**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------



**BÁO CÁO MÔN THAY THẾ**

TP. HCM

20…

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*:**

**“Xây dựng website kinh doanh thuỷ sinh vật cảnh”**

**Giảng viên hướng dẫn : ThS. HUỲNH TRUNG TRỤ**

**Sinh viên thực hiện : NGUYỀN QUỐC NHẬT – N18DCCN142**

**PHẠM ĐỨC PHÚ PHÚC – N18DCCN154**

**Lớp : D18CQCP02-N**

**Khoá : 2018 – 2023**

**Hệ : ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 11/2023**

**LỜI CẢM ƠN**

Trong suốt thời gian hơn 4 năm học tập và rèn luyện tại Học viện Công nghệ Bưu chính Viễn thông cơ sở tại thành phố Hồ Chí Minh cho đến nay, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của quý thầy cô và bạn bè. Với lòng biết ơn sâu sắc và chân thành nhất, nhóm em xin gửi đến quý thầy cô ở khoa Công nghệ thông tin 2 cùng với tri thức và tâm huyết của mình để truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường.

Chúng em cũng xin bày tỏ lòng biết ơn đến ban lãnh đạo của Học viện Công nghệ Bưu chính Viễn thông cơ sở tại thành phố Hồ Chí Minh và các khoa phòng ban chức năng đã trực tiếp và gián tiếp giúp đỡ em trong suốt quá trình học tập và nghiên cứu đề tài này.

Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế của sinh viên, bài báo cáo này không thể tránh được những thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các quý thầy cô để em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công tác thực tế sau này.

Chúng em xin chân thành cảm ơn!

TP. HCM, tháng 11 năm 2023

**MỤC LỤC**

[**CHƯƠNG I:** **GIỚI THIỆU ĐỀ TÀI** 1](#_Toc150200937)

[**CHƯƠNG II:** **CƠ SỞ LÝ THUYẾT** 2](#_Toc150200938)

[**1.** **Ngôn ngữ lập trình** 2](#_Toc150200939)

[**1.1** **Ngôn ngữ Java** 2](#_Toc150200940)

[**1.2** **Ngôn ngữ TypeScript** 3](#_Toc150200941)

[**2.** **Hệ quản trị cơ sở dữ liệu PostgreSQL** 3](#_Toc150200942)

[**3.** **Công cụ lập trình** 4](#_Toc150200943)

[**3.1** **IntelliJ IDEA** 4](#_Toc150200944)

[**3.2** **Microsoft Visual Studio Code** 5](#_Toc150200945)

[**3.3** **Postman** 6](#_Toc150200946)

[**3.4** **DBeaver** 7](#_Toc150200947)

[**4.** **Framework** 8](#_Toc150200948)

[**4.1** **Spring Boot** 8](#_Toc150200949)

[**4.2** **VueJS** 9](#_Toc150200950)

[**4.3** **NextJS** 10](#_Toc150200951)

[**5.** **Tổng quan về RESTful API** 11](#_Toc150200952)

[**CHƯƠNG III:** **KHẢO SÁT HỆ THỐNG** 12](#_Toc150200953)

[**1.** **Khảo sát thực tế** 12](#_Toc150200954)

[**1.1** **Nghiệp vụ phía khách hàng** 12](#_Toc150200955)

[**1.2** **Nghiệp vụ phía nhân viên** 13](#_Toc150200956)

[**2.** **Các yêu cầu của hệ thống** 14](#_Toc150200957)

[**2.1** **Yêu cầu chức năng nghiệp vụ** 14](#_Toc150200958)

[**2.2** **Yêu cầu phi chức năng của hệ thống** 17](#_Toc150200959)

[**2.3** **Yêu cầu về chất lượng** 17](#_Toc150200960)

[**CHƯƠNG IV:** **PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG** 18](#_Toc150200961)

[**1.** **Sơ đồ tổng quan hệ thống** 18](#_Toc150200962)

[**2.** **Mô hình chức năng nghiệp vụ (BFD – Business Function Diagram)** 18](#_Toc150200963)

[**3.** **Mô hình cộng tác (Collaboration Diagram)** 20](#_Toc150200964)

[**3.1** **Mô hình cộng tác cho bộ phận quản lý** 20](#_Toc150200965)

[**3.2** **Mô hình công tác cho bộ phận nhân viên** 21](#_Toc150200966)

[**3.3** **Mô hình cộng tác cho bộ phận khách hàng** 22](#_Toc150200967)

[**4.** **Thiết lập mô hình thực thể ERD (ERD – Entity Relationship Diagram)** 24](#_Toc150200968)

[**4.1** **Xác định các thực thể** 24](#_Toc150200969)

[**4.2** **Mô hình thực thể kết hợp ERD** 24](#_Toc150200970)

[**4.3** **Mô hình dữ liệu quan hệ dạng chuẩn 3** 26](#_Toc150200971)

[**5.** **Thiết kế dữ liệu** 26](#_Toc150200972)

[**5.1** **Mô hình Diagram** 26](#_Toc150200973)

[**5.2** **Từ điển dữ liệu** 27](#_Toc150200974)

[**CHƯƠNG V:** **XÂY DỰNG ỨNG DỤNG** 33](#_Toc150200975)

[**1.** **Xây dựng Web Service** 33](#_Toc150200976)

[**1.1** **Công nghệ sử dụng** 33](#_Toc150200977)

[**1.2** **Sơ đồ chung cho thiết kế xử lý** 33](#_Toc150200978)

[**1.3** **Cấu hình application.properties** 35](#_Toc150200979)

[**2.** **Xây dựng giao diện** 36](#_Toc150200980)

[**2.1** **Thiết giao diện quản lý** 36](#_Toc150200981)

[**2.2** **Thiết kế giao diện khách hàng** 36](#_Toc150200982)

[**CHƯƠNG VI:** **KẾT LUẬN** 37](#_Toc150200983)

[**1.** **Đánh giá** 37](#_Toc150200984)

[**1.1** **Kết quả đạt được** 37](#_Toc150200985)

[**1.2** **Ưu điểm và khuyết điểm** 37](#_Toc150200986)

[**2.** **Hướng phát triển** 37](#_Toc150200987)

[**MỤC TÀI LIỆU THAM KHẢO** 38](#_Toc150200988)

**DANH MỤC CÁC BẢNG**

[Bảng 3.1 Yêu cầu chức năng nghiệp vụ cho bộ phận quản lý 16](#_Toc150202754)

[Bảng 3.2 Yêu cẩu chức năng nghiệp vụ cho bộ phận khách hàng 17](#_Toc150202755)

[Bảng 3.3 Yêu cầu phi chức năng của hệ thống 17](#_Toc150202756)

[Bảng 3.4 Yêu cầu về chất lượng 17](#_Toc150202757)

[Bảng 4.1 Users 28](#_Toc150202761)

[Bảng 4.2 Role 28](#_Toc150202762)

[Bảng 4.3 Product 28](#_Toc150202763)

[Bảng 4.4 Category 29](#_Toc150202764)

[Bảng 4.5 Product Image 29](#_Toc150202765)

[Bảng 4.6 Order Supplier 29](#_Toc150202766)

[Bảng 4.7 Import Form 29](#_Toc150202767)

[Bảng 4.8 Order Customer 30](#_Toc150202768)

[Bảng 4.9 Order Status 30](#_Toc150202769)

[Bảng 4.10 Cart Detail 31](#_Toc150202770)

[Bảng 4.11 User Role 31](#_Toc150202771)

[Bảng 4.12 Import Detail 31](#_Toc150202772)

[Bảng 4.13 Order Supplier Detail 31](#_Toc150202773)

[Bảng 4.14 Price Detail 32](#_Toc150202774)

[Bảng 4.15 Deduction 32](#_Toc150202775)

[Bảng 5.1 Mô tả package 35](#_Toc150202776)

**DANH MỤC HÌNH VÀ SƠ ĐỒ**

**DANH MỤC CÁC CỤM TỪ VIẾT TẮT**

* API: Application Programming Interface
* App: Application
* DFD: Data Flow Diagram
* ERD: Entity Relationship Diagram
* HTTP: HyperText Transfer Protocol
* IDE: Integrated development environment
* DB: Database
* REST: Representational State Transfer
* UI: User Interface

1. **GIỚI THIỆU ĐỀ TÀI**

Ngày nay, ứng dụng công nghệ thông tin và việc tin học hóa được xem là một trong những yếu tố mang tính quyết định trong hoạt động của các chính phủ, tổ chức, cũng như của các công ty, nó đóng vai trò hết sức quan trọng, có thể tạo ra những bước đột phá mạnh mẽ. Cùng với sự phát triển không ngừng về kỹ thuật máy tính và mạng điện tử, công nghệ thông tin cũng được những công nghệ có đẳng cấp cao và lần lượt chinh phục hết đỉnh cao này đến đỉnh cao khác.

Mạng Internet là một trong những sản phẩm có giá trị hết sức lớn lao và ngày càng trở nên một công cụ không thể thiếu, là nền tảng chính cho sự truyền tải, trao đổi thông tin trên toàn cầu. Giờ đây, mọi việc liên quan đến thông tin trở nên thật dễ dàng cho người sử dụng: chỉ cần có một máy tính kết nối Internet và một dòng dữ liệu truy tìm thì gần như lập tức... cả thế giới về vấn đề mà bạn đang quan tâm sẽ hiện ra, có đầy đủ thông tin, hình ảnh và thậm chí đôi lúc có cả những âm thanh nếu bạn cần...

Bằng Internet, chúng ta đã thực hiện được nhiều công việc với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Chính điều này, đã thúc đẩy sự khai sinh và phát triển của thương mại điện tử và chính phủ điện tử trên khắp thế giới, làm biến đổi đáng kể bộ mặt văn hóa, nâng cao chất lượng cuộc sống con người. Trong hoạt động sản xuất, kinh doanh, giờ đây, thương mại điện tử đã khẳng định được vai trò xúc tiến và thúc đẩy sự phát triển của doanh nghiệp. Đối với một cửa hàng hay shop, việc quảng bá và giới thiệu đến khách hàng các sản phẩm mới đáp ứng được nhu cầu của khác hàng sẽ là cần thiết. Vậy phải quảng bá thế nào? Đó là xây dựng được một Website cho cửa hàng của mình quảng bá tất cả các sản phẩm của mình bán.

Vì vậy, nhóm em đã thực hiện đồ án “ XÂY DỰNG WEBSITE KINH DOANH THUỶ SINH VẬT CẢNH” dưới sự hướng dẫn của thầy Huỳnh Trung Trụ. Người chủ cửa hàng sẽ đưa các sản phẩm mà cửa hang bán lên website và quản lý bằng website đó, khách hàng có thể đặt và mua hàng trên website mà không cần đến cửa hà

1. **CƠ SỞ LÝ THUYẾT**

Đề tài này được thực hiện trên ngôn ngữ lập trình Java và ngôn ngữ TypeScript; Cơ sở dữ liệu được xây dựng trên hệ quản trị cơ sở dữ liệu PostgreSQL; Công cụ lập trình là IntelliJ IDEA, Microsoft Visual Studio Code, Postman, DBeaver; Framework hỗ trợ là VueJS và NextJS.

1. **Ngôn ngữ lập trình**
2. **Ngôn ngữ Java**

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak.

Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Java được tạo ra với tiêu chí “Viết (code) một lần, thực thi khắp nơi” (Write Once, Run Anywhere – WORA). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó.



**Những đặc điểm của ngôn ngữ lập trình Java:**

- Đơn giản và quen thuộc: Vì Java kế thừa trực tiếp từ C/C++ nên nó có những đặc điểm của ngôn ngữ này, Java đơn giản vì mặc dù dựa trên cơ sở C++ nhưng Sun đã cẩn thận lược bỏ các tính năng khó nhất của của C++ để làm cho ngôn ngữ này dễ sử dụng hơn.

- Hướng đối tượng và quen thuộc.

- Mạnh mẽ (thể hiện ở cơ chế tự động thu gom rác - Garbage Collection) và an toàn.

- Kiến trúc trung lập, độc lập nền tảng và có tính khả chuyển (Portability).

- Hiệu suất cao.

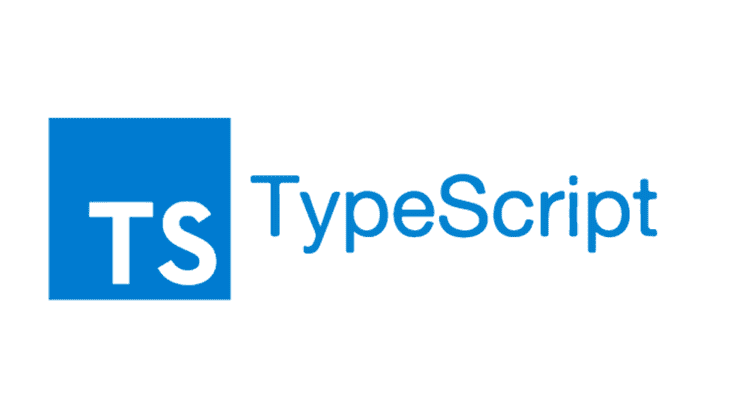
- Máy ảo (biên dịch và thông dịch).

- Phân tán.

- Đa nhiệm: Ngôn ngữ Java cho phép xâ dựng trình ứng dụng, trong đó nhiều quá trình có thể xảy ra đồng thời. Tính đa nhiệm cho phép các nhà lập trình có thể biên soạn phần mềm đáp ứng tốt hơn, tương tác tốt hơn và thực hiện theo thời gian thực.

1. **Ngôn ngữ TypeScript**

TypeScript là một ngôn ngữ lập trình phát triển bởi Microsoft. Nó là một phần mở rộng của JavaScript, được thiết kế để giúp lập trình viên phát triển ứng dụng web và ứng dụng phía máy chủ JavaScript lớn và phức tạp hơn một cách dễ dàng và an toàn hơn.



**Các đặc điểm chính của TypeScript bao gồm:**

* Kiểu dữ liệu tĩnh: TypeScript hỗ trợ kiểu dữ liệu tĩnh, cho phép xác định kiểu dữ liệu của biến, tham số và giá trị trả về từ hàm. Điều này giúp phát hiện lỗi trước khi chương trình chạy, cung cấp tính tin cậy hơn trong mã nguồn.
* Hỗ trợ ES6 và ESNext: TypeScript hỗ trợ các tính năng và cú pháp của các phiên bản mới của ECMAScript, bao gồm cả ES6 và các phiên bản sau này, giúp viết mã hiện đại và sạch hơn.
* Tích hợp tốt với JavaScript: Có thể sử dụng mã JavaScript hiện có trong dự án TypeScript mà không cần chuyển đổi hoặc viết lại từ đầu.
* Hệ thống mô-đun: TypeScript có một hệ thống mô-đun mạnh mẽ cho phép quản lý và tổ chức mã nguồn dễ dàng hơn.
* Hỗ trợ khám phá và tự động hoàn thành: TypeScript cung cấp tính năng khám phá mã nguồn và tự động hoàn thành mã, giúp viết mã nhanh chóng và hiệu quả hơn.
* Cộng đồng mạnh mẽ: TypeScript có một cộng đồng lớn và đang phát triển mạnh mẽ, với nhiều thư viện và framework hỗ trợ.
* TypeScript giúp giảm thiểu lỗi logic, tăng tính bảo mật, và cải thiện khả năng bảo trì trong các dự án phức tạp, đặc biệt là trong các ứng dụng web và ứng dụng phía máy chủ lớn.

1. **Hệ quản trị cơ sở dữ liệu PostgreSQL**

PostgreSQL là một hệ quản trị cơ sở dữ liệu (Database Management System - DBMS) mã nguồn mở và mạnh mẽ. Nó là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System - RDBMS) được phát triển bởi cộng đồng và được chính thức hỗ trợ bởi PostgreSQL Global Development Group.



**PostgreSQL có nhiều tính năng mạnh mẽ, bao gồm:**

* Tích hợp ACID: PostgreSQL tuân thủ nguyên tắc ACID (Atomicity, Consistency, Isolation, Durability), đảm bảo tính toàn vẹn của dữ liệu trong quá trình ghi và đọc dữ liệu.
* Hỗ trợ nhiều loại dữ liệu: Nó hỗ trợ nhiều loại dữ liệu bao gồm số nguyên, số thập phân, văn bản, hình ảnh, âm thanh và nhiều loại dữ liệu tùy chỉnh.
* Ngôn ngữ truy vấn SQL: PostgreSQL sử dụng ngôn ngữ truy vấn SQL mạnh mẽ cho việc tạo, truy vấn và quản lý dữ liệu.
* Hệ thống mô-đun và phần mềm mở rộng: Có thể mở rộng khả năng của PostgreSQL bằng cách sử dụng các phần mềm mở rộng (extensions) và hệ thống mô-đun (plugins).
* Hỗ trợ khóa ngoại và ràng buộc: PostgreSQL hỗ trợ ràng buộc tham chiếu và khóa ngoại, giúp đảm bảo tính nhất quán của dữ liệu.
* Hỗ trợ trình tự và xử lý giong: PostgreSQL hỗ trợ trình tự (triggers) và xử lý giong (stored procedures) cho việc tự động hóa các hoạt động trên cơ sở dữ liệu.
* Hỗ trợ giao diện đồ họa và dòng lệnh: PostgreSQL có các giao diện đồ họa và dòng lệnh để quản lý cơ sở dữ liệu.
* Mã nguồn mở và cộng đồng mạnh mẽ: Sự phát triển và hỗ trợ cho PostgreSQL đến từ một cộng đồng rộng lớn và nhiều công ty và tổ chức lớn.

1. **Công cụ lập trình**
2. **IntelliJ IDEA**

Phần mềm IntelliJ IDEA, sản phẩm nổi tiếng của JetBrains đã nhận được rất nhiều giải thưởng. Phần mềm được thiết kế để cải tiến năng suất cho các nhà phát triển. IntelliJ IDEA cung cấp trình soạn thảo thông minh, trình phân tích mã và tập hợp mạnh mẽ của refactorings hỗ trợ một loạt các ngôn ngữ lập trình, các khuôn khổ và công nghệ, và đã sẵn sàng để sử dụng.

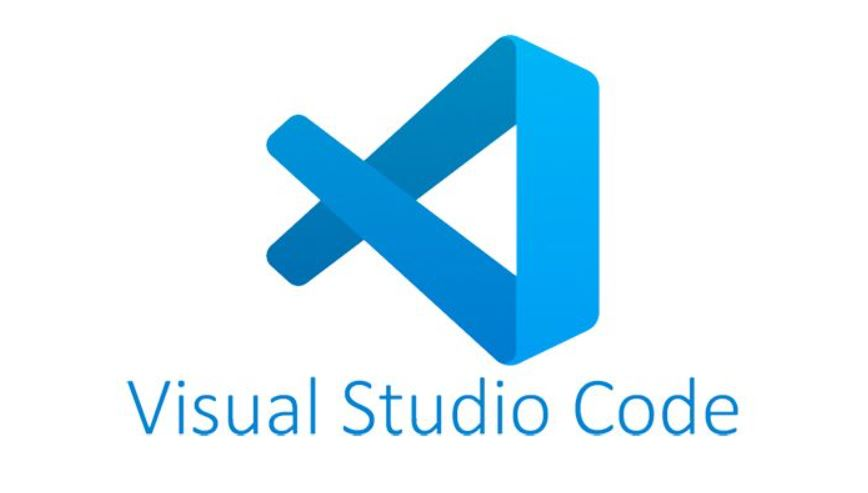


**Lợi ích chính của phần mềm IntelliJ IDEA:**

* Cho phép các nhà phát triển tập trung phát triển và quản lý tất cả các tác vụ thông thường
* Cho phép viết, gỡ lỗi, tái cấu trúc, kiểm tra và tìm hiểu mã của bạn mà không bị ảnh hưởng.
* Xử lý liền mạch cơ sở mã hỗn hợp của Java, Ruby, Groovy, Python và Scala.
* Tự động duy trì chất lượng mã.
* Theo dõi và sửa lỗi trên tất cả các cấp độ – từ các câu đến kiến trúc tổng thể.
* Tạo mã “sạch”, nhanh chóng thực hiện mã trong thời gian ngắn nhất.
* Được thiết kế để làm việc trên tất cả các quy mô – từ cá nhân đến doanh nghiệp.
* Hỗ trợ tất cả các ngôn ngữ, công nghệ và framework chính.
* Làm việc với các hệ thống điều khiển phiên bản phổ biến và TeamCity, server tích hợp liên tục

1. **Microsoft Visual Studio Code**

Microsoft Visual Studio Code (VS Code) là một môi trường phát triển tích hợp (Integrated Development Environment - IDE) mã nguồn mở phát triển bởi Microsoft. Nó được thiết kế nhằm hỗ trợ phát triển ứng dụng và mã nguồn một cách dễ dàng, hiệu quả và mở rộng. Visual Studio Code chủ yếu được sử dụng cho việc phát triển ứng dụng web và phát triển ứng dụng máy tính cá nhân.



**Visual Studio Code có những đặc điểm chính sau:**

* Mã nguồn mở: VS Code là một sản phẩm mã nguồn mở, có nghĩa là người dùng có thể tải và sử dụng nó miễn phí và thậm chí thay đổi mã nguồn mở nếu cần.
* Hỗ trợ nhiều ngôn ngữ lập trình: VS Code hỗ trợ nhiều ngôn ngữ lập trình phổ biến, bao gồm JavaScript, TypeScript, Python, Java, C++, C#, và nhiều ngôn ngữ khác. Bạn có thể mở rộng hỗ trợ cho ngôn ngữ tùy chỉnh.
* Giao diện người dùng thân thiện: VS Code có giao diện người dùng trực quan, dễ sử dụng và đề cao hiệu suất.
* Hỗ trợ mở rộng và phần mềm mở rộng: Có thể mở rộng tính năng của VS Code bằng cách cài đặt các phần mềm mở rộng (extensions) từ cộng đồng. Các phần mềm mở rộng này cung cấp tính năng bổ sung như hỗ trợ cho các framework, ngôn ngữ, và công cụ phát triển cụ thể.
* Debugging tích hợp: VS Code cung cấp tích hợp với các trình gỡ lỗi (debugger) cho nhiều ngôn ngữ, cho phép xác định và sửa lỗi trong mã nguồn dễ dàng hơn.
* Hỗ trợ kiểm soát phiên bản: Có thể tích hợp VS Code với các hệ thống kiểm soát phiên bản như Git để quản lý và theo dõi sự thay đổi trong mã nguồn.
* Hỗ trợ tích hợp dự án: VS Code cho phép quản lý và làm việc với nhiều dự án khác nhau, giúp bạn tổ chức và duyệt qua mã nguồn dễ dàng.

1. **Postman**

Postman là một công cụ cho phép chúng ta thao tác với API, phổ biến nhất là REST. Postman hiện là một trong những công cụ phổ biến nhất được sử dụng trong thử nghiệm các API. Với Postman, ta có thể gọi Rest API mà không cần viết dòng code nào.

Postman hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, PATCH, DELETE, …). Bên cạnh đó, nó còn cho phép lưu lại lịch sử các lần request, rất tiện cho việc sử dụng lại khi cần.



**Những lợi ích khi sử dụng Postman:**

* Sử dụng Collections (Bộ sưu tập) – Postman cho phép người dùng tạo bộ sưu tập cho các lệnh gọi API của họ. Mỗi bộ sưu tập có thể tạo các thư mục con và nhiều yêu cầu (request). Điều này giúp việc tổ chức các bộ thử nghiệm.
* Collaboration – Collections và environment có thể được import hoặc export giúp chia sẻ tệp dễ dàng.
* API Testing – Test trạng thái phản hồi HTTP.
* Gỡ lỗi – Bảng điều khiển Postman giúp kiểm tra dữ liệu nào đã được truy xuất giúp dễ dàng gỡ lỗi kiểm tra.

1. **DBeaver**

DBeaver là phần mềm quản lý cơ sở dữ liệu hỗ trợ JDBC driver.

DBeaver rất trực quan và dễ sử dụng để kết nối với nhiều cơ sở dữ liệu khác nhau, chạy truy vấn, tạo ERD.

DBeaver hỗ trợ các hệ quản trị cơ sở dữ liệu Oracle, MySQL, IBM DB2, SQL Server, PostgreSQL, Java DB…

Chúng ta có thể tải xuống tại dbeaver.io/download/ có phiên bản miễn phí trên các hệ điều hành Windows, Mac OS X, Linux.

****

**Một số lợi ích và tính năng của ứng dụng này:**

* Câu lệnh SQL / thực thi tập lệnh.
* Tự động điền và liên kết siêu dữ liệu trong trình soạn thảo SQL.
* Giao diện được thiết kế hiện đại và cẩn thận, có thể cuộn trang để xem danh sách kết quả 1 cách tiện lợi.
* Xuất dữ liệu (bảng, kết quả truy vấn).
* Tìm kiếm các đối tượng cơ sở dữ liệu (bảng, cột, ràng buộc, thủ tục).
* DBeaver tiêu thụ ít bộ nhớ hơn nhiều so với các chương trình phổ biến khác (SQuirreL, DBVisualizer).
* Tất cả các hoạt động cơ sở dữ liệu từ xa hoạt động ở chế độ mở khóa, vì vậy DBeaver không bị sập nếu máy chủ cơ sở dữ liệu không phản hồi hoặc có sự cố mạng liên quan.

1. **Framework**
2. **Spring Boot**

Spring Boot là một dự án con của Spring Framework, được phát triển bởi Pivotal (một công ty con của VMware). Nó là một framework phát triển ứng dụng Java cho việc xây dựng các ứng dụng web và ứng dụng phía máy chủ một cách dễ dàng và nhanh chóng. Spring Boot giúp tối ưu hóa quá trình phát triển và triển khai ứng dụng Java bằng cách cung cấp nhiều tính năng và cấu hình mặc định, giúp lập trình viên tập trung vào việc phát triển logic ứng dụng chứ không phải là cấu hình phức tạp.



**Các đặc điểm chính của Spring Boot bao gồm:**

* Tự cấu hình (Auto-Configuration): Spring Boot tự động cấu hình các thành phần cơ bản của ứng dụng dựa trên các thư viện và phụ thuộc bạn sử dụng trong dự án. Điều này giúp giảm sự phức tạp trong việc cấu hình và đảm bảo tích hợp dễ dàng.
* Standalone: Ứng dụng Spring Boot có thể chạy độc lập, không cần một máy chủ ứng dụng ngoài như Tomcat. Spring Boot bao gồm một máy chủ web nhúng, giúp triển khai ứng dụng dễ dàng hơn.
* Spring Boot Starter: Spring Boot cung cấp các Starter là các bộ cài đặt chuẩn để phát triển ứng dụng cho các mục đích cụ thể như ứng dụng web, dự án thực hiện giao diện RESTful, dự án thực hiện xử lý dữ liệu, v.v.
* Hỗ trợ tích hợp với Spring Framework: Spring Boot là sự mở rộng của Spring Framework và hoàn toàn tương thích với nó. Người dùng có thể sử dụng các thành phần của Spring Framework và Spring Boot trong cùng một dự án.
* Hệ thống quản lý cấu hình: Spring Boot cung cấp hệ thống quản lý cấu hình dựa trên các tệp cấu hình (properties), biến môi trường và nhiều nguồn dữ liệu khác.

1. **VueJS**

VueJS (đôi khi được viết tắt là Vue) là một framework JavaScript mã nguồn mở được sử dụng để xây dựng giao diện người dùng và ứng dụng web tương tác. Nó được thiết kế để dễ dàng tích hợp vào các dự án hiện có và giúp phát triển ứng dụng web đơn trang (Single Page Applications - SPAs) một cách dễ dàng và hiệu quả.

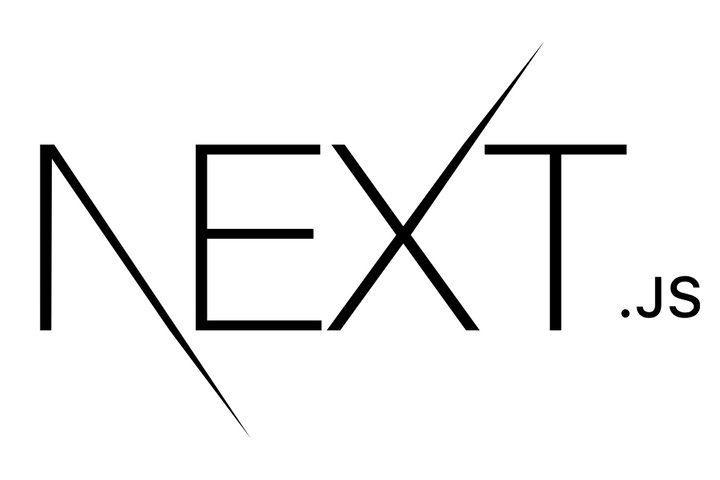


**VueJS có những đặc điểm chính sau:**

* Mã nguồn mở: Vue.js là một dự án mã nguồn mở, nghĩa là có thể sử dụng nó miễn phí, thậm chí tham gia đóng góp vào việc phát triển nếu bạn muốn.
* Dễ học và sử dụng: Vue.js được thiết kế với cú pháp đơn giản và dễ hiểu, với tài liệu hướng dẫn chi tiết, giúp lập trình viên nhanh chóng làm quen và sử dụng.
* Tích hợp dễ dàng: Vue.js có khả năng tích hợp vào các dự án hiện có một cách dễ dàng. Lập trình viên có thể sử dụng Vue.js như một thư viện độc lập hoặc kết hợp với các framework khác như Laravel hoặc Ruby on Rails.
* Tích hợp với Virtual DOM: Vue.js sử dụng Virtual DOM để tối ưu hóa hiệu suất của ứng dụng. Khi dữ liệu thay đổi, Vue.js cập nhật một bản sao ảo của DOM trước khi cập nhật DOM thực tế, giúp giảm tiêu tốn tài nguyên và làm cho ứng dụng chạy nhanh hơn.
* Cấu trúc thành phần: Vue.js sử dụng cấu trúc thành phần, cho phép xây dựng giao diện người dùng bằng cách kết hợp các thành phần độc lập với nhau. Điều này giúp tạo ra mã nguồn dễ quản lý và tái sử dụng.
* Hệ thống giao diện người dùng linh hoạt: Vue.js cho phép xây dựng các ứng dụng với giao diện người dùng động và tương tác một cách dễ dàng. Lập trình viên có thể thêm các hiệu ứng, xử lý sự kiện và tương tác người dùng một cách linh hoạt.

1. **NextJS**

Next.js là một framework phát triển ứng dụng web phía máy chủ và phía máy khách (server-side rendering - SSR) cho React, một thư viện JavaScript phát triển bởi Facebook. Next.js giúp tạo ra các ứng dụng web hiệu suất cao, có thời gian tải nhanh, và dễ bảo trì. Nó cung cấp một cách tiếp cận hiệu quả để xây dựng các ứng dụng web đa trang (multi-page applications) với React.



**Các đặc điểm chính của Next.js bao gồm:**

* Server-Side Rendering (SSR): Next.js cho phép tạo ra ứng dụng với SSR, giúp trình duyệt lấy dữ liệu từ máy chủ và tạo HTML trang trước khi hiển thị cho người dùng. Điều này cải thiện tốc độ tải trang và SEO.
* Phát triển tĩnh (Static Site Generation - SSG): Next.js cung cấp khả năng xây dựng các trang web tĩnh tại thời điểm xây dựng (build time), giúp giảm tải máy chủ và tăng hiệu suất.
* Tích hợp dễ dàng với React: Next.js tích hợp tốt với React, cho phép sử dụng các thành phần và khái niệm của React một cách dễ dàng.
* Hệ thống định tuyến động: Next.js cung cấp hệ thống định tuyến động mạnh mẽ giúp quản lý định tuyến và chuyển hướng trang một cách dễ dàng.
* Quản lý trạng thái ứng dụng (App State Management): Next.js không yêu cầu sử dụng một thư viện quản lý trạng thái cụ thể, nhưng lập trình viên có thể tích hợp các thư viện quản lý trạng thái như Redux hoặc Mobx theo ý muốn.
* Cấu hình mặc định thông minh: Next.js cung cấp các giải pháp cấu hình mặc định thông minh, nhưng bạn có thể tùy chỉnh cấu hình nếu cần.
* Tự động tối ưu hóa hiệu suất: Next.js tự động tối ưu hóa hiệu suất cho ứng dụng bằng cách sử dụng hệ thống định tuyến và cách chuyển đổi động.

1. **Tổng quan về RESTful API**

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.

API (Application Programming Interface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML.

REST (REpresentational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.

Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API.

1. **KHẢO SÁT HỆ THỐNG**
2. **Khảo sát thực tế**

Về nghiệp vụ của một website kinh doanh thủy sinh vật cảnh, có thể liệt kê ra những vấn đề sau:

1. **Nghiệp vụ phía khách hàng**
2. **Xem và tìm kiếm các loại thủy sinh vật cảnh**

* Khách hàng có thể xem thông tin các thủy sinh vật cảnh hiện có trong cửa hàng, mỗi sản phầm bao gồm: tên, giá, số lượng còn lại, mô tả, hình ảnh, loại (động vật, thực vật), môi trường sống (nước mặn, nước ngọt), nhiệt độ môi trường, độ pH, tầng nước (đáy, ở giữa, trên mặt nước), hình thức sinh sản (để con, để trứng), loại thức ăn (thực vật, côn trùng…), chiều dài tối đa…
* Khách hàng có thể tìm kiếm nhanh một sản phẩm muốn mua, bằng cách nhập tên sản phầm cần tìm trong ô tìm kiếm.
* Có bộ lọc tìm kiếm và sắp xếp các sản phẩm, theo một hoặc nhiều tiêu chí, ví dụ: loại là động vật, tên từ A-Z, giá từ thấp đến cao, số lượng mua từ cao đến thấp, môi trường sống là nước ngọt,…

1. **Đặt hàng và thanh toán**

* Khách hàng sau khi đã tìm thấy sản phẩm thích hợp, sẽ lựa chọn thêm mẫu mã, màu sắc muốn mua nếu có (ví dụ: cá màu vàng, cá màu xanh…), nhập số lượng… sau đó chọn thêm sản phẩm vào giỏ hàng (để mua sau), hoặc chọn mua ngay để chuyển luôn sang màn hình thanh toán.
* Khách hàng có thể đặt hàng nhiều lần với nhiều đơn mua hàng (giữa khách hàng và cửa hàng), đơn mua hàng có các thông tin bao gồm ngày đặt, họ tên, địa chỉ, số điện thoại, ngày giao, trạng thái của đơn (chưa duyệt, đang giao, đã giao, đã huỷ) và một đơn đặt chỉ thuộc về một khách hàng.
* Một đơn mua hàng có nhiều sinh vật cảnh khác nhau và một sinh vật cảnh có thể được đặt với nhiều đơn mua.
* Khách hàng cần cho biết các thông tin: địa chỉ giao hàng, số điện thoại liên hệ, người nhận hàng… để shop có thể giao hàng. Khách hàng có thể chọn một địa chỉ giao hàng đã lưu lại trong sổ địa chỉ, hoặc tạo mới địa chỉ nếu chưa có.
* Người dùng có thể lựa chọn hình thức thanh toán bằng tiền mặt hoặc ví điện tử, thanh toán tại thời điểm giao hàng hoặc trả trước qua ví điện tử.

1. **Xem và thao tác với đơn hàng**

* Khách hàng có thể xem trạng thái của các đơn hàng đã đặt, ví dụ: đang chờ xác nhận, đã xác nhận, đang chuẩn bị hàng, đang giao, đã giao, đã hủy.
* Khách hàng có thể yêu cầu hủy đơn nếu đổi ý không muốn mua (đơn chưa được duyệt).
* Khi nhận được hàng thì khách hàng sẽ chọn xác nhận là hàng đã được giao, trường hợp khách hàng không xác nhận, trong khi nhân viên giao hàng xác nhận đã giao, thì đơn hàng chuyển sang trạng thái chờ xác nhận (bởi khách hàng).
* Khách hàng có thể trả hàng (nếu sản phẩm cho phép trả hàng) trong thởi gian quy định), nếu sản phẩm đã mua bị hỏng, hoặc không đảm bảo chất lượng, không đúng mô tả… và đưa ra ý kiến phản hồi cho cửa hàng.
* Đối với các đơn hàng đã nhận, khách hàng có thể đánh giá và nhận xét.

1. **Nghiệp vụ phía nhân viên**
2. **Quản lý nhân viên (đối với nhân viên quản lý)**

* Xem danh sách nhân viên.
* Chỉnh sửa thông tin nhân viên.

1. **Quản lý khách hàng**

Xem danh sách khách hàng.

1. **Nhập hàng từ nhà cung cấp**

* Xem danh sách các nhà cung cấp.
* Đặt hàng và nhập hàng từ nhà cung cấp.

1. **Quản lý thể loại sản phẩm và các sản phẩm**

* Xem các thể loại.
* Chỉnh sửa thể loại
* Xem danh sách sản phẩm..
* Chỉnh sửa sản phẩm.

1. **Quản lý các đợt khuyến mãi**

* Cho phép thêm các đợt khuyến mãi trong khoảng thời gian nhất định, mỗi đợt khuyến mãi chứa danh sách các sản phẩm tham gia khuyến mãi và phần trăm khuyến mãi của mỗi sản phẩm.
* Tạm ngưng hoặc hiệu chỉnh khuyến mãi khi cần thiết.

1. **Quản lý các đơn đặt hàng**

* Xem trạng thái đơn hàng.
* Duyệt đơn đặt hàng, chấp nhận hủy đơn nếu khách hàng yêu cầu.
* Xác nhận đã giao hàng cho khách.
* Mỗi đơn mua sẽ có một hoá đơn tương ứng được lập bởi nhân viên.

1. **Báo cáo**

* Thống kê doanh thu theo ngày, tháng, năm.
* Báo cáo chi tiết doanh số của từng sản phẩm.
* Thống kê theo sản phẩm.
* Xuất số liệu hoặc báo cáo ra file pdf hoặc excel.

1. **Các yêu cầu của hệ thống**
2. **Yêu cầu chức năng nghiệp vụ**
3. **Bộ phận quản lý**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Yêu cầu** |
| 1 | Xem danh sách nhân viên | Tra cứu |  |
| 2 | Thêm nhân viên | Lưu trữ | Phải có các thông tin bao gồm: email, mật khẩu, họ, tên, giới tính, số điện thoại, địa chỉ, CCCD, trạng thái |
| 3 | Sửa thông tin nhân viên | Lưu trữ | Trừ mã nhân viên không được sửa, còn lại được sửa |
| 4 | Xem danh sách khách hàng | Tra cứu |  |
| 5 | Xem danh sách thể loại sản phẩm | Tra cứu |  |
| 6 | Thêm thể loại sản phảm | Lưu trữ | Phải có các thông tin: Tên thể loại  Thông tin không bắt buộc: Mô tả |
| 7 | Sửa thể loại sản phẩm | Lưu trữ | Trừ mã thể loại không được chỉnh sửa, còn lại được phép chỉnh sửa |
| 8 | Xóa thể loại sản phẩm | Lưu trữ | Không được xóa thể loại đã có sản phẩm |
| 9 | Xem danh sách sản phẩm | Tra cứu |  |
| 10 | Thêm mới sản phẩm | Lưu trữ | Phải có các thông tin: Tên, trạng thái, loại  Các thông tin không bắt buộc bao gồm: Mô tả, môi trường sống, nhiệt độ môi trường, độ PH, vị trí trong bể, hình thức sinh sản, loại thức ăn, kích thước tối đa |
| 11 | Chỉnh sửa sản phẩm | Lưu trữ | Trừ mã sinh vật cảnh không được chỉnh sửa, còn lại được phép chỉnh sửa |
| 12 | Xem hình ảnh của sản phẩm | Tra cứu |  |
| 13 | Thêm mới hình ảnh sản phẩm | Lưu trữ | Thêm mới hình cho sản phẩm bằng cách thêm vào một liên kết dẫn đến hình ảnh đó, hoặc tải hình ảnh lên từ thiết bị, có thể thêm nhiều hình ảnh cho 1 sản phẩm |
| 14 | Xoá hình ảnh sản phẩm | Lưu trữ |  |
| 15 | Xem danh sách giá | Tra cứu |  |
| 16 | Cập nhật giá mới | Lưu trữ | Cập nhật giá và ngày áp dụng cho giá đó |
| 17 | Giảm trừ số lượng tồn | Lưu trữ | Giảm trừ trong trong trường hợp sinh vật cảnh bị chết đi |
| 18 | Xem các loại đơn hàng | Tra cứu |  |
| 19 | Duyệt/huỷ đơn hàng | Lưu trữ |  |
| 20 | Xem thống kê doanh thu | Tra cứu | Phải nhập khoảng thời gian để thống kê |
| 21 | Xuất file Excel của thống kê | Tra cứu |  |
| 22 | Đăng nhập | Lưu trữ | Phải nhập đúng email và mật khẩu |
| 23 | Đăng xuất | Lưu trữ |  |
| 24 | Lấy lại mật khẩu | Lưu trữ | Phải nhập đúng email của tài khoản để lấy lại mật khẩu (Mật khẩu mới được gửi qua email) |

Bảng 3.1 Yêu cầu chức năng nghiệp vụ cho bộ phận quản lý

1. **Bộ phận khách hàng**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Yêu cầu** | |
| 1 | Xem danh sách sản phẩm | Tra cứu |  |
| 2 | Xem chi tiết thông tin một sản phẩm | Tra cứu |  |
| 3 | Thêm sản phẩm vào giỏ hàng | Lưu trữ | Sản phẩm phải có số lượng tồn lớn hơn 0 mới được phép thêm vào giỏ hàng |
| 4 | Xoá sản phẩm khỏi giỏ hàng | Lưu trữ |  |
| 5 | Đặt hàng | Lưu trữ | Khi đặt hàng, phải nhập đầy đủ các thông tin sau: Tên người nhận, địa chỉ, số điện thoại  Thông tin không bắt buộc: Email của người nhận |
| 6 | Xem đơn hàng | Tra cứu |  |
| 7 | Huỷ đơn | Lưu trữ | Chỉ được huỷ khi đơn hàng đó chưa được nhân viên của cửa hàng xác nhận |
| 8 | Đăng nhập | Lưu trữ | Phải nhập đúng email và mật khẩu |
| 9 | Đăng xuất | Lưu trữ |  |
| 10 | Lấy lại mật khẩu | Lưu trữ | Phải nhập đúng email của tài khoản để lấy lại mật khẩu (Mật khẩu mới được gửi qua email) |

Bảng 3.2 Yêu cẩu chức năng nghiệp vụ cho bộ phận khách hàng

1. **Yêu cầu phi chức năng của hệ thống**

|  |  |  |
| --- | --- | --- |
| **STT** | **Nội dung** | **Mô tả chi tiết** |
| 1 | Phân quyền sử dụng | - Trang web có 2 trang đăng nhập: một cho nhân viên cửa hàng và một cho khách hàng  - Nhân viên quản lý được phép sử dụng tất cả các chức năng của ngưởi quản lý  - Nhân viên thường không được quản lý nhân viên khác và thống kê  - Khách hàng được sử dụng các chức năng dành cho khách hàng: Xem sản phẩm, quản lý giỏ hàng, đơn hàng |

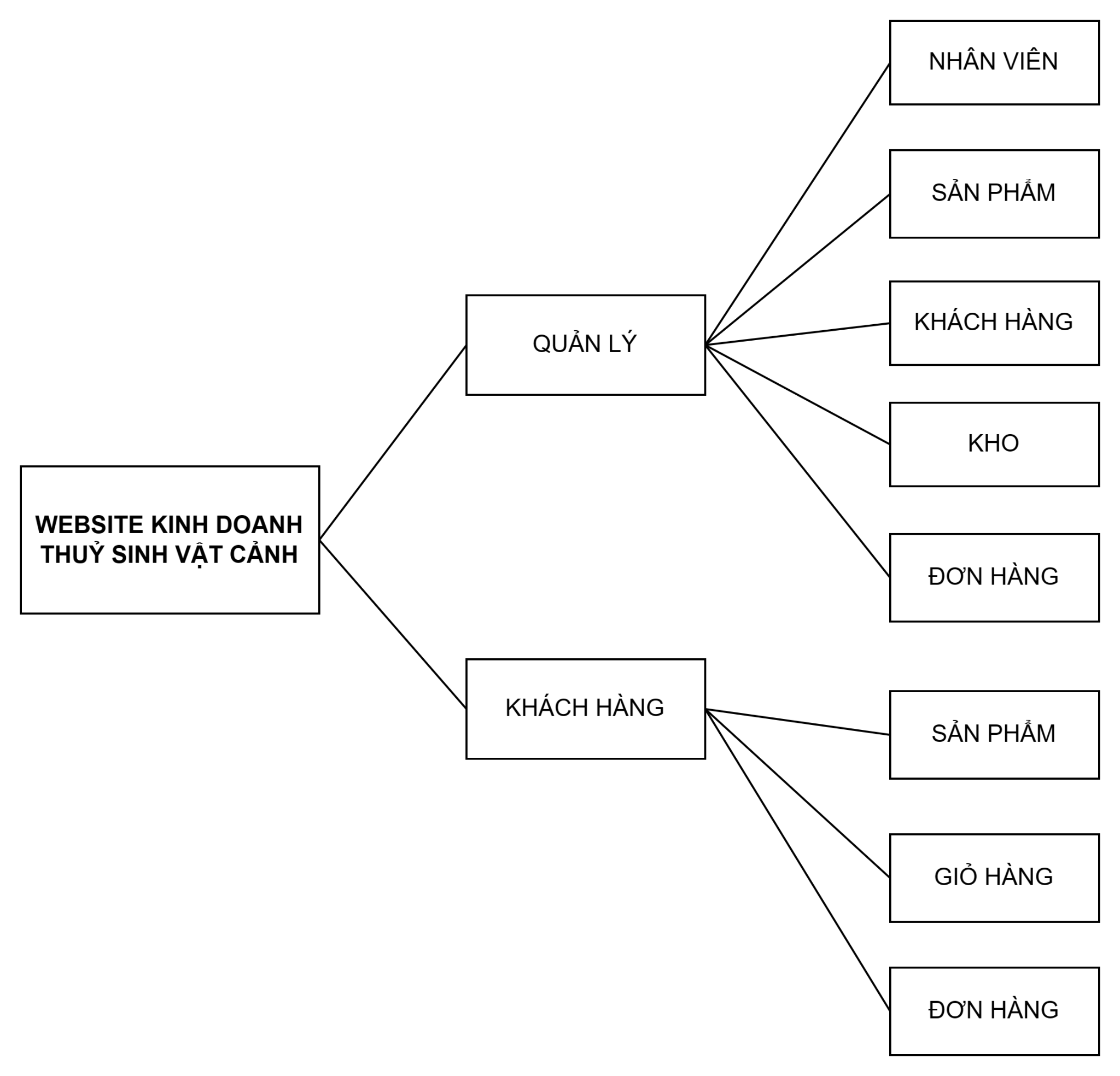
Bảng 3.3 Yêu cầu phi chức năng của hệ thống

1. **Yêu cầu về chất lượng**

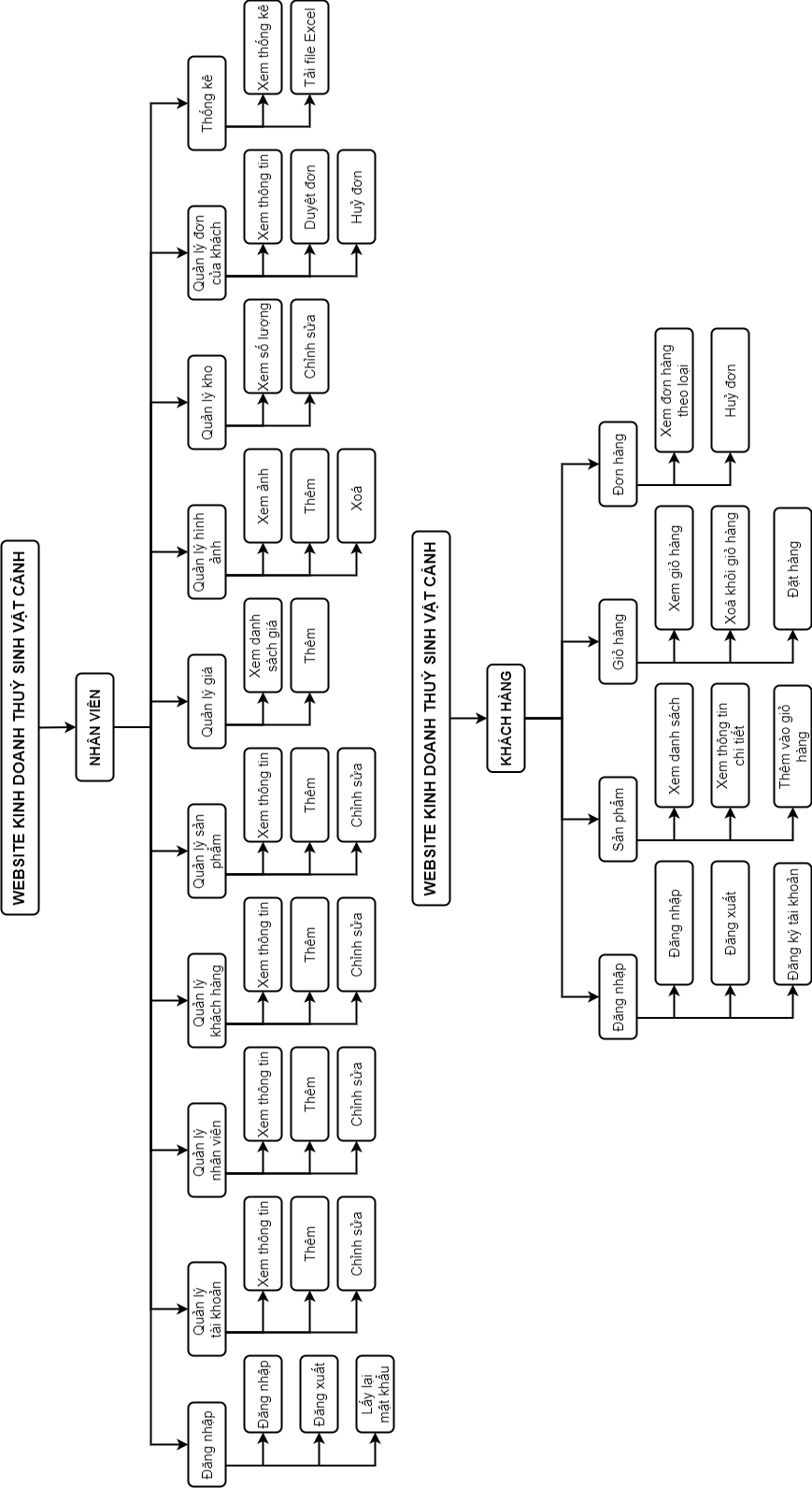
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Nội dung** | **Tiêu chuẩn** | **Mô tả chi tiết** |
| 1 | Trang dành cho nhân viên | Tiện dụng  Chính xác | Giao diện trực quan, dễ sử dụng  Việc lưu trữ và thay đổi thông tin khi thao tác các chức năng liên quan đến thêm – sửa – xoá phải hoàn toàn chính xác |
| 2 | Trang dành cho khách hàng | Tiện dụng  Chính xác | Giao diện trực quan, dễ sử dụng  Các thao tác đặt hàng phải nhanh chóng, chính xác |

Bảng 3.4 Yêu cầu về chất lượng

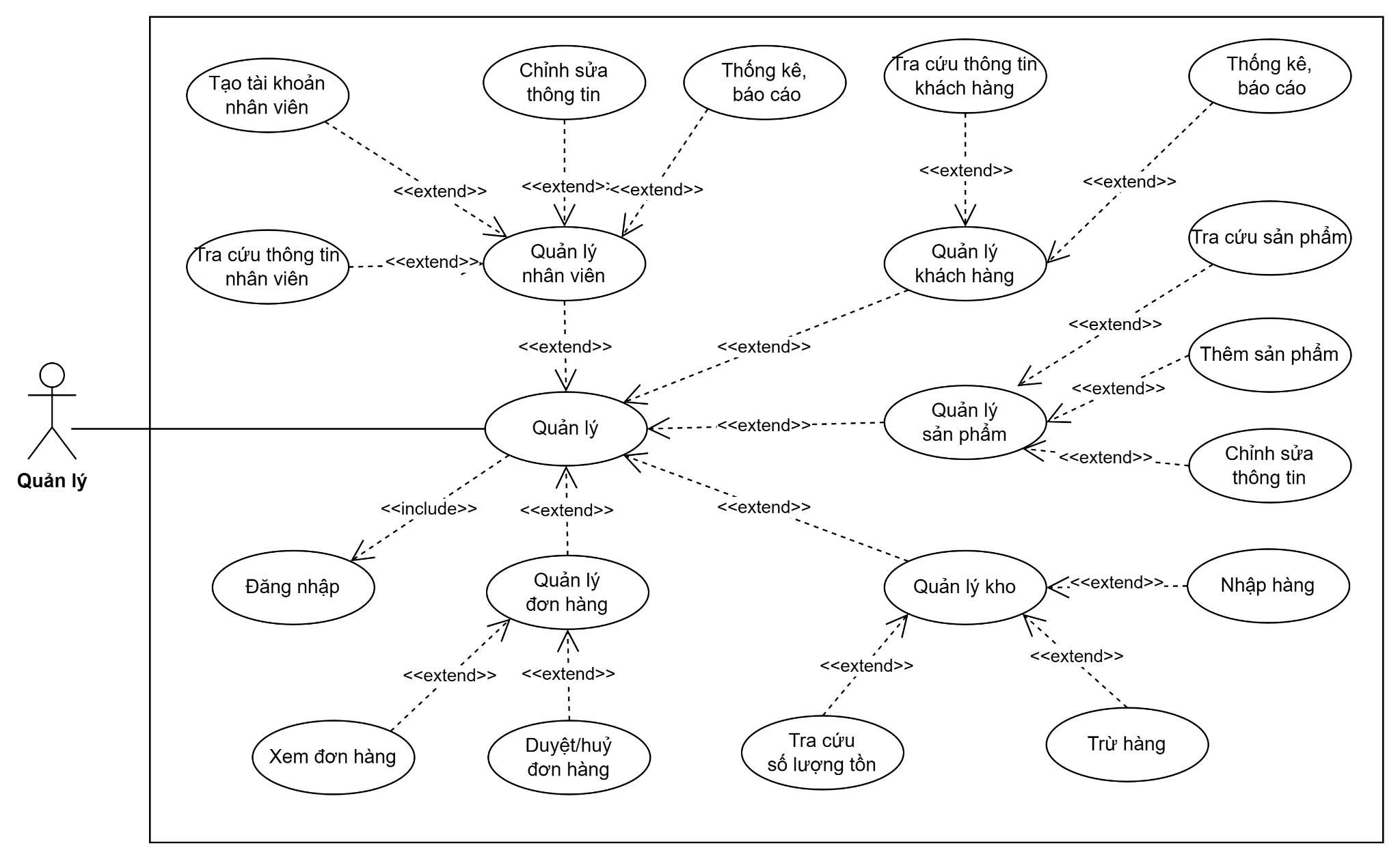
1. **PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**
2. **Sơ đồ tổng quan hệ thống**



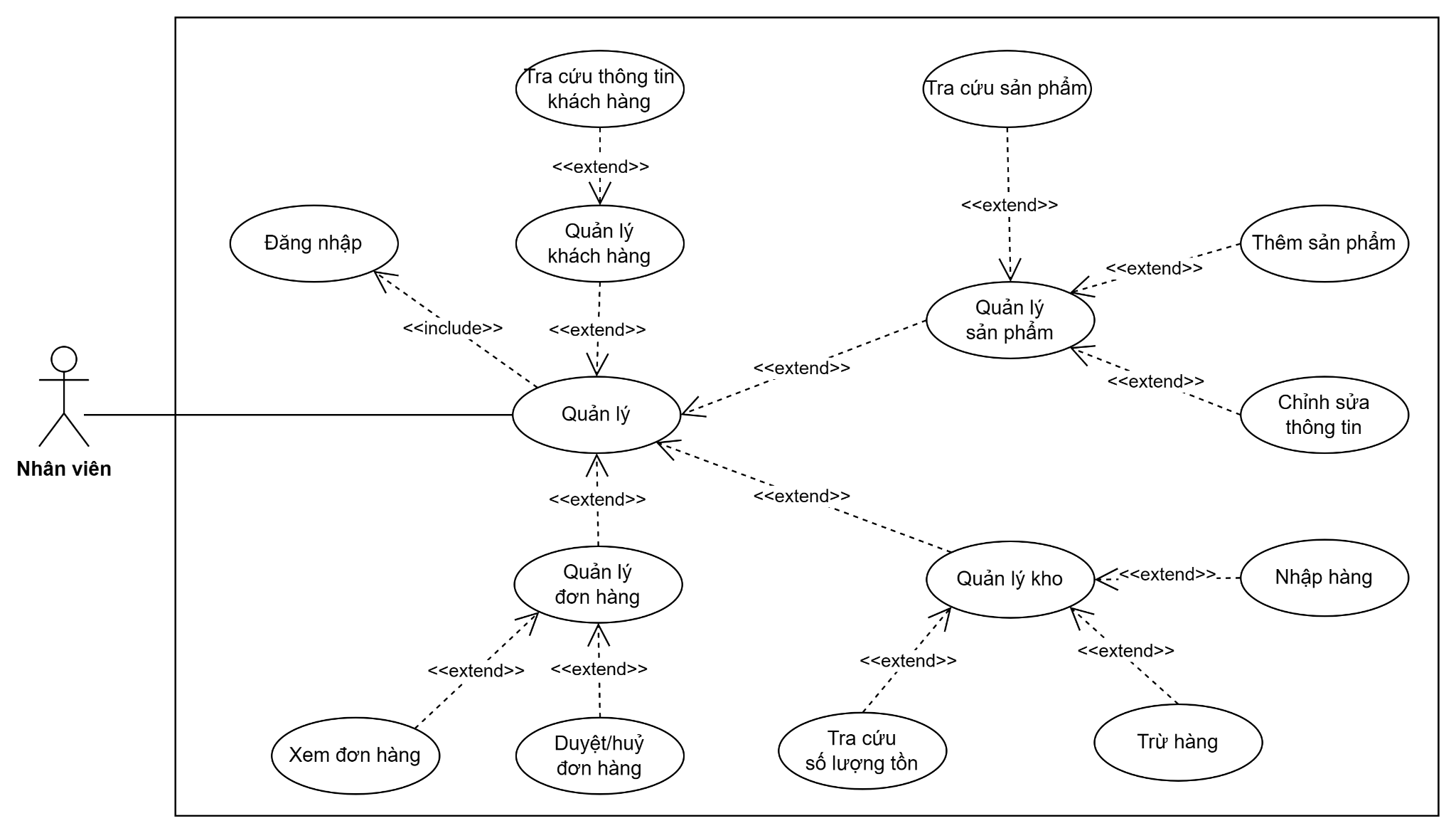
1. **Mô hình chức năng nghiệp vụ (BFD – Business Function Diagram)**

****

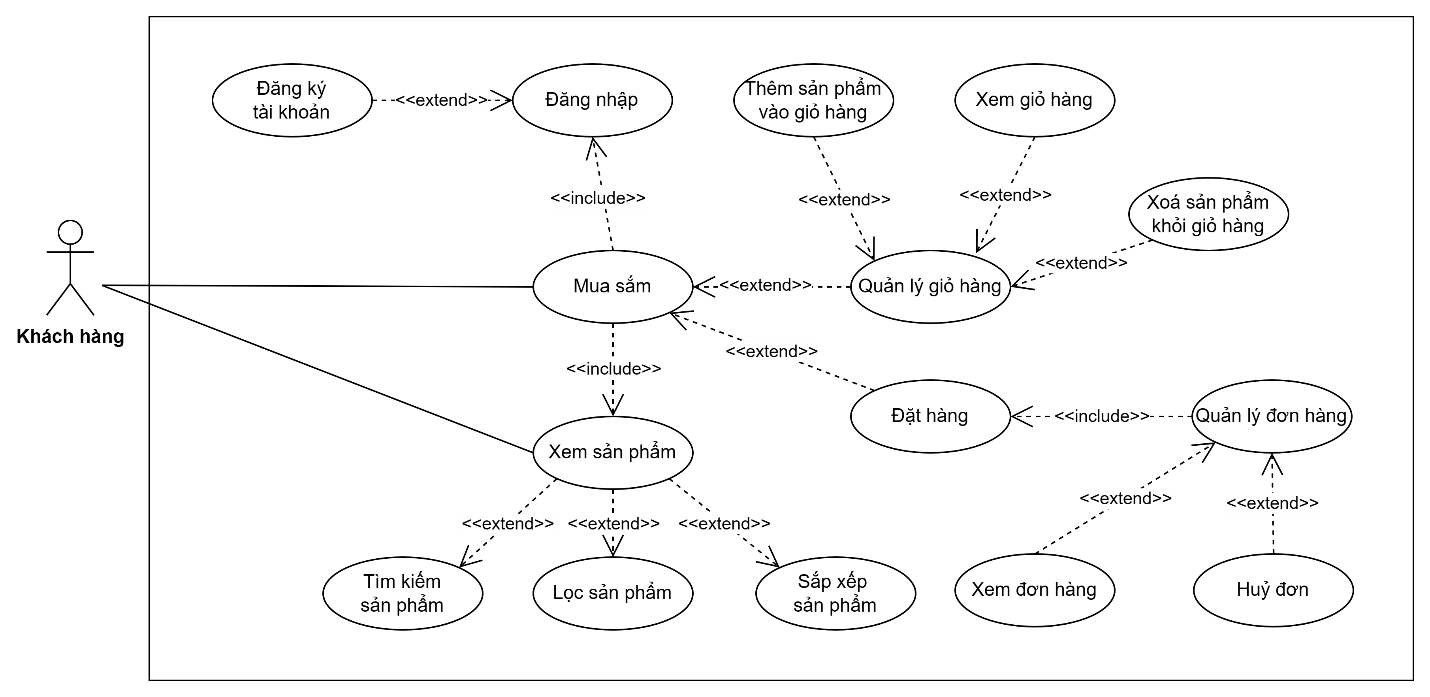
1. **Mô hình cộng tác (Collaboration Diagram)**
2. **Mô hình cộng tác cho bộ phận quản lý**



1. **Mô hình công tác cho bộ phận nhân viên**



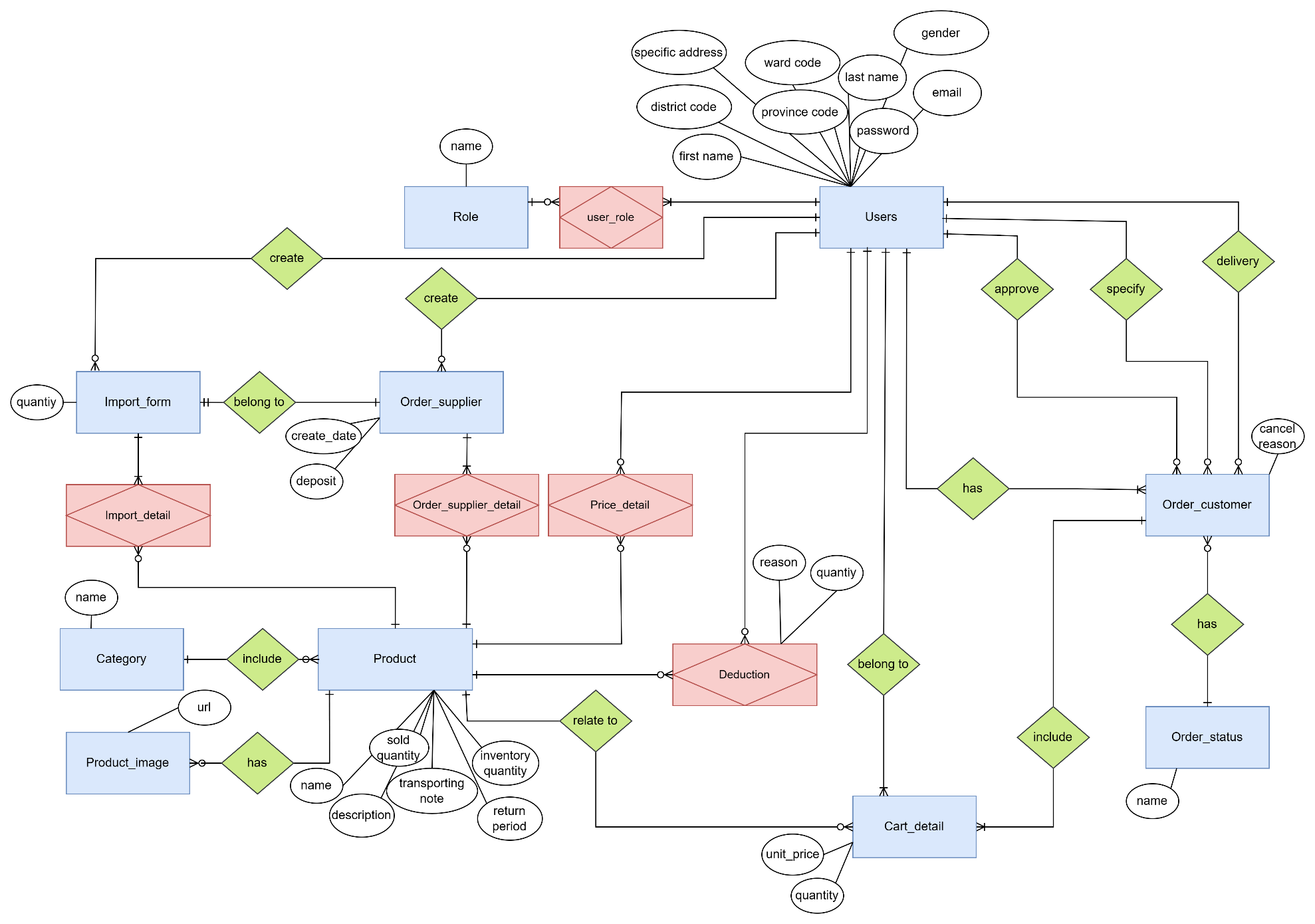
1. **Mô hình cộng tác cho bộ phận khách hàng**



1. **Thiết lập mô hình thực thể ERD (ERD – Entity Relationship Diagram)**
2. **Xác định các thực thể**

* **Users** (**id**, avatar, first\_name, last\_name, gender, birthday, email, password, phone, address, status, token, token\_creation\_date, created\_at, updated\_at)
* **Role** (**id**, name)
* **Product** (**id**, name, description, food\_type, habitat, inventory\_quanity, max\_size, ph, position, temperature, reproduction\_method, sold\_quantity, created\_at, updated\_at)
* **Category** (**id**, name, created\_at, updated\_at)
* **Product\_image** (**id**, url)
* **Order**\_**supplier** (**id**, supplier\_name, order\_date, delivery\_date, status, , created\_at, updated\_at)
* **Import**\_**form** (**id**, create\_date, created\_at, updated\_at)
* **Order**\_**customer** (**id**, delivery\_address, delivery\_date, delivery\_mail, delivery\_name, delivery\_phone, order\_date, total\_price, created\_at, updated\_at)
* **Order**\_**status** (**id**, name, num\_order)
* **Cart\_detail** (**id**, price, quantity, created\_at, updated\_at)

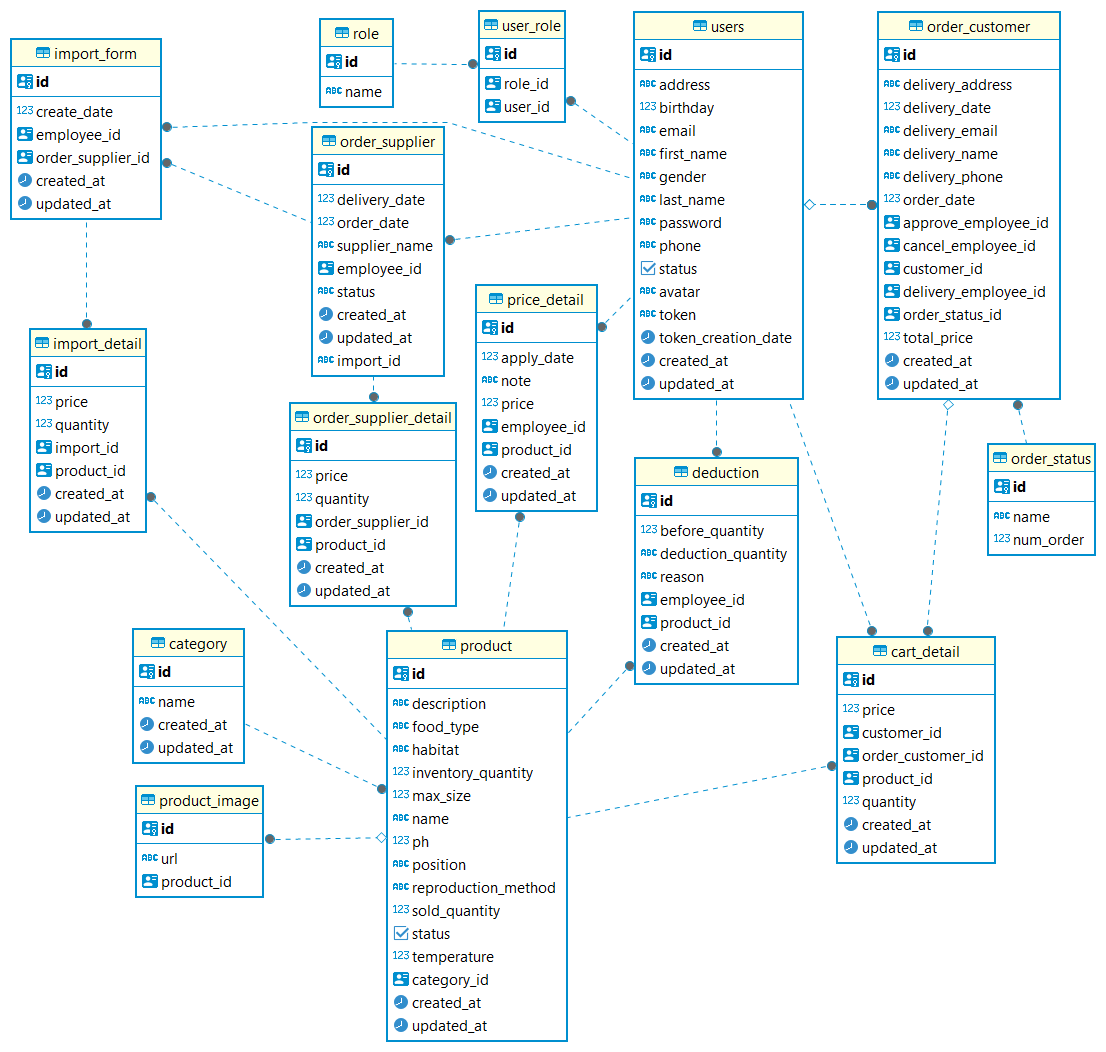
1. **Mô hình thực thể kết hợp ERD**



1. **Mô hình dữ liệu quan hệ dạng chuẩn 3**

* **Users** (**id**, avatar, first\_name, last\_name, gender, birthday, email, password, phone, address, status, token, token\_creation\_date, created\_at, updated\_at)
* **Role** (**id**, name)
* **Product** (**id**, name, description, food\_type, habitat, inventory\_quanity, max\_size, ph, position, temperature, reproduction\_method, sold\_quantity, status, created\_at, updated\_at, *category\_id*)
* **Category** (**id**, name, created\_at, updated\_at)
* **Product**\_**image** (**id**, url, *product\_id*)
* **Order**\_**supplier** (**id**, supplier\_name, order\_date, delivery\_date, status, created\_at, updated\_at, *employee\_id*)
* **Import**\_**form** (**id**, create\_date, created\_at, updated\_at, *employee\_id*, *order\_supplier\_id*)
* **Order**\_**customer** (**id**, delivery\_address, delivery\_date, delivery\_mail, delivery\_name, delivery\_phone, order\_date, total\_price, created\_at, updated\_at, *approve\_employee\_id, cancel\_employee\_id, delivery\_employee\_id, customer\_id, order\_status\_id*)
* **Order**\_**status** (**id**, name, num\_order)
* **Cart\_detail** (**id**, price, quantity, created\_at, updated\_at, *customer\_id, product\_id, order\_customer\_id*)
* **User\_role** (**id,** *role\_id, user\_id*)
* **Import\_detail** (**id,** price, quantity, created\_at, updated\_at**,** *import\_id, product\_id*)
* **Order\_supplier\_detail** (**id,** price, quantity, created\_at, updated\_at, *order\_supplier\_id, product\_id*)
* **Price\_detail** (**id,** price, apply\_date, note, created\_at, updated\_at, *employee\_id, product\_id*)
* **Deduction** (**id,**  before\_quantity, deduction\_quantity, reason, created\_at, updated\_at, *employee\_id, product\_id*)

1. **Thiết kế dữ liệu**
2. **Mô hình Diagram**



1. **Từ điển dữ liệu**

* **Users** (**id**, avatar, first\_name, last\_name, gender, birthday, email, password, phone, address, status, token, token\_creation\_date, created\_at, updated\_at)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | avatar | varchar | 255 |  |  |
| 3 | first\_name | varchar | 255 |  |  |
| 4 | last\_name | varchar | 255 |  |  |
| 5 | gender | varchar | 255 |  |  |
| 6 | birthday | int8 |  |  |  |
| 7 | email | varchar | 255 |  |  |
| 8 | password | varchar | 255 |  |  |
| 9 | phone | varchar | 255 |  |  |
| 10 | address | varchar | 255 |  |  |
| 11 | status | bool |  |  |  |
| 12 | token | varchar | 255 |  |  |
| 13 | token\_creation\_date | timestamp |  |  |  |
| 14 | created\_at | timestamp |  |  |  |
| 15 | updated\_at | timestamp |  |  |  |

Bảng 4.1 Users

* **Role** (**id**, name)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | name | varchar | 255 |  |  |

Bảng 4.2 Role

* **Product** (**id**, name, description, food\_type, habitat, inventory\_quanity, max\_size, ph, position, temperature, reproduction\_method, sold\_quantity, status, created\_at, updated\_at, *category\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **id** | UUID |  | Khoá chính |  |
| 2 | name | varchar | 255 |  |  |
| 3 | description | varchar | 255 |  |  |
| 4 | food\_type | varchar | 255 |  |  |
| 5 | habitat | varchar | 255 |  |  |
| 6 | inventory\_quanity | int4 |  |  |  |
| 7 | max\_size | float8 |  |  |  |
| 8 | ph | float8 |  |  |  |
| 9 | position | varchar | 255 |  |  |
| 10 | temperature | float8 |  |  |  |
| 11 | reproduction\_method | varchar | 255 |  |  |
| 12 | sold\_quantity | int4 |  |  |  |
| 13 | status | bool |  |  |  |
| 14 | created\_at | timestamp |  |  |  |
| 15 | updated\_at | timestamp |  |  |  |
| 16 | *category\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.3 Product

* **Category** (**id**, name, created\_at, updated\_at)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | name | varchar | 255 |  |  |
| 3 | created\_at | timestamp |  |  |  |
| 4 | updated\_at | timestamp |  |  |  |

Bảng 4.4 Category

* **Product\_image** (**id**, url, *product\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | url | varchar | 255 |  |  |
| 3 | *product\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.5 Product Image

* **Order**\_**supplier** (**id**, supplier\_name, order\_date, delivery\_date, status, created\_at, updated\_at, *employee\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | supplier\_name | varchar | 255 |  |  |
| 3 | order\_date | int8 |  |  |  |
| 4 | delivery\_date | int8 |  |  |  |
| 5 | status | varchar | 255 |  |  |
| 6 | created\_at | timestamp |  |  |  |
| 7 | updated\_at | timestamp |  |  |  |
| 8 | *employee\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.6 Order Supplier

* **Import**\_**form** (**id**, create\_date, created\_at, updated\_at, *employee\_id*, *order\_supplier\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | create\_date | int8 |  |  |  |
| 3 | created\_at | timestamp |  |  |  |
| 4 | updated\_at | timestamp |  |  |  |
| 5 | *employee\_id* | UUID |  | Khoá ngoại |  |
| 6 | *order\_supplier\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.7 Import Form

* **Order**\_**customer** (**id**, delivery\_address, delivery\_date, delivery\_mail, delivery\_name, delivery\_phone, order\_date, total\_price, created\_at, updated\_at, *approve\_employee\_id, cancel\_employee\_id, delivery\_employee\_id, customer\_id, order\_status\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | delivery\_address | varchar | 255 |  |  |
| 3 | delivery\_mail | varchar | 255 |  |  |
| 4 | delivery\_date | int8 |  |  |  |
| 5 | delivery\_name | varchar | 255 |  |  |
| 6 | delivery\_phone | varchar | 255 |  |  |
| 7 | order\_date | int8 |  |  |  |
| 8 | total\_price | int8 |  |  |  |
| 9 | created\_at | timestamp |  |  |  |
| 10 | updated\_at | timestamp |  |  |  |
| 11 | *approve\_employee\_id* | UUID |  | Khoá ngoại |  |
| 12 | *cancel\_employee\_id* | UUID |  | Khoá ngoại |  |
| 13 | *delivery\_employee\_id* | UUID |  | Khoá ngoại |  |
| 14 | *customer\_id* | UUID |  | Khoá ngoại |  |
| 15 | *order\_status\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.8 Order Customer

* **Order**\_**status** (**id**, name, num\_order)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | name | varchar | 255 |  |  |
| 3 | Num\_order | int4 |  |  |  |

Bảng 4.9 Order Status

* **Cart\_detail** (**id**, price, quantity, created\_at, updated\_at, *customer\_id, product\_id, order\_customer\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | price | int8 |  |  |  |
| 3 | quantity | int4 |  |  |  |
| 4 | created\_at | timestamp |  |  |  |
| 5 | updated\_at | timestamp |  |  |  |
| 6 | *customer\_id* | UUID |  | Khoá ngoại |  |
| 7 | *product\_id* | UUID |  | Khoá ngoại |  |
| 8 | *order\_customer\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.10 Cart Detail

* **User\_role** (**id,** *role\_id, user\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | *role\_id* | UUID |  | Khoá ngoại |  |
| 3 | *user\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.11 User Role

* **Import\_detail** (**id,** price, quantity, created\_at, updated\_at**,** *import\_id, product\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | price | int8 |  |  |  |
| 3 | quantity | int4 |  |  |  |
| 4 | created\_at | timestamp |  |  |  |
| 5 | updated\_at | timestamp |  |  |  |
| 6 | *import\_id* | UUID |  | Khoá ngoại |  |
| 7 | *product\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.12 Import Detail

* **Order\_supplier\_detail** (**id,** price, quantity, created\_at, updated\_at, *order\_supplier\_id, product\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | price | int8 |  |  |  |
| 3 | quantity | int4 |  |  |  |
| 4 | created\_at | timestamp |  |  |  |
| 5 | updated\_at | timestamp |  |  |  |
| 6 | *order\_supplier\_id* | UUID |  | Khoá ngoại |  |
| 7 | *product\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.13 Order Supplier Detail

* **Price\_detail** (**id,** price, apply\_date, note, created\_at, updated\_at, *employee\_id, product\_id*)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | price | int8 |  |  |  |
| 3 | apply\_date | int8 |  |  |  |
| 4 | note | varchar | 255 |  |  |
| 5 | created\_at | timestamp |  |  |  |
| 6 | updated\_at | timestamp |  |  |  |
| 7 | *employee\_id* | UUID |  | Khoá ngoại |  |
| 8 | *product\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.14 Price Detail

* **Deduction** (**id,**  before\_quantity, deduction\_quantity, reason, created\_at, updated\_at, *employee\_id, product\_id*)

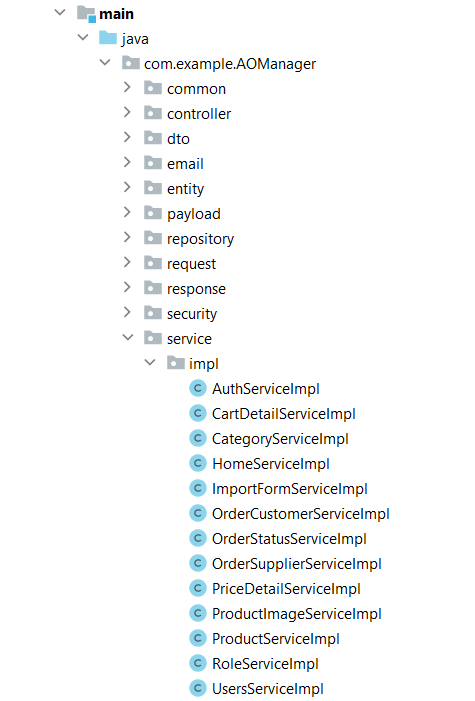
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Độ dài | Loại khoá | Ghi chú |
| 1 | **Id** | UUID |  | Khoá chính |  |
| 2 | before\_quantity | int4 |  |  |  |
| 3 | deduction\_quantity | int4 |  |  |  |
| 4 | reason | varchar | 255 |  |  |
| 5 | created\_at | timestamp |  |  |  |
| 6 | updated\_at | timestamp |  |  |  |
| 7 | *employee\_id* | UUID |  | Khoá ngoại |  |
| 8 | *product\_id* | UUID |  | Khoá ngoại |  |

Bảng 4.15 Deduction

1. **XÂY DỰNG ỨNG DỤNG**
2. **Xây dựng Web Service**
3. **Công nghệ sử dụng**

* Spring Boot: Tạo và phát triển ứng dụng nhanh
* Spring Web: Tạo Web App, ở đây gọi là web service và dùng tiêu chuẩn Restful API.
* Spring JPA: Tương tác với cơ sở dữ liệu
* PostgreSQL: Là một hệ quản trị cơ sở dữ liệu quan hệ - đối tượng.

1. **Sơ đồ chung cho thiết kế xử lý**

****

****

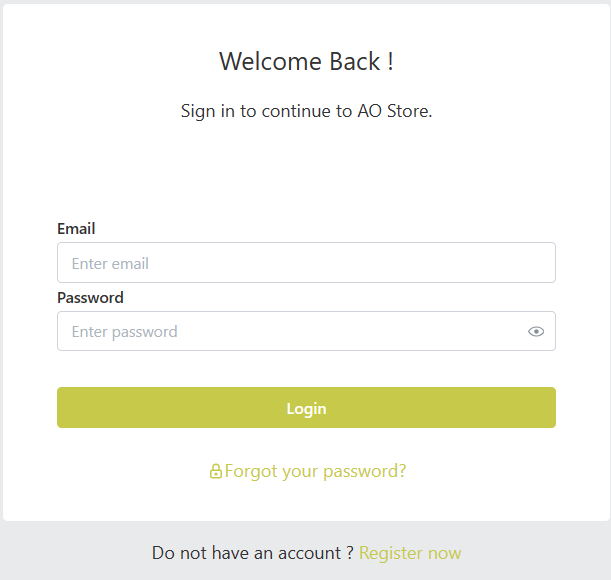
|  |  |
| --- | --- |
| **Tên gói (package)** | **Mô tả** |
| common | Chứa mọi thứ dùng chung cho toàn bộ project, bao bồm các hàm dùng chung (hàm format keyword tìm kiếm, hàm chuẩn hoá tên…), các phương thức check thông tin đầu vào (kiểm tra chuỗi UUID hợp lệ, kiểm tra ngày tháng hợp lệ…) và các message dùng chung. |
| controller | Nơi tiếp nhận các request |
| dto | Phục cho cho việc trả dữ liệu về cho client, giảm bớt lượng thông tin không cần thiết và hỗ trợ tăng cường bảo mật. |
| email | Khai báo các lớp và phương thức phục vụ cho việc gửi mail. |
| entity | Khi đối tượng hoá các thực thể trong CSDL, mỗi thực thể sẽ là một class trong package này. |
| payload | Quản lý đăng nhập, đăng ký, đổi mật khẩu có token. |
| repository | Nơi chứa các Interface để tương tác với cơ sở dữ liệu thông qua thư viện JPA. |
| request | Mỗi class trong package này là một thực thể chứa thông tin của yêu cầu khi gọi API. |
| response | Định nghĩa các lớp nhằm thống nhất định dạng trả về khi gọi API. |
| security | Định nghĩa các lớp, các phương thức liên quan đến xử lý xác thực. |
| service | Nơi định nghĩa chi tiết các hàm xử lý logic. |

Bảng 5.1 Mô tả package

1. **Cấu hình application.properties**

****

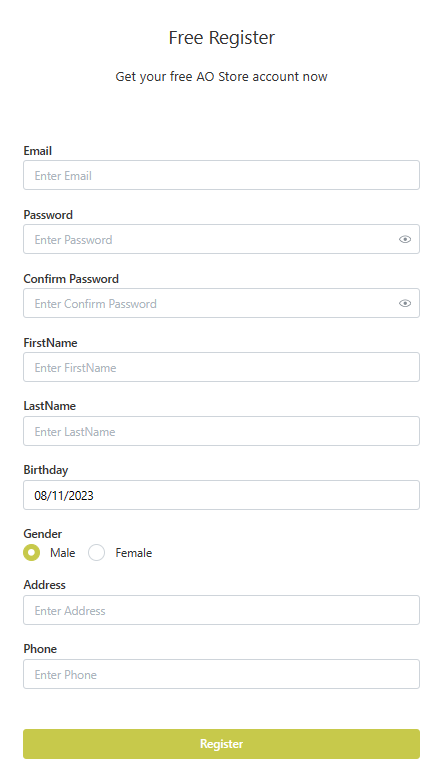
1. **Xây dựng giao diện**
2. **Thiết giao diện quản lý**
3. **Giao diện…**
4. **Giao diện…**
5. **Giao diện…**
6. **Thiết kế giao diện khách hàng**
7. **Giao diện đăng nhập (Sign In)**

****

***Mô tả giao diện:***

* Dùng để khách hàng đăng nhập vào trang web
* Khi nhấn nút đăng nhập, hệ thống sẽ kiểm tra email và password mà khách hàng nhập vào, sau đó sẽ kiểm tra tài khoản đã tồn tại trong hệ thống hay chưa
* Nếu có lỗi thì thông báo lỗi tại màn hình này
* Nếu không có lỗi, đăng nhập thành công sẽ chuyển đến màn hình trang chủ
* Khi nhấn link quên mật khẩu sẽ dẫn đến trang lấy lại mật khẩu
* Khi nhấn link đăng ký tài khoản mới, sẽ dẫn đến trang đăng ký tài khoản mới

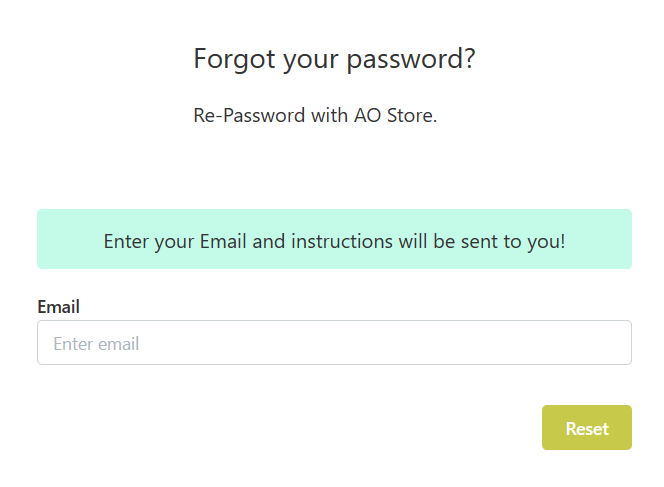
1. **Giao diện đăng ký (Sign Up)**

****

***Mô tả giao diện:***

* Dùng để khách hàng đăng kí tài khoản mới
* Kiểm tra các lỗi cơ bản:
* Không điền đầy đủ thông tin (Trừ birthday, tất cả các fields phải được nhập đầy đủ)
* Định dạng email, password, kiểm tra password trùng hay không?
* Khi nhấn button Đăng kí, hệ thống bắt đầu xử lí, kiểm tra lỗi người dùng nhập, kiểm tra lỗi đăng kí tài khoản
* Nếu có lỗi thì thông báo lỗi tại màn hình này
* Nếu không có lỗi, thông báo đăng nhập thành công, ghi nhớ đăng nhập và chuyển sang màn hình trang chủ

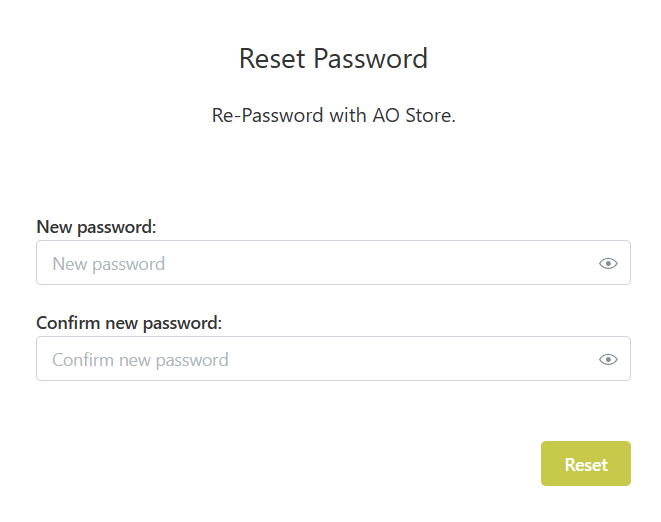
1. **Giao diện quên mật khẩu (Forgot Password)**



***Mô tả giao diện:***

* Dùng để khách hàng nhận link đặt lại mật khẩu khi quên mật khẩu
* Khách hàng nhập email vào ô text và ấn button reset để đặt lại mật khẩu
* Nếu email chưa được nhập mà nhấn nút reset thì sẽ có thông báo yêu cầu nhập email
* Nếu email đã được nhập, hệ thống sẽ kiểm tra email đó có tồn tại trong cơ sở dữ liệu hay không
* Nếu email nhập vào không tồn tại trong CSDL → Thông báo khách hàng
* Nếu email có tồn tại, gửi đến email của người dùng đường dẫn đến trang dặt lại mật khẩu

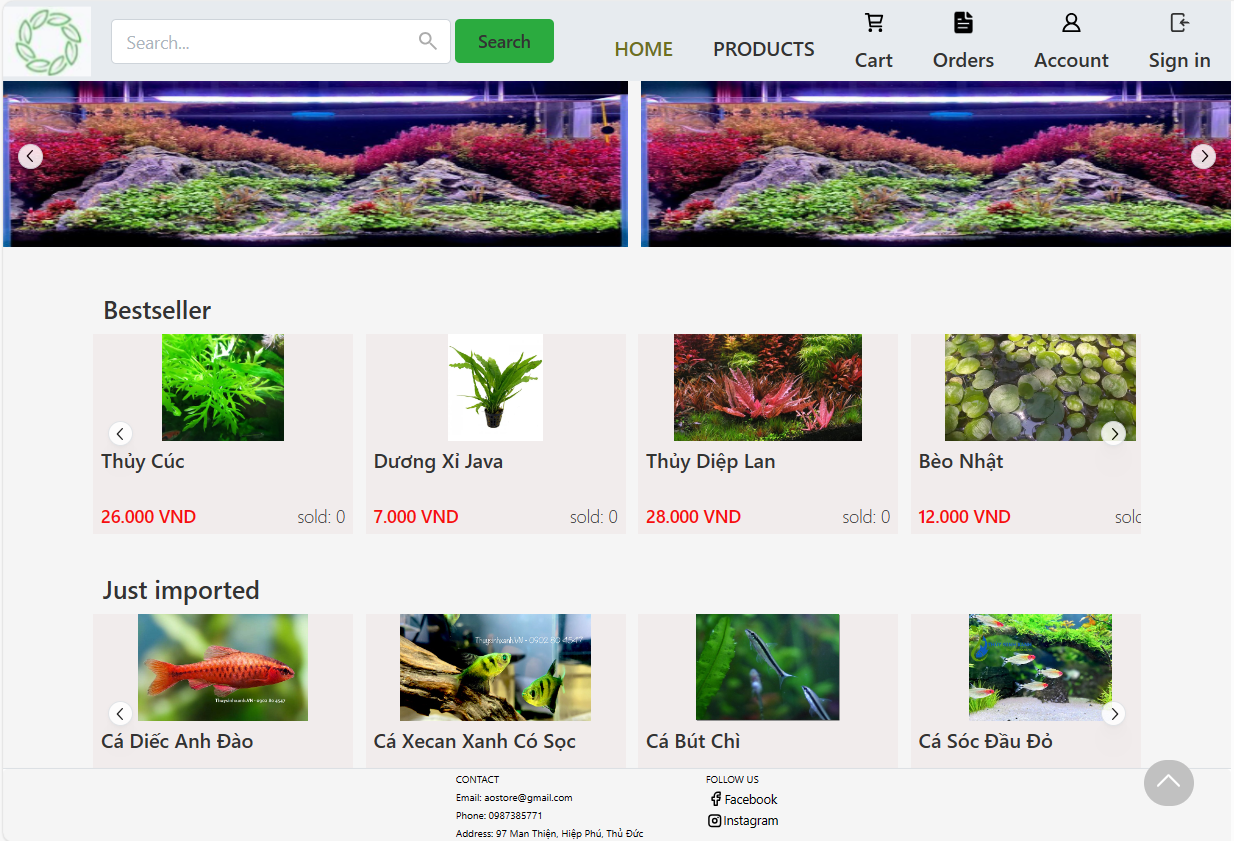
1. **Giao diện đặt lại mật khẩu (Reset Password)**



***Mô tả giao diện:***

* Dùng để khách hàng đặt lại mật khẩu khi quên mật khẩu
* Khách hàng nhập mật khẩu mới và xác nhận lại mật khẩu vào 2 ô text rồi nhấn nút reset để đặt lại mật khẩu
* Nếu mật khẩu chưa được nhập thì sẽ có thông báo yêu cầu nhập mật khẩu
* Nếu cả 2 ô text đã được nhập thì sẽ tiến hành kiểm tra
* Nếu mật khẩu mới và xác nhận mật khẩu lai không khớp thì sẽ thông báo đến người dùng
* Nếu cả hai mật khẩu trùng khớp, tiến hành đặt lại mật khẩu, thông báo đến người dùng

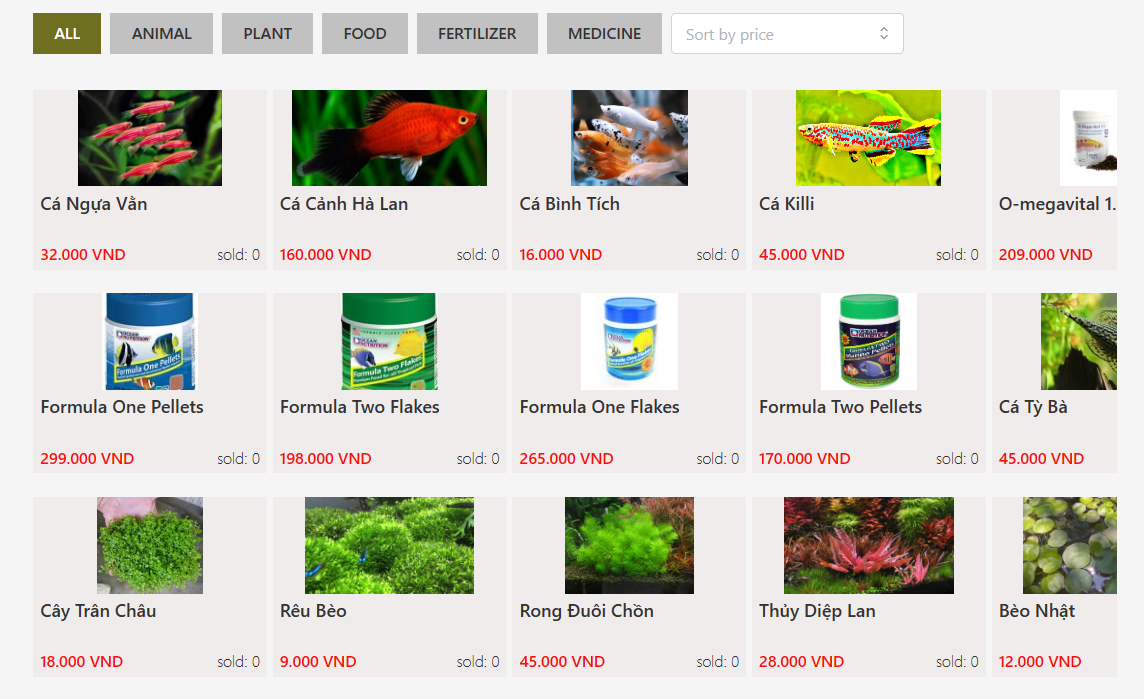
1. **Giao diện trang chủ (Home)**



***Mô tả giao diện***: Màn hình này chia ra làm 3 khu vực chính

* Khu vực các sản phẩm bán chạy: Thanh trượt qua lại chứa các sản phẩm bán chạy trong thời gian gần đây → Khi nhấn vào một sản phẩm ở thanh trượt này, khách hàng sẽ được đưa đến trang thông tin chi tiết của sản phẩm đó
* Khu vực các sản phẩm mới nhập về: Thanh trượt qua lại chứa các sản phẩm mới được cửa hàng nhập về → Khi nhấn vào một sản phẩm ở thanh trượt này, khách hàng sẽ được đưa đến trang thông tin chi tiết của sản phẩm đó
* Khu vực menu chính của màn hình (Xuất hiện xuyên suốt ở hầu hết các màn hình của khách hàng)
* Logo của cửa hàng
* Ô nhập các từ khoá để tìm kiếm sản phẩm
* Nút tìm kiếm: Sau khi khách hàng nhập từ khoá ở ô tìm kiếm, nhấn nút này để trang web tìm các sản phẩm trùng khớp với từ khoá.
* Nút trang chủ: Khi nhấn nút này, nếu hiện tại đang ở trang chủ thì không có gì thay đổi, nếu ở trang khác trang chủ thì sẽ được điều hướng về trang chủ
* Nút danh sách sản phẩm: Click nút này sẽ đưa khách hàng đến trang danh sách sản phẩm trong cửa hàng
* Nút giỏ hàng: Click nút này sẽ đưa khách hàng đến trang giỏ hàng
* Nút đơn hàng: Click nút này sẽ đưa khách hàng đến trang giỏ hàng
* Nút xem thông tin tài khoản các nhân: Đưa khách hàng đến trang có đầy đủ các thông tin cá nhân của khách hàng
* Nút đăng xuất: Đăng xuất tài khoản, điều hướng khách hàng đến màn hình đăng nhập
* Khu vực chân trang: chứa các thông tin liên hệ của nhà bán hàng

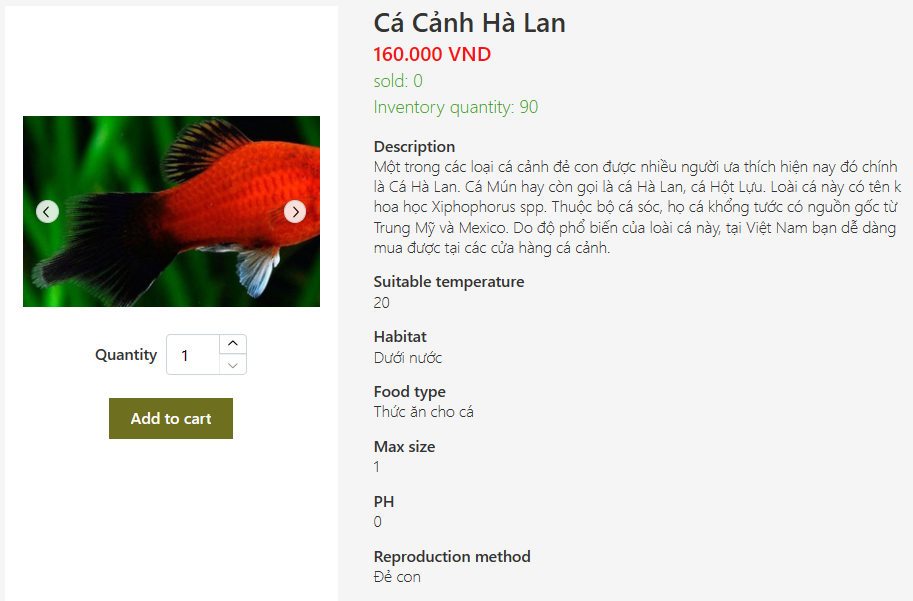
1. **Giao diện danh sách sản phẩm (List Product)**

****

***Mô tả giao diện***:

* Nút xem tất cả sản phẩm: Khi click nút này, trang web sẽ hiện ra tất cả sản phẩm có trong cửa hàng, không lọc theo bất kì tiêu chí nào
* Nút xem theo động vật: Khi click nút này, sẽ chỉ hiện ra các sản phẩm có loại là động vật
* Nút xem theo thực vật: Khi click nút này, sẽ chỉ hiện ra các sản phẩm có loại là thực vật
* Nút xem thức ăn: Các loại thức ăn dành cho cá
* Nút xem phân bón: Các loại phân bón dành cho cây thuỷ sinh
* Nút xem thuốc: Các loại thuốc trị bệnh cho cá và cho cây thuỷ sinh
* Sắp xếp sản phẩm: Sản phẩm sẽ được sắp xếp theo giá tăng dần hoặc giá giảm dần
* Sản phẩm: Khi click vào mốt sản phẩm, sẽ đưa khách hàng đến trang thông tin chi tiết của sản phẩm đó

1. **Giao diện chi tiết một sản phẩm (Detail Product)**

****

***Mô tả giao diện***:

* Khu vực chứa thông tin chi tiết của một sản phẩm, tại đây có các thông tin:
* Tên của sản phẩm
* Giá của sản phẩm
* Số lượng sản phẩm đã bán ra
* Số lượng sản phẩm còn trong kho
* Mô tả về sản phẩm để người dùng biết chính xác sản phẩm mà mình muốn mua hoặc biết thêm thông tin về sản phẩm này trong trường hợp trước đó chưa từng biết về sản phẩm này
* Các thông tin khác: Có thể là môi trường sống, nhiệt độ thích hợp, độ PH, vị trí trong bể, thức ăn, hình thức sinh sản, kích thước tối đa
* Các thành phần còn lại trên màn hình
* Hình ảnh của sản phẩm
* Nút mũi tên chuyển tiếp canh bên hình ảnh: Do là một sản phẩm có nhiều hình minh hoạ khác nhau nên các hình sẽ được đặt trong một thanh trượt, và có thể trượtqua lai bằng hai mũi tên hai bên
* Số lượng trước khi thêm vào giỏ hàng: Có thể nhập tay hoặc điều chỉnh tăng giảm băng mũi tên lên xuống ngay cạnh bên
* Nút thêm vào giỏ hàng: Khi click nút này, sản phẩm mà khách hàng đang xem sẽ được thêm vào giỏ hàng với số lượng đã được điều chỉnh trên màn hình

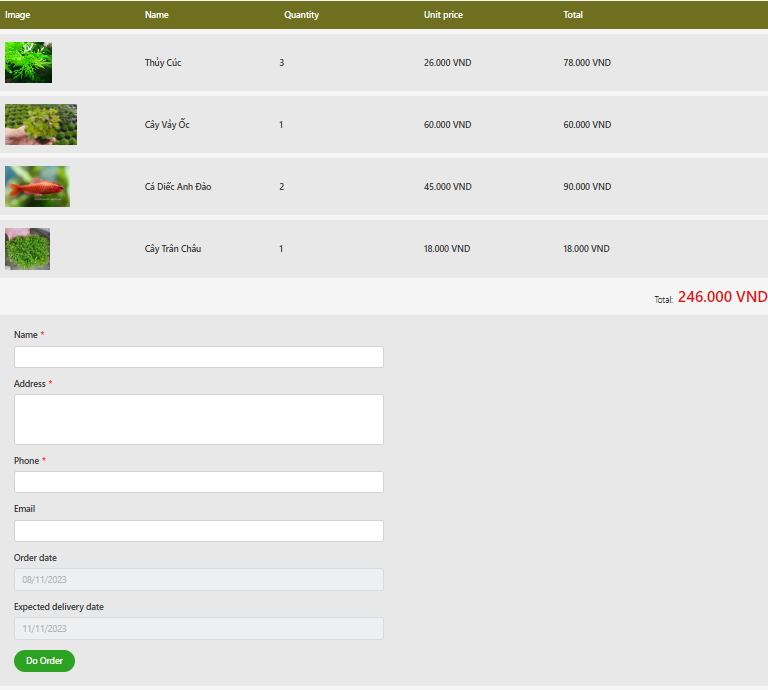
1. **Giao diện giỏ hàng (Cart)**



***Mô tả giao diện***:

* Mỗi một sản phẩm sẽ được hiện một dòng trong danh sách, trong đó có chứa các thông tin: Hình ảnh của sản phẩm, tên sản phẩm, só lượng mua, đơn giá của sản phẩm, thành tiền tính trên số lượng và đơn giá
* Trên mỗi hàng còn có nút xoá sản phẩm ra khỏi giỏ hàng
* Phía dưới là tổng tiền của các sản phẩm trong giỏ hàng
* Nút đặt hàng: Khi click nút này, khách hàng sẽ được điều hướng đến trang nhâp các thông tin cần thiết để tiến hành đặt hàng

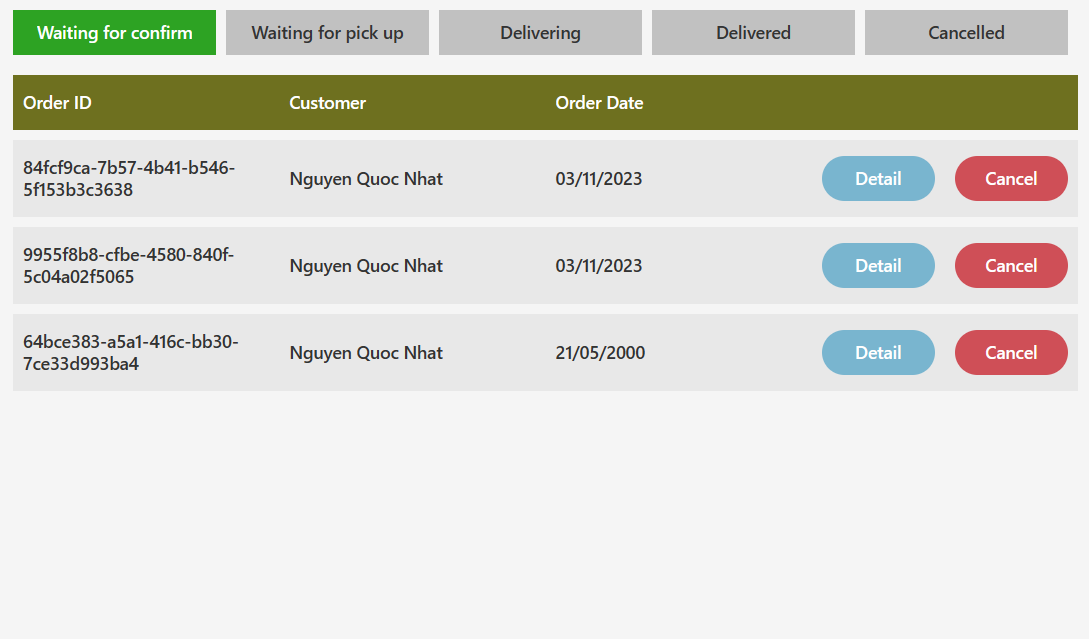
1. **Giao diện đặt hàng (Order)**



***Mô tả giao diện:***

* Khu vực danh sách và thông tin của các sản phẩm trong đơn hàng này (Tương tự như mô tả của giỏ hàng, được mô tả ở sheet Cart)
* Khu vực nhập thông tin của người nhận hàng
* Người dùng nhập thông tin của người nhận vào các trường cần thiết (Ngoại trừ ngày đặt hàng và ngày giao dự kiến)
* Nút xác nhận đặt hàng: Khi click vào nút này, các thông tin sẽ được kiểm tra, nếu thông tin nhập vào chưa hợp lệ thì sẽ thông báo tới khách hàng để chỉnh sửa, nếu thông tin hợp lệ sẽ tiến hành tạo một đơn hàng với trạng thái là đang chờ xác nhận

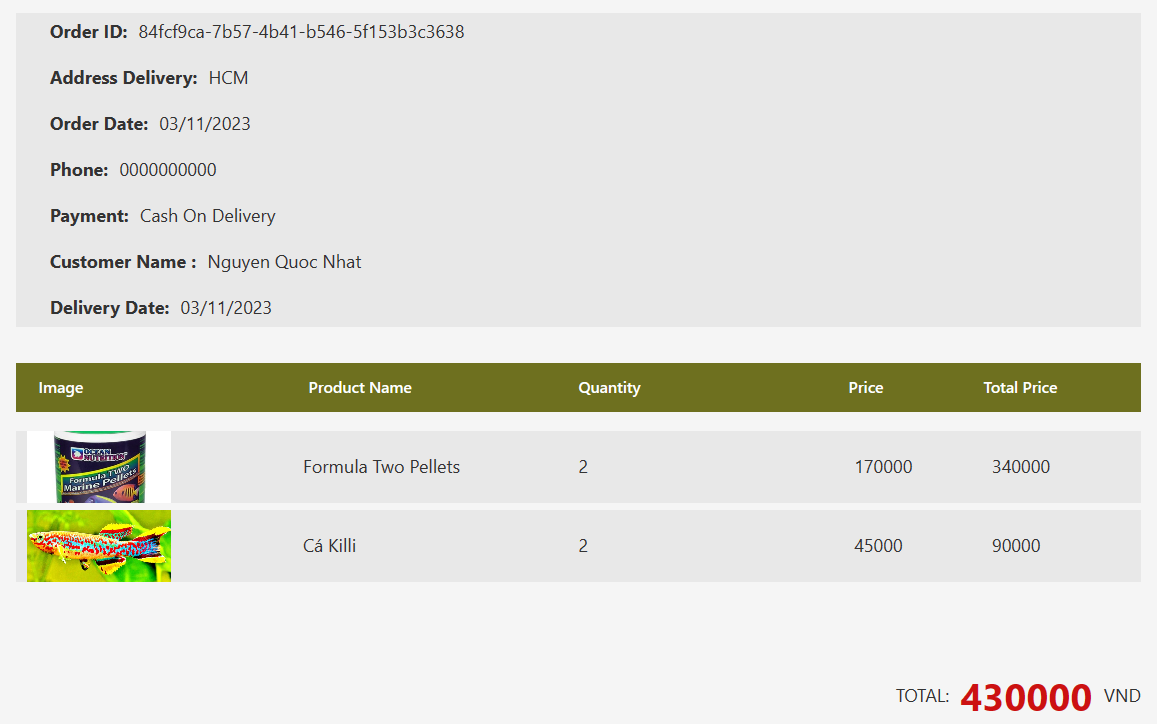
1. **Giao diện danh sách đơn đặt hàng (List Order)**



***Mô tả giao diện:***

* Hàng nút phía trên cho phép khách hàng chọn loại đơn hàng theo trạng thái mà khách hàng muốn xem, bao gồm các trạng thái: Chờ xác nhận, chờ lấy hàng, đang giao hàng, đã giao hàng, đã huỷ
* Mỗi một hàng trong danh sách là một đơn hàng, tại mỗi đơn hàng thì sẽ có hai nút lựa chọn
* Nút xem chi tiết đơn hàng: Click nút này sẽ điều hướng đến trang chi tiết đơn hàng
* Nút huỷ đơn hàng: Sau khi click nút này sẽ có một confirm message, nếu ng dùng xác nhận huỷ, đơn hàng sẽ chuyển sang trạng thái đã huỷ

1. **Giao diện chi tiết đơn đặt hàng (Detail Order)**

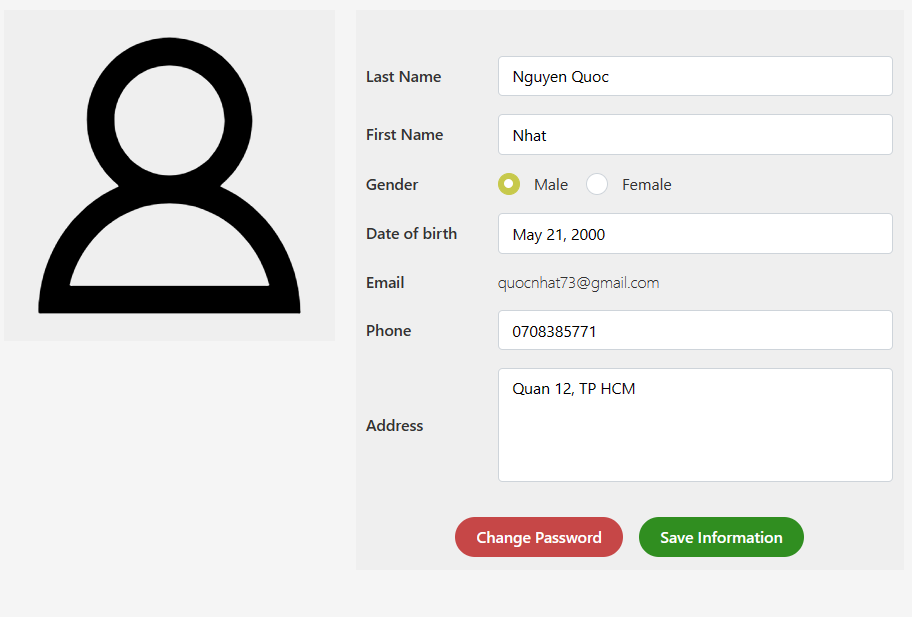


***Mô tả giao diện:***

Thông tin chi tiết của một đơn hàng, bao gồm:

* Mã đơn hàng
* Tên người nhận
* Địa chỉ nhận hàng
* Ngày đặt
* Ngày giao dự kiến
* Số điện thoại người nhận hàng
* Hình thức thanh toán
* Danh sách các sản phẩm trong đơn hàng, mỗi sản phẩm gồm các thông tin:
* Hình ảnh sản phẩm
* Tên sản phẩm
* Số lượng
* Đơn giá
* Thành tiền
* Tổng tiền của đơn hàng

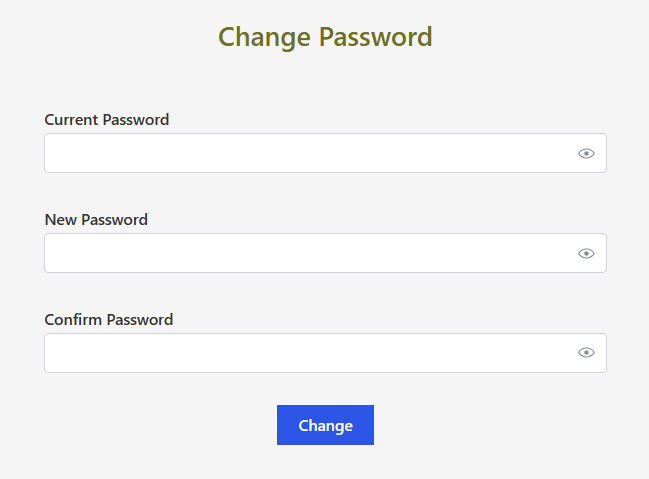
1. **Giao diện thông tin cá nhân (Personal Information)**

****

***Mô tả giao diện:***

* Thông tin của khách hàng bao gồm: tên, giới tính, ngày sinh, số điện thoại, email, địa chỉ
* Nút sửa thông tin: Người dùng có thể sửa các thônh tin của bản thân rồi nhấn nút sửa thông tin để cập nhật thông tin
* Nút đổi mật khẩu: Khi click nút này, sẽ đưa người dùng đến với giao diện đổi mật khẩu

1. **Giao diện đổi mật khẩu (Change Password)**

****

***Mô tả giao diện*:**

* Trong trường hợp khách hàng chưa nhập đủ các thông tin cần thiết để đổi mật khẩu mà click vào nút đổi mật khẩu thí sẽ có thông báo lỗi đến khách hàng
* Khi nhập đủ các thông tin cần thiết, trang web sẽ kiểm tra các thông tin nhập vào có hợp lệ hay không
* Nếu các thông tin nhập vào hợp lệ thì sẽ kiểm tra mật khẩu mới nhập vào 2 lần có trùng khớp nhau hay không, nếu trùng khớp thì tiến hành đổi mật khẩu cho khách hàng và thông báo đến cho khách hàng

1. **KẾT LUẬN**
2. **Đánh giá**
3. **Kết quả đạt được**

Xây dựng được hai trang web với tương đối các chức năng cơ bản dành cho nhân viên quản lý và khách hàng cửa hệ thống kinh doanh thuỷ sinh vật cảnh.

1. **Ưu điểm và khuyết điểm**

* **Ưu điểm:** Quản lý tốt hơn việc quản lý bằng sổ sách; Có tương đối đủ các chức năng cơ bản để quản lý một trang bán thuỷ sinh vật cảnh; Giao diện đơn giản dễ sử dụng.
* **Khuyết điểm:** Chưa thanh toán online; Giao diện tuy đơn giản dễ sử dụng nhưng chưa thật sự bắt mắt và tối ưu.

1. **Hướng phát triển**

* Trên cơ sở kế thừa những gì đã đạt được, tiếp tục sửa chữa và khắc phục hạn chế mà đồ án đang hiện có.
* Hoàn thiện chương trình một cách tối ưu nhất để đem vào sử dụng thực tế và cho người dùng sử dụng một cách thuận tiện nhất.

# **MỤC TÀI LIỆU THAM KHẢO**

1. Tìm hiểu về VueJS

* https://viblo.asia/p/tim-hieu-ve-vuejs-co-ban-p1-Do754bpWZM6
* https://topdev.vn/blog/gioi-thieu-framework-vue-js/

1. Tìm hiểu về NextJS

* <https://viblo.asia/p/tim-hieu-ve-nextjs-phan-1-V3m5WQkwZO7>
* https://kb.pavietnam.vn/nextjs-la-gi-1-vai-uu-nhuoc-diem-cua-nextjs.html

1. Tìm hiểu về PostgreSQL

* https://viblo.asia/p/tim-hieu-he-quan-tri-co-so-du-lieu-postgresql-m68Z0eLdlkG
* https://fptcloud.com/postgresql-la-gi/

1. Tìm hiểu về cách vẽ sơ đồ cộng tác

* <https://voer.edu.vn/m/bieu-do-cong-tac-collaboration-diagram/81ff0884>
* <https://vi.strephonsays.com/sequence-dia0gram-and-vs-collaboration-diagram-9260>