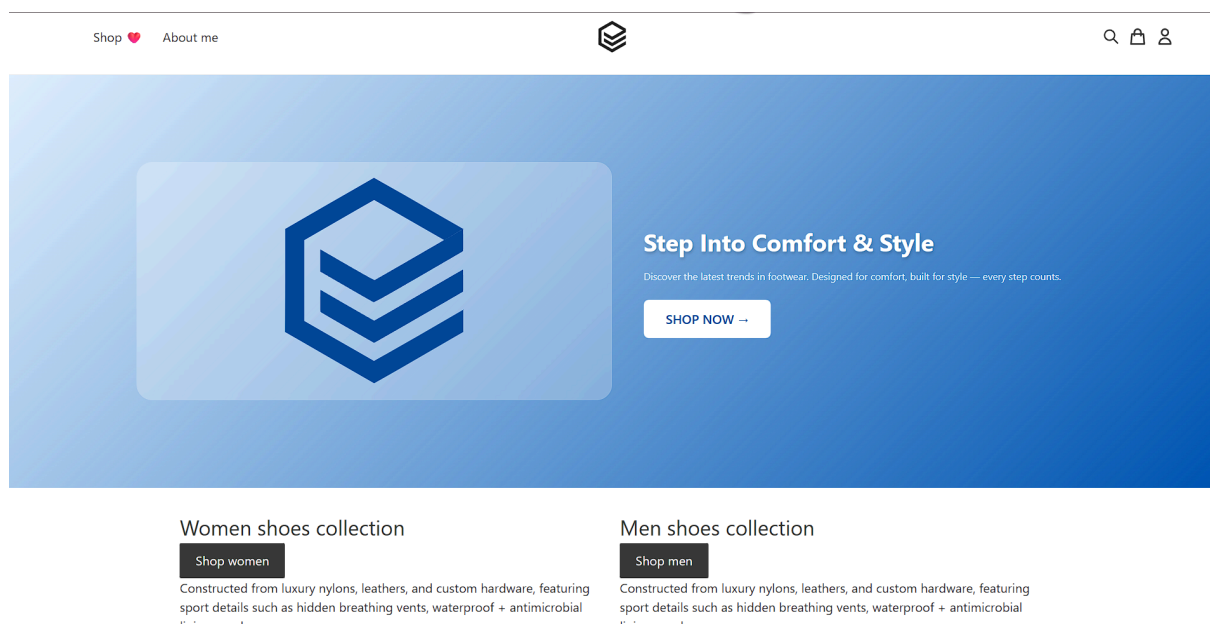
Phần này trình bày các giao diện chính của hệ thống EverShop, bao gồm **giao diện chính người dùng (Storefront)** dành cho khách hàng và **giao diện quản trị**

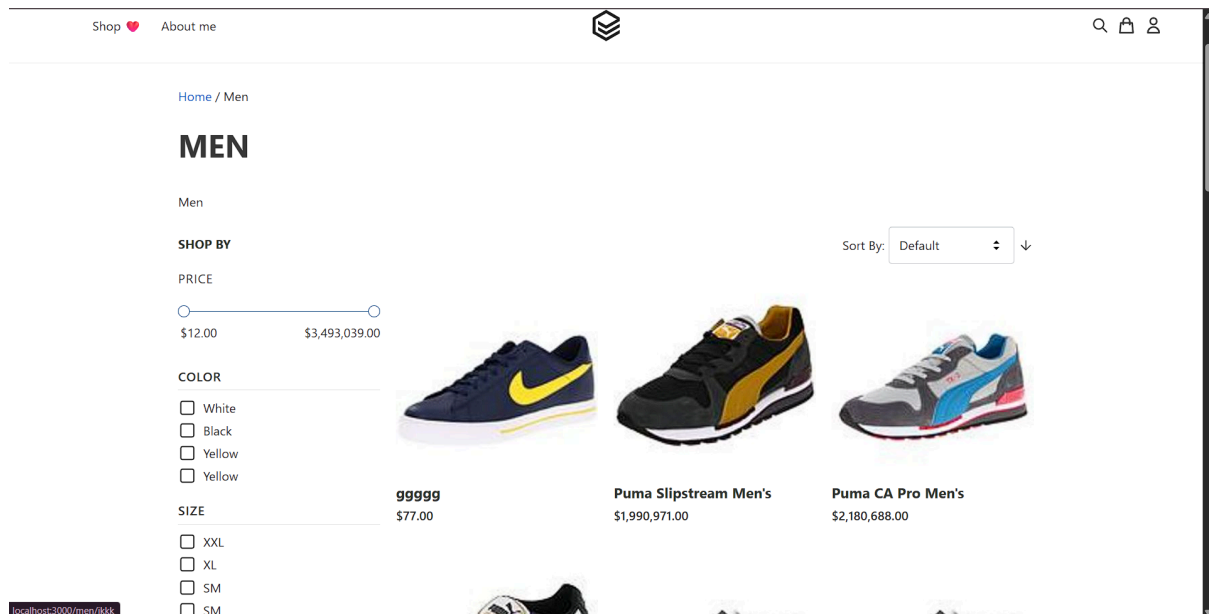
**(Admin Panel)** dành cho người quản lý. Mỗi giao diện của ever-shop được thiết kế hiện đại, trực quan và đáp ứng đầy đủ các chức năng cốt lõi của một hệ thống thương mại điện tử (chi tiết giao diện được thể hiện trên web).

### 6.5.1 Giao diện người dùng (Storefront)

#### Đặc điểm chính:

- Trang chủ hiển thị danh mục sản phẩm nổi bật, banner khuyến mãi và thanh tìm kiếm.
- Trang danh mục cho phép lọc sản phẩm theo **giá, thương hiệu, màu sắc hoặc kích cỡ**.
- Trang chi tiết sản phẩm hiển thị **hình ảnh, mô tả, giá, tình trạng hàng và nút “Thêm vào giỏ hàng”**.
- Giỏ hàng (Cart) thể hiện rõ **tổng giá trị đơn hàng, số lượng, phí vận chuyển** và cho phép người dùng chỉnh sửa trực tiếp.
- Trang thanh toán (Checkout) hỗ trợ nhập thông tin giao hàng, chọn phương thức thanh toán và xác nhận đơn hàng.



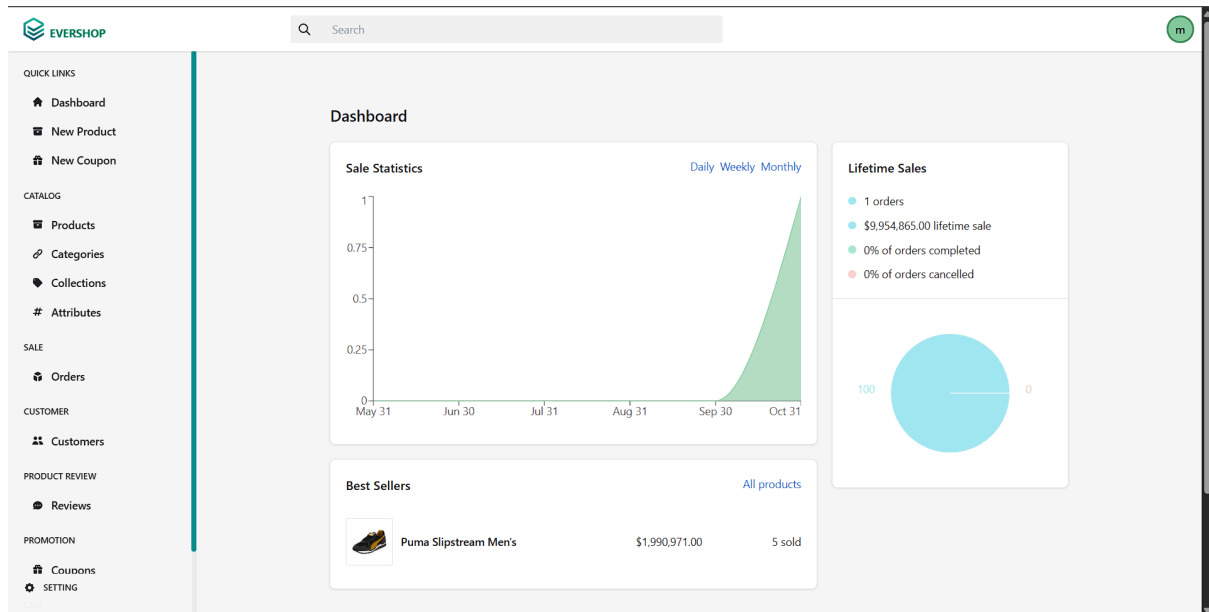


## 6.5.2 Giao diện quản trị (Admin Panel)

Giao diện quản trị được thiết kế dành cho người vận hành hệ thống, giúp **quản lý toàn bộ dữ liệu và hoạt động kinh doanh**.

### Đặc điểm chính:

- **Bảng điều khiển (Dashboard):** hiển thị thống kê tổng quan như số lượng đơn hàng, doanh thu, khách hàng mới.
- **Quản lý sản phẩm:** thêm, sửa, xóa, cập nhật tồn kho, hình ảnh, danh mục và biến thể.
- **Quản lý đơn hàng:** theo dõi trạng thái, cập nhật vận chuyển, tạo hoá đơn và xử lý thanh toán.
- **Quản lý khách hàng:** xem danh sách, lịch sử mua hàng, thông tin liên hệ.
- **Cấu hình cửa hàng:** cài đặt thông tin doanh nghiệp, email, thanh toán (Stripe), vận chuyển, và các module mở rộng.



## 7. TÙY BIẾN, MỞ RỘNG VÀ PHÁT TRIỂN THÊM

### 7.1. Khái niệm về Extension trong EverShop

EverShop sử dụng mô hình **mở rộng qua extension** để thêm hoặc tùy chỉnh tính năng mà không cần thiệp vào mã nguồn lõi.

Mỗi extension được đăng ký thông qua tệp `config/default.json`:

```
{  
  
  "system": {  
  
    "extensions": [  
  
      {  
  
        "name": "google_login",  
  
        "resolve": "node_modules/@evershop/google_login",  
  
        "enabled": true,  
  
        "priority": 10  
  
      }  
  
    ]  
  
  }  
}
```

```
}  
  
}
```

Một extension có thể:

- Thêm trang mới vào Storefront hoặc Admin.
- Mở rộng schema GraphQL (thêm field, mutation, query).
- Gắn logic backend mới (API, middleware, hooks).
- Bổ sung UI/UX ở cấp module hoặc theme.

Cấu trúc thư mục tiêu chuẩn của extension:

extensions/

└─ your-extension/

```
├─ dist/          # Thư mục mã đã biên dịch (JS)  
├─ src/           # Thư mục mã nguồn TypeScript  
│ └─ api/         # Định nghĩa API hoặc endpoint backend  
│ └─ crons/       # Các tác vụ định kỳ (cron jobs)  
│ └─ graphql/     # Schema & resolver mở rộng GraphQL  
│ └─ pages/       # Trang giao diện React (Storefront/Admin)  
│ └─ subscribers/ # Đăng ký các sự kiện (event hooks)  
│ └─ bootstrap.ts # Điểm khởi tạo chính của extension  
├─ package.json   # Thông tin extension, scripts, dependencies  
├─ Readme.md      # Tài liệu mô tả, hướng dẫn sử dụng  
└─ tsconfig.json  # Cấu hình TypeScript (compile sang dist/)
```

## 7.2. Quy trình tạo một Extension mới

### **Bước 1 – Tạo thư mục extension:**

```
cd extensions  
  
mkdir product_review  
  
cd product_review  
  
npm init -y
```

### **Bước 2 – Đăng ký trong config/default.json:**

```
{  
  "system": {  
    "extensions": [  
      { "name": "product_review", "resolve":  
"extensions/product_review", "enabled": true }  
    ]  
  }  
}
```

### **Bước 3 – Cấu hình backend và giao diện:**

- Trong thư mục api/ định nghĩa REST hoặc GraphQL API.
- Trong pages/, thêm các component React cho giao diện.
- Trong graphql/, mở rộng type hoặc query.

Ví dụ thêm trường mới vào Product:

```
extend type Product {  
  reviews: [Review]  
}
```

### 7.3 Extension đã mở rộng 1: google\_login

**Mục tiêu:** Cho phép khách hàng sử dụng tài khoản Google để đăng nhập vào cửa hàng thay vì phải tạo tài khoản thủ công.

**Hướng dẫn cài đặt & cấu hình:**

- Cài đặt: `npm install @evershop/google_login npm`

**Kích hoạt extension trong file config/default.json:**

```
{  
  "system": {  
    "extensions": [  
      {  
        "name": "google_login",  
        "resolve": "node_modules/@evershop/google_login",  
        "enabled": true,  
        "priority": 10  
      }  
    ]  
  }  
}
```

**Cấu hình trong file .env:**

```
GOOGLE_LOGIN_CLIENT_ID="YOUR_GOOGLE_CLIENT_ID"  
GOOGLE_LOGIN_CLIENT_SECRET="YOUR_GOOGLE_CLIENT_SECRET"  
GOOGLE_LOGIN_SUCCESS_REDIRECT_URL="http://localhost:3000"
```

GOOGLE\_LOGIN\_FAILURE\_REDIRECT\_URL="http://localhost:3000/account/login"

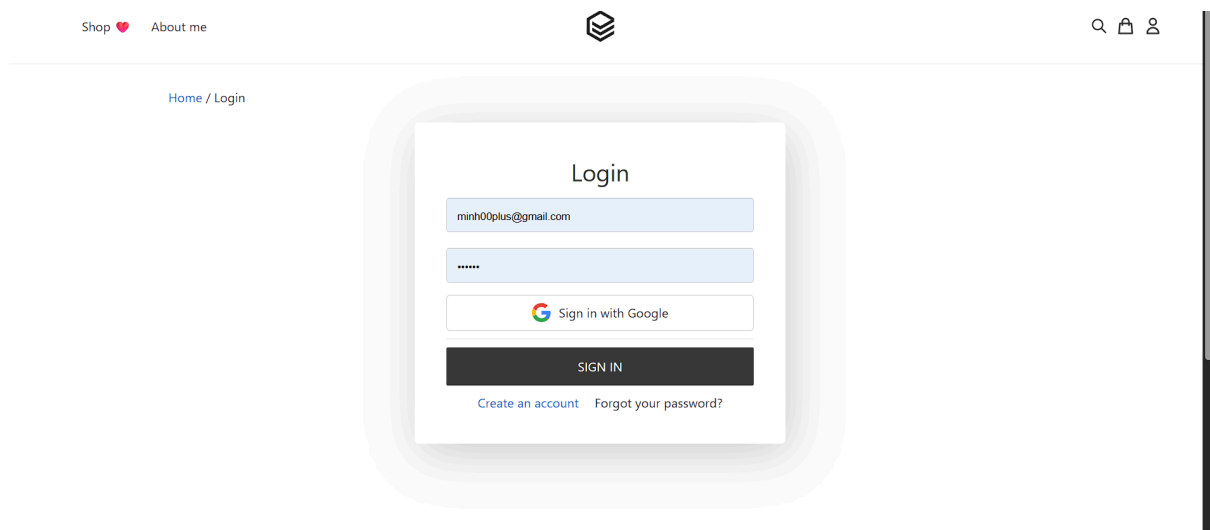
### Quy trình hoạt động:

1. Người dùng chọn “Đăng nhập bằng Google” tại trang /account/login.
2. Hệ thống chuyển hướng tới OAuth endpoint của Google (qua thư viện nội bộ của extension).
3. Sau khi xác thực thành công, Google trả về token và thông tin người dùng. Extension kiểm tra: nếu khách hàng đã tồn tại thì cập nhật, nếu chưa thì tạo mới trong bảng customer.
4. Hệ thống đăng nhập người dùng và chuyển hướng về URL thành công (ví dụ trang chủ) hoặc URL thất bại nếu có lỗi.

### Lưu ý kỹ thuật:

- Đảm bảo callback URL đã đăng ký trong dự án Google API Console trùng với GOOGLE\_LOGIN\_SUCCESS\_REDIRECT\_URL.
- Kiểm tra bảo mật: state parameter, xác minh token, tránh lỗ hổng giao thức OAuth2.
- Kiểm thử đầy đủ trong môi trường dev và production.

### Ảnh minh họa sau khi cài đặt thành công extension Google login



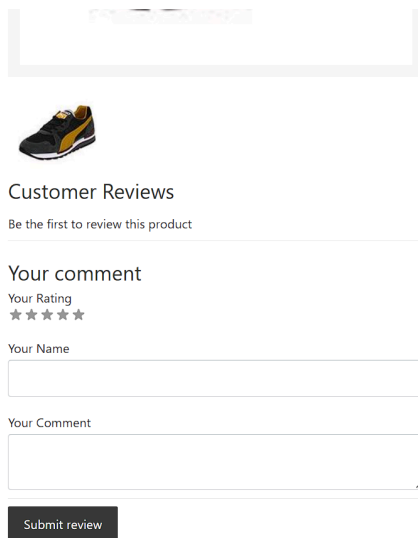


## 7.4 Extension đã mở rộng 2: product\_review

**Mục tiêu:** Cho phép khách hàng đăng đánh giá (review) và chấm sao (rating) cho từng sản phẩm, giúp tăng độ tin cậy và tương tác trên cửa hàng.

**Hướng dẫn cài đặt:**

- Cài đặt: `npm install @evershop/product_review`
- Cài đặt và kích hoạt tương tự cách extension Google Login (thêm vào `config/default.json`).



The screenshot shows a web form for customer reviews. At the top, there is a placeholder for a product image. Below it, the text "Customer Reviews" is displayed, followed by the prompt "Be the first to review this product". The form includes a section for "Your comment" with a "Your Rating" section showing five stars. Below the rating is a text input field for "Your Name" and a larger text area for "Your Comment". A "Submit review" button is located at the bottom of the form.

## 7.5. Trang tùy chỉnh: About-Me Page

### Mục tiêu

Xây dựng trang giới thiệu cá nhân tại đường dẫn `/about-me`, nhằm thể hiện thông tin nhà phát triển hoặc nhóm dự án, đồng thời tăng tính nhận diện thương hiệu của cửa hàng.

### 1 Tạo cấu trúc thư mục extension

Trong thư mục dự án EverShop, tạo cấu trúc sau:

```
extensions/
```

```
└─ aboutMe/
```

```
    └─ dist/
```

```
    # Mã JS đã biên dịch
```

```

└─ src/
    └─ pages/
        └─ frontStore/
            └─ aboutMe/
                └─ Aboutme.tsx          # Component
React của trang About-Me

                └─ route.json          # Đăng ký
router cho trang

```

## 2 Thêm cấu hình extension vào file config/default.json

Sau khi tạo extension aboutMe, cần **đăng ký extension này trong hệ thống EverShop** để nó được nạp khi khởi động ứng dụng.

Mở file **config/default.json** tại thư mục gốc của dự án và thêm đoạn cấu hình sau vào phần "extensions":

```

{
  "system": {
    "extensions": [
      {
        "name": "aboutMe",
        "resolve": "extensions/aboutMe",
        "enabled": true,
        "priority": 10
      }
    ]
  }
}

```

### Giải thích:

- **name:** Tên định danh của extension, nên trùng với tên thư mục (aboutMe).
- **resolve:** Đường dẫn tới vị trí extension trong thư mục dự án.  
**enabled:** Nếu đặt là true, extension sẽ được kích hoạt khi khởi động EverShop.
- **priority:** Mức độ ưu tiên tải extension (số càng cao → tải sớm hơn).

### Kết quả:

Khi khởi động lại EverShop, hệ thống sẽ tự động nhận diện extension aboutMe, biên dịch code TypeScript từ src/ sang dist/, và kích hoạt route /about-me trong storefront.

### 3 Cấu hình route cho trang About-Me

Trong file src/pages/frontStore/aboutMe/route.json, thêm:

```
{  
  "methods": [ "GET" ],  
  "path": "/about-me"  
}
```

### Giải thích:

- **methods:** Xác định phương thức HTTP mà route hỗ trợ. Ở đây là GET vì trang này chỉ cần hiển thị nội dung.
- **path:** Là đường dẫn URL mà người dùng truy cập — trong trường hợp này là /about-me.



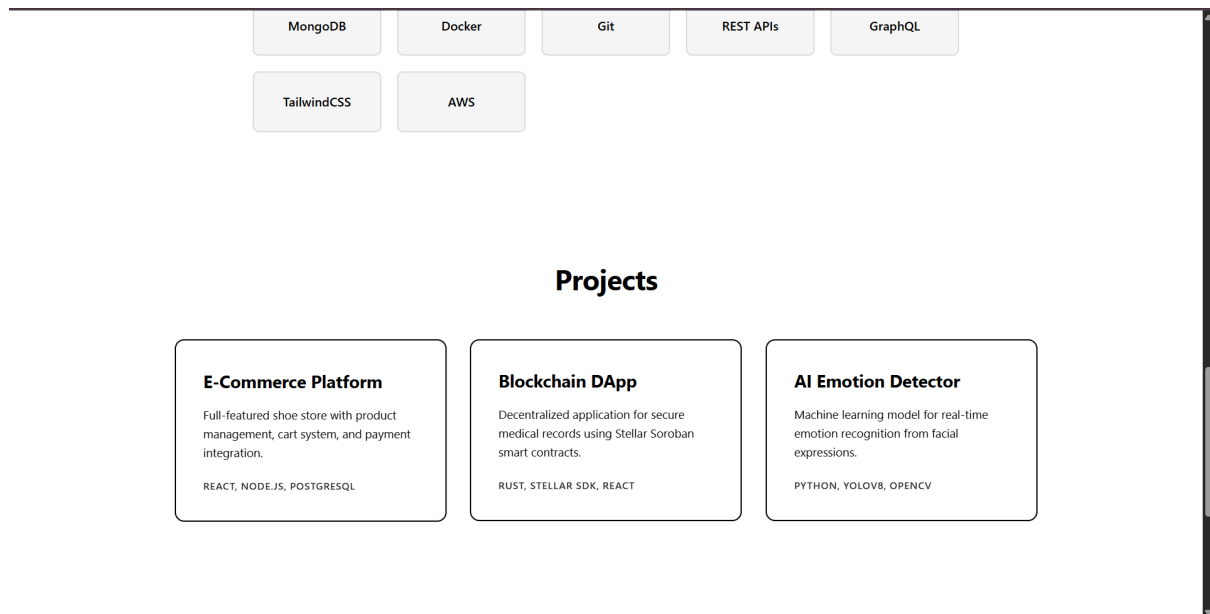
## About Me

Hi, I'm Minh — a Computer Science student at Phenikaa University with a passion for technology and innovation. I specialize in full-stack development, focusing on creating efficient, user-friendly applications.

My expertise spans from front-end frameworks like React to back-end systems with Node.js, along with blockchain development and AI/ML models. I'm always eager to learn and take on new challenges.

## Skills

React	Node.js	TypeScript	Python	PostgreSQL
MongoDB	Docker	Git	REST APIs	GraphQL



## 8. KẾT QUẢ, ĐÁNH GIÁ VÀ KHUYẾN NGHỊ

### 8.1 Kết quả đạt được khi triển khai demo hoặc prototype

Khi triển khai một store demo hoặc prototype EverShop, có thể đạt được các kết quả sau:

**Thiết lập nhanh chóng:** Nhờ hệ thống cài đặt tự động, có thể có một cửa hàng hoạt động đầy đủ chức năng chỉ trong vài phút. So sánh với Magento hoặc các nền tảng khác, điều này tiết kiệm thời gian đáng kể.

**Giao diện hiện đại:** Admin panel được xây dựng bằng React với giao diện hiện đại, tương tự như các ứng dụng SaaS phổ biến. Người dùng sẽ cảm thấy quen thuộc khi sử dụng.

**Hoạt động mượt mà:** Với công nghệ Node.js và GraphQL, storefront hoạt động rất mượt mà, với thời gian tải trang ngắn và trải nghiệm người dùng tốt.

**Khả năng mở rộng:** Nhờ kiến trúc mô-đun, có thể dễ dàng thêm các tính năng mới hoặc tùy biến giao diện mà không cần lo lắng về việc phá vỡ mã cốt lõi.

**Hỗ trợ đa phương thức thanh toán:** EverShop đi kèm với hỗ trợ tích hợp cho Stripe, và có thể dễ dàng thêm các phương thức thanh toán khác thông qua extension.

## 8.2 Đánh giá ưu điểm và hạn chế của nền tảng

### Ưu điểm:

1. **Kiến trúc mô-đun mạnh mẽ:** EverShop được thiết kế từ đầu với kiến trúc mô-đun, làm cho việc mở rộng và bảo trì dễ dàng hơn so với các nền tảng khác.
2. **Công nghệ hiện đại:** Sử dụng Node.js, React, PostgreSQL, và GraphQL — đây là những công nghệ phổ biến và được hỗ trợ tốt bởi cộng đồng.
3. **Admin panel tuyệt vời:** Admin panel được xây dựng bằng React với giao diện hiện đại, tương tự Shopify.
4. **Tốc độ cao:** EverShop xử lý request nhanh chóng nhờ Node.js và GraphQL.
5. **Mã nguồn mở:** Tự do sử dụng, sửa đổi, và phân phối. Không phải lo lắng về chi phí giấy phép.
6. **Cộng đồng đang phát triển:** Mặc dù nhỏ hơn so với Magento hay WooCommerce, cộng đồng EverShop đang phát triển nhanh chóng.

### Hạn chế:

1. **Cộng đồng nhỏ:** So sánh với Magento hay WooCommerce, cộng đồng của EverShop nhỏ hơn, có thể khó tìm các extension hoặc hỗ trợ.
2. **Tài liệu có sẵn hạn chế:** Mặc dù tài liệu chính thức tốt, tài nguyên học tập bên thứ ba còn hạn chế.
3. **Yêu cầu kiến thức kỹ thuật:** EverShop yêu cầu người sử dụng có kiến thức về JavaScript, React, và Node.js để tùy biến hoặc mở rộng. Điều này không phù hợp cho người không có kiến thức kỹ thuật.
4. **Kiến trúc monolithic:** Mặc dù EverShop có kiến trúc mô-đun, nó vẫn là một ứng dụng monolithic. Điều này có thể là một hạn chế đối với các doanh nghiệp rất lớn cần kiến trúc microservices.
5. **Tính năng B2B hạn chế:** EverShop tập trung vào B2C (business-to-consumer). Tính năng B2B (business-to-business) còn hạn chế.

## 8.3 Khuyến nghị và hướng phát triển tiếp theo

### Khuyến nghị cho người lựa chọn EverShop:

- **Phù hợp cho:** Startup, doanh nghiệp vừa, nhà phát triển muốn có nền tảng hiện đại với kiến trúc mô-đun tốt.
- **Không phù hợp cho:** Doanh nghiệp lớn cần kiến trúc phức tạp, người không có kiến thức kỹ thuật.

### Hướng phát triển tiếp theo cho EverShop:

1. **Phát triển cộng đồng:** EverShop cần tăng tích cực trong cộng đồng, tạo nhiều event, tutorial, và case study để thu hút nhiều nhà phát triển hơn.

2. **Tài liệu chi tiết hơn:** Cần viết thêm tài liệu chi tiết hơn về các chức năng phổ biến, cách tạo extension, cách tạo theme, v.v.
3. **Hỗ trợ B2B:** Phát triển các tính năng B2B như multi-vendor, wholesale pricing, v.v.
4. **Kiến trúc Headless:** Hỗ trợ tốt hơn cho kiến trúc headless commerce, cho phép sử dụng EverShop backend với các frontend khác nhau (Next.js, Vue.js, v.v.).
5. **Performance Optimization:** Cải thiện hiệu suất thêm bằng caching, CDN integration, v.v.
6. **AI/ML Integration:** Tích hợp các tính năng AI/ML như product recommendations, chatbot, v.v.

## 9. KẾT LUẬN

EverShop là một nền tảng thương mại điện tử mã nguồn mở hiện đại, được xây dựng với công nghệ tiên tiến như Node.js, React, PostgreSQL, và GraphQL. Nó cung cấp một kiến trúc mô-đun mạnh mẽ, cho phép các nhà phát triển dễ dàng mở rộng và tùy biến nền tảng theo nhu cầu của họ.

Điểm mạnh của EverShop là kiến trúc tốt, giao diện admin hiện đại, và API đầy đủ. Tuy nhiên, nó vẫn còn những hạn chế như cộng đồng nhỏ, tài liệu hạn chế, và yêu cầu kiến thức kỹ thuật cao.

Với sự phát triển tiếp tục, đặc biệt là trong việc phát triển cộng đồng, viết tài liệu chi tiết, và thêm các tính năng mới, EverShop có tiềm năng trở thành một lựa chọn hàng đầu cho các doanh nghiệp vừa và các nhà phát triển muốn xây dựng cửa hàng thương mại điện tử hiện đại.

Báo cáo này đã cung cấp cái nhìn toàn diện về EverShop, từ kiến trúc hệ thống, chức năng sẵn có, use-case thực tế, cho đến quy trình cài đặt và triển khai. Hy vọng rằng báo cáo này sẽ giúp sinh viên, giảng viên, và các nhà phát triển hiểu rõ hơn về nền tảng này và có thể sử dụng nó một cách hiệu quả.

## 10. TÀI LIỆU THAM KHẢO

1. EverShop Official Website: <https://evershop.io/>
2. EverShop Documentation: <https://evershop.io/docs/>
3. EverShop GitHub Repository:  
<https://github.com/evershopcommerce/evershop>
4. Node.js Official Documentation: <https://nodejs.org/docs/>
5. PostgreSQL Official Documentation: <https://www.postgresql.org/docs/>
6. React Official Documentation: <https://react.dev/>

7. GraphQL Official Documentation: <https://graphql.org/learn/>
8. Express.js Official Documentation: <https://expressjs.com/>
9. Stripe Official Documentation: <https://stripe.com/docs>
10. Docker Official Documentation: <https://docs.docker.com/>
11. Nginx Official Documentation: <https://nginx.org/en/docs/>
12. PM2 Official Documentation: <https://pm2.keymetrics.io/docs/>
13. Comparison of eCommerce Platforms 2025 - Various sources
14. Best Practices for Node.js Applications - Community resources