**BÁO CÁO ĐỀ TÀI 01**

**KIỂM THỬ PHẦN MỀM**

**GV hướng dẫn: TS. Đỗ Như Tài**

**Lớp: DCT122C3 – Mã học phần: 841408**

**Sinh viên: Trịnh Long Phát – 3122411150**

**Lê Hồng Phát – 3122411145**

**Trương Phú Kiệt – 3122411109**

**Trà Đức Toàn – 3122411217**

**Thành phố Hồ Chí Minh, tháng 9 năm 2025**

**MỤC LỤC**

[Danh mục hình ảnh 6](#_Toc210763806)

[Introduction 7](#_Toc210763807)

[Business content 8](#_Toc210763808)

[Conceptual model 10](#_Toc210763809)

[Use case hệ thống bán hàng 11](#_Toc210763810)

[User Story 11](#_Toc210763811)

[1. Khách hàng mới 11](#_Toc210763812)

[2. Khách hàng quen thuộc 12](#_Toc210763813)

[3. Quản trị viên 13](#_Toc210763814)

[Use Case: Quản lý tài khoản khách hàng 15](#_Toc210763815)

[1. Pre-Condition 15](#_Toc210763816)

[2. Trigger 15](#_Toc210763817)

[3. Basic Flow 15](#_Toc210763818)

[4. Alternative Flow 16](#_Toc210763819)

[5. Business Rule 16](#_Toc210763820)

[6. Activity Diagram 17](#_Toc210763821)

[Use Case: Quản lý thanh toán 18](#_Toc210763822)

[1. Pre-Condition 18](#_Toc210763823)

[2. Trigger 18](#_Toc210763824)

[3. Basic Flow 18](#_Toc210763825)

[4. Alternative Flow 19](#_Toc210763826)

[5. Business Rule 19](#_Toc210763827)

[6. Activity Diagram 20](#_Toc210763828)

[Sơ đồ khối (Block Diagram) 21](#_Toc210763829)

[1. Mô tả tổng quan 21](#_Toc210763830)

[2. Giải thích chi tiết các thành phần trong sơ đồ 21](#_Toc210763831)

[3. Luồng hoạt động của hệ thống 24](#_Toc210763832)

[4. Ưu điểm của kiến trúc 24](#_Toc210763833)

[Deployment View 25](#_Toc210763834)

[1. Mô tả tổng quan 25](#_Toc210763835)

[2. Các giai đoạn triển khai 25](#_Toc210763836)

[3. Quy trình CI/CD tổng thể 27](#_Toc210763837)

[4. Ưu điểm của mô hình triển khai 28](#_Toc210763838)

[Communication View 29](#_Toc210763839)

[1. Giới thiệu 30](#_Toc210763840)

[2. Mô hình giao tiếp tổng quan 30](#_Toc210763841)

[3. Các nhóm API chính 30](#_Toc210763842)

[3.1. Product (Sản phẩm) 30](#_Toc210763843)

[3.2. User (Người dùng) 31](#_Toc210763844)

[3.3. Detail Order (Chi tiết đơn hàng) 31](#_Toc210763845)

[3.4. Order (Đơn hàng) 31](#_Toc210763846)

[3.5. Delivery (Giao hàng) 31](#_Toc210763847)

[4. Cách thức giao tiếp 32](#_Toc210763848)

[5. Luồng xử lý ví dụ 32](#_Toc210763849)

[Decomposition View: C4 33](#_Toc210763850)

[C1 - System Context 33](#_Toc210763851)

[C2 - Containers 34](#_Toc210763852)

[1. Mô tả tổng quan hệ thống 34](#_Toc210763853)

[2. Các tác nhân bên ngoài và vai trò 34](#_Toc210763854)

[3. Mối quan hệ giữa các thành phần 35](#_Toc210763855)

[4. Tóm tắt vai trò của hệ thống 36](#_Toc210763856)

[C3 - Components 37](#_Toc210763857)

[1. Mô tả tổng quan 37](#_Toc210763858)

[2. Các Container chính trong hệ thống 38](#_Toc210763859)

[3. Luồng giao tiếp giữa các Container 39](#_Toc210763860)

[4. Mối liên kết giữa các thành phần 40](#_Toc210763861)

[5. Ưu điểm của mô hình Container 40](#_Toc210763862)

[C4 - Code 41](#_Toc210763863)

[1. Mô tả tổng quan 41](#_Toc210763864)

[2. Danh sách các bảng và mô tả chi tiết 41](#_Toc210763865)

[3. Quan hệ giữa các bảng 46](#_Toc210763866)

[4. Tóm tắt cấu trúc hệ thống dữ liệu 46](#_Toc210763867)

[Test Plan 47](#_Toc210763868)

[1. Introduction 47](#_Toc210763869)

[1.1 Purpose 47](#_Toc210763870)

[1.2 Background 47](#_Toc210763871)

[1.3 Scope 48](#_Toc210763872)

[1.4 Project Identification 51](#_Toc210763873)

[1.5 References 53](#_Toc210763874)

[2. Feature to be tested 54](#_Toc210763875)

[2.1 Functionality 54](#_Toc210763876)

[2.2 Usability 56](#_Toc210763877)

[2.3 Design Constraints 57](#_Toc210763878)

[2.4 Interfaces 58](#_Toc210763879)

[3. Feature not to be tested 60](#_Toc210763880)

[3.1 Setup a user account 60](#_Toc210763881)

[3.2 Remove user account 60](#_Toc210763882)

[4. Test Strategy 60](#_Toc210763883)

[4.1 Testing Types 60](#_Toc210763884)

[4.2 Tools 62](#_Toc210763885)

[5. Resources 62](#_Toc210763886)

[5.1 Roles 62](#_Toc210763887)

[6. Project Milestones 64](#_Toc210763888)

[7. Deliverables 65](#_Toc210763889)

[7.1 Test Model 65](#_Toc210763890)

[7.2 Test Logs 65](#_Toc210763891)

[7.3 Bug Reports 65](#_Toc210763892)

# **Danh mục hình ảnh**

[Hình 1. Conceptual model 10](#_Toc210762106)

[Hình 2. Use case hệ thống bán hàng 11](#_Toc210762107)

[Hình 3. Activity diagram quản lý tài khoản 17](#_Toc210762108)

[Hình 4. Activity diagram quản lý thanh toán 20](#_Toc210762109)

[Hình 5. Sơ đồ khối 21](#_Toc210762110)

[Hình 6. Deployment view 25](#_Toc210762111)

[Hình 7. Communication view 29](#_Toc210762112)

[Hình 8. C1 - System context 33](#_Toc210762113)

[Hình 9. C2 - Containers 34](#_Toc210762114)

[Hình 10. C3 - Components 37](#_Toc210762115)

[Hình 11. C4 - Code 41](#_Toc210762116)

# **Giới thiệu**

Đề tài : Website bán quần áo

Ngày nay, công nghệ thông tin đã có những bước phát triển mạnh mẽ trong mọi phương diện nói chung ví dụ như: đời sống, công việc, giải trí, truyền thông, ... Và riêng với bán hàng, so với cách bán truyền thống thì nay doanh nghiệp, cửa hàng nhỏ lẻ nào cũng có một website để quáng bá, bán hàng trực tuyến sản phẩm và tương tác với người dùng. Nắm bắt được nhu cầu đó, nhóm em quyết định thực hiện đề tài: Xây dựng Website bán quần áo sử dụng công nghệ ReactJS & NodeJS ( API, Socket ). Khi sử dụng trang web khách hàng sẽ cảm nhận được sự mới mẻ và thuận tiện của Website mang lại. Và website cũng dễ dàng cung cấp thông tin chi tiết sản phẩm giúp khách hàng có thể thanh toán trực tiếp qua paypal hoặc ship cod.

# **Bối cảnh thực tế**

Website **Fear Of God** hỗ trợ các kịch bản kinh doanh chính bao gồm: Danh mục sản phẩm, Giỏ hàng, Quy trình đặt hàng, Thanh toán, Giao hàng, Trò chuyện trực tuyến, Xác nhận Email và Kiểm soát truy cập.

**Với Danh mục sản phẩm**, Người mua có thể duyệt danh sách quần áo với các chức năng lọc, tìm kiếm, phân loại và sắp xếp theo tên sản phẩm, loại sản phẩm, giá. Người mua cũng có thể xem chi tiết sản phẩm bằng cách nhấn vào sản phẩm, trong trang chi tiết sẽ hiển thị: tên sản phẩm, mô tả, giá, tình trạng còn hàng, hình ảnh sản phẩm. Quản trị viên (SysAdmin) có thể quản lý sản phẩm với khả năng thêm, sửa, xóa, phân loại và phân trang sản phẩm.

**Với Giỏ hàng & Quy trình đặt hàng**, Người mua có thể thêm sản phẩm vào giỏ từ trang danh mục hoặc trang chi tiết sản phẩm. Sau khi chọn, người mua sẽ thấy các sản phẩm trong giỏ cùng bảng tóm tắt bao gồm: tổng tiền sản phẩm, phí vận chuyển, khuyến mãi (nếu có) và tổng đơn hàng. Khi người mua thêm hoặc xóa sản phẩm, giỏ hàng và bảng tóm tắt sẽ được cập nhật tự động. Cuối cùng, người mua có thể tiến hành đặt hàng. Quản trị viên có thể xem tất cả đơn hàng và cập nhật trạng thái.

**Với Quy trình thanh toán**, sau khi người mua nhấn nút đặt hàng, hệ thống sẽ kiểm tra thông tin sản phẩm và cung cấp nhiều hình thức thanh toán như PayPal hoặc Thanh toán khi nhận hàng (COD). Sau khi thanh toán thành công, hệ thống sẽ gửi email xác nhận cho người mua.

**Với Giao hàng**, hệ thống sẽ tính phí vận chuyển dựa trên khoảng cách từ cửa hàng đến địa chỉ khách hàng. **Google Map API** được tích hợp để hiển thị tuyến đường giao hàng và ước tính chi phí. Quản trị viên có thể theo dõi và quản lý thông tin giao hàng.

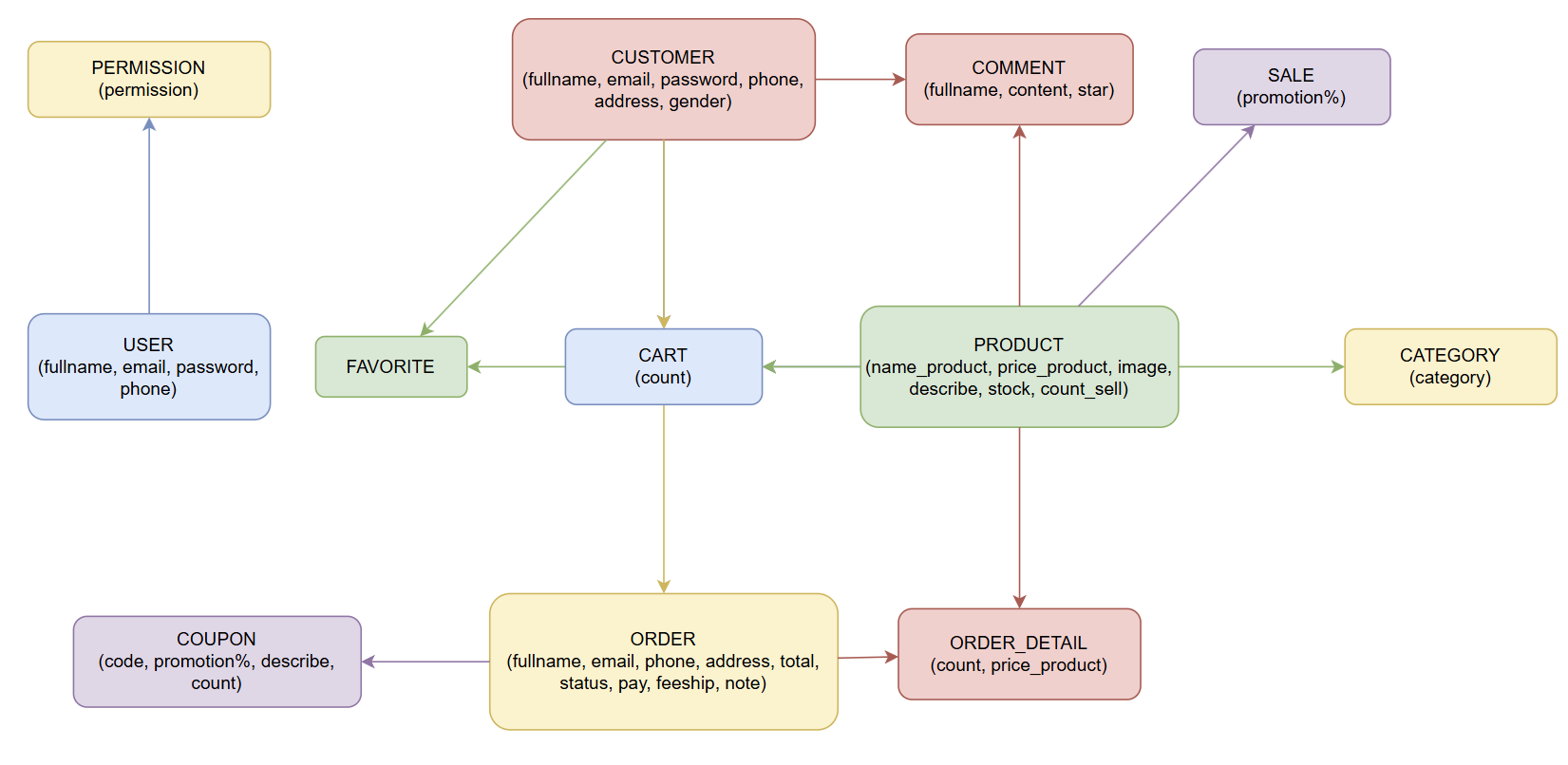
**Với Trò chuyện trực tuyến (Live Chat)**, Người mua có thể tương tác trực tiếp với nhân viên tư vấn qua chat thời gian thực sử dụng WebSocket, giúp nâng cao trải nghiệm và dịch vụ hỗ trợ.

**Với Xác nhận Email**, sau khi đơn hàng được đặt, người mua sẽ tự động nhận được email xác nhận chứa thông tin chi tiết: mã đơn hàng, danh sách sản phẩm, phương thức thanh toán và địa chỉ giao hàng.

**Với Kiểm soát truy cập**, Người mua hoặc Quản trị viên có thể đăng nhập/đăng xuất hệ thống. Nếu là Người mua, hệ thống sẽ chuyển đến trang danh mục sản phẩm; nếu là Quản trị viên, hệ thống sẽ chuyển đến trang quản trị.

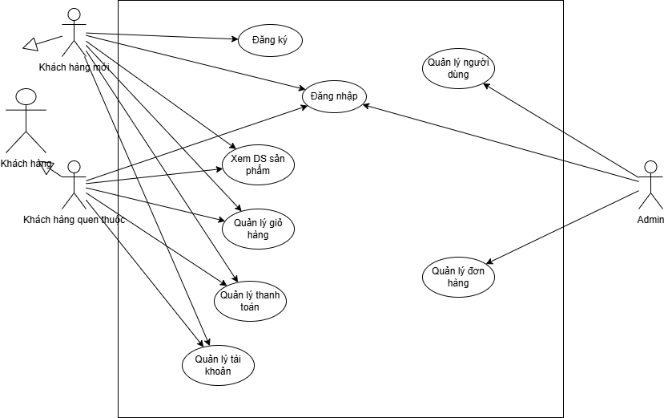
Một số tác vụ khởi tạo cần thực hiện khi Website **Fear Of God** bắt đầu hoạt động gồm: tạo tài khoản Quản trị viên, tài khoản mẫu cho Người mua, và dữ liệu mẫu cho sản phẩm quần áo, loại sản phẩm.

# **Conceptual model**



Hình 1. Conceptual model

# **Use case hệ thống bán hàng**



Hình 2. Use case hệ thống bán hàng

## **User Story**

### **1. Khách hàng mới**

**US-01: Đăng ký tài khoản**

Với vai trò là khách hàng mới, tôi muốn đăng ký tài khoản để có thể mua hàng và theo dõi đơn hàng của mình.

**Tiêu chí chấp nhận:**

* Biểu mẫu đăng ký gồm họ tên, email, số điện thoại và mật khẩu.
* Email phải là duy nhất.
* Mật khẩu phải có ít nhất 8 ký tự.
* Sau khi đăng ký thành công, hệ thống gửi email xác thực tài khoản.

**US-02: Đăng nhập**

Với vai trò là khách hàng mới, tôi muốn đăng nhập sau khi đăng ký để có thể bắt đầu mua hàng trên hệ thống.

**Tiêu chí chấp nhận:**

* Cho phép đăng nhập bằng email và mật khẩu.
* Nếu nhập sai quá 5 lần, hệ thống sẽ khóa tài khoản trong 15 phút.
* Có tùy chọn ghi nhớ đăng nhập để không cần nhập lại lần sau.

**US-03: Xem danh sách sản phẩm**

Với vai trò là khách hàng, tôi muốn xem danh sách các sản phẩm đang có để chọn được sản phẩm cần mua.

**Tiêu chí chấp nhận:**

* Cho phép tìm kiếm, lọc theo danh mục hoặc sắp xếp theo giá và độ mới.
* Mỗi sản phẩm hiển thị ảnh, tên, giá và tình trạng tồn kho.

### **2. Khách hàng quen thuộc**

**US-04: Quản lý giỏ hàng**

Với vai trò là khách hàng quen thuộc, tôi muốn thêm, cập nhật hoặc xóa sản phẩm trong giỏ hàng để chuẩn bị đơn hàng trước khi thanh toán.

**Tiêu chí chấp nhận:**

* Khi thêm sản phẩm, giỏ hàng được cập nhật ngay lập tức.
* Nếu số lượng sản phẩm vượt quá tồn kho hoặc nhỏ hơn 1, hệ thống hiển thị thông báo lỗi.
* Tổng tiền trong giỏ hàng tự động cập nhật khi có thay đổi.

**US-05: Quản lý thanh toán**

Với vai trò là khách hàng quen thuộc, tôi muốn thực hiện thanh toán và chọn phương thức thanh toán phù hợp để hoàn tất đơn hàng.

**Tiêu chí chấp nhận:**

* Cho phép nhập địa chỉ giao hàng và chọn hình thức thanh toán (COD, ví điện tử, thẻ ngân hàng...).
* Hệ thống kiểm tra thông tin hợp lệ trước khi xử lý thanh toán.
* Sau khi thanh toán thành công, hệ thống hiển thị thông báo và gửi email xác nhận.

**US-06: Quản lý tài khoản**

Với vai trò là khách hàng quen thuộc, tôi muốn cập nhật thông tin cá nhân để đảm bảo địa chỉ giao hàng và liên hệ luôn chính xác.

**Tiêu chí chấp nhận:**

* Cho phép thay đổi họ tên, số điện thoại và địa chỉ.
* Sau khi cập nhật, hệ thống ghi nhận thay đổi và gửi email xác nhận.

**US-07: Xem và đánh giá sản phẩm**

Với vai trò là khách hàng quen thuộc, tôi muốn xem và viết đánh giá cho sản phẩm để chia sẻ trải nghiệm mua hàng của mình.

**Tiêu chí chấp nhận:**

* Chỉ khách hàng đã mua sản phẩm mới được phép đánh giá.
* Đánh giá gồm số sao (1–5) và nội dung bình luận.
* Hệ thống hiển thị đánh giá công khai trên trang sản phẩm.

### **3. Quản trị viên**

**US-08: Quản lý người dùng**

Với vai trò là quản trị viên, tôi muốn quản lý tài khoản khách hàng để có thể thêm, sửa, xóa hoặc khóa tài khoản khi cần thiết.

**Tiêu chí chấp nhận:**

* Hiển thị danh sách người dùng gồm tên, email và trạng thái tài khoản.
* Email không được trùng lặp.
* Chỉ quản trị viên có quyền thay đổi thông tin người dùng.
* Hệ thống ghi lại nhật ký khi có thay đổi.

**US-09: Quản lý đơn hàng**

Với vai trò là quản trị viên, tôi muốn xem và cập nhật trạng thái đơn hàng để theo dõi tiến trình giao hàng và xử lý các vấn đề phát sinh.

**Tiêu chí chấp nhận:**

* Hiển thị danh sách đơn hàng gồm mã đơn, ngày đặt, trạng thái và tổng tiền.
* Cho phép thay đổi trạng thái đơn hàng như: đang xử lý, đã giao, đã hủy.
* Gửi thông báo qua email cho khách hàng khi trạng thái thay đổi.

# **Use Case: Quản lý tài khoản khách hàng**

## **1. Pre-Condition**

**Actor:** Admin được phép quản lý tài khoản của các khách hàng, có thể thêm tài khoản mới, kích hoạt / khóa / xóa tài khoản khách hàng.

**Điều kiện:** Admin đã đăng nhập vào hệ thống.

## **2. Trigger**

Để quản lý khách hàng, Admin chọn chức năng **"User"** trên trang quản lý.

## **3. Basic Flow**

| **Bước** | **Mô tả hành động** |
| --- | --- |
| **3.1** | Hệ thống hiển thị màn hình **quản lý User**. |
| **3.2** | Person có thể xem thông tin tài khoản trên màn hình này, hoặc thực hiện các hành động **"Thêm / Sửa / Xóa"**. |
| **3.3** | Nếu thêm tài khoản, Actor chọn vào nút **Thêm** và nhập thông tin tài khoản mới. |
| **3.4** | Hệ thống thực hiện kiểm tra tài khoản mới nhập. **Sau đó lưu vào hệ thống.** *(Alternative: Khi kiểm tra không hợp lệ → xem mục 4)* |
| **3.5** | Nếu sửa tài khoản, Actor chọn nút **Sửa** và cập nhật **role** tài khoản mới. |
| **3.6** | Người dùng chọn **xác nhận thay đổi**, sau đó hệ thống **lưu vào cơ sở dữ liệu**. |
| **3.7** | Nếu xóa tài khoản, Actor chọn nút **Delete** để xóa tài khoản. |
| **3.8** | Người dùng chọn **xác nhận thay đổi**, sau đó hệ thống **lưu lại kết quả**. |
| **3.9** | Hệ thống quay lại **màn hình quản lý User**. |

## **4. Alternative Flow**

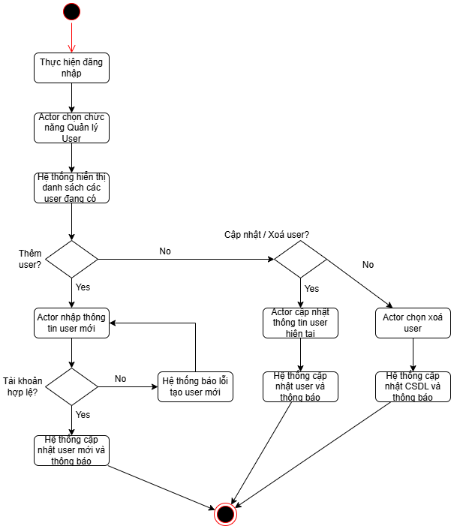
| **Tình huống** | **Mô tả** |
| --- | --- |
| **4.1** | Khi kiểm tra không hợp lệ: • Hệ thống hiển thị thông báo lỗi cho Person. • Cho phép Person nhập lại thông tin tài khoản. • Quay trở lại bước **3.3**. |

## **5. Business Rule**

Yêu cầu về việc thông báo lỗi nhập sai dữ liệu:

* Màn hình hiển thị thông báo lỗi về **tài khoản hoặc mật khẩu**.
* Các trường dữ liệu không hợp lệ sẽ được **bôi đỏ**.
* Khi con trỏ chuột focus vào trường dữ liệu đầu tiên không hợp lệ, hệ thống sẽ **highlight** trường đó.
* Khi user chỉnh sửa đúng dữ liệu, **màu sắc của trường dữ liệu trở lại bình thường.**

## **6. Activity Diagram**



Hình . Activity diagram quản lý tài khoản

# **Use Case: Quản lý thanh toán**

## **1. Pre-Condition**

**Actor:** Person có thể quản lý thanh toán khi mua sản phẩm trong giỏ hàng, có thể chọn phương thức thanh toán và nhập địa chỉ giao hàng.

**Điều kiện:** Person đã đăng nhập vào hệ thống và chọn mua sản phẩm trong giỏ hàng.

## **2. Trigger**

Để quản lý thanh toán, Person chọn chức năng **"Thanh toán"** khi mua sản phẩm trong giỏ hàng.

## **3. Basic Flow**

| **Bước** | **Mô tả hành động** |
| --- | --- |
| **3.1** | Hệ thống hiển thị màn hình **quản lý thanh toán**. |
| **3.2** | Person nhập **thông tin giao hàng** và **địa chỉ nhận hàng**. |
| **3.3** | Hệ thống kiểm tra thông tin giao hàng và **tính phí ship**.  *(Alternative: Khi kiểm tra địa chỉ không hợp lệ → xem mục 4)* |
| **3.4** | Actor chọn **phương thức thanh toán**. |
| **3.5** | Nếu thanh toán **COD**, đơn hàng được lưu vào **CSDL** và hệ thống **thông báo thành công**. |
| **3.6** | Nếu thanh toán **chuyển khoản**, Actor thực hiện quá trình chuyển khoản. |
| **3.7** | Hệ thống kiểm tra **thông tin giao dịch chuyển khoản**.  *(Alternative: Khi kiểm tra chuyển khoản không hợp lệ → xem mục 4)* |
| **3.8** | Đơn hàng được lưu vào **CSDL** và hệ thống **thông báo thanh toán thành công**. |

## **4. Alternative Flow**

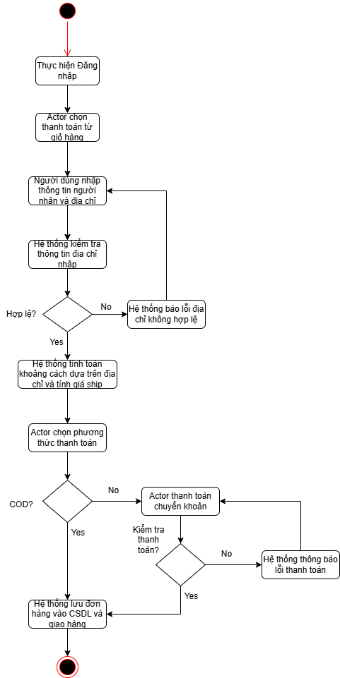
| **Tình huống** | **Mô tả** |
| --- | --- |
| **4.1** | Khi kiểm tra địa chỉ **không hợp lệ**: • Hệ thống hiển thị thông báo lỗi. • Cho phép Actor nhập lại địa chỉ mới. |
| **4.2** | Khi kiểm tra **chuyển khoản không hợp lệ**: • Hệ thống thông báo đơn hàng chưa thanh toán thành công. • Cho phép Actor **thực hiện thanh toán lại**. |

## **5. Business Rule**

Yêu cầu về việc thông báo lỗi nhập sai dữ liệu:

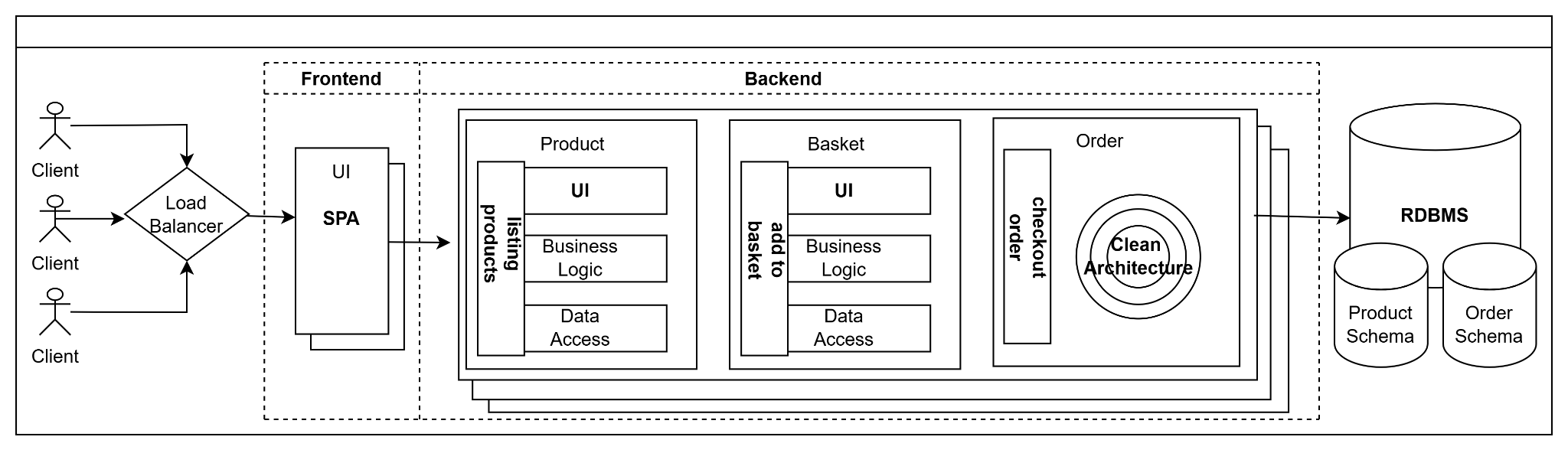
* Màn hình hiển thị thông báo lỗi về **địa chỉ giao hàng hoặc thông tin thanh toán**.
* Các trường dữ liệu không hợp lệ sẽ được **bôi đỏ**.
* Khi con trỏ chuột focus vào trường dữ liệu đầu tiên không hợp lệ, hệ thống sẽ **highlight** trường đó.
* Khi user sửa đúng dữ liệu, **màu sắc của trường dữ liệu trở lại bình thường.**

## **6. Activity Diagram**



Hình . Activity diagram quản lý thanh toán

# **Sơ đồ khối (Block Diagram)**



Hình 5. Sơ đồ khối

### **1. Mô tả tổng quan**

Hệ thống website bán quần áo được thiết kế dựa trên kiến trúc đa tầng (Multi-tier Architecture), chia thành các lớp xử lý độc lập gồm:

* Lớp giao diện người dùng (Frontend – SPA)
* Lớp nghiệp vụ (Business Logic – Backend)
* Lớp dữ liệu (Database – RDBMS)

Kiến trúc này giúp hệ thống dễ mở rộng, dễ bảo trì và có thể triển khai độc lập từng phần, đồng thời đảm bảo hiệu suất hoạt động ổn định khi lượng người dùng tăng.

### **2. Giải thích chi tiết các thành phần trong sơ đồ**

**2.1. Người dùng (User)**

Là người sử dụng cuối cùng của hệ thống (khách hàng, nhân viên bán hàng hoặc quản trị viên).

Người dùng truy cập vào website thông qua trình duyệt web hoặc thiết bị di động, tương tác với giao diện SPA (Single Page Application).

**2.2. Load Balancer (Bộ cân bằng tải)**

Đây là thành phần trung gian có nhiệm vụ:

* Phân phối đều lưu lượng truy cập từ người dùng đến các máy chủ backend.
* Giảm tải cho hệ thống khi có nhiều người dùng truy cập cùng lúc.
* Tăng tính sẵn sàng (Availability) – nếu một máy chủ bị lỗi, Load Balancer sẽ chuyển hướng sang máy khác.

**2.3. SPA (Single Page Application – ReactJS)**

Đây là lớp giao diện người dùng của website bán quần áo, được xây dựng bằng ReactJS.

Vai trò chính:

* Hiển thị danh sách sản phẩm, chi tiết sản phẩm, giỏ hàng, thanh toán, quản lý tài khoản.
* Gửi các yêu cầu HTTP đến Backend thông qua API để lấy dữ liệu.
* Quản lý trạng thái hiển thị và phản hồi nhanh nhờ cơ chế SPA – chỉ tải lại phần nội dung thay đổi, không tải lại toàn trang.

Ví dụ: Khi khách hàng thêm sản phẩm vào giỏ hàng, ReactJS chỉ cập nhật phần giỏ hàng trên giao diện mà không cần tải lại toàn bộ trang.

**2.4. Các Module trong Backend**

Backend được chia thành 3 module chính tương ứng với các nghiệp vụ của hệ thống:

**a. Product Module – Quản lý sản phẩm**

* UI: Giao diện quản trị để thêm, sửa, xóa và xem danh sách sản phẩm.
* Business Logic:
  + Xử lý logic khi người dùng lọc, tìm kiếm hoặc phân loại sản phẩm.
  + Tính toán giá sau khuyến mãi, xử lý tồn kho, quản lý ảnh sản phẩm.
* Data Access:
  + Giao tiếp trực tiếp với cơ sở dữ liệu (Product Schema).
  + Lấy dữ liệu sản phẩm, danh mục, khuyến mãi từ bảng Product, Category, Sale.

**b. Basket Module – Giỏ hàng**

* UI: Hiển thị danh sách sản phẩm khách hàng đã thêm vào giỏ.
* Business Logic:
  + Tính tổng tiền, số lượng và các mã giảm giá áp dụng.
  + Cập nhật số lượng sản phẩm trong giỏ khi khách hàng thêm/xóa sản phẩm.
* Data Access:
  + Lưu tạm thông tin giỏ hàng vào cơ sở dữ liệu hoặc session người dùng.

**c. Order Module – Đặt hàng**

* Checkout Order:
  + Xử lý quy trình thanh toán (COD, ví điện tử, PayPal, v.v.).
  + Kiểm tra tính hợp lệ của đơn hàng, tạo chi tiết đơn hàng và tính phí giao hàng.
* Clean Architecture:
  + Lõi hệ thống (core business logic) tách biệt với phần giao diện và dữ liệu.
  + Dễ bảo trì, dễ mở rộng, và có thể tái sử dụng trong các nền tảng khác (mobile, POS…).

Ví dụ: Khi người dùng xác nhận đơn hàng, module “Order” kiểm tra thông tin người dùng, địa chỉ, tính toán phí giao hàng (Google Maps API), gửi email xác nhận (SMTP), rồi lưu đơn vào cơ sở dữ liệu.

**2.5. RDBMS (Cơ sở dữ liệu quan hệ)**

Cơ sở dữ liệu được thiết kế theo mô hình Relational Database Management System (RDBMS) với hai schema chính:

| Schema | Chức năng |
| --- | --- |
| Product Schema | Quản lý dữ liệu về sản phẩm, danh mục, khuyến mãi. |
| Order Schema | Quản lý thông tin đơn hàng, chi tiết đơn hàng, thanh toán, khách hàng. |

Cơ sở dữ liệu có thể triển khai bằng MongoDB hoặc MySQL, được truy cập thông qua lớp Data Access trong backend.

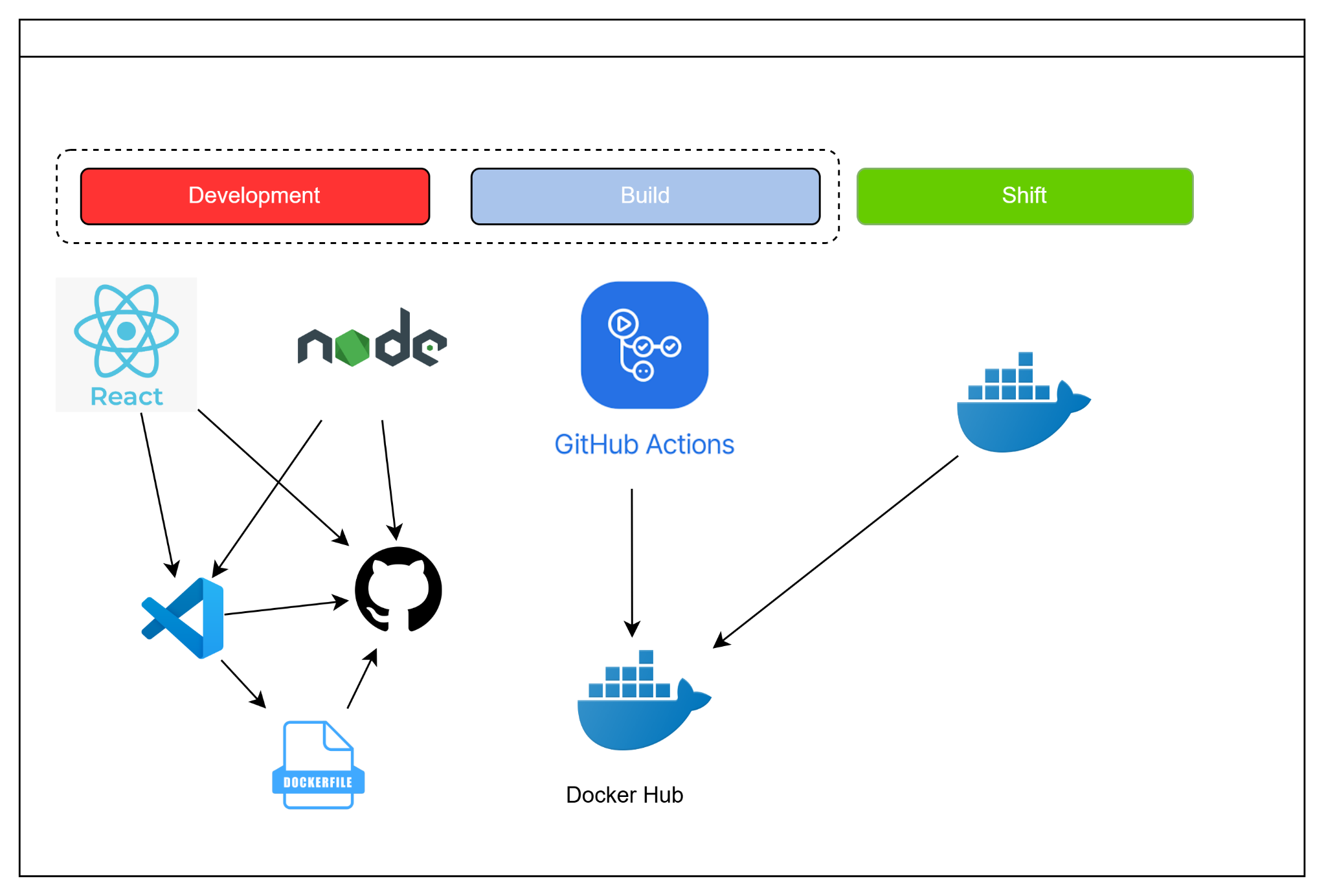
### **3. Luồng hoạt động của hệ thống**

* 1. Khách hàng truy cập website (SPA) để xem sản phẩm.
  2. SPA gửi yêu cầu đến Backend API để lấy danh sách sản phẩm từ Product Schema.
  3. Khi khách hàng thêm sản phẩm vào giỏ hàng, Basket Module xử lý và lưu lại tạm.
  4. Khi người dùng đặt hàng, Order Module kiểm tra dữ liệu, tạo đơn, gọi PayPal API hoặc SMTP để xác nhận thanh toán.
  5. CSDL RDBMS lưu toàn bộ dữ liệu giao dịch, đảm bảo tính toàn vẹn và truy xuất lịch sử mua hàng.

### **4. Ưu điểm của kiến trúc**

* Phân tầng rõ ràng: Dễ dàng phát triển, bảo trì và kiểm thử độc lập.
* Khả năng mở rộng cao: Có thể thêm module mới (ví dụ: giao hàng, đánh giá sản phẩm) mà không ảnh hưởng phần khác.
* Đáp ứng nhiều người dùng: Load Balancer giúp hệ thống xử lý đồng thời hàng trăm yêu cầu.
* Hiệu suất cao: SPA giảm tải cho server, chỉ gọi API khi cần.
* Tuân thủ nguyên tắc Clean Architecture: Dễ bảo trì, tách biệt nghiệp vụ với dữ liệu và giao diện.

# **Deployment View**



Hình 6. Deployment view

### **1. Mô tả tổng quan**

Hệ thống KTPM FEAR OF GOD được triển khai theo quy trình CI/CD (Continuous Integration / Continuous Deployment) nhằm tự động hóa việc phát triển, build, đóng gói và triển khai ứng dụng web bán quần áo.

Kiến trúc triển khai bao gồm ba giai đoạn chính:

* Development (Phát triển)
* Build (Xây dựng và đóng gói ứng dụng)
* Shift (Triển khai vận hành hệ thống)

Toàn bộ quy trình được thiết lập để đảm bảo hệ thống hoạt động ổn định, dễ mở rộng và dễ cập nhật phiên bản.

### **2. Các giai đoạn triển khai**

**2.1. Development (Phát triển ứng dụng)**

Giai đoạn này diễn ra trên môi trường của lập trình viên (local environment).  
Các công cụ và thành phần chính bao gồm:

* ReactJS:
  + Dùng để xây dựng giao diện người dùng (UI).
  + Quản lý các trang như trang sản phẩm, giỏ hàng, thanh toán, đăng ký, đăng nhập.
* Node.js:
  + Xử lý logic nghiệp vụ ở phía backend.
  + Tạo API giao tiếp giữa frontend và cơ sở dữ liệu (MongoDB).
* Visual Studio Code (VS Code):
  + Môi trường phát triển chính để viết và gỡ lỗi mã nguồn.
* Dockerfile:
  + Được tạo trong dự án để định nghĩa cách ứng dụng sẽ được đóng gói trong container.
  + Mô tả cách cài đặt Node.js, copy mã nguồn, cài đặt dependencies và khởi chạy ứng dụng.

Quy trình tại giai đoạn Development:

1. Lập trình viên viết code và chạy thử ứng dụng trên môi trường local.
2. Code được commit lên GitHub repository.
3. Mỗi lần có thay đổi, GitHub Actions sẽ tự động kích hoạt quá trình build.

**2.2. Build (Xây dựng và kiểm thử tự động)**

* GitHub Actions:
  + Là công cụ tự động hóa quá trình CI/CD.
  + Khi có commit mới, GitHub Actions sẽ:
    - Kiểm tra cú pháp và dependencies.
    - Build ứng dụng React và Node.js.
    - Chạy test tự động (nếu có).
    - Đóng gói ứng dụng thành image Docker.
* Docker Hub:
  + Nơi lưu trữ image Docker sau khi build thành công.
  + Image này chứa toàn bộ mã nguồn backend và frontend đã sẵn sàng để triển khai.

Quy trình tại giai đoạn Build:

1. GitHub Actions build dự án và tạo Docker image.
2. Image được push lên Docker Hub.
3. Mọi máy chủ (server) có thể pull image này để triển khai.

**2.3. Shift (Triển khai và vận hành hệ thống)**

* Docker Container (Production):
  + Máy chủ triển khai sẽ pull image từ Docker Hub về.
  + Container được chạy để khởi động hệ thống (React + Node + MongoDB).
* Quy trình vận hành:
  + Container chạy frontend (ReactJS) – hiển thị giao diện website.
  + Container chạy backend (Node.js) – xử lý yêu cầu người dùng, truy cập cơ sở dữ liệu.
  + Dữ liệu được lưu trong MongoDB (hoặc RDBMS trong mô hình tổng thể).

Ưu điểm:

* Dễ dàng cập nhật ứng dụng bằng cách chỉ cần build lại image mới.
* Triển khai nhanh chóng, đồng nhất trên mọi môi trường (Windows, Linux, Cloud...).
* Tự động hóa toàn bộ quy trình build và deploy.

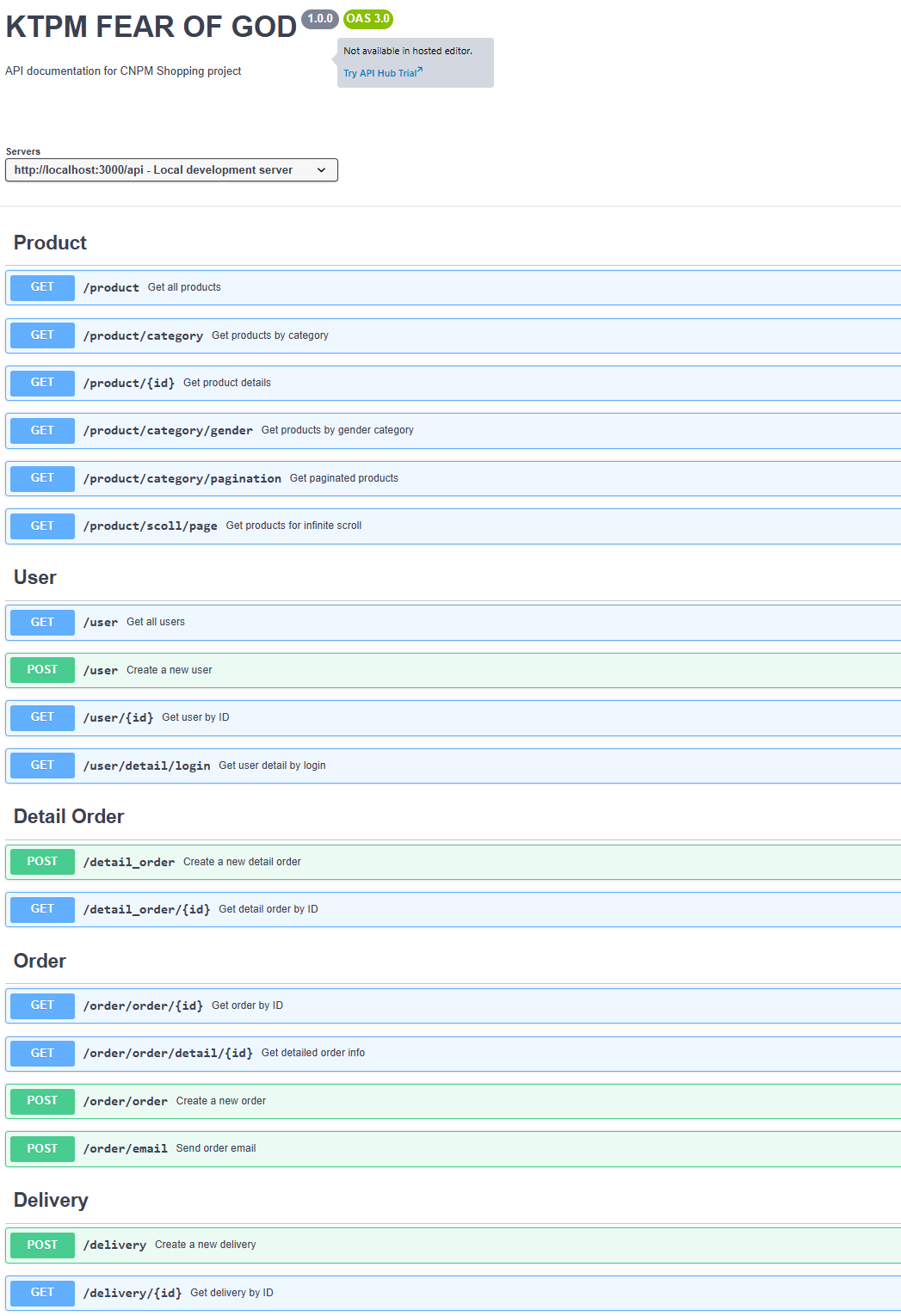
### **3. Quy trình CI/CD tổng thể**

* 1. Lập trình viên phát triển tính năng mới và push code lên GitHub.
  2. GitHub Actions tự động kích hoạt pipeline: build – test – tạo Docker image.
  3. Image được lưu trữ tại Docker Hub.
  4. Máy chủ triển khai (production server) pull image về và chạy container mới.
  5. Ứng dụng ReactJS + Node.js + MongoDB được khởi chạy, phục vụ người dùng cuối.

### **4. Ưu điểm của mô hình triển khai**

| Tiêu chí | Lợi ích đạt được |
| --- | --- |
| Tự động hóa | Không cần triển khai thủ công, giảm sai sót con người |
| Đồng nhất môi trường | Mọi nơi chạy cùng một image, loại bỏ lỗi “chạy được ở máy tôi” |
| Tốc độ triển khai cao | Build và deploy trong vài phút nhờ Docker |
| Dễ bảo trì | Có thể rollback hoặc cập nhật nhanh khi lỗi phát sinh |
| Tính mở rộng cao | Dễ dàng scale container khi lượng truy cập tăng |

# **Communication View**



Hình 7. Communication view

## **1. Giới thiệu**

Communication View mô tả cách các thành phần trong hệ thống KTPM FEAR OF GOD (hệ thống bán hàng trực tuyến) giao tiếp với nhau thông qua API (Application Programming Interface).

Các API này cho phép frontend (ứng dụng React) và backend (Node.js) trao đổi dữ liệu với cơ sở dữ liệu (Database) một cách thống nhất và có kiểm soát.

Toàn bộ API được triển khai tại server: **http://localhost:3000/api**

## **2. Mô hình giao tiếp tổng quan**

1. Client (React SPA) gửi các yêu cầu HTTP (GET, POST) đến server.
2. Server Node.js tiếp nhận, xử lý nghiệp vụ và truy xuất dữ liệu từ cơ sở dữ liệu.
3. Server trả phản hồi (Response) lại cho client ở dạng JSON.
4. Các module (Product, User, Order, Delivery, Detail Order) giao tiếp qua RESTful API.

## **3. Các nhóm API chính**

### **3.1. Product (Sản phẩm)**

Các API dùng để lấy thông tin sản phẩm, danh mục và chi tiết hiển thị cho người dùng.

| Phương thức | Endpoint | Chức năng |
| --- | --- | --- |
| GET | /product | Lấy tất cả sản phẩm |
| GET | /product/category | Lọc sản phẩm theo danh mục |
| GET | /product/{id} | Lấy chi tiết sản phẩm theo ID |
| GET | /product/category/gender | Lọc sản phẩm theo giới tính |
| GET | /product/category/pagination | Lấy danh sách sản phẩm phân trang |
| GET | /product/scoll/page | Lấy danh sách sản phẩm cuộn vô hạn (infinite scroll) |

### **3.2. User (Người dùng)**

Các API quản lý thông tin người dùng, đăng nhập và đăng ký tài khoản.

| Phương thức | Endpoint | Chức năng |
| --- | --- | --- |
| GET | /user | Lấy danh sách người dùng |
| POST | /user | Tạo tài khoản người dùng mới |
| GET | /user/{id} | Lấy thông tin người dùng theo ID |
| GET | /user/detail/login | Lấy thông tin người dùng khi đăng nhập |

### **3.3. Detail Order (Chi tiết đơn hàng)**

Các API phục vụ việc lưu trữ và truy xuất thông tin chi tiết của từng đơn hàng.

| Phương thức | Endpoint | Chức năng |
| --- | --- | --- |
| POST | /detail\_order | Tạo chi tiết đơn hàng mới |
| GET | /detail\_order/{id} | Lấy chi tiết đơn hàng theo ID |

### **3.4. Order (Đơn hàng)**

Các API dùng để quản lý và xử lý đơn hàng của khách.

| Phương thức | Endpoint | Chức năng |
| --- | --- | --- |
| GET | /order/order/{id} | Lấy đơn hàng theo ID |
| GET | /order/order/detail/{id} | Lấy thông tin chi tiết đơn hàng |
| POST | /order/order | Tạo đơn hàng mới |
| POST | /order/email | Gửi email xác nhận đơn hàng |

### **3.5. Delivery (Giao hàng)**

Các API đảm nhận việc tạo và truy xuất thông tin giao hàng.

| Phương thức | Endpoint | Chức năng |
| --- | --- | --- |
| POST | /delivery | Tạo thông tin giao hàng mới |
| GET | /delivery/{id} | Lấy thông tin giao hàng theo ID |

## **4. Cách thức giao tiếp**

* Phương thức giao tiếp: RESTful API thông qua HTTP protocol.
* Định dạng dữ liệu: JSON (JavaScript Object Notation).
* Kiểu xác thực: Token-based Authentication (nếu được cấu hình).
* Frontend gọi API: bằng fetch hoặc axios trong React.
* Backend xử lý: thông qua Express.js, liên kết với cơ sở dữ liệu MySQL hoặc MongoDB.

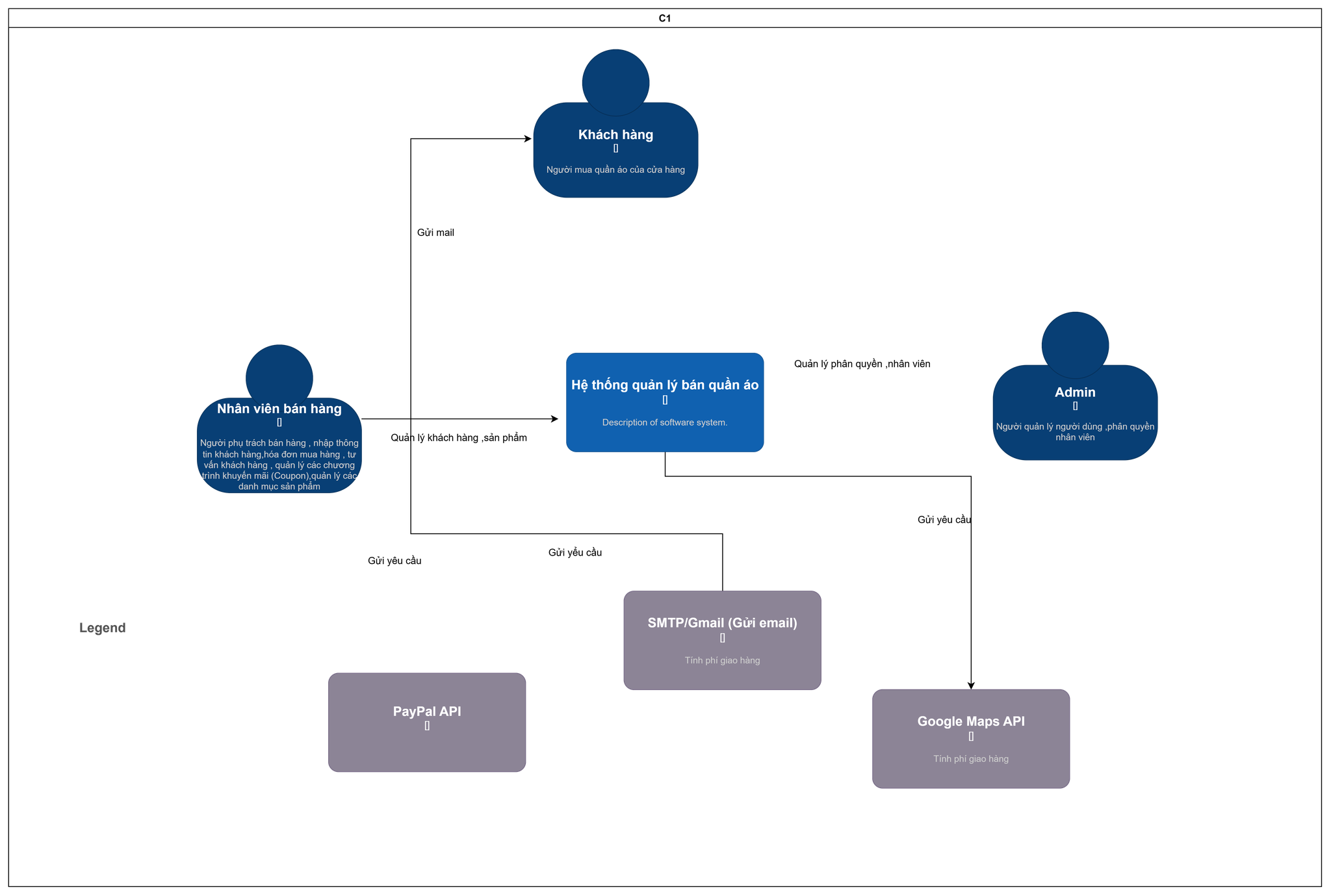
## **5. Luồng xử lý ví dụ**

Trường hợp: Người dùng đặt hàng

1. Người dùng đăng nhập → Gửi yêu cầu POST /user/detail/login.
2. Hệ thống trả về thông tin người dùng và token xác thực.
3. Người dùng chọn sản phẩm → Gọi GET /product/{id} để lấy chi tiết.
4. Thêm sản phẩm vào giỏ hàng → Tạo mới đơn hàng bằng POST /order/order.
5. Tạo chi tiết đơn hàng bằng POST /detail\_order.
6. Hệ thống gửi email xác nhận bằng POST /order/email.
7. Cuối cùng, tạo giao hàng bằng POST /delivery.

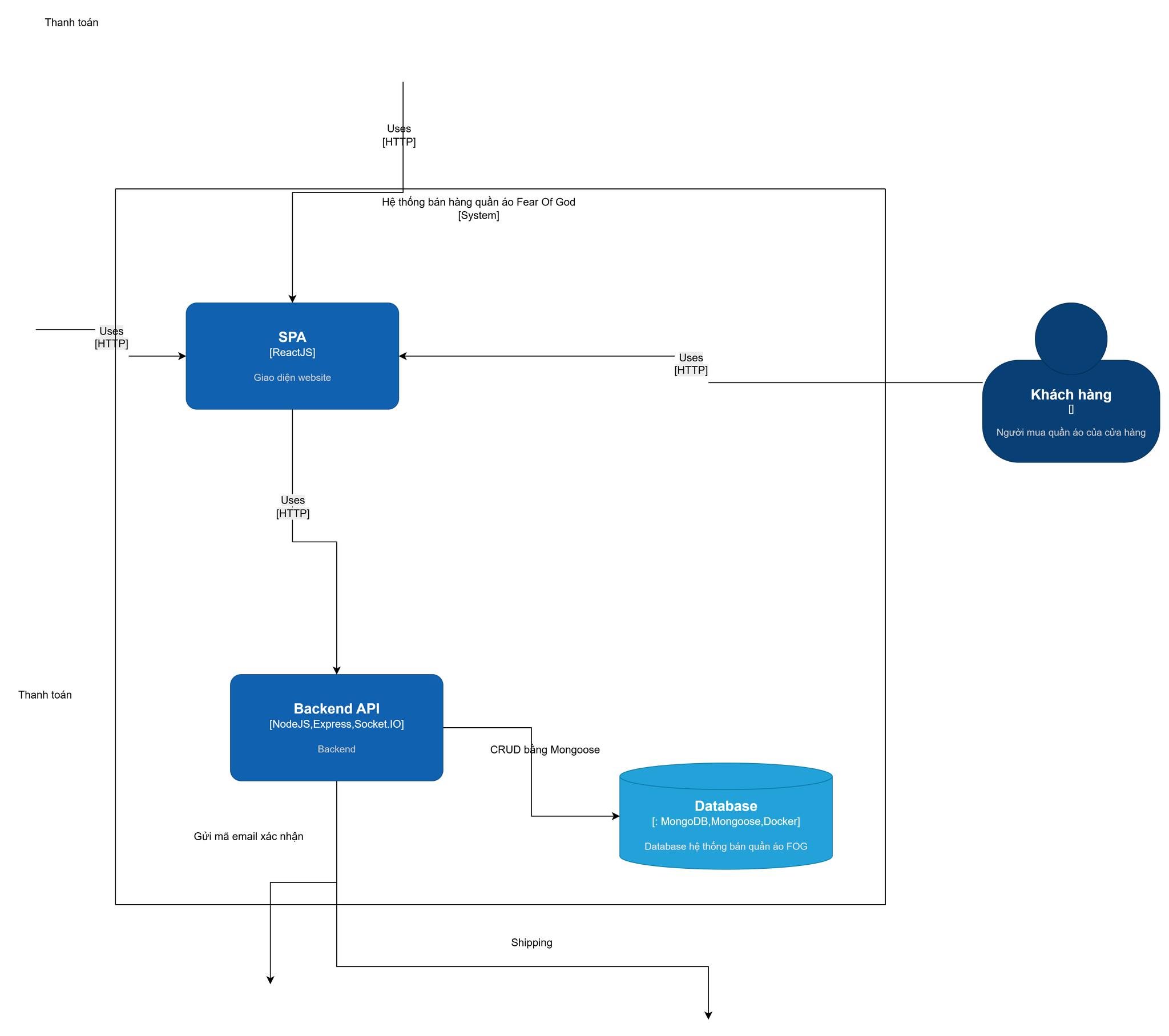
# **Decomposition View: C4**

## **C1 - System Context**



Hình 8. C1 - System context

## **C2 - Containers**



Hình 9. C2 - Containers

### **1. Mô tả tổng quan hệ thống**

Hệ thống “Quản lý bán quần áo” là một ứng dụng hỗ trợ các hoạt động bán hàng trực tuyến, bao gồm quản lý sản phẩm, đơn hàng, khách hàng, nhân viên và các chức năng thanh toán – giao hàng. Hệ thống phục vụ cho ba nhóm người dùng chính: Khách hàng, Nhân viên bán hàng và Quản trị viên (Admin).

Hệ thống cũng tích hợp các dịch vụ bên ngoài như PayPal API, SMTP/Gmail và Google Maps API để hỗ trợ thanh toán, gửi email và tính phí giao hàng.

### **2. Các tác nhân bên ngoài và vai trò**

Khách hàng

* Là người mua sản phẩm từ cửa hàng quần áo.
* Có thể đăng ký tài khoản, đăng nhập, tìm kiếm và xem thông tin sản phẩm.
* Thêm sản phẩm vào giỏ hàng, thực hiện thanh toán trực tuyến và theo dõi đơn hàng.

Nhân viên bán hàng

* Là người chịu trách nhiệm cập nhật thông tin sản phẩm và đơn hàng.
* Quản lý thông tin khách hàng, xử lý khiếu nại hoặc hỗ trợ các yêu cầu đặt hàng.
* Theo dõi tình trạng giao hàng và phối hợp với các bộ phận khác để hoàn tất đơn.

Quản trị viên (Admin)

* Là người có quyền cao nhất trong hệ thống.
* Quản lý người dùng, phân quyền cho nhân viên, giám sát hoạt động của hệ thống.
* Có quyền thêm, sửa, xóa dữ liệu và truy cập toàn bộ các báo cáo.

Các dịch vụ bên ngoài

1. PayPal API
   * Được sử dụng để xử lý thanh toán trực tuyến cho khách hàng.
   * Giúp đảm bảo an toàn trong các giao dịch mua bán.
2. SMTP/Gmail API
   * Dùng để gửi email xác nhận đơn hàng hoặc thông báo khuyến mãi cho khách hàng.
   * Cũng có thể gửi thông báo khi đơn hàng thay đổi trạng thái.
3. Google Maps API
   * Hỗ trợ tính toán khoảng cách và chi phí giao hàng dựa trên địa chỉ của khách hàng.
   * Giúp xác định tuyến giao hàng hợp lý.

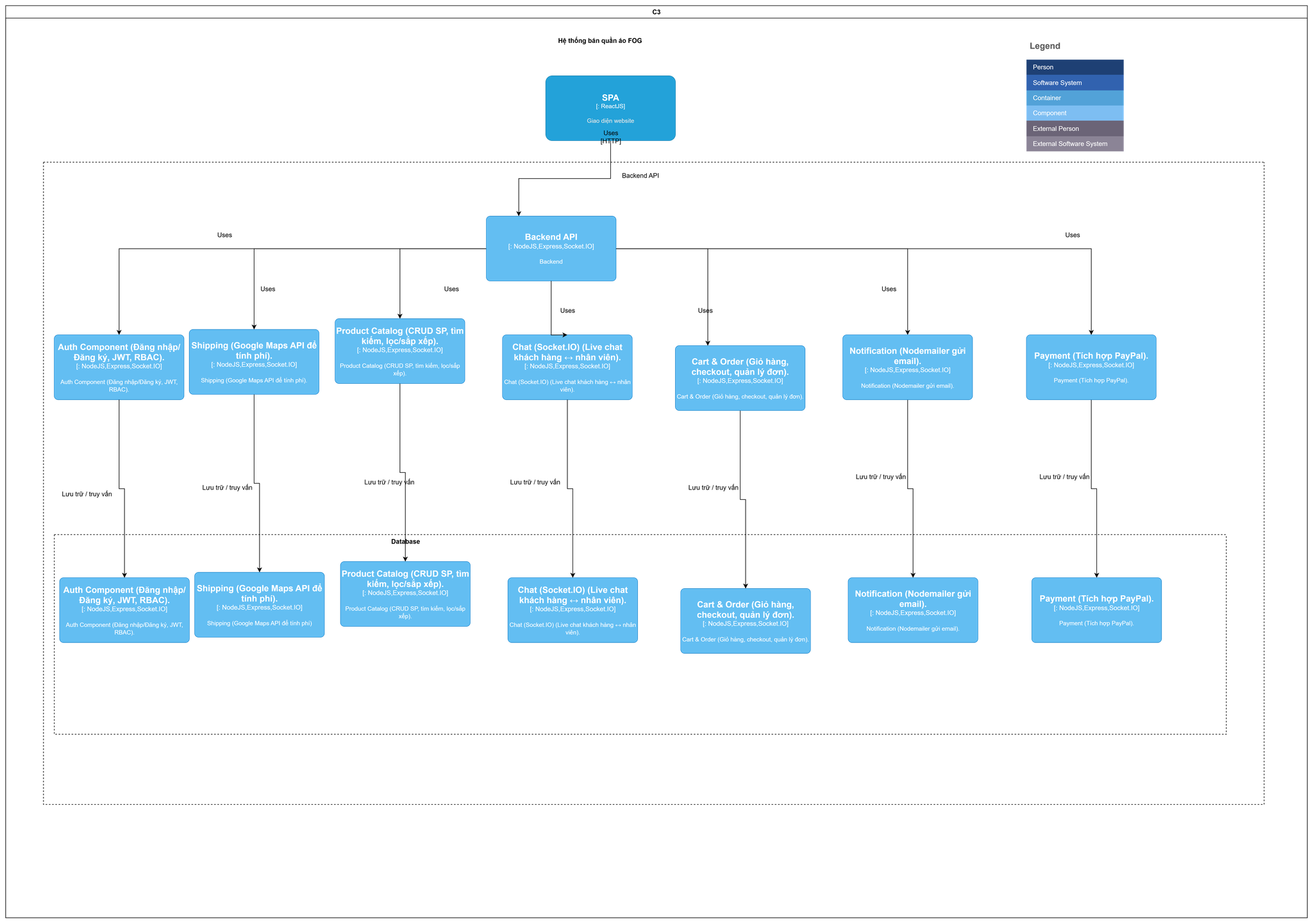
### **3. Mối quan hệ giữa các thành phần**

* Khách hàng tương tác trực tiếp với hệ thống qua giao diện website hoặc ứng dụng.
* Nhân viên bán hàng thao tác trong hệ thống để xử lý và cập nhật đơn hàng.
* Admin giám sát và điều hành toàn bộ hoạt động của hệ thống.
* Hệ thống sử dụng PayPal API, SMTP/Gmail API và Google Maps API để mở rộng chức năng.

### **4. Tóm tắt vai trò của hệ thống**

Hệ thống “Quản lý bán quần áo” đóng vai trò trung tâm, là cầu nối giữa người dùng nội bộ (Admin, nhân viên) và người dùng bên ngoài (khách hàng), đồng thời đảm bảo việc liên kết với các dịch vụ bên thứ ba một cách an toàn và hiệu quả.

## **C3 - Components**



Hình 10. C3 - Components

### **1. Mô tả tổng quan**

Hệ thống “Quản lý bán quần áo – KTPM FEAR OF GOD” được thiết kế theo mô hình 3 lớp (Three-tier Architecture) gồm:

* Frontend (SPA – ReactJS): Xử lý giao diện người dùng.
* Backend API (Node.js + Express): Xử lý logic nghiệp vụ và cung cấp dữ liệu thông qua API.
* Database (MongoDB): Lưu trữ toàn bộ dữ liệu của hệ thống.

Mô hình Container giúp chia hệ thống thành các thành phần riêng biệt, có thể triển khai, bảo trì và mở rộng độc lập, đồng thời đảm bảo khả năng tích hợp với các dịch vụ bên ngoài.

### **2. Các Container chính trong hệ thống**

**2.1. SPA (Single Page Application) – ReactJS**

Vai trò:

* Là giao diện website mà khách hàng trực tiếp sử dụng để tương tác với hệ thống.
* Hiển thị sản phẩm, giỏ hàng, thông tin tài khoản và cho phép người dùng thực hiện đặt hàng hoặc thanh toán.

Chức năng chính:

* Gửi yêu cầu HTTP đến Backend API để lấy dữ liệu sản phẩm, người dùng và đơn hàng.
* Xử lý hiển thị giao diện và phản hồi người dùng theo thời gian thực.
* Quản lý phiên đăng nhập (token) và lưu tạm dữ liệu trên trình duyệt (local storage).

Công nghệ sử dụng:

* ReactJS
* HTML5, CSS3, JavaScript (ES6)
* Axios / Fetch API để gọi dữ liệu từ backend

**2.2. Backend API – Node.js, Express, Socket.IO**

Vai trò:

* Là thành phần trung tâm xử lý toàn bộ nghiệp vụ của hệ thống.
* Đóng vai trò là cầu nối giữa giao diện người dùng và cơ sở dữ liệu.

Chức năng chính:

* Nhận và xử lý các yêu cầu từ SPA thông qua HTTP.
* Quản lý tài khoản, sản phẩm, đơn hàng, và xử lý thanh toán.
* Tích hợp các API bên ngoài như PayPal, Google Maps và SMTP (Gmail).
* Quản lý session, xác thực người dùng, và đồng bộ hóa dữ liệu theo thời gian thực với Socket.IO.

Công nghệ sử dụng:

* Node.js
* Express.js
* Socket.IO
* Mongoose (kết nối MongoDB)
* JSON Web Token (JWT) cho xác thực

**2.3. Database – MongoDB (Mongoose, Docker)**

Vai trò:

* Lưu trữ toàn bộ dữ liệu của hệ thống bao gồm sản phẩm, người dùng, đơn hàng, chi tiết đơn hàng và lịch sử giao dịch.

Chức năng chính:

* Cung cấp dữ liệu cho Backend thông qua các truy vấn (query).
* Lưu trữ dữ liệu dạng NoSQL, giúp truy xuất linh hoạt và tốc độ cao.
* Đảm bảo tính toàn vẹn dữ liệu và hỗ trợ sao lưu, khôi phục dễ dàng.

Công nghệ sử dụng:

* MongoDB
* Mongoose (ORM để thao tác dữ liệu trong Node.js)
* Docker (đóng gói và triển khai cơ sở dữ liệu nhanh chóng)

### **3. Luồng giao tiếp giữa các Container**

1. Khách hàng truy cập website thông qua trình duyệt và gửi yêu cầu tới SPA (ReactJS).
2. SPA gọi các API qua giao thức HTTP đến Backend (Node.js + Express).
3. Backend xử lý yêu cầu, truy cập Database (MongoDB) để lấy hoặc ghi dữ liệu.
4. Kết quả được trả về SPA, hiển thị lại cho người dùng.
5. Nếu có sự kiện thời gian thực (như cập nhật trạng thái đơn hàng), Socket.IO sẽ truyền dữ liệu ngay mà không cần tải lại trang.

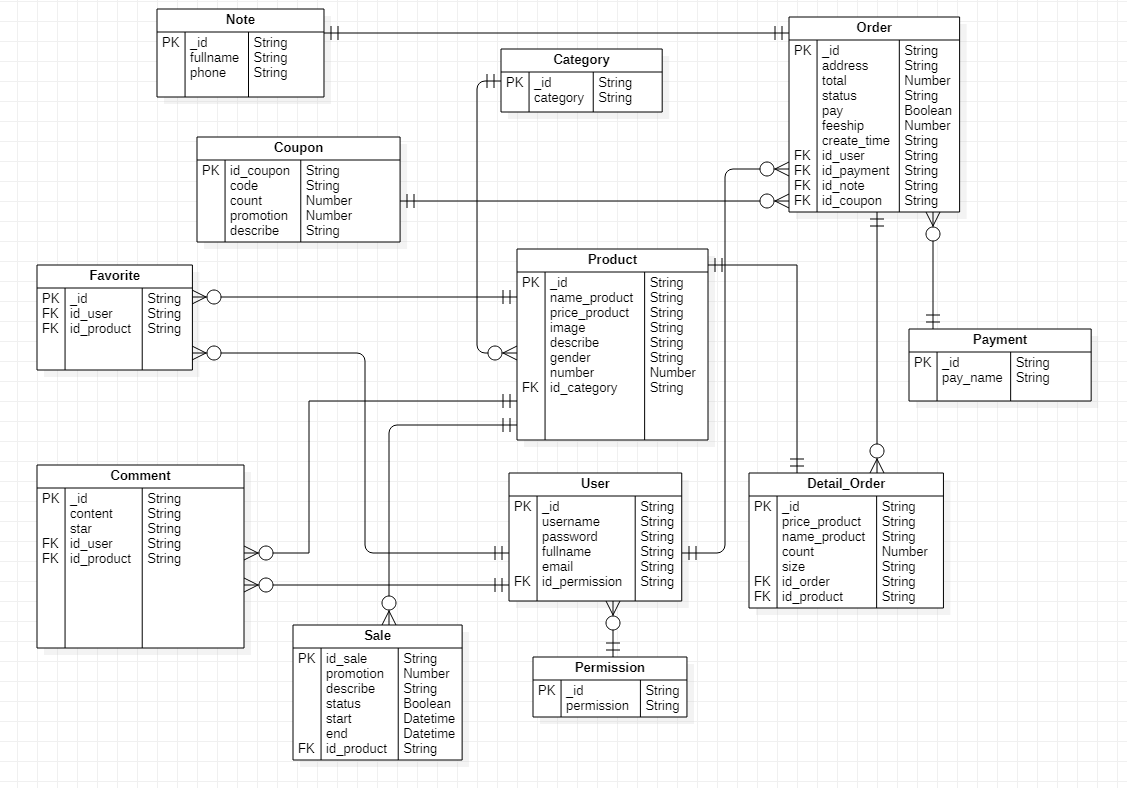
### **4. Mối liên kết giữa các thành phần**

| Thành phần | Kết nối đến | Giao thức / Cách thức | Mục đích |
| --- | --- | --- | --- |
| Khách hàng | SPA | HTTP/HTTPS | Giao tiếp người dùng – giao diện |
| SPA | Backend API | HTTP/RESTful API | Gửi yêu cầu, nhận dữ liệu |
| Backend API | Database | Mongoose Query | Truy vấn, cập nhật dữ liệu |
| Backend API | Dịch vụ bên ngoài (PayPal, Gmail, Google Maps) | API Integration | Thanh toán, gửi mail, định vị |

### **5. Ưu điểm của mô hình Container**

* Tách biệt rõ ràng giữa giao diện, nghiệp vụ và dữ liệu, dễ dàng bảo trì.
* Có thể mở rộng độc lập từng thành phần, ví dụ: scale backend hoặc database khi tải lớn.
* Tương thích với Docker, dễ triển khai lên các môi trường khác nhau.
* Hỗ trợ thời gian thực thông qua Socket.IO, nâng cao trải nghiệm người dùng.

## **C4 - Code**



Hình 11. C4 - Code

### **1. Mô tả tổng quan**

Mô hình **C4 – Code** thể hiện cấu trúc chi tiết của các **lớp dữ liệu (entities)** trong hệ thống “Quản lý bán quần áo”.

Mỗi bảng (class) biểu diễn cho một **thực thể nghiệp vụ (business entity)** như người dùng, sản phẩm, đơn hàng, danh mục, v.v...  
Các thực thể được liên kết với nhau thông qua **quan hệ khóa chính (PK)** và **khóa ngoại (FK)**, đảm bảo tính toàn vẹn và nhất quán dữ liệu.

### **2. Danh sách các bảng và mô tả chi tiết**

**2.1. User (Người dùng)**

* **Mô tả:** Lưu thông tin tài khoản người dùng, bao gồm khách hàng, nhân viên và admin.
* **Thuộc tính:**
  + id (PK): String
  + username: String
  + password: String
  + fullname: String
  + email: String
  + id\_permission (FK): String → liên kết bảng **Permission**
* **Quan hệ:**
  + 1–n với **Order** (một user có nhiều đơn hàng)
  + 1–n với **Comment**, **Favorite**

**2.2. Permission (Phân quyền)**

* **Mô tả:** Xác định quyền truy cập của người dùng (Admin, Nhân viên, Khách hàng).
* **Thuộc tính:**
  + id (PK): String
  + permission: String

**2.3. Product (Sản phẩm)**

* **Mô tả:** Lưu thông tin chi tiết của sản phẩm được bán trong cửa hàng.
* **Thuộc tính:**
  + id (PK): String
  + name\_product: String
  + price\_product: String
  + image: String
  + describe: String
  + gender: String
  + number: Number
  + id\_category (FK): String → liên kết bảng **Category**
* **Quan hệ:**
  + 1–n với **Sale**
  + 1–n với **Comment**
  + 1–n với **Favorite**
  + 1–n với **Detail\_Order**

**2.4. Category (Danh mục)**

* **Mô tả:** Phân loại nhóm sản phẩm (áo, quần, váy, phụ kiện...).
* **Thuộc tính:**
  + id (PK): String
  + category: String
* **Quan hệ:**
  + 1–n với **Product**

**2.5. Sale (Khuyến mãi)**

* **Mô tả:** Lưu thông tin các chương trình giảm giá, khuyến mãi theo sản phẩm.
* **Thuộc tính:**
  + id\_sale (PK): String
  + promotion: Number
  + describe: String
  + status: Boolean
  + start: Datetime
  + end: Datetime
  + id\_product (FK): String → **Product**

**2.6. Coupon (Mã giảm giá)**

* **Mô tả:** Mã giảm giá áp dụng cho toàn bộ đơn hàng.
* **Thuộc tính:**
  + id\_coupon (PK): String
  + code: String
  + count: Number
  + promotion: Number
  + describe: String

**2.7. Order (Đơn hàng)**

* **Mô tả:** Lưu thông tin về các đơn đặt hàng của khách hàng.
* **Thuộc tính:**
  + id (PK): String
  + address: String
  + total: Number
  + status: String
  + pay: Boolean
  + feeship: Number
  + create\_time: Datetime
  + id\_user (FK): String → **User**
  + id\_payment (FK): String → **Payment**
  + id\_note (FK): String → **Note**
  + id\_coupon (FK): String → **Coupon**
* **Quan hệ:**
  + 1–n với **Detail\_Order**

**2.8. Detail\_Order (Chi tiết đơn hàng)**

* **Mô tả:** Lưu thông tin chi tiết các sản phẩm trong mỗi đơn hàng.
* **Thuộc tính:**
  + id (PK): String
  + price\_product: Number
  + name\_product: String
  + count: Number
  + size: String
  + id\_order (FK): String → **Order**
  + id\_product (FK): String → **Product**

**2.9. Payment (Thanh toán)**

* **Mô tả:** Lưu phương thức thanh toán của khách hàng.
* **Thuộc tính:**
  + id (PK): String
  + pay\_name: String

**2.10. Favorite (Yêu thích)**

* **Mô tả:** Danh sách sản phẩm mà khách hàng đánh dấu yêu thích.
* **Thuộc tính:**
  + id (PK): String
  + id\_user (FK): String → **User**
  + id\_product (FK): String → **Product**

**2.11. Comment (Bình luận / Đánh giá)**

* **Mô tả:** Lưu nhận xét, đánh giá của người dùng cho từng sản phẩm.
* **Thuộc tính:**
  + id (PK): String
  + content: String
  + star: String (số sao 1–5)
  + id\_user (FK): String → **User**
  + id\_product (FK): String → **Product**

**2.12. Note (Ghi chú đơn hàng)**

* **Mô tả:** Lưu thông tin ghi chú thêm cho đơn hàng (ví dụ: người nhận, số điện thoại, yêu cầu giao hàng).
* **Thuộc tính:**
  + id (PK): String
  + fullname: String
  + phone: String

### **3. Quan hệ giữa các bảng**

* **User – Permission:** 1–n (một quyền có thể gán cho nhiều người dùng).
* **User – Order:** 1–n (một khách hàng có nhiều đơn hàng).
* **Product – Category:** n–1 (mỗi sản phẩm thuộc một danh mục).
* **Product – Sale:** 1–n (một sản phẩm có thể có nhiều chương trình khuyến mãi).
* **Order – Detail\_Order:** 1–n (một đơn hàng gồm nhiều chi tiết sản phẩm).
* **Product – Detail\_Order:** 1–n (một sản phẩm có thể nằm trong nhiều đơn hàng).
* **User – Favorite:** 1–n (một người dùng có thể thích nhiều sản phẩm).
* **Product – Comment:** 1–n (một sản phẩm có nhiều đánh giá).

### **4. Tóm tắt cấu trúc hệ thống dữ liệu**

| **Nhóm chức năng** | **Bảng liên quan** | **Vai trò chính** |
| --- | --- | --- |
| Quản lý người dùng | User, Permission | Xác thực, phân quyền, thông tin người dùng |
| Quản lý sản phẩm | Product, Category, Sale | Hiển thị, phân loại, khuyến mãi |
| Quản lý đơn hàng | Order, Detail\_Order, Payment, Note, Coupon | Lưu đơn hàng, thanh toán, ghi chú, giảm giá |
| Tương tác người dùng | Comment, Favorite | Đánh giá, yêu thích sản phẩm |

# **Test Plan**

## **1. Introduction**

### **1.1 Purpose**

Mục tiêu của tài liệu này là xác định **chiến lược kiểm thử (Test Strategy)** và **quy trình đảm bảo chất lượng (QA process)** cho ứng dụng **Web Bán Quần Áo** có tích hợp **CI/CD pipeline**.

Tài liệu này thiết lập kế hoạch kiểm thử hệ thống (System Test Plan) nhằm đảm bảo:

* Ứng dụng web thương mại điện tử hoạt động ổn định, đáp ứng các yêu cầu nghiệp vụ về quản lý sản phẩm, giỏ hàng, thanh toán, quản lý người dùng, và khuyến mãi.
* Các thay đổi và cập nhật (deployments) trong quy trình CI/CD được triển khai tự động nhưng vẫn đảm bảo **chất lượng phần mềm** và **không ảnh hưởng đến chức năng hiện có**.
* Tích hợp kiểm thử tự động (automated testing) vào pipeline CI/CD để phát hiện lỗi sớm, giảm thiểu rủi ro khi phát hành.
* Các chức năng chính như đăng nhập, đăng ký, tìm kiếm sản phẩm, đặt hàng, thanh toán và quản lý đơn hàng được kiểm thử toàn diện.
* Đảm bảo ứng dụng có thể mở rộng, bảo mật và duy trì hiệu suất khi số lượng người dùng và đơn hàng tăng.

Đối tượng đọc tài liệu này bao gồm: **đội phát triển (Dev team), QA/Test team, Business Analyst, Quản lý dự án (PM), và các bên liên quan khác**.

### **1.2 Background**

Ứng dụng **Website Bán Quần Áo** được xây dựng bằng **ReactJS (Frontend)** và **NodeJS (Backend – API, Socket)**, kết hợp **MongoDB Atlas** làm cơ sở dữ liệu chính. Hệ thống hỗ trợ mô hình **CI/CD pipeline**, cho phép tự động build, kiểm thử và triển khai nhằm đảm bảo chất lượng phần mềm và rút ngắn thời gian phát hành.

Ứng dụng được phát triển nhằm cung cấp một nền tảng thương mại điện tử hiện đại, nơi khách hàng có thể dễ dàng:

* Xem và tìm kiếm sản phẩm theo danh mục, giới tính, hoặc nhiều tiêu chí khác.
* Thêm sản phẩm vào giỏ hàng, đặt hàng trực tuyến.
* Thanh toán qua **PayPal** hoặc **ship COD**.
* Tra cứu trạng thái vận chuyển trên bản đồ thông qua **Google Map API**.
* Nhận tư vấn trực tiếp qua **live chat (Socket.IO)** và **email xác nhận đơn hàng**.

Các chức năng quản trị (Admin) bao gồm:

* CRUD sản phẩm, người dùng, đơn hàng, phân quyền.
* Quản lý bình luận, đánh giá (rating).
* Theo dõi số lượng hàng tồn kho, doanh thu và báo cáo.

Hệ thống được thiết kế theo mô hình **kết hợp API** với các công nghệ hỗ trợ:

* **NodeJS + Express**: xử lý backend API và WebSocket cho live chat.
* **MongoDB + Mongoose**: quản lý dữ liệu sản phẩm, người dùng, đơn hàng.
* **ReactJS**: giao diện người dùng responsive.
* **Nodemailer**: gửi email xác nhận đơn hàng.
* **Docker**: container hóa các thành phần, tích hợp CI/CD.
* **CI/CD Pipeline**: tự động hóa build, test, deploy.

### **1.3 Scope**

**Testing Phases**

|  |  |  |
| --- | --- | --- |
| **Giai đoạn kiểm thử** | **Đội chịu trách nhiệm** | **Mô tả** |
| Kiểm thử đơn vị  (Unit Testing) | Development Team | Kiểm thử từng **module nhỏ** (ví dụ: API đăng nhập, API thêm sản phẩm, component hiển thị giỏ hàng).  Mục tiêu: đảm bảo mỗi hàm, mỗi đoạn code chạy đúng logic, đầu vào/ra hợp lệ. |
| Kiểm thử tích hợp (Integration Testing) | QA team | Kiểm thử sự **tương tác giữa các module** (ví dụ: khi đặt hàng → giỏ hàng cập nhật → tạo đơn hàng → thanh toán PayPal → gửi email).  Mục tiêu: đảm bảo các thành phần kết nối đúng luồng dữ liệu. |
| Kiểm thử hệ thống (System Testing) | QA team | Kiểm thử **toàn bộ hệ thống** trên môi trường gần giống production (frontend, backend, database, CI/CD pipeline).  Bao gồm: kiểm thử chức năng, bảo mật, giao diện, hiệu năng.  Mục tiêu: đảm bảo hệ thống hoạt động đúng yêu cầu nghiệp vụ ban đầu. |
| Kiểm thử chấp nhận người dùng (UAT) | Người dùng đại diện hoặc khách hàng | Kiểm thử các kịch bản thực tế (ví dụ: khách hàng đăng ký, chọn quần áo, đặt hàng, thanh toán, nhận email xác nhận).  Mục tiêu: xác nhận rằng hệ thống sẵn sàng để triển khai chính thức. |

**Testing Types**

|  |  |  |
| --- | --- | --- |
| **Loại kiểm thử** | **Đội chịu trách nhiệm** | **Mục tiêu** |
| Kiểm thử chức năng (Functionality Testing) | QA team | Đảm bảo các tính năng nghiệp vụ chạy đúng: CRUD sản phẩm, giỏ hàng, thanh toán, quản lý đơn hàng.  Dựa trên yêu cầu đã ghi trong Use Case và ERD. |
| Kiểm thử cơ sở dữ liệu (Database Testing) | QA team + DBA | Kiểm thử tính toàn vẹn dữ liệu (data integrity), truy vấn, và sự đúng đắn khi thêm/xóa/sửa dữ liệu.  Ví dụ: khi hủy đơn hàng, số lượng tồn kho phải được cập nhật lại chính xác. |
| Kiểm thử bảo mật (Security Testing) | QA team | Đảm bảo người dùng không thể truy cập trái phép (ví dụ: bypass login, SQL injection, XSS).  Kiểm thử phân quyền: Admin, Nhân viên, Khách hàng chỉ được truy cập chức năng phù hợp. |
| Kiểm thử giao diện và khả năng sử dụng (GUI and Usability Testing) | QA team | Kiểm tra bố cục, hiển thị sản phẩm, thông báo lỗi rõ ràng, tính dễ dùng.  Ví dụ: hiển thị sản phẩm theo phân loại, nút “Thanh toán” rõ ràng, không lỗi font. |
| Kiểm thử hiệu năng & tải (Performance & Load Testing) | QA team | Đánh giá tốc độ phản hồi khi có nhiều người dùng cùng lúc (ví dụ: 500 user truy cập giỏ hàng).  Kiểm thử CI/CD pipeline: build và test có chạy ổn định khi load cao không. |
| Kiểm thử khói (Smoke Testing) | Dev team + QA team | Kiểm thử nhanh các chức năng chính để đảm bảo build mới không bị “gãy” trước khi QA đi sâu hơn.  Ví dụ: kiểm tra login, thêm sản phẩm, checkout có chạy không. |
| Kiểm thử hồi quy (Regression Testing) | QA team | Thực hiện sau mỗi lần cập nhật, đảm bảo các tính năng cũ không bị lỗi do code mới.  Ví dụ: thêm tính năng khuyến mãi nhưng không làm lỗi giỏ hàng. |
| Kiểm thử tương thích (Compatible Testing) | QA team | Đảm bảo website chạy ổn định trên nhiều **trình duyệt (Chrome, Firefox, Edge)** và thiết bị (desktop, mobile). |
| Kiểm thử tự do (Adhoc Testing) | QA team | Tester kiểm thử theo kinh nghiệm và trực giác, không theo test case cứng nhắc.  Ví dụ: nhập dữ liệu bất thường (số điện thoại = chữ) để kiểm tra phản ứng của hệ thống. |

### **1.4 Project Identification**

The table below identifies the documentation and availability used for developing the **Test Plan:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document (and version /date)** | **Created or Available** | **Received or Reviewed** | **Author or Resource** | **Notes** |
| Requirements Specification | **** Yes  No | **** Yes  No | Nhóm phát triển (Team Members) | Yêu cầu hệ thống từ README và Use Case. |
| Functional Specification | **** Yes  No | **** Yes  No | Nhóm phát triển | Mô tả chức năng: CRUD sản phẩm, giỏ hàng, thanh toán, live chat… |
| Use-Case Reports | **** Yes  No | **** Yes  No | Nhóm phát triển | Bao gồm Use Case cho Khách hàng, Nhân viên, Admin. |
| Project Plan | **** Yes  No | **** Yes  No | Quản lý dự án (PM) | Timeline phát triển + tích hợp CI/CD |
| Design Specifications | **** Yes  No | **** Yes  No | Nhóm phát triển | Thiết kế database (ERD, bảng Product, Order, User…). |
| Prototype |  Yes  No |  Yes  No |  |  |
| User’s Manuals |  Yes No |  Yes  No | README.md trong repo | Hướng dẫn setup MongoDB Atlas, chạy server/client, Docker. |
| Business Model or Flow | Yes  No | Yes  No | README + Use Case | Mô tả quy trình mua hàng, giỏ hàng, thanh toán. |
| Data Model or Flow | Yes  No |  Yes  No | Database schema (MongoDB) | Bảng Product, Category, User, Order, Comment, Favorite, Payment. |
| Business Functions and Rules |  Yes  No |  Yes  No |  |  |
| Project or Business Risk  Assessment |  Yes  No |  Yes  No |  |  |

### **1.5 References**

|  |  |  |
| --- | --- | --- |
| **No.** | **Document** | **Description** |
| 1 | README.md | Mô tả chung đề tài, các chức năng chính (CRUD sản phẩm, giỏ hàng, thanh toán PayPal, live chat, gửi email, vận chuyển qua Google Map API), hướng dẫn cài đặt và công nghệ sử dụng (ReactJS, NodeJS, MongoDB, Docker, CI/CD). |
| 2 | Database ERD & Thiết kế mô hình dữ liệu | Sơ đồ ERD và mô tả quan hệ các bảng (Product, Category, User, Order, Payment, Comment, Favorite). |
| 3 | Use Case Diagrams | Các biểu đồ Use Case cho Khách hàng, Nhân viên, Admin (tham khảo từ README) |

## **2. Feature to be tested**

The listing below identifies those items − use cases, functional requirements, and non-functional requirements − that have been

identified as targets for testing. This list represents what is to be tested.

### **2.1 Functionality**

**2.1.1 Quản lý sản phẩm (Product Management)**

* **CRUD sản phẩm**: Thêm, sửa, xóa, tìm kiếm sản phẩm.
* **Phân loại sản phẩm**: theo Category, Giới tính, Giá.
* **Phân trang và lọc sản phẩm** (paging, filtering, sorting).
* **Xem chi tiết sản phẩm**: tên, mô tả, hình ảnh, giá, số lượng tồn kho.

**2.1.2 Quản lý người dùng (User Management)**

* **Đăngký/Đăng nhập/Đăng xuất**.
* **Đặt lại mật khẩu (Reset password)**.
* **Phân quyền người dùng**: Khách hàng, Nhân viên bán hàng, Admin.
* **Quản lý hồ sơ cá nhân (Profile)**: cập nhật tên, email, mật khẩu.

**2.1.3 Giỏ hàng và đơn hàng (Cart & Order Management)**

* Thêm/Xóa sản phẩm trong giỏ hàng.
* Cập nhật số lượng sản phẩm trong giỏ hàng.
* Tạo đơn hàng (Order): nhập thông tin giao hàng, tính tổng tiền.
* Theo dõi trạng thái đơn hàng: Chưa xác nhận → Đã xác nhận → Đang giao → Hoàn thành/Hủy.
* Xem lịch sử mua hàng (Order history).

**2.1.4 Thanh toán (Payment)**

* Thanh toán PayPal (Sandbox).
* Thanh toán khi nhận hàng (COD).
* Xác nhận đơn hàng qua email (Nodemailer).
* Kiểm tra dữ liệu thanh toán được lưu chính xác vào MongoDB.

**2.1.5 Vận chuyển và giao hàng (Delivery)**

* **Tính phí vận chuyển** dựa trên khoảng cách bằng **Google Map API**.
* **Hiển thị tuyến đường từ cửa hàng → khách hàng**.
* **Quản lý phương thức giao hàng** (Delivery info trong Order).

**2.1.6 Live Chat & Hỗ trợ khách hàng**

* **Tính phí vận chuyển** dựa trên khoảng cách bằng **Google Map API**.
* **Hiển thị tuyến đường từ cửa hàng → khách hàng**.
* **Quản lý phương thức giao hàng** (Delivery info trong Order).

**2.1.7 Bình luận & đánh giá sản phẩm (Comment & Rating)**

* **Khách hàng thêm bình luận (Comment)**.
* **Đánh giá sản phẩm bằng sao (1–5)**.
* **Admin kiểm duyệt/xóa bình luận** nếu vi phạm.

**2.1.8 Quản trị hệ thống (Admin Management)**

* **Quản lý sản phẩm**: CRUD toàn bộ sản phẩm.
* **Quản lý người dùng**: tạo tài khoản nhân viên, phân quyền, khóa/mở khóa user.
* **Quản lý đơn hàng**: xác nhận, thay đổi trạng thái, hủy đơn hàng.
* **Quản lý bình luận, yêu thích (Favorite)**.
* **Xem báo cáo/thống kê** (doanh thu, số lượng bán ra, sản phẩm còn hàng/hết hàng).

### **2.2 Usability**

Các kiểm thử về khả năng sử dụng (Usability) sẽ tập trung vào trải nghiệm người dùng của website bán quần áo. Mục tiêu là đảm bảo giao diện trực quan, dễ sử dụng, hiển thị đúng nội dung, và tương thích trên nhiều thiết bị.

2.2.1 Responsive Design Support

* Xác minh website hiển thị đúng trên **desktop, tablet, mobile**.
* Đảm bảo bố cục (layout), hình ảnh sản phẩm, nút chức năng (mua hàng, giỏ hàng, thanh toán) không bị vỡ layout.
* Kiểm tra hiển thị ở nhiều độ phân giải màn hình (FullHD, 2K, Mobile 360px).

**2.2.2 Multi-session / Login Support**

* Đảm bảo người dùng có thể **đăng nhập, đăng xuất nhiều lần** mà không mất dữ liệu giỏ hàng/đơn hàng.
* Kiểm tra việc duy trì **session** sau khi reload trang hoặc quay lại từ tab khác.
* Kiểm tra session timeout (người dùng không hoạt động trong 30 phút).

**2.2.3 Font & Language Display**

* Kiểm tra **font chữ** hiển thị rõ ràng, dễ đọc.
* Đảm bảo không bị lỗi font khi hiển thị **tiếng Việt có dấu** (Unicode).
* Kiểm tra hiển thị ký tự đặc biệt trong tên sản phẩm, bình luận, địa chỉ giao hàng.

**2.2.4 Navigation & User Flow**

* Đảm bảo người dùng có thể điều hướng dễ dàng giữa **Trang chủ → Danh mục → Sản phẩm chi tiết → Giỏ hàng → Thanh toán**.
* Kiểm tra link, button, menu, breadcrumbs hoạt động đúng.
* Đảm bảo luồng người dùng không bị “kẹt” (dead end).

**2.2.5 Accessibility & Error Messages**

* Đảm bảo form nhập liệu (đăng ký, đăng nhập, thanh toán) có thông báo lỗi rõ ràng khi nhập sai định dạng (ví dụ: email không hợp lệ, số điện thoại thiếu).
* Kiểm tra màu sắc, độ tương phản (contrast ratio) đảm bảo khả năng đọc.
* Các thông báo thành công/thất bại (toast, popup) phải hiển thị đúng và dễ hiểu.

**2.2.6 Browser Compatibility**

* Kiểm thử hiển thị và usability trên các trình duyệt chính: **Chrome, Firefox, Edge**.
* Kiểm tra một số trình duyệt mobile (Chrome Android, Safari iOS).

### **2.3 Design Constraints**

**2.3.1 Maximum message length**

Xác minh các trường nhập liệu có giới hạn hợp lý:

* **Tên người dùng**: tối đa 50 ký tự.
* **Mật khẩu**: tối thiểu 8 ký tự, tối đa 64 ký tự, bao gồm chữ hoa, chữ thường, số, ký tự đặc biệt.
* **Tên sản phẩm**: tối đa 100 ký tự.
* **Mô tả sản phẩm**: tối đa 500 ký tự.
* **Địa chỉ giao hàng**: tối đa 200 ký tự.

Đảm bảo hệ thống hiển thị thông báo lỗi rõ ràng nếu người dùng vượt quá giới hạn.

**2.3.2 Test environment**

Kiểm thử sẽ được tiến hành trong các môi trường sau:

* **Frontend**: ReactJS chạy trên NodeJS, truy cập qua trình duyệt Chrome, Firefox, Edge.
* **Backend**: NodeJS + Express API, chạy trên Docker container.
* **Database**: MongoDB Atlas (cluster cloud).
* **CI/CD Pipeline**: GitHub Actions/GitLab CI/Jenkins (tùy thiết lập nhóm).
* **Payment Sandbox**: PayPal sandbox để kiểm thử thanh toán.
* **Email Service**: SMTP (Nodemailer, Gmail/SendGrid).
* **Map API**: Google Map API để hiển thị tuyến đường và tính phí giao hàng.

**2.3.3 Characters Sets & Localization**

* Hệ thống phải hỗ trợ **Unicode UTF-8** để hiển thị chính xác **tiếng Việt có dấu** và ký tự đặc biệt.
* Tất cả dữ liệu (tên sản phẩm, email, địa chỉ, bình luận) phải lưu trữ và truy xuất được trong MongoDB mà không mất ký tự.
* Website hiện chỉ hỗ trợ **tiếng Việt** và **tiền tệ VND/USD qua PayPal**.
* Các ngôn ngữ và tiền tệ khác chưa nằm trong phạm vi kiểm thử (Not in Scope).

### **2.4 Interfaces**

**2.4.1 User Interfaces**

Các kiểm thử giao diện người dùng (UI) sẽ tập trung vào việc đảm bảo website hiển thị đúng và dễ sử dụng trên cả **giao diện khách hàng (User)** và **giao diện quản trị (Admin)**.

Giao diện Khách hàng

* **Trang chủ**: hiển thị sản phẩm nổi bật, danh mục sản phẩm.
* **Trang danh mục (Category Page)**: lọc, phân loại, phân trang sản phẩm.
* **Trang chi tiết sản phẩm**: hình ảnh, tên, mô tả, giá, tồn kho, nút “Thêm vào giỏ hàng”.
* **Giỏ hàng (Cart Page)**: danh sách sản phẩm đã chọn, số lượng, tổng tiền, nút “Thanh toán”.
* **Thanh toán (Checkout Page)**: nhập thông tin người nhận, chọn phương thức thanh toán (PayPal, COD).
* **Theo dõi đơn hàng (Order Tracking)**: hiển thị trạng thái đơn hàng và phí vận chuyển trên Google Map.
* **Live Chat**: khung chat với nhân viên tư vấn (Socket.IO).
* **Đăng ký/Đăng nhập/Quên mật khẩu**: form nhập liệu với xác thực dữ liệu đầu vào (validation).

Giao diện Quản trị (Admin Panel)

* **Quản lý sản phẩm**: CRUD sản phẩm (tên, giá, hình ảnh, mô tả, tồn kho).
* **Quản lý người dùng**: phân quyền (Khách hàng, Nhân viên, Admin).
* **Quản lý đơn hàng**: xác nhận, thay đổi trạng thái, hủy đơn hàng
* **Quản lý bình luận/đánh giá**: xem, xóa, xử lý các comment không hợp lệ.
* **Thống kê/Báo cáo**: doanh thu, sản phẩm bán chạy, hàng tồn kho.

**2.4.2 System Interfaces**

Các giao diện tích hợp giữa hệ thống và dịch vụ bên ngoài:

* **Payment Gateway (PayPal Sandbox)**
  + Kiểm thử tích hợp API PayPal ho thanh toán trực tuyến.
  + Đảm bảo dữ liệu giao dịch lưu vào hệ thống sau khi thanh toán thành công.
* **Email Service (Nodemailer, Gmail/SendGrid)**
  + Gửi email xác nhận đơn hàng cho khách hàng.
  + Kiểm thử nội dung email, độ trễ và độ tin cậy.
* **Google Map API**
  + Hiển thị tuyến đường vận chuyển từ cửa hàng đến khách hàng.
  + Tính toán phí vận chuyển dựa trên khoảng cách.
* **CI/CD Pipeline Integration**
  + Tích hợp GitHub Actions/GitLab CI/Jenkins.
  + Tự động chạy unit test, regression test trước khi deploy.

## **3. Feature not to be tested**

### **3.1 Setup a user account**

* + Kiểm thử khôi phục sau sự cố
  + Các tính năng nâng cấp tương lai như AI Recommendation, Chatbot
  + Hệ thống chỉ hỗ trợ chức năng **đăng ký tài khoản** theo form chuẩn (username, email, password).
  + Các chức năng nâng cao như thiết lập nhiều tài khoản, lưu “remember password” hay tích hợp tài khoản mạng xã hội (Google, Facebook) **không nằm trong phạm vi kiểm thử**.

## **3.2 Remove user account**

* + - Chức năng **xóa tài khoản vĩnh viễn** (account deletion) hiện **chưa được phát triển** trong phiên bản này.
    - Do đó, kiểm thử sẽ **không bao gồm** việc xác minh quy trình xóa tài khoản, xử lý dữ liệu liên quan hay phục hồi dữ liệu sau khi xóa.

## **4. Test Strategy**

### **4.1 Testing Types**

**4.1.1 Function and Usability Testing**

|  |  |
| --- | --- |
| Test Objective: | Để đảm bảo chức năng mục tiêu thử nghiệm phù hợp, bao gồm điều hướng, nhập dữ liệu, xử lý và truy xuất |
| Technique: | Thực thi từng **use case**, luồng xử lý hoặc chức năng, với cả dữ liệu hợp lệ và dữ liệu không hợp lệ, để xác minh:   * Kết quả mong đợi xảy ra khi sử dụng dữ liệu hợp lệ (ví dụ: thêm sản phẩm vào giỏ hàng thành công). * Thông báo lỗi hoặc cảnh báo phù hợp hiển thị khi sử dụng dữ liệu không hợp lệ (ví dụ: nhập sai định dạng email khi đăng ký). |
| Completion Criteria: | • Tất cả các thử nghiệm theo kế hoạch đã được thực hiện.  • Tất cả các lỗi được xác định đã được khắc phục. |
| Special Considerations: | Xác định hoặc mô tả những vấn đề (nội bộ hoặc bên ngoài) có thể ảnh hưởng đến việc triển khai và thực hiện kiểm thử chức năng, ví dụ: môi trường kiểm thử chưa ổn định, dữ liệu mẫu chưa đầy đủ, hoặc tích hợp với hệ thống thanh toán bên thứ ba (Paypal) và API bản đồ (Google Maps). |

**4.1.2 User Interface Testing**

Kiểm thử khả dụng sẽ được thực hiện bằng cách xác minh rằng thông tin hiển thị trên từng màn hình (trang chủ, danh mục sản phẩm, chi tiết sản phẩm, giỏ hàng, thanh toán, hồ sơ người dùng) là chính xác và rõ ràng. Các menu, nút bấm, biểu tượng và liên kết điều hướng sẽ được kiểm tra để đảm bảo trải nghiệm duyệt và mua hàng diễn ra thuận lợi.

Tầm quan trọng đặc biệt được đặt vào:

* Hình ảnh sản phẩm (chất lượng, phóng to, carousel hiển thị nhiều ảnh).
* Nội dung mô tả, giá và khuyến mãi được trình bày chính xác.
* Trình bày danh mục, bộ lọc, sắp xếp sản phẩm (theo giới tính, kích thước, giá).
* Quy trình giỏ hàng và thanh toán rõ ràng, dễ sử dụng.
* Thông báo phản hồi (ví dụ: “Thêm sản phẩm vào giỏ thành công”, “Đặt hàng thành công”, “Thanh toán thất bại”).
* Tính nhất quán của bố cục, font chữ, màu sắc, nút và các thành phần giao diện.
* Các khía cạnh về khả năng truy cập như độ tương phản màu, độ dễ đọc của chữ, và gợi ý kiểm tra lỗi trong form (ví dụ: đăng nhập, đăng ký, thanh toán).

Kiểm thử giao diện và tính khả dụng sẽ được tiến hành song song với kiểm thử chức năng trong mọi bản build, bắt đầu từ bản build đầu tiên (chu kỳ kiểm thử 2 tuần) cho đến bản build cuối cùng trước khi phát hành.

### **4.2 Tools**

Các tool được tích hợp trong dự án:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Tool** | **Vendor/In-house** | **Version** |
| Bug Tracking | Mantis |  | Lastest |
| Project Management | Github |  | Lastest |

## **5. Resources**

### **5.1 Roles**

Các vai trò được sử dụng trong dự án:

|  |  |  |
| --- | --- | --- |
| **Human Resources** | | |
| Employee | Minimum Resources Recommended  (number of full-time roles allocated) | Specific Responsibilities/Job Description |
| Test Developer | 1 | * Xác định, ưu tiên và triển khai các test case. * Xây dựng kế hoạch kiểm thử (Test Plan). * Đánh giá hiệu quả của hoạt động kiểm thử. * Hỗ trợ nhóm phát triển trong việc chuẩn bị dữ liệu kiểm thử. |
| Tester | 2 | * Thực hiện kiểm thử theo các test case. * Viết và cập nhật đặc tả test case. * Thực thi test case và ghi nhận kết quả. * Ghi log lỗi vào hệ thống (GitHub Issues). * Theo dõi việc sửa lỗi và xác nhận lại (re-test). * Đề xuất chỉnh sửa/tài liệu nếu phát hiện thiếu sót. |

## **6. Project Milestones**

Hoạt động và mốc kiểm thử phụ thuộc vào tiến trình phát triển. Pha xây dựng được chia thành 3 build chính, sau đó là bản release candidate và bản release chính thức. Mỗi build bao gồm chu kỳ kiểm thử đầy đủ: lập kế hoạch, thiết kế, thực thi và đánh giá.

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone Task** | **Effort** | **Start Date** | **End Date** |
| **Build 1** – Plan Test, Design Test, Execute Test, Evaluate Test | 5 ngày | 01/10/2025 | 05/10/2025 |
| **Build 2** – Plan Test, Design Test, Execute Test, Evaluate Test | 5 ngày | 06/10/2025 | 10/10/2025 |
| **Build 3** – Plan Test, Design Test, Execute Test, Evaluate Test | 5 ngày | 11/10/2025 | 15/10/2025 |
| **Release Candidate** – Full System Test, Regression Test, UAT | 4 ngày | 16/10/2025 | 19/10/2025 |
| **Release Version –** Final Acceptance, Test Summary Report | 3 ngày | 10/10/2025 | 22/10/2025 |

## **7. Deliverables**

### **7.1 Test Model**

Mô hình kiểm thử bao gồm toàn bộ **test case, test scenario, quy trình kiểm thử và test script** được xây dựng cho từng chức năng của hệ thống. Các tài liệu này sẽ đảm bảo rằng mọi yêu cầu (requirement) đều được ánh xạ (traceability) với các trường hợp kiểm thử tương ứng.

### **7.2 Test Logs**

Kết quả thực thi test case sẽ được ghi lại trong **Test Execution Log**. Nhật ký kiểm thử (test log) sẽ lưu trữ các thông tin: ngày kiểm thử, người thực hiện, môi trường kiểm thử, dữ liệu sử dụng và trạng thái (Pass/Fail). Tài liệu này sẽ được trình bày dưới dạng **Word/Excel** để tiện theo dõi và báo cáo tiến độ.

### **7.3 Bug Reports**

Các lỗi (defects/bugs) được phát hiện trong quá trình kiểm thử sẽ được ghi nhận và theo dõi thông qua hệ thống **Jira hoặc Mantis**. Báo cáo lỗi sẽ bao gồm các thông tin: mô tả lỗi, bước tái hiện (steps to reproduce), môi trường xảy ra, mức độ nghiêm trọng (severity), mức độ ưu tiên (priority), trạng thái xử lý (open, in progress, fixed, closed).