|  |  |
| --- | --- |
|  |  |
| HỌ VÀ TÊN SINH VIÊN: PHẠM XUÂN TRƯỜNG | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC CÔNG NGHỆ THÔNG TIN |
|  |
| **XÂY DỰNG WEBSITE BÁN ĐẶC SẢN QUÊ HƯƠNG(CÁC LOẠI NEM) SỬ DỤNG ASP.NET CORE VÀ ANGULAR** |
|  |
|  |
| **CBHD:ThS. Vũ Duy Giang** |
| NGÀNH CÔNG NGHỆ THÔNG TIN | **Sinh viên: Phạm Xuân Trường** |
| **Mã số sinh viên:2020602253** |
|  |
|  |
|  |
| Hà Nội – Năm 2024 |
|  |

**MỤC LỤC**

[L](#_Toc12111)**[ỜI CẢM ƠN](#_Toc12111)** [8](#_Toc12111)

**[MỞ ĐẦU](#_Toc14745)** [9](#_Toc14745)

[1. Giới thiệu 9](#_Toc7785)

[1.1. Lý do chọn đề tài 9](#_Toc17440)

[1.2. Tính cấp thiết của đề tài 10](#_Toc30970)

[2. Mục tiêu và phương pháp nghiên cứu 11](#_Toc16672)

[2.1. Mục tiêu của đề tài 11](#_Toc27264)

[2.2. Nội dung nghiên cứu 11](#_Toc1818)

[2.3. Phương pháp luận và phương pháp nghiên cứu 11](#_Toc19597)

**[CHƯƠNG 1: TỔNG QUAN VỀ WEBSITE BÁN NEM](#_Toc14882)** [15](#_Toc14882)

[1.1. Mục tiêu của website 15](#_Toc18564)

[1.2. Nghiên cứu thị trường 15](#_Toc9515)

[1.2.1. Phân tích thị trường nem tại Việt Nam 15](#_Toc4163)

[1.2.2. Đối tượng khách hàng 16](#_Toc17025)

[1.3. Lợi ích của việc xây dựng website 16](#_Toc11558)

[1.3.1. Lợi ích cho doanh nghiệp 17](#_Toc5454)

[1.3.2. Lợi ích cho khách hàng 17](#_Toc8191)

[1.4. Mô tả website 18](#_Toc17444)

**[CHƯƠNG 2: CÔNG NGHỆ ASP.NET CORE VÀ ANGULAR](#_Toc8698)** [20](#_Toc8698)

[2.1. Giới thiệu về ASP.NET Core 20](#_Toc9432)

[2.1.1. Khái niệm ASP.NET Core 20](#_Toc5517)

[2.1.2. Hoàn cảnh ra đời ASP.NET Core 21](#_Toc11316)

[2.1.3. Lý do ASP.NET Core phổ biến 21](#_Toc28633)

[2.2. Giới thiệu về ABP Framework 23](#_Toc2127)

[2.2.1. Khái niệm ABP Framework 23](#_Toc26155)

[2.2.2. Ưu điểm cúa ABP Framework 24](#_Toc8843)

[2.3. Giới thiệu về Angular 25](#_Toc28288)

[2.3.1. Khái niệm Angular 25](#_Toc31048)

[2.3.2. Ưu điểm của Angular 25](#_Toc21865)

[2.4. Giới thiệu về SQL Server 27](#_Toc13193)

[2.4.1. Khái niệm SQL Server 27](#_Toc18892)

[2.4.2. Ưu điểm của SQL Server 27](#_Toc7307)

[2.5. Giới thiệu về PrimeNG 28](#_Toc11625)

[2.5.1. Khái niệm PrimeNG 28](#_Toc17003)

[2.5.2. Ưu điểm của PrimeNG 29](#_Toc17347)

[2.6. Giới thiệu về RESTful API 30](#_Toc30999)

[2.6.1. Khái niệm RESTful API 30](#_Toc32180)

[2.6.2 Diễn giải các thành phần 30](#_Toc20477)

[2.6.3. Hoạt động của RESTful 31](#_Toc3828)

[2.6.4. Ưu điểm RESTful API 32](#_Toc28174)

[2.7. Mô hình Clean Architecture trong ASP.NET Core 33](#_Toc1141)

[2.7.1. Khái niệm Clean Architecture 33](#_Toc20239)

[2.7.2. Cấu trúc Clean Architecture 33](#_Toc2818)

[2.7.3. Ưu điểm của mô hình Clean Architecture 34](#_Toc29779)

[2.8. Chatbot DialogFlow 35](#_Toc20365)

[2.8.1. Ưu điểm Chatbot DialogFlow 35](#_Toc27724)

[2.8.2. Cách hoạt động 36](#_Toc28411)

**[CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI](#_Toc28565)** [38](#_Toc28565)

[3.1. Mô tả sơ bộ bài toán 38](#_Toc22919)

[3.2. Các yêu cầu hệ thống 38](#_Toc1891)

[3.2.1. Yêu cầu về chức năng 38](#_Toc28134)

[3.2.2. Yêu cầu về phi chức năng 39](#_Toc4233)

[3.3. Mô hình hóa Use case 40](#_Toc29390)

[3.3.1. Use case tổng quan 40](#_Toc18277)

[3.3.2. Actor Quản lý 41](#_Toc2459)

[3.3.3. Actor Khách hàng 41](#_Toc11382)

[3.4. Đặc tả Use case 42](#_Toc19229)

[3.4.1. Use case Đăng Nhập 42](#_Toc20078)

[3.4.2. Use case Đăng Ký 42](#_Toc2008)

[3.4.3. Use case Quản lý sản phẩm 43](#_Toc11767)

[3.4.4. Use case Quản lý đơn hàng 45](#_Toc12088)

[3.4.5. Use case Quản lý tài khoản 46](#_Toc20616)

[3.4.6. Use case Quản lý loại sản phẩm 48](#_Toc15760)

[3.4.7. Use case Thêm sản phẩm vào giỏ hàng 49](#_Toc765)

[3.4.8. Use case Xem chi tiết sản phẩm 50](#_Toc20186)

[3.4.9. Use case Xem sản phẩm theo loại sản phẩm 51](#_Toc4556)

[3.4.10. Use case Đặt hàng 52](#_Toc9350)

[3.5. Phân tích Use case 53](#_Toc11283)

[3.5.1. Use case Đăng Nhập 53](#_Toc7918)

[3.5.2. Use case Đăng Ký 54](#_Toc4043)

[3.5.3. Use case Quản lý đơn hàng 55](#_Toc28443)

[3.5.4. Use case Quản lý tài khoản 57](#_Toc20710)

[3.5.5. Use case Quản lý sản phẩm 58](#_Toc11265)

[3.5.6. Use case Thêm sản phẩm vào giỏ hàng 59](#_Toc23084)

[3.5.7. Use case Tìm kiếm 60](#_Toc3684)

[3.5.8. Use case Xem chi tiết sản phẩm 61](#_Toc32260)

[3.6. Mô hình hóa cơ sở dữ liệu 62](#_Toc28920)

[3.6.1. Các yêu cầu về dữ liệu 62](#_Toc24214)

[3.6.2. Biểu đồ thực thể liên kết 63](#_Toc24192)

[3.6.3. Mô tả văn tắt các thực thể 63](#_Toc28644)

[3.6.4. Mô tả chi tiết các bảng 65](#_Toc1534)

[3.7. Giao diện màn hình 70](#_Toc11216)

[3.7.1. Giao diện trang Admin 71](#_Toc27945)

[3.7.2. Giao diện trang Public 73](#_Toc3879)

**[KẾT LUẬN](#_Toc15297)** [76](#_Toc15297)

**[TÀI LIỆU THAM KHẢO](#_Toc30430)** [78](#_Toc30430)

**DANH MỤC CÁC BẢNG**

[Bảng 3.1 . Bảng mô tả chi tiết bảng sản phẩm 65](#_Toc32342)

[Bảng 3.2 . Bảng mô tả chi tiết bảng loại sản phẩm 65](#_Toc529)

[Bảng 3.3 . Bảng mô tả chi tiết bảng thuộc tính sản phẩm 66](#_Toc19637)

[Bảng 3.4 . Bảng mô tả chi tiết bảng thuộc tính kiểu số nguyêm 66](#_Toc13022)

[Bảng 3.5 . Bảng mô tả chi tiết bảng thuộc tính kiểu Varchar 67](#_Toc15402)

[Bảng 3.6 . Bảng mô tá chi tiết bảng thuộc tính kiểu số thập phân 67](#_Toc19706)

[Bảng 3.7 . Bảng mô tả chi tiết bảng thuộc tính kiểu văn bản 67](#_Toc11062)

[Bảng 3.8 . Bảng mô tả chi tiết bảng thuộc tính dạng ngày giờ 67](#_Toc7701)

[Bảng 3.9 . Bảng mô tả chi tiết bảng nhà sản xuất 68](#_Toc10552)

[Bảng 3.10 . Bảng mô tả chi tiết bảng đơn hàng 68](#_Toc9632)

[Bảng 3.11 . Bảng mô tả chi tiết bảng chi tiết đơn hàng 69](#_Toc20287)

[Bảng 3.12 . Bảng mô tả chi tiết bảng khuyến mãi 70](#_Toc21564)

**DANH MỤC HÌNH ẢNH**

[Hình 2.1 . Tổng quan ASP.NET Core 20](#_Toc4181)

[Hình 2.2 . Giới thiệu ABP Framework 24](#_Toc25738)

[Hình 2.3 . Giới thiệu Angular 25](#_Toc19610)

[Hình 2.4 . Giới thiệu SQL Server 27](#_Toc30051)

[Hình 2.5 . Giới thiệu PrimeNG 29](#_Toc18233)

[Hình 2.6 . Cấu trúc mô hình Clean Architecture 34](#_Toc1310)

[Hình 3.1 . Use case tổng quan 40](#_Toc713)

[Hình 3.2 . Use case quản trị viên 41](#_Toc4113)

[Hình 3.3 . Use case khách hàng 41](#_Toc27757)

[Hình 3.4 . Biểu đồ trình tự Use case Đăng Nhập 53](#_Toc27557)

[Hình 3.5 . Biểu đồ lớp Use case Đăng Nhập 54](#_Toc31982)

[Hình 3.6 . Biểu đồ trình tự Use case Đăng Ký 54](#_Toc4103)

[Hình 3.7 . Biểu đồ lớp Use case Đăng Ký 55](#_Toc28685)

[Hình 3.8 . Biểu đồ trình tự Use case Quản lý đơn hàng 55](#_Toc10784)

[Hình 3.9 . Biểu đồ lớp Quản lý đơn hàng 56](#_Toc1559)

[Hình 3.10 . Biểu đồ trình tự Quản lý tài khoản 57](#_Toc2504)

[Hình 3.11 . Biểu đồ lớp Quản lý tài khoản 57](#_Toc953)

[Hình 3.12 . Biểu đồ trình tự Quản lý sản phẩm 58](#_Toc16438)

[Hình 3.13 . Biểu đồ lớp Quản lý sản phẩm 59](#_Toc25622)

[Hình 3.14 . Biểu đồ trình tự Use case Thêm sản phẩm vào giỏ hàng 59](#_Toc699)

[Hình 3.15 . Biểu đồ lớp Use case Thêm sản phẩm vào giỏ hàng 60](#_Toc28951)

[Hình 3.16 . Biểu đồ trình tự Use case Tìm kiếm 60](#_Toc9649)

[Hình 3.17 . Biểu đồ lớp Use case Tìm kiếm 61](#_Toc21817)

[Hình 3.18 . Biểu đồ trình tự Use case Xem chi tiết sản phẩm 61](#_Toc5624)

[Hình 3.19 . Biểu dồ lớp Use case Xem chi tiết sản phẩm 62](#_Toc12057)

[Hình 3.20 . Biểu đồ thực thể liên kết 63](#_Toc6523)

[Hình 3.21 . Giao diện trang thống kê 71](#_Toc22352)

[Hình 3.22 . Giao diện trang quản lý sản phẩm 71](#_Toc29160)

[Hình 3.23 . Giao diện trang quản lý nhà sản xuất 72](#_Toc32436)

[Hình 3.24 . Giao diện trang quản lý loại sản phẩm 72](#_Toc11631)

[Hình 3.25 . Giao diện trang quản lý thuộc tính 72](#_Toc12439)

[Hình 3.26 . Giao diện trang quản lý tài khoản 72](#_Toc6807)

[Hình 3.27 . Giao diện trang quản lý mã giảm giá 73](#_Toc23885)

[Hình 3.28 . Giao diện trang quản lý đơn hàng 73](#_Toc18141)

[Hình 3.29 . Giao diện trang chủ 73](#_Toc28136)

[Hình 3.30 . Giao diện xem chi tiết sản phẩm 74](#_Toc20637)

[Hình 3.31 . Giao diện trang Login 74](#_Toc17734)

[Hình 3.32 . Giao diện Giỏ hàng 75](#_Toc4150)

[Hình 3.33 . Giao diện thanh toán 75](#_Toc21070)

[Hình 3.34 . Giao diện xem đơn hàng 75](#_Toc23658)

**LỜI CẢM ƠN**

Em xin chân thành cảm ơn khoa khoa Công Nghệ Thông Tin cùng tất cả các thầy giáo, cô giáo đã tận tình giảng dạy và giúp đỡ em trong suốt quá trình học tập, nghiên cứu.

Đặc biệt em xin bày tỏ lòng biết ơn sâu sắc đến thầy Vũ Duy Giang – Người đã trực tiếp giảng dạy và hướng dẫn tạo mọi điều kiện thuận lợi giúp đỡ em trong quá trình thực hiện đề tài. Thầy đã giúp em rất nhiều trong việc chọn đề tài và các công cụ thực hiện. Những vốn kiến thức được tiếp thu trong quá trình hướng dẫn và giúp đỡ của thầy không chỉ là nền tảng cho quá trình nghiên cứu khóa luận mà còn là hành trang quý báu để em bước vào đời một cách vững vàng và tự tin hơn.

Tuy nhiên vì kiến thức chuyên môn còn hạn chế và bản thân còn thiếu nhiều kinh nghiệm thực tiễn nên nội dung của báo cáo không tránh khỏi những thiếu sót, em rất mong nhận sự góp ý, chỉ bảo thêm của quý thầy cô hướng dẫn và thầy cô phản biện để báo cáo này được hoàn thiện hơn.

**MỞ ĐẦU**

1. **Giới thiệu**
   1. **Lý do chọn đề tài**

Em đã chọn đề tài phát triển một trang web bán nem, đặc sản của quê hương với những lý do rất đặc biệt và ý nghĩa sâu sắc. Trước hết, việc này không chỉ là về kinh doanh mà còn về việc gìn giữ và phát triển văn hóa quê hương mình. Nem, không chỉ là một phần của ẩm thực mà còn là một biểu tượng của sự đa dạng và sự đặc biệt trong văn hóa ẩm thực của quê hương em.

Bằng cách tạo ra một trang web chuyên biệt cho nem đặc sản, em muốn truyền tải thông điệp về giá trị và ý nghĩa của nem trong việc kết nối con người với quê hương, với những giá trị văn hóa và truyền thống mà nem mang lại. Qua trang web này, em mong muốn tạo ra một không gian kỷ niệm, nơi mà mỗi người có thể tìm thấy hương vị quê hương, nhớ về những kỷ niệm đẹp và tạo ra những trải nghiệm ẩm thực đặc biệt.

Ngoài ra, việc phát triển một trang web bán nem cũng mang lại cơ hội kinh doanh và tiếp cận thị trường mở rộng. Nem, với hương vị độc đáo và sự phong phú trong hương liệu, có thể thu hút không chỉ người dân địa phương mà còn những người yêu thích ẩm thực truyền thống và đang tìm kiếm trải nghiệm mới lạ.

Tuy nhiên, không chỉ dừng lại ở việc kinh doanh, em còn muốn tạo ra một kết nối chặt chẽ hơn giữa người sản xuất và người tiêu dùng. Em sẽ đưa ra thông tin về nguồn gốc, quy trình sản xuất và các thông tin liên quan khác, giúp người tiêu dùng có cái nhìn rõ ràng hơn về sản phẩm mình đang sử dụng. Điều này không chỉ tạo ra sự tin tưởng và uy tín mà còn thúc đẩy sự tương tác và giao lưu giữa cộng đồng người sản xuất và người tiêu dùng.

Cuối cùng, thông qua trang web này, em hy vọng tạo ra một không gian giao lưu và chia sẻ giữa những người yêu thích nem, từ việc chia sẻ công thức, kỹ thuật chế biến đến việc gợi ý về cách thưởng thức và kết hợp nem với các món ăn khác. Điều này sẽ giúp tạo ra một cộng đồng sôi động và giàu tính tương tác, góp phần vào việc duy trì và phát triển văn hóa ẩm thực đặc biệt của quê hương. Đó là lý do tại sao em đã chọn đề tài này, với niềm đam mê và sứ mệnh về văn hóa và kinh doanh.

* 1. **Tính cấp thiết của đề tài**

Việc phát triển một trang web bán nem đặc sản của quê hương không chỉ là một ý tưởng kinh doanh mà còn mang lại những lợi ích văn hóa, kinh tế đáng kể. Trong thời đại hiện đại, việc bảo tồn và phát triển di sản văn hóa địa phương là một nhiệm vụ cấp thiết. Nem đặc sản của quê hương không chỉ là một phần của ẩm thực mà còn là biểu tượng của sự đa dạng và sự đặc biệt trong văn hóa ẩm thực. Tuy nhiên, những giá trị này đang dần mất đi trong bối cảnh thị trường ẩm thực đang thay đổi nhanh chóng và sự xuất hiện của các sản phẩm thương mại hóa.

Việc phát triển một trang web chuyên biệt để bán nem không chỉ giúp bảo tồn mà còn thúc đẩy sự phát triển của các loại nem truyền thống, giúp giới trẻ hiểu và yêu quý những giá trị văn hóa này. Đồng thời, trang web cung cấp một nền tảng để người sản xuất và nhà làm nem tiếp cận một thị trường rộng lớn hơn, giúp tăng cơ hội kinh doanh và thu nhập cho họ.

Thúc đẩy mua sắm trực tuyến cũng là một yếu tố quan trọng. Trong bối cảnh cuộc sống hối hả, việc có thể mua nem đặc sản từ quê hương chỉ với một vài thao tác trên internet là điều mà nhiều người tiêu dùng đang tìm kiếm. Điều này không chỉ đáp ứng nhu cầu của họ mà còn giúp tạo ra một trải nghiệm mua sắm thuận tiện và độc đáo.

Cuối cùng, việc phát triển một trang web bán nem cũng mang lại lợi ích kinh tế địa phương. Bằng cách tạo ra một kênh tiếp thị và bán hàng trực tuyến cho nem đặc sản, chúng ta có thể giúp tăng cường kinh tế địa phương, tạo ra thu nhập cho cộng đồng và giúp duy trì sự phát triển bền vững. Với những lợi ích văn hóa, kinh tế và xã hội mà nó mang lại, việc phát triển một trang web bán nem đặc sản của quê hương không chỉ là cần thiết mà còn là một bước quan trọng trong việc bảo tồn và phát triển di sản văn hóa của quê hương.

1. **Mục tiêu và phương pháp nghiên cứu**
   1. **Mục tiêu của đề tài**

Tìm hiểu về công nghệ ASP.Net Core và Angular. Xây dựng website dạng modern web. Với backend là các service xây dựng theo Restful API.

Dự án nhằm tạo ra một nền tảng trực tuyến để các cửa hàng kinh doanh nem có thể quản lý bán hàng, lưu trữ dữ liệu và thực hiện các tác vụ quản lý và chăm sóc khách hàng một cách hiệu quả.

Trình bày báo cáo khoa học và hợp lý với nội dung trong đề tài khoá luận tốt nghiệp.

* 1. **Nội dung nghiên cứu**

- Tổng quan về Angular và Web API ASP.NET Core.

- Tìm hiểu về ABP FrameWork.

- Phân tích yêu cầu và thiết kế hệ thống.

- Triển khai và phát triển phần mềm.

* 1. **Phương pháp luận và phương pháp nghiên cứu**

**2.3.1. Phương pháp luận**

- Nghiên cứu thị trường và phân tích yêu cầu:

+ Tiến hành nghiên cứu thị trường để hiểu về xu hướng mua sắm trực tuyến và nhu cầu của khách hàng về nem đặc sản.

+ Phân tích yêu cầu chức năng và phi chức năng của trang web .

- Thiết kế kiến trúc hệ thống:

+ Xác định kiến trúc tổng quan của hệ thống là Microservices, bao gồm phân chia thành phần front-end và back-end.

+ Thiết kế cấu trúc dữ liệu cho cả phần front-end (Angular cho Admin App và Razor Pages cho Public App) và back-end (Web API ASP.NET Core).

- Triển khai giao diện người dùng (UI/UX):

+ Phát triển giao diện người dùng cho Admin App sử dụng Angular, bao gồm thiết kế trang thống kê, trang quản lý sản phẩm, quản lý đơn hàng, quản lý thuộc tính, ….

+ Sử dụng Razor Pages để phát triển giao diện người dùng cho Public App.

- Triển khai back-end (API):

+ Xây dựng các endpoint của Web API ASP.NET Core để cung cấp dữ liệu cho giao diện người dùng ở Admin App, bao gồm các chức năng quản lý sản phẩm, đơn hàng, ….

- Kiểm thử và debug:

+ Tiến hành kiểm thử chức năng và hiệu suất cho cả phần front-end và back-end của hệ thống.

+ Debug lỗi và sửa chữa các vấn đề phát sinh trong quá trình triển khai.

- Triển khai và đánh giá:

+ Triển khai trang web trên môi trường sản phẩm thực tế và đảm bảo sự ổn định và khả dụng của hệ thống.

+ Đánh giá hiệu suất của hệ thống dựa trên các tiêu chí như tốc độ tải trang, thời gian phản hồi và độ tin cậy của các tính năng.

- Tổng kết và tài liệu hóa:

+ Tổng kết kết quả phát triển và đánh giá mức độ đạt được các mục tiêu đã đề ra.

+ Tài liệu hóa quy trình phát triển, bao gồm tài liệu hướng dẫn sử dụng, mã nguồn và các tài liệu kỹ thuật.

**2.3.2. Phương pháp nghiên cứu**

- Nghiên cứu thị trường và phân tích yêu cầu:

+ Tiến hành khảo sát thị trường để hiểu về nhu cầu và xu hướng mua sắm nem đặc sản trực tuyến.

+ Phân tích yêu cầu chức năng và phi chức năng của trang web bán nem, xác định các tính năng quan trọng như quản lý sản phẩm, đơn hàng, khách hàng.

- Nghiên cứu công nghệ:

+Nghiên cứu về Angular và Web API ASP.NET Core, bao gồm các tính năng, ưu điểm, hạn chế và cách thức triển khai.

+So sánh và lựa chọn công nghệ phù hợp cho dự án, dựa trên yêu cầu chức năng và mục tiêu đề ra.

- Thiết kế kiến trúc hệ thống:

+ Sử dụng cấu trúc ứng dụng là Microservices.

+ Xác định kiến trúc tổng quan của hệ thống, bao gồm cấu trúc front-end và back-end.

+ Sử dụng Angular để làm front-end cho phần giao diện của Admin App.

+ Sử dụng Razor Pages làm front-end cho phần giao diện của Public App.

- Triển khai và phát triển phần mềm:

+ Phát triển giao diện người dùng phía Admin sử dụng Angular, bao gồm các trang chính như thống kê, trang quản lý sản phẩm,….

+ Xây dựng Web API ASP.NET Core để cung cấp dữ liệu cho giao diện người dùng trang Admin.

- Tích hợp và kiểm thử:

+ Tích hợp các thành phần của hệ thống và kiểm tra tính tương thích và ổn định.

+ Tiến hành kiểm thử chức năng và hiệu suất cho cả phần front-end và back-end của trang web.

- Triển khai và đánh giá:

+ Triển khai trang web trên môi trường sản phẩm thực tế và đảm bảo sự ổn định và khả dụng của hệ thống.

+ Đánh giá hiệu suất của hệ thống dựa trên các tiêu chí như tốc độ tải trang, thời gian phản hồi và độ tin cậy của các tính năng.

- Tổng kết và đánh giá kết quả:

+ Tổng kết quá trình phát triển dự án và đánh giá mức độ đạt được các mục tiêu đã đề ra.

+ Đề xuất các cải tiến và hướng phát triển tương lai cho hệ thống.

# CHƯƠNG 1: TỔNG QUAN VỀ WEBSITE BÁN NEM

* 1. **Mục tiêu của website**
* Bán hàng trực tuyến: Tạo nền tảng để khách hàng có thể mua sản phẩm trực tuyến một cách thuận tiện.
* Cung cấp thông tin: Cung cấp thông tin chi tiết về các sản phẩm nem truyền thống.
* Đưa món nem truyền thống tiếp cận được với nhiều người hơn trên mọi miền Tổ Quốc.
* Hỗ trợ khách hàng: Cung cấp dịch vụ hỗ trợ và chăm sóc khách hàng.
  1. **Nghiên cứu thị trường**

**1.2.1. Phân tích thị trường nem tại Việt Nam**

Thị trường nem truyền thống tại Việt Nam hiện nay đang chứng kiến sự bùng nổ về nhu cầu và đa dạng hóa sản phẩm. Các sản phẩm nem như nem chua, nem bùi, thịt chua,… đang trở thành những mặt hàng thiết yếu trong các bữa ăn hàng ngày và các dịp lễ tết. Sự đổi mới liên tục và cạnh tranh khốc liệt trong ngành công nghiệp thực phẩm thúc đẩy các doanh nghiệp không ngừng cải tiến và cung cấp các sản phẩm với chất lượng cao và hương vị đặc trưng.

Ngoài ra, xu hướng tiêu dùng của người dân cũng thay đổi mạnh mẽ khi họ có xu hướng tìm kiếm những sản phẩm nem có nguồn gốc rõ ràng, đảm bảo vệ sinh an toàn thực phẩm, và giá cả hợp lý.

Các cơ sở sản xuất nem từ quy mô nhỏ đến lớn đều đang góp phần vào nguồn cung cấp phong phú. Các thương hiệu nem nổi tiếng như nem chua Thanh Hóa, nem Ninh Hòa, nem Lai Vung luôn có uy tín và thị phần lớn trong lòng người tiêu dùng. Bên cạnh đó, sự hỗ trợ từ các chính sách phát triển nông nghiệp và làng nghề truyền thống cũng tạo điều kiện thuận lợi cho sự phát triển của thị trường nem.

Sự đa dạng hóa sản phẩm và nhu cầu ngày càng tăng đã và đang thúc đẩy các doanh nghiệp trong ngành phải nỗ lực không ngừng để đáp ứng kỳ vọng của người tiêu dùng, đồng thời đảm bảo chất lượng sản phẩm và tuân thủ các quy định về an toàn thực phẩm. Trong bối cảnh thị trường ngày càng phát triển, các doanh nghiệp sản xuất nem cần tiếp tục đổi mới, nâng cao chất lượng và tạo ra những sản phẩm mới để duy trì và mở rộng thị phần.

**1.2.2. Đối tượng khách hàng**

Đối tượng khách hàng của website bán nem truyền thống rất đa dạng, bao gồm:

- Người tiêu dùng cá nhân: Những người tìm kiếm các sản phẩm nem để phục vụ cho bữa ăn gia đình hàng ngày, các bữa tiệc, hoặc các dịp lễ tết. Họ quan tâm đến hương vị truyền thống, chất lượng sản phẩm và sự tiện lợi khi mua sắm trực tuyến.

- Nhà hàng và quán ăn: Các nhà hàng, quán ăn có nhu cầu mua sắm nem số lượng lớn để phục vụ thực khách. Họ cần các sản phẩm nem chất lượng cao, đảm bảo an toàn thực phẩm và có nguồn cung ổn định.

- Nhóm tuổi: Khách hàng ở nhiều độ tuổi khác nhau, từ các gia đình, người lớn tuổi yêu thích hương vị truyền thống đến những người trẻ tuổi tìm kiếm các món ăn đặc sản và tiện lợi.

- Sở thích và hành vi: Những người yêu thích ẩm thực truyền thống, đam mê các món ăn đặc sản của Việt Nam. Họ thường xuyên tìm kiếm các sản phẩm mới, đa dạng hóa khẩu vị và có xu hướng mua sắm trực tuyến để tiết kiệm thời gian.

Website bán nem truyền thống cần đáp ứng nhu cầu đa dạng của khách hàng, cung cấp thông tin chi tiết về sản phẩm, đảm bảo chất lượng và vệ sinh an toàn thực phẩm, cũng như tạo điều kiện mua sắm thuận tiện với dịch vụ giao hàng nhanh chóng và chính sách chăm sóc khách hàng tốt.

* 1. **Lợi ích của việc xây dựng website**

**1.3.1. Lợi ích cho doanh nghiệp**

Lợi ích cho doanh nghiệp khi làm website bán nem truyền thống bao gồm:

- Mở rộng thị trường: Tiếp cận được nhiều khách hàng hơn, không bị giới hạn bởi địa lý. Website cho phép doanh nghiệp bán hàng không chỉ trong nước mà còn có thể xuất khẩu nem truyền thống ra quốc tế.

- Tăng cường thương hiệu: Xây dựng và phát triển thương hiệu trực tuyến, tạo sự nhận diện và tin tưởng từ khách hàng. Một website chuyên nghiệp giúp nâng cao uy tín của doanh nghiệp và tạo ấn tượng tích cực đối với người tiêu dùng.

- Tiết kiệm chi phí: Giảm chi phí vận hành so với cửa hàng truyền thống, tối ưu hóa quá trình quản lý và bán hàng. Không cần chi trả cho mặt bằng và các chi phí liên quan, doanh nghiệp có thể tập trung vào việc cải thiện chất lượng sản phẩm và dịch vụ.

- Cải thiện dịch vụ khách hàng: Website cho phép doanh nghiệp cung cấp thông tin chi tiết về sản phẩm, hướng dẫn sử dụng và chăm sóc khách hàng trực tuyến. Điều này giúp nâng cao trải nghiệm mua sắm và xây dựng mối quan hệ lâu dài với khách hàng.

- Tăng doanh thu: Khả năng tiếp cận một lượng lớn khách hàng trực tuyến và cung cấp các chương trình khuyến mãi, giảm giá giúp doanh nghiệp tăng doanh số bán hàng và lợi nhuận.

- Linh hoạt trong kinh doanh: Doanh nghiệp có thể dễ dàng cập nhật thông tin sản phẩm, giá cả và các chương trình khuyến mãi một cách nhanh chóng và hiệu quả, giúp tăng cường tính cạnh tranh trên thị trường.

**1.3.2. Lợi ích cho khách hàng**

Lợi ích cho khách hàng khi mua sắm trên website bán nem truyền thống bao gồm:

- Tiện lợi: Khách hàng có thể mua sắm bất kỳ lúc nào, ở bất kỳ đâu chỉ cần có kết nối internet. Điều này giúp tiết kiệm thời gian và công sức so với việc phải đến cửa hàng trực tiếp.

- Đa dạng sản phẩm: Khách hàng dễ dàng so sánh và lựa chọn sản phẩm từ nhiều nhà cung cấp khác nhau. Website thường cung cấp nhiều loại nem với thông tin chi tiết về sản phẩm, giúp khách hàng có nhiều sự lựa chọn phù hợp với nhu cầu của mình.

- Dịch vụ hỗ trợ: Khách hàng hưởng lợi từ các dịch vụ hỗ trợ khách hàng trực tuyến, bao gồm tư vấn sản phẩm, giải đáp thắc mắc, và hướng dẫn sử dụng. Các chương trình khuyến mãi và ưu đãi thường xuyên trên website giúp khách hàng tiết kiệm chi phí mua sắm.

- Thông tin minh bạch: Website cung cấp thông tin chi tiết về sản phẩm, nguồn gốc, thành phần và cách bảo quản, giúp khách hàng yên tâm về chất lượng và an toàn thực phẩm.

- Giao hàng tận nơi: Dịch vụ giao hàng nhanh chóng và tiện lợi giúp khách hàng nhận được sản phẩm ngay tại nhà mà không cần phải di chuyển, đặc biệt hữu ích trong các dịp lễ tết hoặc khi cần gấp.

Những lợi ích này giúp nâng cao trải nghiệm mua sắm của khách hàng, đồng thời tạo động lực để họ quay lại mua sắm trên website bán nem truyền thống.

* 1. **Mô tả website**

Website bán nem truyền thống của chúng em mang đến trải nghiệm mua sắm tiện lợi và phong phú cho khách hàng. Với giao diện thân thiện và dễ sử dụng, khách hàng có thể dễ dàng tìm kiếm và lựa chọn từ một loạt các sản phẩm nem đa dạng. Mỗi sản phẩm đều được cung cấp thông tin chi tiết về nguồn gốc, thành phần và hướng dẫn sử dụng, đảm bảo sự minh bạch và an toàn thực phẩm. Dịch vụ giao hàng nhanh chóng và linh hoạt, giúp khách hàng nhận được sản phẩm tươi ngon ngay tại nhà. Với những tính năng trên, website của chúng em mang đến sự tiện lợi tối đa cho mọi khách hàng, từ cá nhân đến nhà hàng và quán ăn.

Một số chức năng của website:

- Admin:

+ Thống kê

+ Quản lý sản phẩm

+ Quản lý mã giảm giá

+ Quản lý tài khoản người dùng.

- Khách hàng:

+ Xem chi tiết sản phẩm

+ Thêm vào giỏ hàng

+ Đặt hàng

+ Xem đơn hàng

# CHƯƠNG 2: CÔNG NGHỆ ASP.NET CORE VÀ ANGULAR

**2.1. Giới thiệu về ASP.NET Core**

**2.1.1. Khái niệm ASP.NET Core**

ASP.NET Core là một framework mã nguồn mở, đa nền tảng hiện đại được sử dụng để xây dựng các ứng dụng kết nối đám mây, bao gồm web apps, IoT, và backend cho di động. Ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc phiên bản đầy đủ của .NET Framework, mang lại sự linh hoạt tối đa cho việc triển khai trên đám mây hoặc tại chỗ.

Với thiết kế module hóa, ASP.NET Core tối ưu hóa tài nguyên và chi phí phát triển, cho phép bạn dễ dàng điều chỉnh và mở rộng giải pháp của mình. Các ứng dụng ASP.NET Core có thể phát triển và chạy trên nhiều nền tảng như Windows, Mac, và Linux.

Điểm nổi bật nhất của ASP.NET Core là việc chuyển sang mã nguồn mở, một bước tiến lớn so với các phiên bản trước đây. Điều này không chỉ thúc đẩy tính minh bạch và sự đóng góp từ cộng đồng mà còn phản ánh xu hướng phát triển chung của các ngôn ngữ lập trình hiện nay. Việc trở thành mã nguồn mở giúp ASP.NET Core thu hút nhiều lập trình viên và doanh nghiệp, đẩy mạnh sự phát triển và đổi mới trong lĩnh vực này.

Graphical user interface

Description automatically generated

**Hình 2.1****. Tổng quan ASP.NET Core**

**2.1.2. Hoàn cảnh ra đời ASP.NET Core**

ASP.NET Core là phiên bản mới của ASP.NET, được phát triển bởi Microsoft và phát hành dưới dạng mã nguồn mở trên GitHub. ASP.NET Core được sử dụng để phát triển các ứng dụng web và có thể tương thích với nhiều hệ điều hành khác nhau như Windows, Mac, và Linux, kể cả trên nền tảng MVC. Ban đầu được gọi là ASP.NET 5, phiên bản này sau đó đã được đổi tên thành ASP.NET Core.

ASP.NET Core được thiết kế để tối ưu hóa framework phát triển cho các ứng dụng có thể chạy tại chỗ hoặc triển khai trên đám mây. Nó bao gồm các thành phần theo hướng module nhằm giảm thiểu tài nguyên và tiết kiệm chi phí phát triển. Việc ASP.NET Core là mã nguồn mở cũng phản ánh xu hướng hiện tại của các ngôn ngữ lập trình, tập trung vào sự minh bạch và đóng góp từ cộng đồng.

Sử dụng ASP.NET Core, chúng ta có thể tận dụng các tính năng sau:

* Hợp nhất việc xây dựng giao diện web (UI) và web APIs.
* Tích hợp các framework phía client hiện đại và các quy trình phát triển.
* Hệ thống cấu hình dựa trên môi trường đám mây.
* Dependency injection được tích hợp sẵn.
* HTTP request được tối ưu nhẹ hơn.
* Có thể host trên IIS hoặc tự host trong process của riêng chúng ta.
* Xây dựng trên .NET Core, hỗ trợ quản lý phiên bản ứng dụng.
* Chuyển các thực thể, thành phần, module thành các gói NuGet.
* Các công cụ mới để đơn giản hóa quá trình phát triển web hiện đại.
* Khả năng chạy đa nền tảng (Windows, Mac, và Linux).
* Mã nguồn mở và tập trung vào cộng đồng.

ASP.NET Core là một bước tiến lớn trong việc phát triển ứng dụng web hiện đại, mang lại sự linh hoạt và hiệu quả cao cho các nhà phát triển.

**2.1.3. Lý do ASP.NET Core phổ biến**

ASP.NET Core là một framework mã nguồn mở, đa nền tảng được phát triển bởi Microsoft, hỗ trợ phát triển trên nhiều hệ điều hành như Windows, macOS, và Linux. Điều này cho phép các nhà phát triển lựa chọn môi trường làm việc ưa thích của mình mà không gặp phải các ràng buộc về hệ điều hành. Sự linh hoạt này không chỉ tăng cường sự tiện lợi mà còn mở rộng phạm vi của các dự án, cho phép nhóm phát triển làm việc đồng bộ trên nhiều hệ thống khác nhau.

ASP.NET Core được tối ưu hóa để có hiệu suất vượt trội so với các phiên bản trước của ASP.NET. Nó cung cấp khả năng xử lý đồng thời nhiều yêu cầu (request) và phản hồi (response) một cách hiệu quả, giúp ứng dụng hoạt động nhanh chóng và mượt mà. Hệ thống tối ưu hóa bộ nhớ và quản lý luồng dữ liệu thông minh giúp giảm thời gian phản hồi và tăng cường khả năng xử lý tải cao, làm cho ASP.NET Core trở thành lựa chọn lý tưởng cho các ứng dụng yêu cầu hiệu suất cao.

Với kiến trúc modular, ASP.NET Core cho phép các nhà phát triển lựa chọn và tích hợp chỉ những thành phần cần thiết cho ứng dụng của họ. Điều này không chỉ giúp giảm kích thước của ứng dụng mà còn tối ưu hóa việc sử dụng tài nguyên hệ thống. Các module có thể được thêm vào hoặc loại bỏ một cách linh hoạt, giúp dễ dàng mở rộng hoặc thu hẹp phạm vi ứng dụng theo yêu cầu cụ thể của dự án.

ASP.NET Core liên tục cập nhật và tích hợp các công nghệ mới như Docker, Kubernetes, và các công nghệ containerization khác. Điều này giúp đơn giản hóa việc triển khai và quản lý ứng dụng trên các môi trường đám mây (cloud) và tại chỗ (on-premises). Khả năng tích hợp với các công nghệ tiên tiến này không chỉ giúp tiết kiệm thời gian và công sức trong quá trình triển khai mà còn đảm bảo rằng ứng dụng của bạn luôn ở trạng thái sẵn sàng và linh hoạt trước các thay đổi trong công nghệ.

ASP.NET Core cung cấp một framework mạnh mẽ để phát triển các dịch vụ web RESTful thông qua ASP.NET Core Web API. Điều này giúp tạo ra các ứng dụng web API linh hoạt, dễ dàng tích hợp với các ứng dụng khác. Khả năng hỗ trợ JSON và XML cùng với các công cụ xây dựng API mạnh mẽ giúp giảm bớt sự phức tạp trong việc phát triển và duy trì các dịch vụ web, đồng thời đảm bảo tính nhất quán và dễ dàng mở rộng.

ASP.NET Core đi kèm với các tính năng bảo mật mạnh mẽ như xác thực (authentication) và ủy quyền (authorization), bảo vệ chống lại các cuộc tấn công phổ biến như CSRF (Cross-Site Request Forgery) và XSS (Cross-Site Scripting). Các tính năng này giúp bảo vệ ứng dụng khỏi các lỗ hổng bảo mật, đảm bảo rằng dữ liệu và hệ thống của bạn luôn an toàn. Ngoài ra, ASP.NET Core cung cấp các cơ chế mã hóa và bảo mật dữ liệu tiên tiến, giúp đảm bảo rằng thông tin nhạy cảm luôn được bảo vệ ở mức cao nhất.

ASP.NET Core không chỉ là một bước tiến lớn trong việc phát triển ứng dụng web hiện đại mà còn mang lại sự linh hoạt và hiệu quả cao cho các nhà phát triển. Với những ưu điểm vượt trội về hiệu suất, tính năng bảo mật, và khả năng hỗ trợ các công nghệ mới, ASP.NET Core xứng đáng là lựa chọn hàng đầu cho các dự án phát triển ứng dụng web và dịch vụ web hiện nay.

**2.2. Giới thiệu về ABP Framework**

**2.2.1. Khái niệm ABP Framework**

ABP Framework là một nền tảng phát triển ứng dụng mạnh mẽ và linh hoạt, được thiết kế để xây dựng các ứng dụng doanh nghiệp hiện đại với cấu trúc tốt và hiệu quả. ABP Framework cung cấp một loạt các tính năng và công cụ hỗ trợ lập trình viên, bao gồm quản lý người dùng, phân quyền, kiểm tra, và xử lý ngoại lệ, giúp tiết kiệm thời gian và công sức trong quá trình phát triển phần mềm. Nền tảng này hỗ trợ cả ứng dụng Web và ứng dụng di động, đồng thời tích hợp tốt với các công nghệ phổ biến như ASP.NET Core, Entity Framework Core, và Angular. ABP Framework tuân theo các nguyên tắc thiết kế hiện đại như Dependency Injection, Unit of Work, và Repository Pattern, đảm bảo rằng các ứng dụng được xây dựng có khả năng mở rộng, bảo trì dễ dàng và hiệu suất cao. Với tài liệu chi tiết và cộng đồng hỗ trợ nhiệt tình, ABP Framework là lựa chọn lý tưởng cho các dự án phát triển ứng dụng quy mô lớn và phức tạp.



**Hình 2.2****. Giới thiệu ABP Framework**

**2.2.2. Ưu điểm cúa ABP Framework**

ABP Framework mang lại nhiều ưu điểm vượt trội cho quá trình phát triển ứng dụng.

Thứ nhất, nền tảng này được thiết kế với cấu trúc dự án chuẩn hóa và tổ chức rõ ràng, giúp lập trình viên dễ dàng quản lý mã nguồn, đặc biệt trong các dự án lớn.

Thứ hai, ABP Framework tiết kiệm thời gian và công sức nhờ các tính năng và công cụ có sẵn như quản lý người dùng, phân quyền, xử lý ngoại lệ và kiểm tra, cho phép lập trình viên tập trung vào logic kinh doanh.

Thứ ba, nền tảng này tương thích tốt với các công nghệ hiện đại như ASP.NET Core, Entity Framework Core, và Angular, đảm bảo ứng dụng được xây dựng trên nền tảng vững chắc.

Thứ tư, ABP Framework có tài liệu chi tiết và cộng đồng hỗ trợ mạnh mẽ, giúp lập trình viên dễ dàng nắm bắt và sử dụng.

Thứ năm, ABP Framework tích hợp sẵn các công cụ kiểm tra và bảo mật, đảm bảo ứng dụng phát triển an toàn và tin cậy.

Cuối cùng, khả năng mở rộng cao, tích hợp dễ dàng với các hệ thống khác và cho phép tùy chỉnh theo nhu cầu cụ thể của từng dự án, giúp ABP Framework trở thành lựa chọn lý tưởng cho việc phát triển các ứng dụng doanh nghiệp hiện đại, đảm bảo chất lượng và hiệu suất cao.

**2.3. Giới thiệu về Angular**

**2.3.1. Khái niệm Angular**

Angular là một framework mã nguồn mở được phát triển bởi Google, chuyên dùng để xây dựng các ứng dụng web động và phức tạp. Ra mắt lần đầu vào năm 2010, Angular đã trải qua nhiều phiên bản nâng cấp, với phiên bản hiện tại thường được gọi là Angular (không kèm theo số phiên bản) để phân biệt với AngularJS, phiên bản gốc. Angular sử dụng TypeScript, một phiên bản nâng cao của JavaScript, giúp tăng cường tính bảo mật và khả năng bảo trì của mã nguồn. Với kiến trúc dựa trên các thành phần (components), Angular cho phép tái sử dụng mã nguồn và dễ dàng quản lý các phần của ứng dụng. Framework này cũng tích hợp các công cụ và thư viện mạnh mẽ, bao gồm Angular CLI, giúp đơn giản hóa quy trình phát triển, kiểm thử, và triển khai ứng dụng. Nhờ khả năng tạo ra các ứng dụng web nhanh chóng, linh hoạt và dễ dàng mở rộng, Angular đã trở thành một lựa chọn phổ biến cho các nhà phát triển trên toàn thế giới.



**Hình 2.3****. Giới thiệu Angular**

**2.3.2. Ưu điểm của Angular**

Angular là một framework mạnh mẽ và linh hoạt, cho phép tạo ra các ứng dụng web phức tạp và động với cấu trúc rõ ràng và dễ mở rộng. Sử dụng kiến trúc dựa trên các thành phần (components), Angular giúp mã nguồn được tổ chức tốt, dễ bảo trì và tái sử dụng. Hỗ trợ hai chiều dữ liệu (two-way data binding) giúp đồng bộ hóa dữ liệu giữa mô hình và giao diện người dùng một cách hiệu quả.

Mặc dù có độ phức tạp cao hơn một số framework khác, Angular với sự hỗ trợ mạnh mẽ từ Angular CLI và tài liệu phong phú, giúp việc học và sử dụng trở nên dễ dàng hơn, đặc biệt đối với những người đã quen thuộc với TypeScript và JavaScript.

Angular được tối ưu hóa để đạt hiệu suất cao nhờ cơ chế "change detection" và "ahead-of-time (AOT) compilation", giúp giảm thiểu thời gian render lại DOM và tăng tốc độ xử lý dữ liệu. Ngoài ra, Angular sử dụng "lazy loading" để chỉ tải các module cần thiết, giúp tăng tốc độ tải trang và cải thiện hiệu suất tổng thể của ứng dụng.

Khả năng tích hợp của Angular vào các dự án hiện có hoặc mới xây dựng là rất dễ dàng, tương thích tốt với các công cụ và thư viện như RxJS để xử lý các thao tác không đồng bộ và NGRX để quản lý trạng thái ứng dụng.

Angular cũng cung cấp nhiều tính năng bảo mật mạnh mẽ như kiểm tra đầu vào, bảo vệ chống lại các lỗ hổng XSS (Cross-Site Scripting), và hỗ trợ xác thực (authentication) và ủy quyền (authorization), đảm bảo rằng dữ liệu luôn được an toàn.

Cộng đồng lớn và tích cực của Angular cung cấp vô số tài liệu, hướng dẫn và các dự án mẫu để hỗ trợ các nhà phát triển. Các diễn đàn, nhóm thảo luận và sự kiện liên quan đến Angular diễn ra thường xuyên, tạo điều kiện cho việc học hỏi và chia sẻ kiến thức.

Với những ưu điểm này, Angular đã chứng tỏ mình là một trong những framework mạnh mẽ và linh hoạt nhất để phát triển các ứng dụng web hiện đại.

**2.4. Giới thiệu về SQL Server**

**2.4.1. Khái niệm SQL Server**

SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển. Đây là một phần mềm hệ thống quản lý cơ sở dữ liệu dùng để lưu trữ và truy vấn dữ liệu theo mô hình quan hệ, cho phép quản lý và xử lý dữ liệu một cách hiệu quả. SQL Server sử dụng ngôn ngữ truy vấn cấu trúc (SQL) để thực hiện các thao tác trên cơ sở dữ liệu như thêm, sửa, xóa và truy vấn dữ liệu.



**Hình 2.4****. Giới thiệu SQL Server**

**2.4.2. Ưu điểm của SQL Server**

SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) nổi bật với nhiều ưu điểm vượt trội.

Đầu tiên, SQL Server cung cấp các phiên bản miễn phí như SQL Server Developer và SQL Server Express, cho phép người dùng trải nghiệm đầy đủ các tính năng mà không phải trả phí. Phiên bản Developer cung cấp tất cả các tính năng của phiên bản Enterprise, phù hợp cho phát triển và thử nghiệm, trong khi phiên bản Express là một phiên bản nhẹ hơn, phù hợp cho các ứng dụng nhỏ và người dùng cá nhân.

SQL Server được thiết kế để cung cấp hiệu suất cao với các tính năng tối ưu hóa như in-memory OLTP, columnstore indexes, và query optimization, giúp cải thiện tốc độ xử lý dữ liệu và hiệu năng của các ứng dụng.

Về bảo mật, SQL Server tích hợp các tính năng tiên tiến như mã hóa dữ liệu trong suốt (Transparent Data Encryption), bảo vệ dữ liệu ở cấp độ hàng (Row-Level Security), và luôn mã hóa dữ liệu (Always Encrypted), đảm bảo rằng dữ liệu luôn được bảo vệ khỏi các mối đe dọa bảo mật.

SQL Server cũng tích hợp chặt chẽ với các sản phẩm khác của Microsoft như Azure, Power BI, và Visual Studio, cung cấp một hệ sinh thái hoàn chỉnh cho việc phát triển, triển khai và quản lý các ứng dụng dữ liệu.

Khả năng mở rộng linh hoạt của SQL Server cho phép triển khai trên các máy chủ vật lý, máy chủ ảo hóa, và trên đám mây thông qua Azure SQL Database, đáp ứng nhu cầu của các doanh nghiệp từ nhỏ đến lớn.

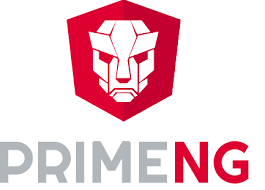
Ngoài ra, SQL Server đi kèm với các công cụ phân tích và báo cáo mạnh mẽ như SQL Server Reporting Services (SSRS) và SQL Server Analysis Services (SSAS), giúp doanh nghiệp khai thác dữ liệu hiệu quả và đưa ra các quyết định chính xác.

Cuối cùng, SQL Server có một cộng đồng người dùng rộng lớn và tài liệu phong phú, bao gồm các hướng dẫn, video, và diễn đàn hỗ trợ, giúp người dùng dễ dàng tìm kiếm giải pháp và nhận hỗ trợ khi cần. Với những ưu điểm này, SQL Server đã chứng tỏ mình là một hệ quản trị cơ sở dữ liệu mạnh mẽ, linh hoạt và an toàn, phù hợp cho mọi loại hình doanh nghiệp.

**2.5. Giới thiệu về PrimeNG**

**2.5.1. Khái niệm PrimeNG**

PrimeNG là một thư viện các thành phần giao diện người dùng (UI components) được phát triển cho các ứng dụng web sử dụng Angular. Được xây dựng bởi PrimeTek, PrimeNG cung cấp một tập hợp phong phú các thành phần UI chất lượng cao, giúp các nhà phát triển dễ dàng tạo ra các giao diện người dùng phức tạp và đẹp mắt.



**Hình 2.5****. Giới thiệu PrimeNG**

**2.5.2. Ưu điểm của PrimeNG**

PrimeNG là một thư viện các thành phần giao diện người dùng (UI components) chất lượng cao, được phát triển đặc biệt cho Angular, giúp các nhà phát triển dễ dàng tạo ra các giao diện người dùng phức tạp và đẹp mắt.

Với hơn 80 thành phần UI phong phú như bảng, biểu đồ, nút, hộp thoại, biểu mẫu và lịch, PrimeNG đáp ứng hầu hết các yêu cầu giao diện trong các ứng dụng web hiện đại.

Thư viện này tích hợp hoàn hảo với Angular, giúp tối ưu hóa quy trình phát triển và cho phép tùy chỉnh giao diện thông qua nhiều chủ đề và CSS tùy chỉnh.

Hiệu suất cao và khả năng phản hồi nhanh của các thành phần PrimeNG đảm bảo các ứng dụng web hoạt động mượt mà và đáp ứng tốt nhu cầu người dùng.

Ngoài ra, PrimeNG đi kèm với tài liệu phong phú, ví dụ và hướng dẫn cụ thể, hỗ trợ đắc lực cho các nhà phát triển trong quá trình sử dụng.

Cộng đồng PrimeNG tích cực và hỗ trợ, cùng với việc thư viện được cập nhật thường xuyên với các thành phần mới, giúp đảm bảo rằng PrimeNG luôn phù hợp với các xu hướng và công nghệ mới nhất trong phát triển web.

Với những ưu điểm này, PrimeNG là lựa chọn lý tưởng cho việc xây dựng các ứng dụng web Angular với giao diện người dùng đẹp mắt, linh hoạt và hiệu suất cao.

**2.6. Giới thiệu về RESTful API**

**2.6.1. Khái niệm RESTful API**

RESTful API là một tiêu chuẩn thiết kế API cho các ứng dụng web, tập trung vào việc quản lý tài nguyên hệ thống một cách hiệu quả. API này chú trọng đến các tài nguyên như tệp văn bản, hình ảnh, âm thanh, video, hoặc dữ liệu động, và các trạng thái của chúng được định dạng và truyền tải qua giao thức HTTP. Với RESTful API, việc truy cập và thao tác với tài nguyên trở nên đơn giản và dễ dàng thông qua các phương thức HTTP tiêu chuẩn như GET, POST, PUT, DELETE. Cách tiếp cận này không chỉ giúp tối ưu hóa việc quản lý và trao đổi dữ liệu giữa các hệ thống mà còn tăng cường tính linh hoạt và khả năng mở rộng của ứng dụng web. RESTful API đã trở thành một chuẩn mực phổ biến trong thiết kế web services, nhờ khả năng dễ dàng tích hợp và khả năng tương thích cao với nhiều nền tảng và ngôn ngữ lập trình khác nhau.

**2.6.2 Diễn giải các thành phần**

API (Application Programming Interface) là tập hợp các quy tắc và giao thức cho phép các ứng dụng hoặc thành phần khác nhau tương tác với nhau. API có thể trả về dữ liệu dưới các định dạng phổ biến như JSON hoặc XML, giúp các ứng dụng sử dụng và hiển thị thông tin một cách hiệu quả.

REST (REpresentational State Transfer) là một kiến trúc thiết kế API, sử dụng các phương thức HTTP đơn giản để tạo ra giao tiếp giữa các máy. Thay vì sử dụng một URL để xử lý thông tin người dùng, REST gửi các yêu cầu HTTP như GET, POST, DELETE, v.v., tới một URL cụ thể để thao tác với dữ liệu.

RESTful API là tiêu chuẩn phổ biến trong thiết kế API cho các ứng dụng web, nhằm quản lý tài nguyên một cách hiệu quả. RESTful API cho phép các ứng dụng web và di động khác nhau giao tiếp thông qua các phương thức HTTP tiêu chuẩn. Một trong những 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 để quản lý tài nguyên.

Điều đặc biệt về RESTful API là nó không quy định logic mã nguồn của ứng dụng và không bị giới hạn bởi bất kỳ ngôn ngữ lập trình hay framework nào. Bất kỳ ngôn ngữ hoặc công nghệ nào đều có thể được sử dụng để thiết kế một RESTful API, giúp tăng tính linh hoạt và khả năng tích hợp cao. RESTful API không chỉ giúp tối ưu hóa việc quản lý và trao đổi dữ liệu giữa các hệ thống mà còn tăng cường khả năng mở rộng và hiệu quả của ứng dụng, làm cho nó trở thành một chuẩn mực phổ biến trong việc xây dựng các dịch vụ web hiện đại.

**2.6.3. Hoạt động của RESTful**

REST hoạt động chủ yếu dựa vào giao thức HTTP và sử dụng các phương thức HTTP tiêu chuẩn để thực hiện các thao tác cơ bản trên tài nguyên (resources). Các phương thức này thường được gọi là CRUD, tương ứng với các hoạt động Create, Read, Update, và Delete – Tạo, Đọc, Sửa, Xóa.

Phương thức GET được sử dụng để truy vấn và trả về một tài nguyên hoặc danh sách các tài nguyên từ máy chủ, giúp người dùng có thể đọc và xem thông tin.  
 Phương thức POST được dùng để tạo mới một tài nguyên trên máy chủ, cho phép người dùng thêm thông tin mới vào hệ thống.

PUT là phương thức dùng để cập nhật thông tin cho một tài nguyên đã tồn tại, giúp duy trì và thay đổi dữ liệu hiện có.

Cuối cùng, phương thức DELETE cho phép xóa một tài nguyên khỏi hệ thống, giúp quản lý và loại bỏ dữ liệu không cần thiết.

Các phương thức này tạo nên nền tảng cho việc xây dựng và quản lý các dịch vụ web hiệu quả và dễ sử dụng. Bằng cách sử dụng các HTTP method một cách rõ ràng và nhất quán, RESTful API cho phép các nhà phát triển xây dựng các ứng dụng web có khả năng mở rộng, dễ bảo trì và linh hoạt.

Hơn nữa, việc định dạng URL và tổ chức các tài nguyên theo chuẩn REST giúp cải thiện sự rõ ràng và dễ hiểu của mã nguồn, làm cho việc phát triển và duy trì hệ thống trở nên dễ dàng hơn. Điều này không chỉ tối ưu hóa quá trình trao đổi dữ liệu giữa client và server mà còn tạo ra một cấu trúc API rõ ràng và hiệu quả, đáp ứng được nhu cầu phức tạp của các ứng dụng web hiện đại.

**2.6.4. Ưu điểm RESTful API**

RESTful API có nhiều ưu điểm vượt trội, giúp nó trở thành lựa chọn phổ biến trong việc thiết kế API cho các ứng dụng web.

Đầu tiên, RESTful API rất đơn giản và dễ sử dụng nhờ việc sử dụng các phương thức HTTP tiêu chuẩn (GET, POST, PUT, DELETE) và định dạng URL dễ hiểu, giúp các nhà phát triển dễ dàng học hỏi và áp dụng.

Thứ hai, RESTful API hiệu quả trong việc truyền dữ liệu, đặc biệt khi sử dụng định dạng JSON, giúp cải thiện hiệu suất của ứng dụng nhờ kích thước nhỏ và khả năng phân tích nhanh. Ngoài ra, RESTful API mang lại sự linh hoạt và khả năng mở rộng cao vì không bị ràng buộc bởi bất kỳ ngôn ngữ lập trình hay framework cụ thể nào, cho phép xây dựng và sử dụng trên nhiều nền tảng khác nhau.

Kiến trúc không trạng thái (stateless) của RESTful API hoạt động tốt với các hệ thống phân tán, mỗi yêu cầu từ client đến server đều chứa tất cả thông tin cần thiết để hiểu và xử lý yêu cầu, giúp cải thiện khả năng mở rộng và giảm thiểu sự phức tạp của hệ thống.

Hơn nữa, cách tổ chức tài nguyên và định dạng URL rõ ràng của RESTful API giúp mã nguồn dễ bảo trì và nâng cấp, cho phép thêm mới hoặc cập nhật các tính năng mà không ảnh hưởng đến các phần khác của hệ thống.

RESTful API cũng hỗ trợ caching, giúp giảm tải cho máy chủ và tăng tốc độ phản hồi cho người dùng, đặc biệt hữu ích đối với các ứng dụng cần xử lý lượng lớn yêu cầu từ client.

Cuối cùng, RESTful API có khả năng tích hợp cao, dễ dàng kết nối với các ứng dụng và dịch vụ khác, mang lại sự tiện lợi và hiệu quả trong phát triển và triển khai hệ thống.

**2.7. Mô hình Clean Architecture trong ASP.NET Core**

**2.7.1. Khái niệm Clean Architecture**

Clean Architecture là một mẫu kiến trúc phần mềm được thiết kế để tạo ra các hệ thống phần mềm có cấu trúc rõ ràng, dễ hiểu và dễ bảo trì. Trong ASP.NET Core, mô hình Clean Architecture giúp tổ chức mã nguồn theo cách tách biệt rõ ràng giữa các tầng (layers) của ứng dụng, giúp dễ dàng kiểm soát sự phụ thuộc và cải thiện khả năng mở rộng và bảo trì.

**2.7.2. Cấu trúc Clean Architecture**

Mô hình Clean Architecture trong ASP.NET Core là một phương pháp thiết kế phần mềm giúp tổ chức mã nguồn một cách rõ ràng và tách biệt các thành phần khác nhau của ứng dụng.

Entities (Thực thể) là các đối tượng cơ bản trong ứng dụng, thường đại diện cho dữ liệu trong cơ sở dữ liệu hoặc trong hệ thống. Các thực thể chỉ định rõ cấu trúc dữ liệu và không chứa logic kinh doanh, giúp chúng độc lập và dễ dàng tái sử dụng trong nhiều ngữ cảnh khác nhau.

Use Cases (Các trường hợp sử dụng) chứa logic kinh doanh của ứng dụng. Mỗi use case thường tương ứng với một tính năng hoặc chức năng cụ thể và không phụ thuộc vào các thành phần cụ thể của framework hay lớp dữ liệu. Điều này giúp duy trì tính nhất quán và dễ dàng kiểm thử.

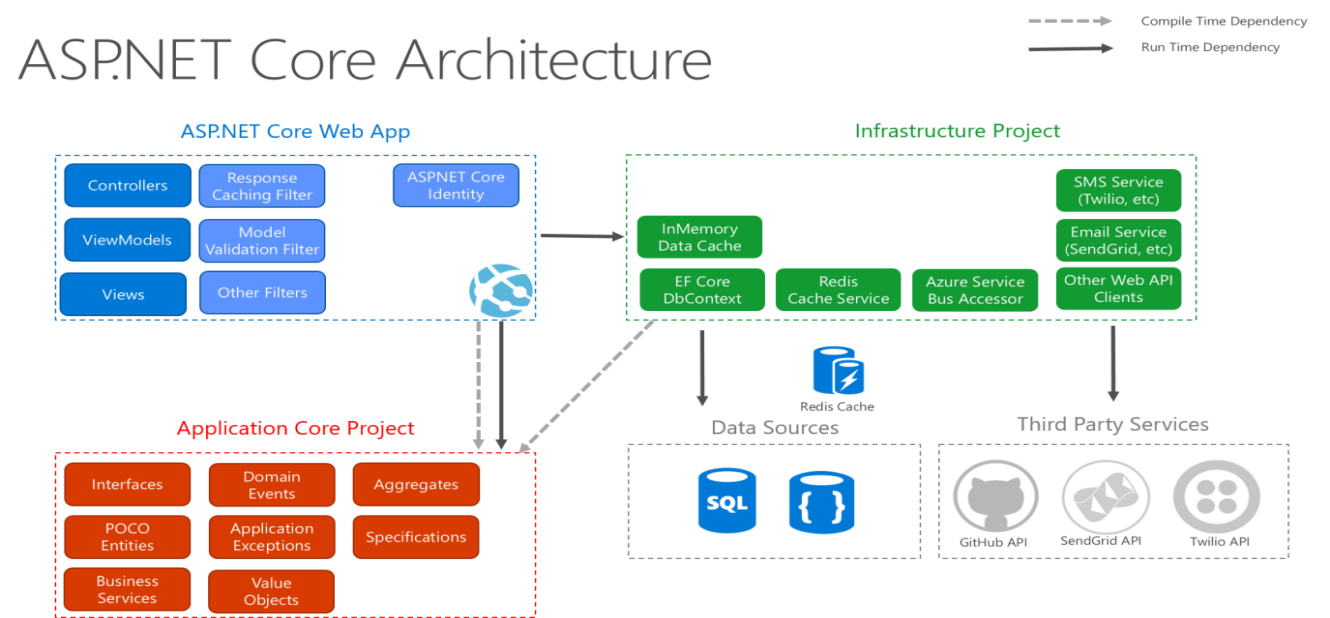
Interfaces (Giao diện) định nghĩa các phương thức mà các lớp ngoài cần triển khai. Việc sử dụng giao diện tạo ra sự tách biệt giữa các lớp và tăng tính linh hoạt của hệ thống, cho phép dễ dàng thay thế hoặc mở rộng các thành phần mà không ảnh hưởng đến toàn bộ hệ thống.

Repositories (Kho dữ liệu) là các lớp cung cấp giao diện để truy cập dữ liệu từ cơ sở dữ liệu hoặc bất kỳ nguồn dữ liệu nào khác. Chúng triển khai các giao diện được định nghĩa trong lớp Interfaces, đóng vai trò như một cầu nối giữa logic kinh doanh và lớp dữ liệu, giúp tách biệt các thao tác dữ liệu khỏi logic ứng dụng.

Controllers (Bộ điều khiển) chịu trách nhiệm điều hướng yêu cầu HTTP đến các use case tương ứng và trả về phản hồi cho người dùng. Chúng là điểm vào của các yêu cầu từ client và giúp quản lý luồng dữ liệu giữa người dùng và hệ thống.

Presenters/Views (Trình chiếu/Hiển thị) chịu trách nhiệm hiển thị dữ liệu cho người dùng, thường là các trang HTML hoặc các thành phần giao diện người dùng (UI components) trong ứng dụng web. Chúng đảm bảo rằng dữ liệu được hiển thị một cách trực quan và dễ hiểu cho người dùng cuối.

Bằng cách tách biệt các thành phần này, Clean Architecture giúp tạo ra các ứng dụng dễ bảo trì, dễ kiểm thử và có khả năng mở rộng cao, đảm bảo rằng mỗi thành phần của hệ thống có thể phát triển và thay đổi một cách độc lập.



**Hình 2.6****. Cấu trúc mô hình Clean Architecture**

**2.7.3. Ưu điểm của mô hình Clean Architecture**

Mô hình Clean Architecture mang lại nhiều ưu điểm vượt trội trong thiết kế và phát triển phần mềm.

Đầu tiên, nó giúp tách biệt rõ ràng các thành phần của ứng dụng như Entities, Use Cases, Interfaces, Repositories, Controllers, và Presenters/Views, làm cho việc quản lý và duy trì mã nguồn trở nên dễ dàng hơn. Sự tách biệt này cũng cải thiện khả năng kiểm thử từng thành phần một cách độc lập, giúp phát hiện sớm các lỗi và đảm bảo chất lượng phần mềm.

Một ưu điểm nổi bật khác là khả năng bảo trì và mở rộng dễ dàng, cho phép thay đổi hoặc bổ sung các tính năng mới mà không ảnh hưởng đến toàn bộ hệ thống, giảm thiểu rủi ro và chi phí.

Mô hình này không bị ràng buộc bởi bất kỳ framework hay công nghệ cụ thể nào, mang lại sự linh hoạt cao và cho phép chuyển đổi hoặc cập nhật công nghệ mà không cần thay đổi cấu trúc toàn bộ ứng dụng.

Clean Architecture tập trung vào logic nghiệp vụ, giữ cho mã nguồn rõ ràng và dễ hiểu, đồng thời tách biệt các chi tiết triển khai kỹ thuật. Điều này giúp đảm bảo rằng các vấn đề nghiệp vụ chính được giải quyết một cách hiệu quả.

Bên cạnh đó, khả năng tái sử dụng các thành phần như Entities và Use Cases trong nhiều dự án khác nhau giúp tiết kiệm thời gian và công sức trong quá trình phát triển.

Cuối cùng, mô hình này khuyến khích các thực hành tốt trong phát triển phần mềm, như nguyên tắc SOLID, tạo ra các ứng dụng bền vững và dễ bảo trì trong tương lai.

Với những ưu điểm này, Clean Architecture đã trở thành lựa chọn phổ biến cho các dự án lớn và phức tạp, yêu cầu tính bền vững và khả năng mở rộng cao.

**2.8. Chatbot DialogFlow**

**2.8.1. Ưu điểm Chatbot DialogFlow**

Chatbot DialogFlow là một nền tảng phát triển chatbot được cung cấp bởi Google, cho phép tạo ra các chatbot và các ứng dụng đàm thoại tự nhiên. Dưới đây là các đặc điểm chính của DialogFlow:

- Xử lý ngôn ngữ tự nhiên (NLP): DialogFlow sử dụng công nghệ NLP để hiểu và xử lý ngôn ngữ tự nhiên của người dùng. Nó có khả năng nhận diện ý định (intents) và thực thể (entities) từ câu hỏi hoặc yêu cầu của người dùng.

- Tích hợp dễ dàng: DialogFlow có thể tích hợp với nhiều nền tảng khác nhau. Điều này giúp tạo ra các chatbot có thể hoạt động trên nhiều kênh giao tiếp.

- Hỗ trợ đa ngôn ngữ: DialogFlow hỗ trợ nhiều ngôn ngữ, cho phép phát triển chatbot phục vụ người dùng trên toàn thế giới.

- Giao diện người dùng thân thiện: Nền tảng cung cấp một giao diện đồ họa dễ sử dụng để xây dựng và quản lý các chatbot mà không cần nhiều kiến thức lập trình.

- Machine Learning: DialogFlow sử dụng các mô hình học máy để cải thiện khả năng hiểu và phản hồi của chatbot theo thời gian. Nó có khả năng học hỏi từ các tương tác với người dùng để trở nên thông minh hơn.

- Tích hợp API: Ngoài việc sử dụng giao diện người dùng, DialogFlow còn cung cấp các API để các nhà phát triển có thể tích hợp sâu hơn và tùy chỉnh theo nhu cầu cụ thể của ứng dụng.

- Quản lý hội thoại: DialogFlow cho phép quản lý các luồng hội thoại phức tạp với người dùng, bao gồm cả việc duy trì ngữ cảnh giữa các câu hỏi và câu trả lời.

DialogFlow là một công cụ mạnh mẽ và linh hoạt cho việc phát triển các chatbot và ứng dụng đàm thoại tự nhiên, giúp doanh nghiệp và nhà phát triển xây dựng các giải pháp tương tác với người dùng một cách hiệu quả.

**2.8.2. Cách hoạt động**

Chatbot hoạt động qua các bước chính sau:

- Nhận đầu vào: Người dùng gửi thông điệp qua kênh giao tiếp như trang web, ứng dụng di động, hoặc mạng xã hội.

- Xử lý ngôn ngữ tự nhiên (NLP): Chatbot phân tích thông điệp để nhận diện ý định (intents) và thực thể (entities) của người dùng.

- Quản lý ngữ cảnh: Chatbot duy trì ngữ cảnh của cuộc hội thoại để hiểu và theo dõi các thông tin liên tiếp.

- Xử lý logic và quyết định: Chatbot áp dụng các quy tắc hoặc truy vấn cơ sở dữ liệu để xác định phản hồi phù hợp.

- Tạo phản hồi: Chatbot tạo ra phản hồi dựa trên kết quả xử lý logic.

- Gửi phản hồi: Phản hồi được gửi lại cho người dùng qua kênh giao tiếp ban đầu.

- Học hỏi và cải thiện: Chatbot ghi nhận và học hỏi từ các tương tác để cải thiện độ chính xác và hiệu suất trong tương lai.

**CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI**

**3.1. Mô tả sơ bộ bài toán**

Website giúp cho người quản lý dễ dàng nắm được các thông tin về khách hàng và sản phẩm. Còn khách hàng dễ dàng tiếp cận đầy đủ thông tin về sản phẩm. Website giúp người dùng tiết kiệm thời gian và sức lao động. Việc áp dụng website sẽ đem lại hiệu quả kinh tế cao, nâng cao năng suất lao động của con người.

**3.2. Các yêu cầu hệ thống**

**3.2.1. Yêu cầu về chức năng**

**Quản lý**

* Quản lý khách hàng
* Quản lý sản phẩm
* Quản lý loại sản phẩm
* Quản lý nhà sản xuất
* Quản lý đơn hàng
* Quản lý Mã giảm giá
* Thống Kê
* Đăng Nhập
* Đăng Ký

**Khách hàng**

* Đăng nhập
* Đăng kí
* Xem chi tiết sản phẩm
* Xem sản phẩm theo loại sản phẩm
* Xem sản phẩm theo nhà sản xuất
* Thêm vào giỏ hàng
* Đặt Hàng
* Đặt câu hỏi với chatbot

**3.2.2. Yêu cầu về phi chức năng**

1. ***Hoạt động***

Hệ thống phải có sự đồng bộ tức là dữ liệu có thể truyền đi giữa các máy tính.

1. ***Hiệu năng:***

Hệ thống phải hoạt động ổn định. Tốc độ cập nhật và tính toán phải nhanh (khoảng 1-2s với một dữ liệu). Tính toán cần chính xác tránh sai hệ thống. Dung lượng chưa thông tin phải tương đối lớn để có thể xử lý được lượng thông tin lớn.

1. ***Bảo mật***

Website cần có hệ thống đăng nhập. Chỉ có các nhân viên quản lý mới có thể đăng nhập vào hệ thống để cập nhật hay chỉnh sửa. Mỗi lần chỉnh sửa cũng sẽ được lưu lại để sử dụng sau này khi có sự cố. …

1. ***Văn hóa và chính trị***

* Ngôn ngữ sử dụng là tiếng Việt.
* Đơn vị thanh toán là tiền tệ có đơn vị VNĐ.
* Phần mềm phải hợp pháp và sử dụng hợp lý

**3.3. Mô hình hóa Use case**

**3.3.1. Use case tổng quan**



**Hình 3.1****. Use case tổng quan**

**3.3.2. Actor Quản lý**



**Hình 3.2****. Use case quản trị viên**

**3.3.3. Actor Khách hàng**



**Hình 3.3****. Use case khách hàng**

**3.4. Đặc tả Use case**

**3.4.1. Use case Đăng Nhập**

Use case này cho phép người dùng đăng nhập vào hệ thống website để thực hiện các chức năng của mình.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi người dùng kích vào ‘đăng nhập’ trên màn hình trang chủ của website. Hệ thống sẽ hiển thị ra một form cho phép người dùng nhập thông tin tài khoản của mình.
2. Người dùng điền thông tin tài khoản và mật khẩu của mình sau đó nhất nút đăng nhập để đăng nhập vào hệ thông. Use case kết thúc.

o ***Luồng rẽ nhánh***

1. Tại bước 2 của luồng cơ bản nếu người dùng nhập sai thông tin tài khoản hoặc mật khẩu, hệ thống sẽ thông báo ‘tài khoản mật khẩu không đúng”.
2. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Người dùng phải đăng kí tài khoản trước đó.

* **Hậu điều kiện**

Không có

* **Mở rộng**

Đăng Kí.

**3.4.2. Use case Đăng Ký**

Use case này cho phép người dùng đăng kí tạo tài khoản để sử dụng các chức năng khác của website.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use này bắt đầu khi người dùng kích vào “Đăng kí’ ở trong giao diện đăng nhập. Hệ thống sẽ hiển thi ra màn hình một form đăng kí bao gồm: email, số điện thoại, tên đăng nhập, mật khẩu, nhập lại mật khẩu, đia chỉ.
2. Người dùng điền thông tin đăng kí lên form đăng kí và nhất nút đăng kí. Use case kết thúc.

o ***Luồng rẽ nhánh***

1. Tại bước 2 của luồng cơ bản nếu người dùng nhập không đầy đủ thông tin hoặc sai định dạng thì hệ thống sẽ yêu cầu nhâp lại. Ngược lại, hệ thống sẽ hiển thị thông báo đăng kí thành công.
2. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Người dùng chưa có tài khoản truy cập hệ thống.

* **Hậu điều kiện**

Nếu đăng kí thành công hệ thống sẽ hiển thị màn hình đăng nhập.

* **Mở rộng**

Đăng Nhập.

**3.4.3. Use case Quản lý sản phẩm**

Use case này cho phép quản trị viên quản lý sản phẩm bao gồm thêm, sửa xóa sản phẩm.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi admin kích vào danh mục “quản lý sản phẩm” trên thanh menu quản trị. Hệ thống sẽ hiện thị ra màn hình một form nhập thêm mới sản phẩm và danh sách các sản phẩm có trong hệ thống.
2. Thêm sản phẩm:

Người quản trị nhập đầy đủ thông tin sản phẩm cần thêm và kích nút thêm. Hệ thống sẽ lưu thông tin sản phẩm mới vào cơ sở dữ liệu đồng thời hiển thị thông tin sản phẩm vừa thêm vào danh sách sản phẩm phía dưới.

1. Sửa sản phẩm:

Người quản trị kích nút sửa trên dòng sản phẩm, hệ thống sẽ hiển thị ra một form cập nhập thông tin sản phẩm. Người quản trị nhập thông tin cần sửa và kích nút sửa. Hệ thống sẽ cập nhật thông tin vào cơ sở dữ liệu và quay về màn hình danh sách sản phẩm.

1. Xóa sản phẩm:

Người quản trị kích vào nút sửa trên dòng sản phẩm. Hệ thống sẽ xóa thông tin sản phẩm được chọn ra khỏi cơ sở dữ liệu và cập nhật lại bảng thông tin sản phẩm.

o ***Luồng rẽ nhánh***

1. Tại bược 2 của luông cơ bản nếu quản trị viên nhập sai định dạng hệ thống sẽ yêu cầu nhập lại thông tin.
2. Tại bước 3 của luồng cơ bản nếu quản trị viên kích nút hủy thì hệ thống sẽ quay lại trang hiển thị danh sách sản phẩm.
3. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Đăng nhập hệ thống bằng tài khoản của quản trị viên.

* **Hậu điều kiện**

Không có.

* **Mở rộng**

Không có.

**3.4.4. Use case Quản lý đơn hàng**

Use case này cho phép quản trị viên quản lý đơn hàng bao gồm xem chi tiết, sửa xóa đơn hàng.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi admin kích vào danh mục “quản lýđơn hàng” trên thanh menu quản trị. Hệ thống sẽ hiện thị ra màn hình danh sách các đơn hàng có trong hệ thống.
2. Xem chi tiết:

Người quản trị kích nút xem chi tiết trên dòng đơn hàng, hệ thống sẽ hiển thị ra một form sửa thông tin đơn hàng và danh sách các đơn hàng.

1. Sửa đơn hàng:

Người quản trị kích nút sửa trên dòng đơn hàng, hệ thống sẽ hiển thị ra một form cập nhập thông tin đơn hàng. Người quản trị nhập thông tin cần sửa và kích nút sửa. Hệ thống sẽ cập nhật thông tin vào cơ sở dữ liệu và quay về màn hình danh sách đơn hàng.

1. Xóa sản phẩm:

Người quản trị kích vào nút sửa trên dòngđơn hàng. Hệ thống sẽ xóa thông tin đơn hàng được chọn ra khỏi cơ sở dữ liệu và cập nhật lại bảng thông tin đơn hàng.

o ***Luồng rẽ nhánh***

1. Tại bược 2 của luông cơ bản nếu quản trị viên nhập sai định dạng hệ thống sẽ yêu cầu nhập lại thông tin.
2. Tại bước 3 của luồng cơ bản nếu quản trị viên kích nút hủy thì hệ thống sẽ quay lại trang hiển thị danh sách đơn hàng.
3. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Đăng nhập hệ thống bằng tài khoản của quản trị viên.

* **Hậu điều kiện**

Không có .

* **Mở rộng**

Không có.

**3.4.5. Use case Quản lý tài khoản**

Use case này cho phép quản trị viên quản lý tài khoản của người dùng bao gồm thêm, sửa xóa tài khoản người dùng.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi admin kích vào danh mục “quản lý tài khoản người dùng” trên thanh menu quản trị. Hệ thống sẽ hiện thị ra màn hình một form nhập thêm mới tài khoản và danh sách các tài khoản người dùng có trong hệ thống.
2. Thêm tài khoản:

Người quản trị nhập đầy đủ thông tin tài khoản cần thêm và kích nút thêm. Hệ thống sẽ lưu thông tin tài khoản mới vào cơ sở dữ liệu đồng thời hiển thị thông tin tài khoản vừa đăng kí vào danh sách tài khoản phía dưới.

1. Sửa tài khoản:

Người quản trị kích nút sửa trên dòng tài khoản, hệ thống sẽ hiển thị ra một form cập nhập thông tin tài khoản. Người quản trị nhập thông tin cần sửa và kích nút sửa. Hệ thống sẽ cập nhật thông tin vào cơ sở dữ liệu và quay về màn hình danh sách tài khoản người dùng.

1. Xóa tài khoản:

Người quản trị kích vào nút sửa trên dòng tài khoản. Hệ thống sẽ xóa thông tin tài khoản được chọn ra khỏi cơ sở dữ liệu và cập nhật lại bảng thông tin tài khoản.

o ***Luồng rẽ nhánh***

1. Tại bược 2 của luông cơ bản nếu quản trị viên nhập sai định dạng hệ thống sẽ yêu cầu nhập lại thông tin.
2. Tại bước 3 của luồng cơ bản nếu quản trị viên kích nút hủy thì hệ thống sẽ quay lại trang hiển thị danh sách tài khoản.
3. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Đăng nhập hệ thống bằng tài khoản của quản trị viên.

* **Hậu điều kiện**

Không có.

* **Mở rộng**

Không có.

**3.4.6. Use case Quản lý loại sản phẩm**

Use case này cho phép quản trị viên quản lý danh mục sản phẩm bao gồm thêm, sửa xóa hãng sản phẩm và mức giá.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi admin kích vào danh mục “quản lý danh mục” trên thanh menu quản trị. Hệ thống sẽ hiện thị ra màn hình một form nhập thêm mới các hãng sản phẩm hoặc các mức giá của sản phẩm và danh sách các hãng sản phẩm, các mức giá sản phẩm có trong hệ thống.
2. Thêm hãng và mức giá sản phẩm:

Người quản trị nhập đầy đủ thông tin hãng sản phẩm cần thêm hoặc mức giá cần thêm và kích nút thêm. Hệ thống sẽ lưu thông tin hãng sản phẩm mới và mức giá mới vào cơ sở dữ liệu đồng thời hiển thị thông tin hãng sản phẩm và mức giá vừa thêm vào danh sách sản phẩm phía dưới.

1. Sửa hãng và mức giá sản phẩm:

Người quản trị kích nút sửa trên dòng hãng sản phẩm hoặc mức giá, hệ thống sẽ hiển thị ra một form cập nhập thông tin hãng sản phẩm hoặc thông tin mức giá. Người quản trị nhập thông tin cần sửa và kích nút sửa. Hệ thống sẽ cập nhật thông tin vào cơ sở dữ liệu và quay về màn hình danh sách quản lí danh mục.

1. Xóa sản phẩm:

Người quản trị kích vào nút sửa trên dòng hãng sản phẩm hoặc mức giá. Hệ thống sẽ xóa thông tin hãng sản phẩm hoặc mức giá được chọn ra khỏi cơ sở dữ liệu và cập nhật lại bảng thông tin hãng sản phẩm hoặc mức giá.

o ***Luồng rẽ nhánh***

1. Tại bược 2 của luông cơ bản nếu quản trị viên nhập sai định dạng hệ thống sẽ yêu cầu nhập lại thông tin.
2. Tại bước 3 của luồng cơ bản nếu quản trị viên kích nút hủy thì hệ thống sẽ quay lại trang hiển thị danh sách các hãng và mức giá.
3. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Đăng nhập hệ thống bằng tài khoản của quản trị viên.

* **Hậu điều kiện**

Không có

* **Mở rộng**

Không có.

**3.4.7. Use case Thêm sản phẩm vào giỏ hàng**

Use case này cho phép người dùng thêm sản phẩm vào giỏ hàng.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi người dùng kích vào nút thêm vào giỏ hàng trên sản phẩm. Hệ thống thêm sản phẩm vào bảng GIOHANG gồm: id sản phẩm, tên sản phẩm, hình ảnh, giá bán, số lượng và hiển thị giỏ hàng. Use case kết thúc.

o ***Luồng rẽ nhánh***

1. Tại bước 1 của luồng cơ bản nếu số lượng sản phẩm được thêm đã hết. Hệ thống hiển thị thông báo đã hết sản phẩm. Use case kết thúc.

2. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với CDSL. Hệ thống hiện thị một thông báo lỗi. Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Số lượng sản phẩm được thêm phải còn trong kho.

* **Hậu điều kiện:**

Không có

* **Mở rộng**

Không có.

**3.4.8. Use case Xem chi tiết sản phẩm**

Use case này giúp xem chi tiết các thông tin liên quan của sản phẩm.

* **Luồng sự kiện**

o ***Luồng cơ bản***

Use case này bắt đầu khi người dùng nhấn vào sản phẩm và chọn “Xem chi tiết” bất kì sản phẩm nào trên giao diện.

Hệ thống sẽ hiển thị toàn bộ thông tin của sản phẩm lên màn hình.

o ***Luồng rẽ nhánh***

Không kết nối được cơ sở dữ liệu tại bất kì thời điểm nào trong quá trình thực hiện use case thì hệ thống sẽ hiển thị báo

“lỗi kết nối” và use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có

* **Mở rộng**

Không có.

**3.4.9. Use case Xem sản phẩm theo loại sản phẩm**

Use case này giúp người dùng xem các sản phẩm theo hãng.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi người dùng trỏ chuột vào “Danh mục sản phẩm” và click chọn hãng bất kì nào trên giao diện.
2. Hệ thống sẽ hiển thị lên màn hình tất cả các sản phẩm của hãng đã chọn.
3. Người dùng xem chi tiết 1 sản phẩm bất kì, usecase kết thúc.

o ***Luồng rẽ nhánh***

1. Tại bước 3, nếu người dùng chọn một hãng khác bất kì trên giao diện màn hình sẽ hiển thị lại danh sách sản phẩm theo hãng mới.
2. Không kết nối được cơ sở dữ liệu tại bất kì thời điểm nào trong quá trình thực hiện use case thì hệ thống sẽ hiển thị báo “lỗi kết nối” và use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Không có

* **Hậu điều kiện**

Không có

* **Mở rộng**

Không có.

**3.4.10. Use case Đặt hàng**

Use case này cho phép khách hàng thực hiện đặt mua hàng trên hệ thống website.

* **Luồng sự kiện**

o ***Luồng cơ bản***

1. Use case này bắt đầu khi người dùng kích vào nút đặt hàng. Hệ thống sẽ hiển thị thông tin đơn hàng gồm người nhận, số điện thoại, địa chỉ yêu cầu người dùng xác nhận để nhận hàng khi được giao.
2. Người dùng xác nhận và kích vào nút “Đặt Hàng”. Hệ thống sẽ hiển thị ra danh sách các sản phẩm khách hàng vừa đặt, đã đặt và trạng thái của đơn hàng. Use case kết thúc.

o ***Luồng rẽ nhánh***

1. Tại bước 1 của luồng cơ bản, nếu người dùng không nhập thông tin người nhận và địa chỉ hệ thống sẽ yêu cầu nhập lại.
2. Tại bất kì bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và Use case kết thúc.

* **Các yêu cầu đặc biệt**

Không có

* **Tiền điều kiện**

Khách hàng phải đăng nhâp trước khi thanh toán.

* **Hậu điều kiện**

Không có

* **Mở rộng**

Không có.

**3.5. Phân tích Use case**

**3.5.1. Use case Đăng Nhập**

Biểu đồ trình tự:



**Hình 3.4****. Biểu đồ trình tự Use case Đăng Nhập**

Biểu đồ lớp:



**Hình 3.5****. Biểu đồ lớp Use case Đăng Nhập**

**3.5.2. Use case Đăng Ký**

Biểu đồ trình tự:



**Hình 3.6****. Biểu đồ trình tự Use case Đăng Ký**

Biểu đồ lớp:



**Hình 3.7****. Biểu đồ lớp Use case Đăng Ký**

**3.5.3. Use case Quản lý đơn hàng**

Biểu đồ trình tự:



**Hình 3.8****. Biểu đồ trình tự Use case Quản lý đơn hàng**

Biểu đồ lớp:



**Hình 3.9****. Biểu đồ lớp Quản lý đơn hàng**

**3.5.4. Use case Quản lý tài khoản**

Biểu đồ trình tự:



**Hình 3.10****. Biểu đồ trình tự Quản lý tài khoản**

Biểu đồ lớp:



**Hình 3.11****. Biểu đồ lớp Quản lý tài khoản**

**3.5.5. Use case Quản lý sản phẩm**

Biểu đồ trình tự:



**Hình 3.12****. Biểu đồ trình tự Quản lý sản phẩm**

Biểu đồ lớp:



**Hình 3.13****. Biểu đồ lớp Quản lý sản phẩm**

**3.5.6. Use case Thêm sản phẩm vào giỏ hàng**

Biểu đồ trình tự:



**Hình 3.14****. Biểu đồ trình tự Use case Thêm sản phẩm vào giỏ hàng**

Biểu đồ lớp:



**Hình 3.15****. Biểu đồ lớp Use case Thêm sản phẩm vào giỏ hàng**

**3.5.7. Use case Tìm kiếm**

Biểu đồ trình tự:



**Hình 3.16****. Biểu đồ trình tự Use case Tìm kiếm**

Biểu đồ lớp:



**Hình 3.17****. Biểu đồ lớp Use case Tìm kiếm**

**3.5.8. Use case Xem chi tiết sản phẩm**

Biểu đồ trình tự:



**Hình 3.18****. Biểu đồ trình tự Use case Xem chi tiết sản phẩm**

Biểu đồ lớp:



**Hình 3.19****. Biểu dồ lớp Use case Xem chi tiết sản phẩm**

**3.6. Mô hình hóa cơ sở dữ liệu**

**3.6.1. Các yêu cầu về dữ liệu**

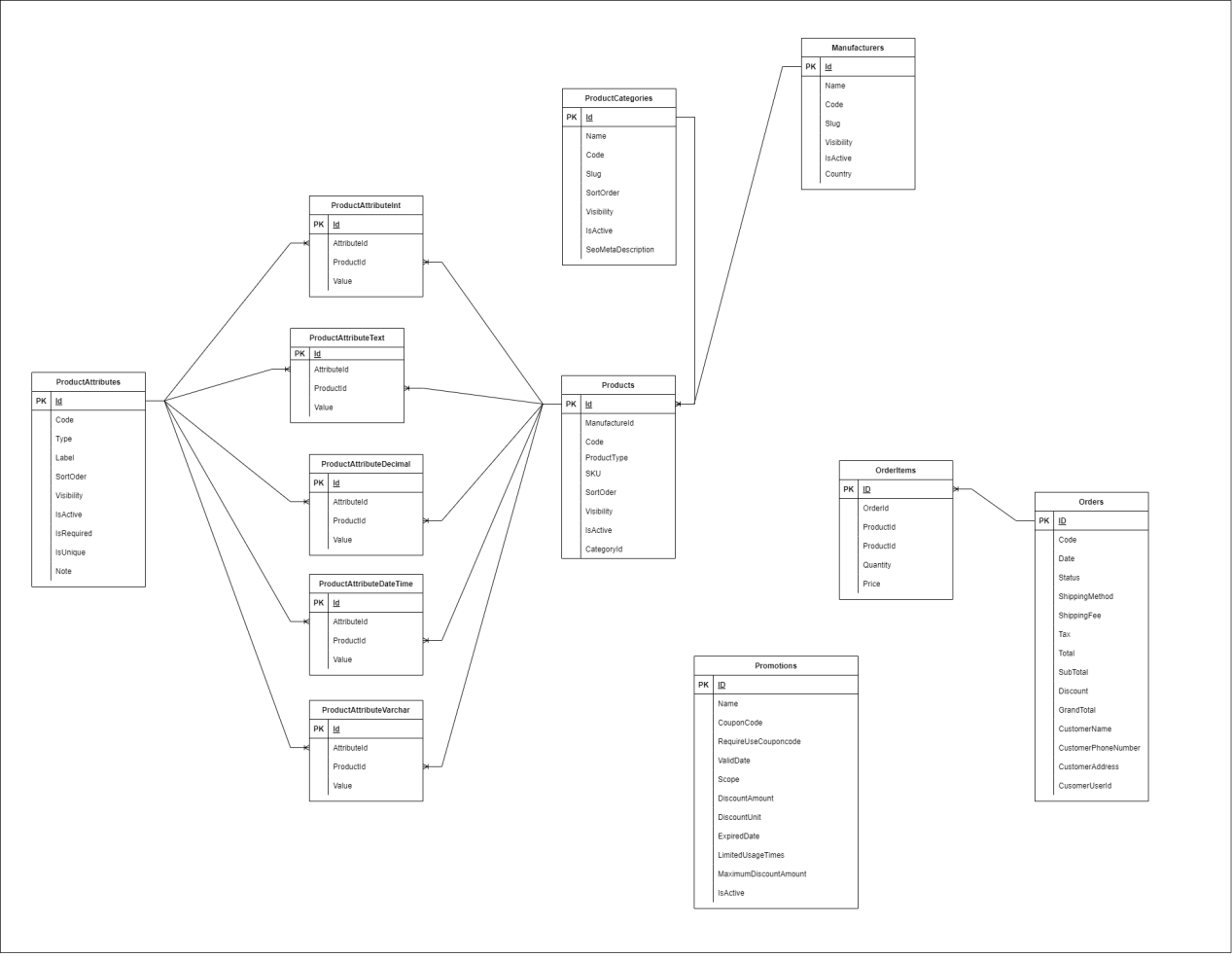
Xây dựng một cơ sở dữ liệu về nem truyền thống. Hệ thống lưu trữ thông tin của người dùng bao gồm: Email, họ tên, mật khẩu, địa chỉ, số điện thoại, …

Mỗi sản phẩm có đều có id riêng để phân biệt với các hàng hóa khác cho phép khách hàng có thể xem cũng như chọn lựa. Các sản phẩm đều có tên sản phẩm, hình ảnh, giá.

Khi khách hàng đặt hàng sẽ có đơn hàng để biết khách hàng mua sản phẩm nào với số lượng và giá bao nhiêu. Đơn hàng bao gồm: id, email, ngày mua, họ tên, số điện thoại, địa chỉ.

Ngoài ra, Website cung cấp tài khoản truy cập cho người quản lý và nhân viên để thực hiện tha tác nghiệp vụ của mình.

**3.6.2. Biểu đồ thực thể liên kết**

****

**Hình 3.20****. Biểu đồ thực thể liên kết**

**3.6.3. Mô tả văn tắt các thực thể**

Dưới đây là ý nghĩa của các bảng trong sơ đồ cơ sở dữ liệu:

- ProductAttributes: Lưu thông tin về các thuộc tính của sản phẩm. Bao gồm mã thuộc tính, loại thuộc tính, nhãn, thứ tự sắp xếp, khả năng hiển thị, trạng thái hoạt động, bắt buộc, duy nhất, và ghi chú.

- ProductAttributesInt: Lưu trữ các giá trị thuộc tính dạng số nguyên của sản phẩm. Bao gồm mã thuộc tính, mã sản phẩm, và giá trị số nguyên.

- ProductAttributesText: Lưu trữ các giá trị thuộc tính dạng văn bản của sản phẩm. Bao gồm mã thuộc tính, mã sản phẩm, và giá trị văn bản.

- ProductAttributesDecimal: Lưu trữ các giá trị thuộc tính dạng số thập phân của sản phẩm. Bao gồm mã thuộc tính, mã sản phẩm, và giá trị số thập phân.

- ProductAttributesDateTime: Lưu trữ các giá trị thuộc tính dạng ngày giờ của sản phẩm. Bao gồm mã thuộc tính, mã sản phẩm, và giá trị ngày giờ.

- ProductAttributesVarchar: Lưu trữ các giá trị thuộc tính dạng varchar của sản phẩm. Bao gồm mã thuộc tính, mã sản phẩm, và giá trị varchar.

- Products: Lưu thông tin về sản phẩm. Bao gồm mã sản phẩm, mã nhà sản xuất, mã loại sản phẩm, SKU, thứ tự sắp xếp, khả năng hiển thị, trạng thái hoạt động, và mã danh mục.

- ProductCategories: Lưu thông tin về danh mục sản phẩm. Bao gồm tên, mã, đường dẫn, thứ tự sắp xếp, khả năng hiển thị, trạng thái hoạt động, và mô tả SEO.

- Manufacturers: Lưu thông tin về nhà sản xuất. Bao gồm tên, mã, đường dẫn, khả năng hiển thị, trạng thái hoạt động, và quốc gia.

- Orders: Lưu thông tin về đơn hàng. Bao gồm mã đơn hàng, ngày, trạng thái, phương thức giao hàng, phí giao hàng, thuế, tổng tiền, tổng phụ, giảm giá, tổng cộng, tên khách hàng, số điện thoại khách hàng, địa chỉ khách hàng, và mã khách hàng.

- OrderItems: Lưu thông tin về các mặt hàng trong đơn hàng. Bao gồm mã mặt hàng, mã đơn hàng, mã sản phẩm, số lượng, và giá.

- Promotions: Lưu thông tin về các chương trình khuyến mãi. Bao gồm tên, mã giảm giá, yêu cầu mã giảm giá, trạng thái hoạt động, phạm vi, số tiền giảm giá, giới hạn giảm giá, ngày hết hạn, số lần sử dụng tối đa, và số tiền giảm giá tối đa.

**3.6.4. Mô tả chi tiết các bảng**

**Bảng 3.1****. Bảng mô tả chi tiết bảng sản phẩm**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| ManufacturerId | Guid() | Mã nhà sản xuất |
| Name | Nvarchar(50) | Tên sản phẩm |
| Code | varchar(50) | Mã code của sản phẩm |
| Slug | varchar(50) | URL-friendly của tên sản phẩm |
| ProductType | int | Kiểu sản phẩm |
| SKU | Varchar(50) | Mã SKU |
| SortOrder | Int | Thứ tự sắp xếp |
| Visibility | bit | Hiển thị |
| IsActive | bit | Kích hoạt |
| CategoryId | Guid() | Id danh mục sản phẩm |
| SeoMetaDescription | Nvarchar(250) | Mô tả meta cho SEO |
| Description | Nvarchar(MAX) | Mô tả |
| ThumbnailPicture | Nvarchar(250) | Hình ảnh |

**Bảng 3.2****. Bảng mô tả chi tiết bảng loại sản phẩm**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| Name | Nvarchar(50) | Tên danh mục |
| Code | varchar(50) | Mã định danh danh mục |
| Slug | varchar(50) | URL-friendly của tên danh mục |
| SortOrder | Int | Thứ tự sắp xếp |
| Visibility | bit | Hiển thị |
| IsActive | bit | Kích hoạt |
| SeoMetaDescription | Nvarchar(250) | Mô tả meta cho SEO |

**Bảng 3.3****. Bảng mô tả chi tiết bảng thuộc tính sản phẩm**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã thuộc tính |
| Code | Varchar(50) | Code thuộc tính |
| Type | int | Kiểu thuộc tính |
| Label | Nvarchar(50) | Nhãn của thuộc tính |
| SortOrder | int | Thứ tự |
| Visibility | bit | Hiển thị |
| IsActive | bit | Kích hoạt |
| IsRequired | bit | Bắt buộc |
| IsUnique | bit | Duy Nhất |
| Note | Nvarchar(MAX) | Ghi chú |

**Bảng 3.4****. Bảng mô tả chi tiết bảng thuộc tính kiểu số nguyêm**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| AttributeId | Guid() | Mã thuộc tính |
| ProductId | Guid() | Mã sản phẩm |
| Value | int | Giá trị |

**Bảng 3.5****. Bảng mô tả chi tiết bảng thuộc tính kiểu Varchar**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| AttributeId | Guid() | Mã thuộc tính |
| ProductId | Guid() | Mã sản phẩm |
| Value | Nvarchar(500) | Giá trị |

**Bảng 3.6****. Bảng mô tá chi tiết bảng thuộc tính kiểu số thập phân**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| AttributeId | Guid() | Mã thuộc tính |
| ProductId | Guid() | Mã sản phẩm |
| Value | Decimal(18,2) | Giá trị |

**Bảng 3.7****. Bảng mô tả chi tiết bảng thuộc tính kiểu văn bản**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| AttributeId | Guid() | Mã thuộc tính |
| ProductId | Guid() | Mã sản phẩm |
| Value | Nvarchar(MAX) | Giá trị |

**Bảng 3.8****. Bảng mô tả chi tiết bảng thuộc tính dạng ngày giờ**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| AttributeId | Guid() | Mã thuộc tính |
| ProductId | Guid() | Mã sản phẩm |
| Value | Nvarchar(MAX) | Giá trị |

**Bảng 3.9****. Bảng mô tả chi tiết bảng nhà sản xuất**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| Name | Nvarchar(50) | Tên nhà sản xuất |
| Code | Varchar(50) | Mã định danh |
| Slug | Varchar(50) | URL-friendly của tên nhà sản xuất |
| Visibility | bit | Hiển Thị |
| IsActive | bit | Kích Hoạt |
| Country | Nvarchar(MAX) | Địa chỉ |

**Bảng 3.10****. Bảng mô tả chi tiết bảng đơn hàng**

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **Id** | Guid() | Mã |
| Code | varchar(50) | Mã định danh đơn hàng |
| Status | int | Trạng Thái |
| PaymentMethod | int | Hình thức thanh toán |
| ShippingFee | int | Phí ship |
| Tax | float | Thuế |
| Total | float | Giá trị đơn hàng trước thuế |
| Subtotal | float | Giá trị đơn hàng sau thuế |
| Discount | float | Tổng giảm |
| GrandTotal | float | Tổng giá trị cuối cùng |
| CustomerName | Nvarchar(50) | Tên khách hàng đặt hàng |
| CustomerPhoneNumber | Nvarchar(50) | Số điện thoại khách hàng |
| CustomerAddress | Nvarchar(250) | Địa chỉ |
| CustomerUserId | Guid() | Mã Khách hàng |

**Bảng 3.11****. Bảng mô tả chi tiết bảng chi tiết đơn hàng**

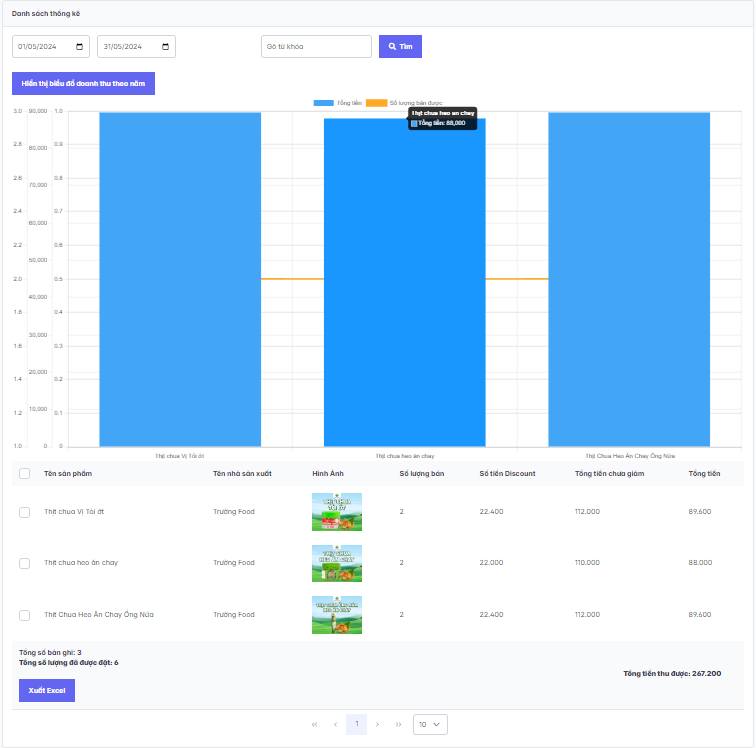
|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu** | **Mô Tả** |
| **OrderId** | Guid() | Mã đơn hàng |
| ProductId | Guid() | Mã sản phẩm |
| SKU | Varchar(50) | Mã SKU |
| Quantity | int | Số lượng |
| Price | float | Giá |

**Bảng 3.12****. Bảng mô tả chi tiết bảng khuyến mãi**

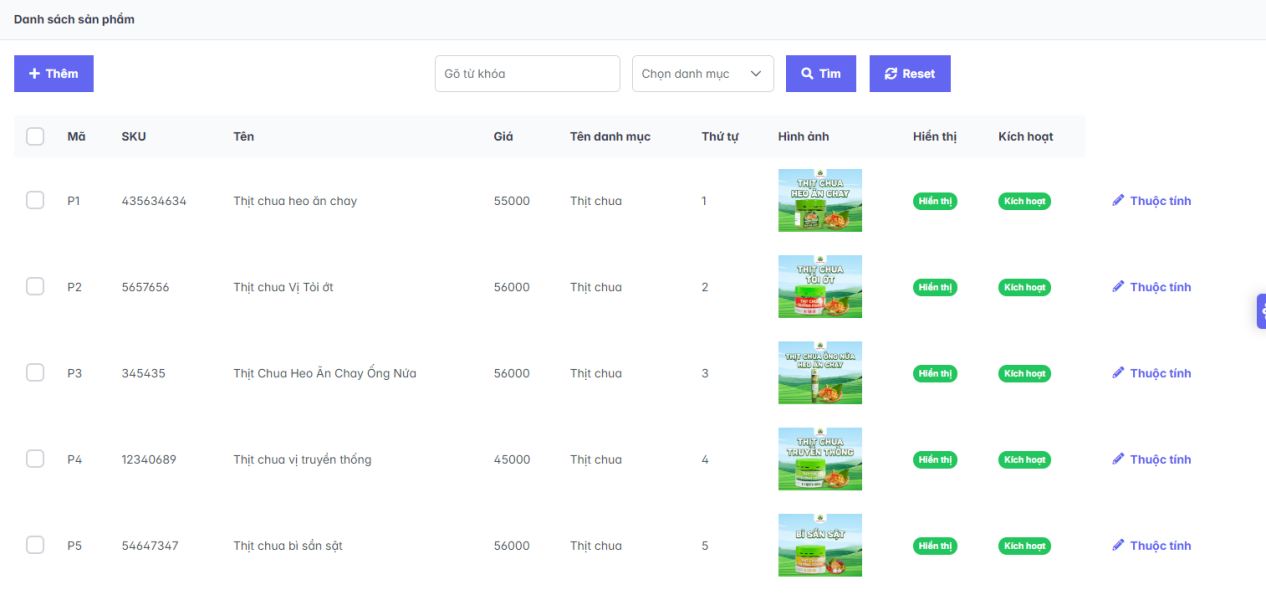
|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu | Mô Tả |
| **Id** | Nvarchar(50) | Mã |
| Name | Nvarchar(50) | Tên khuyến mãi |
| CouponCode | Varchar(50) | Mã giảm giá |
| ValidDate | datetime | Ngày bắt đầu |
| ExpriedDate | datetime | Ngày hết hạn |
| DiscountMount | float | Tỉ lệ giảm |
| IsActive | bit | Hoạt động |

**3.7. Giao diện màn hình**

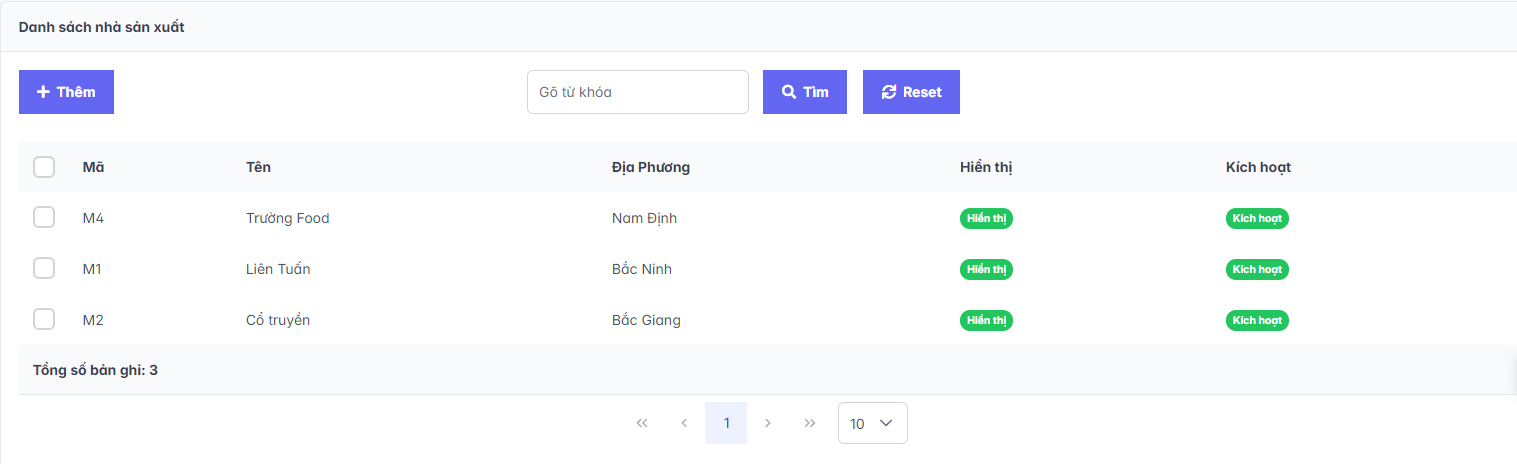
**3.7.1. Giao diện trang Admin**



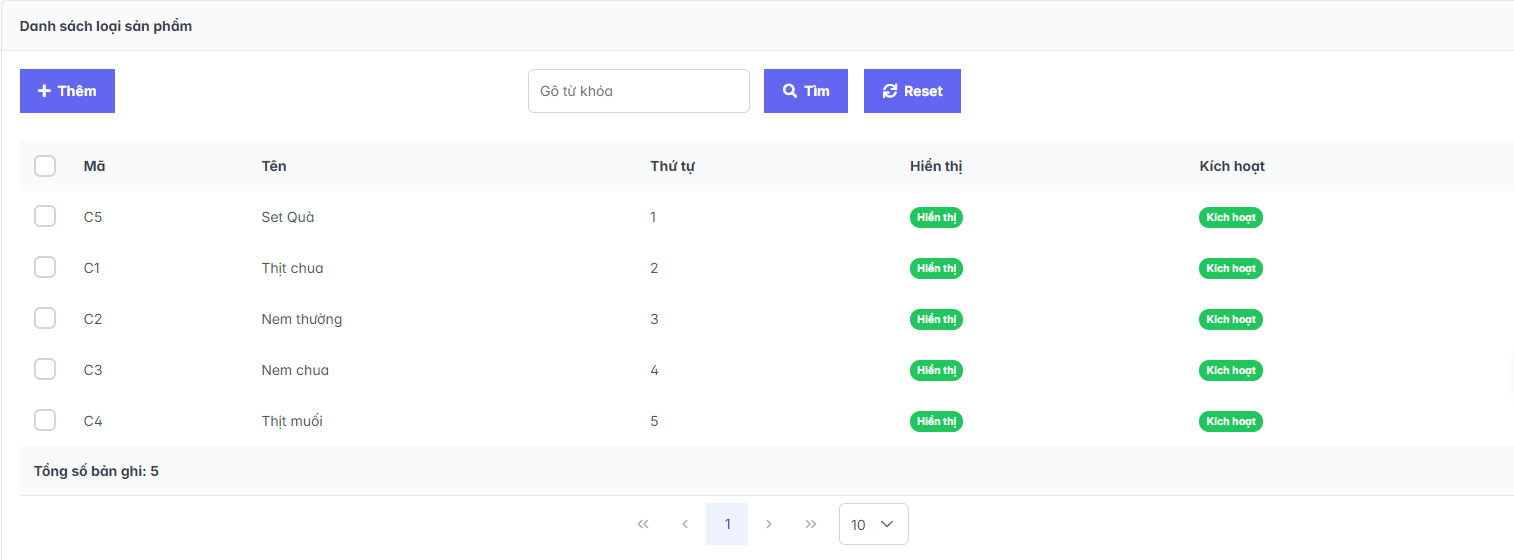
**Hình 3.21****. Giao diện trang thống kê**



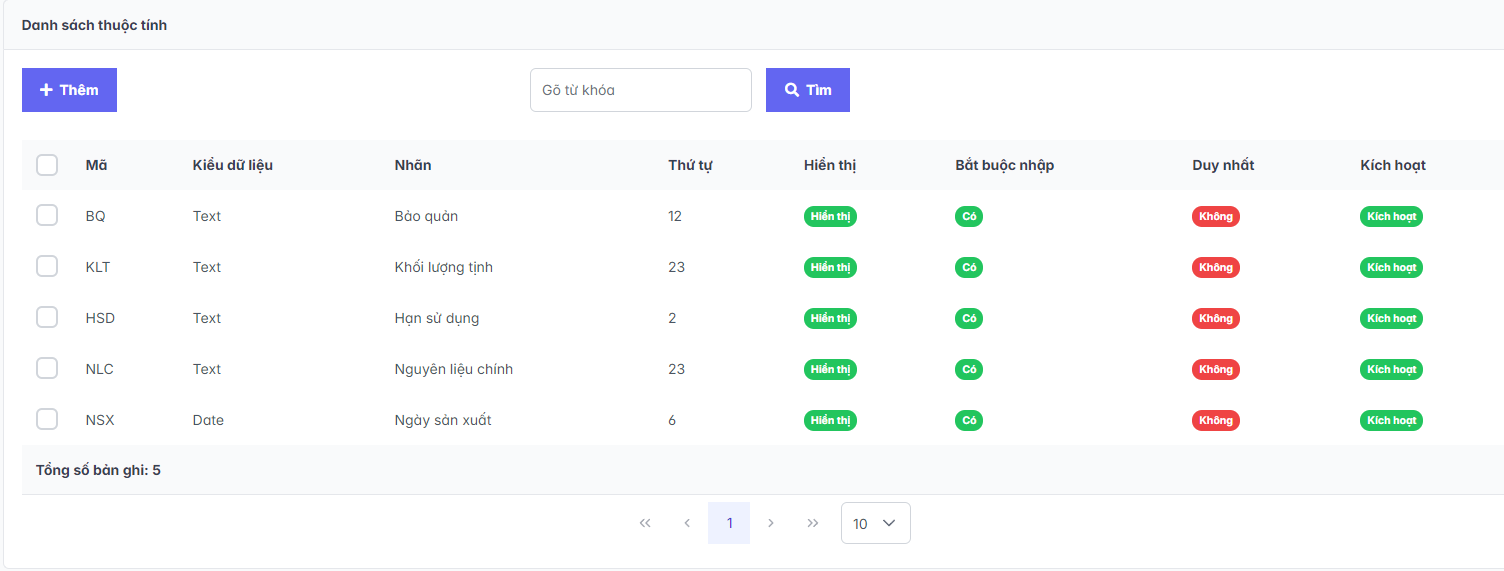
**Hình 3.22****. Giao diện trang quản lý sản phẩm**



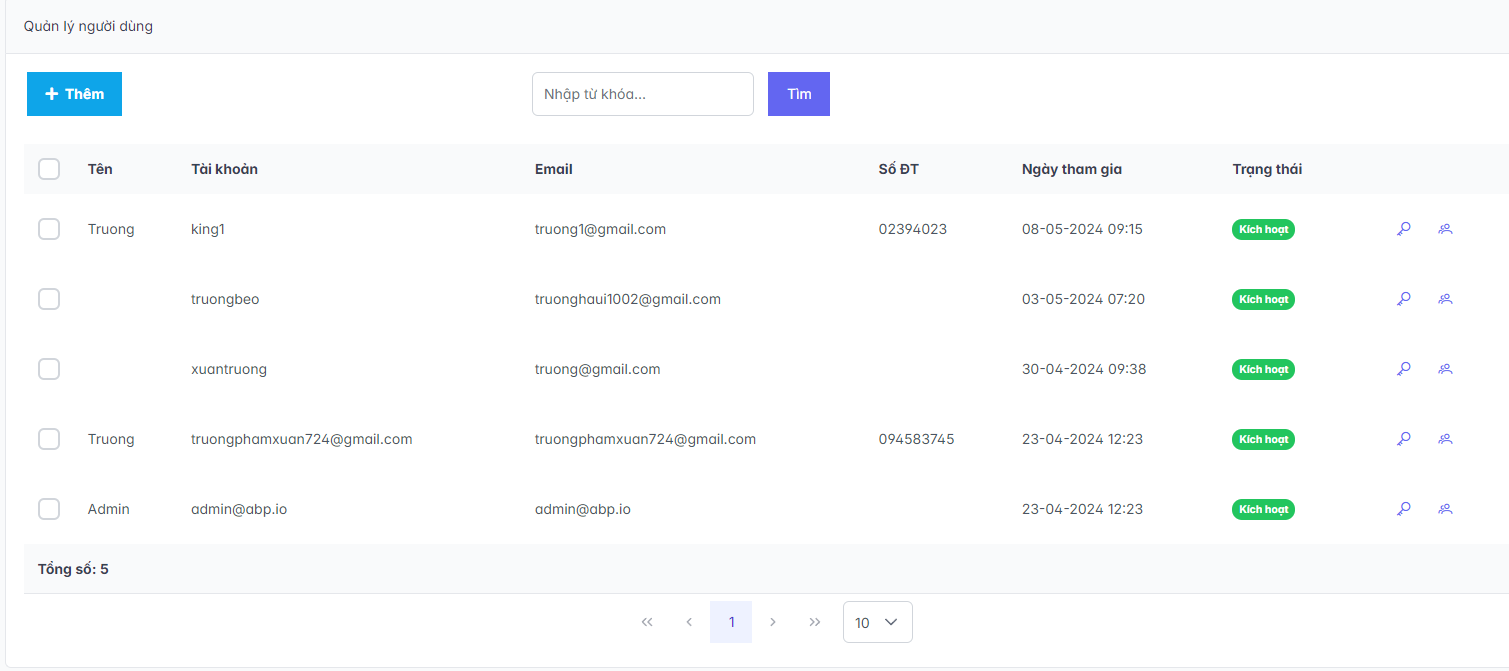
**Hình 3.23****. Giao diện trang quản lý nhà sản xuất**



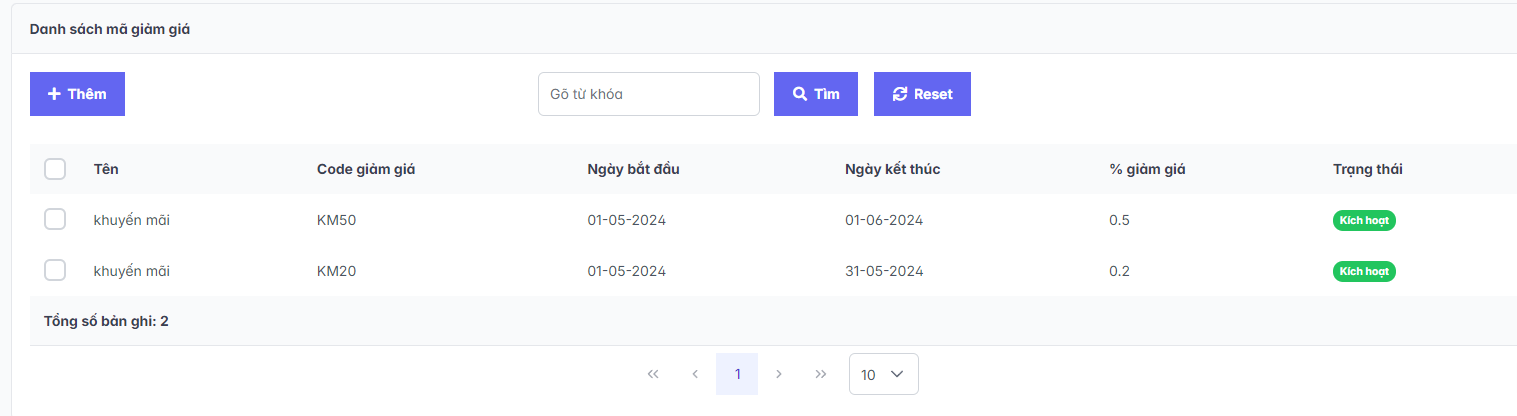
**Hình 3.24****. Giao diện trang quản lý loại sản phẩm**



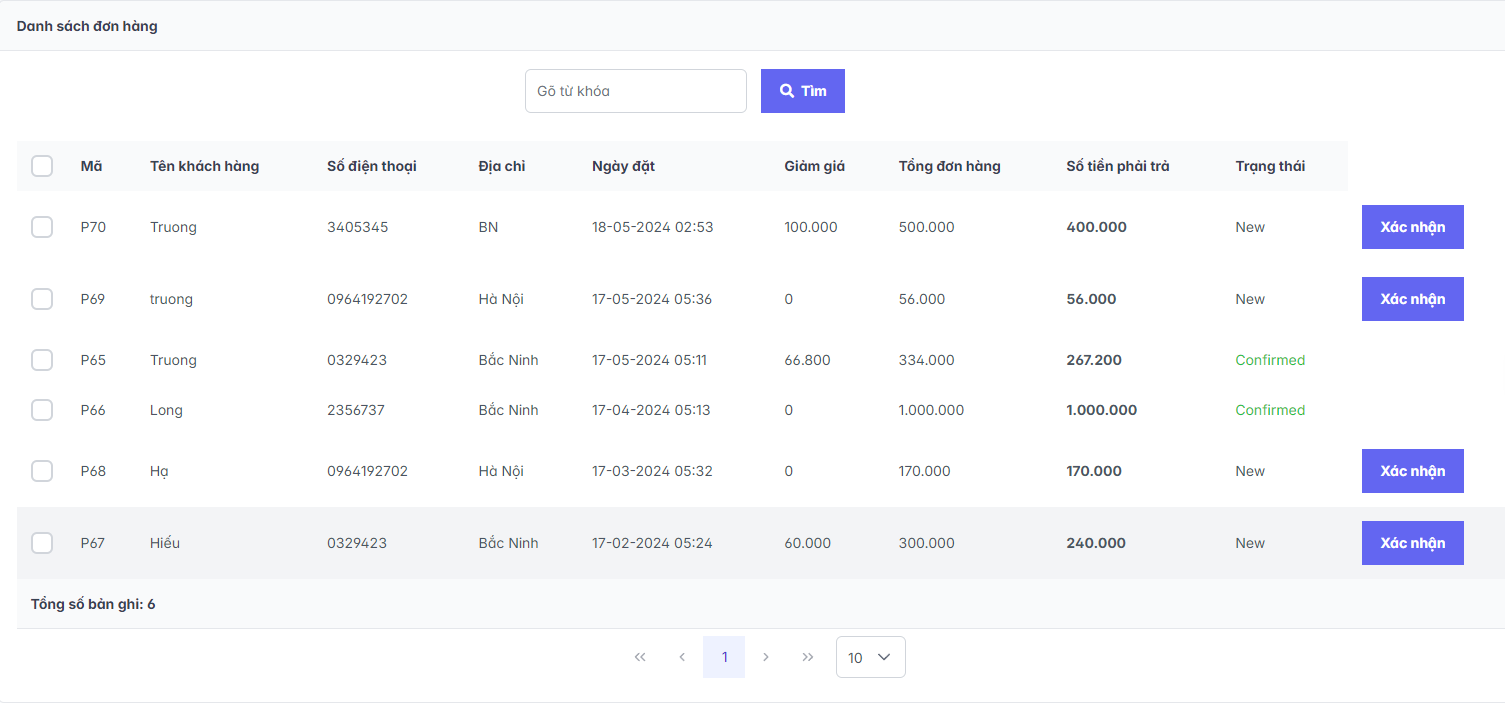
**Hình 3.25****. Giao diện trang quản lý thuộc tính**



**Hình 3.26****. Giao diện trang quản lý tài khoản**

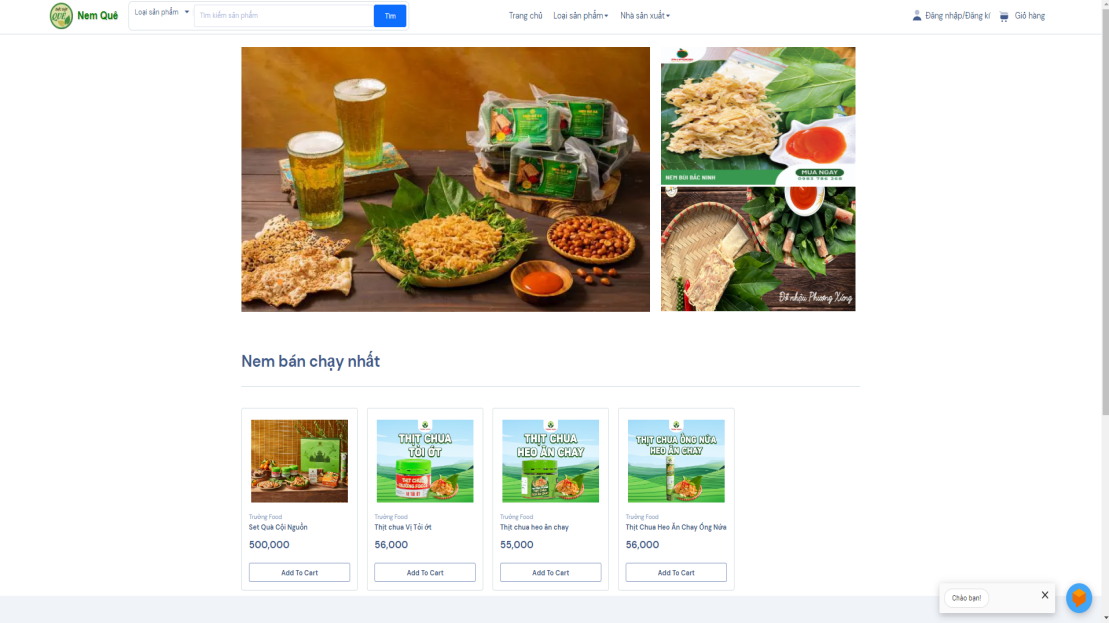


**Hình 3.27****. Giao diện trang quản lý mã giảm giá**

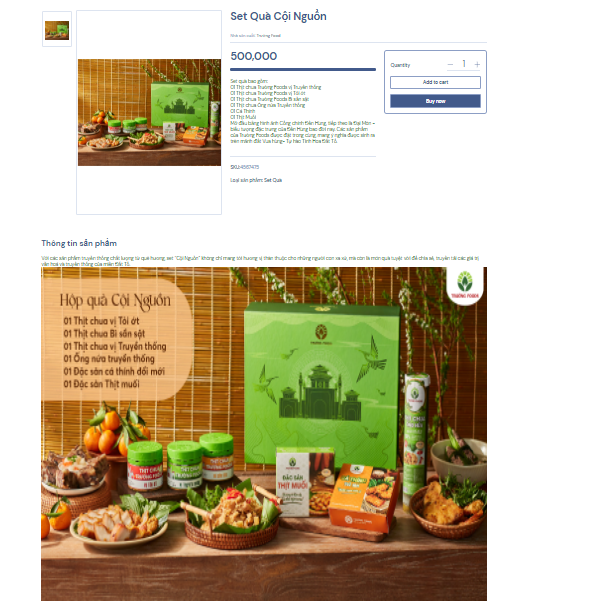


**Hình 3.28****. Giao diện trang quản lý đơn hàng**

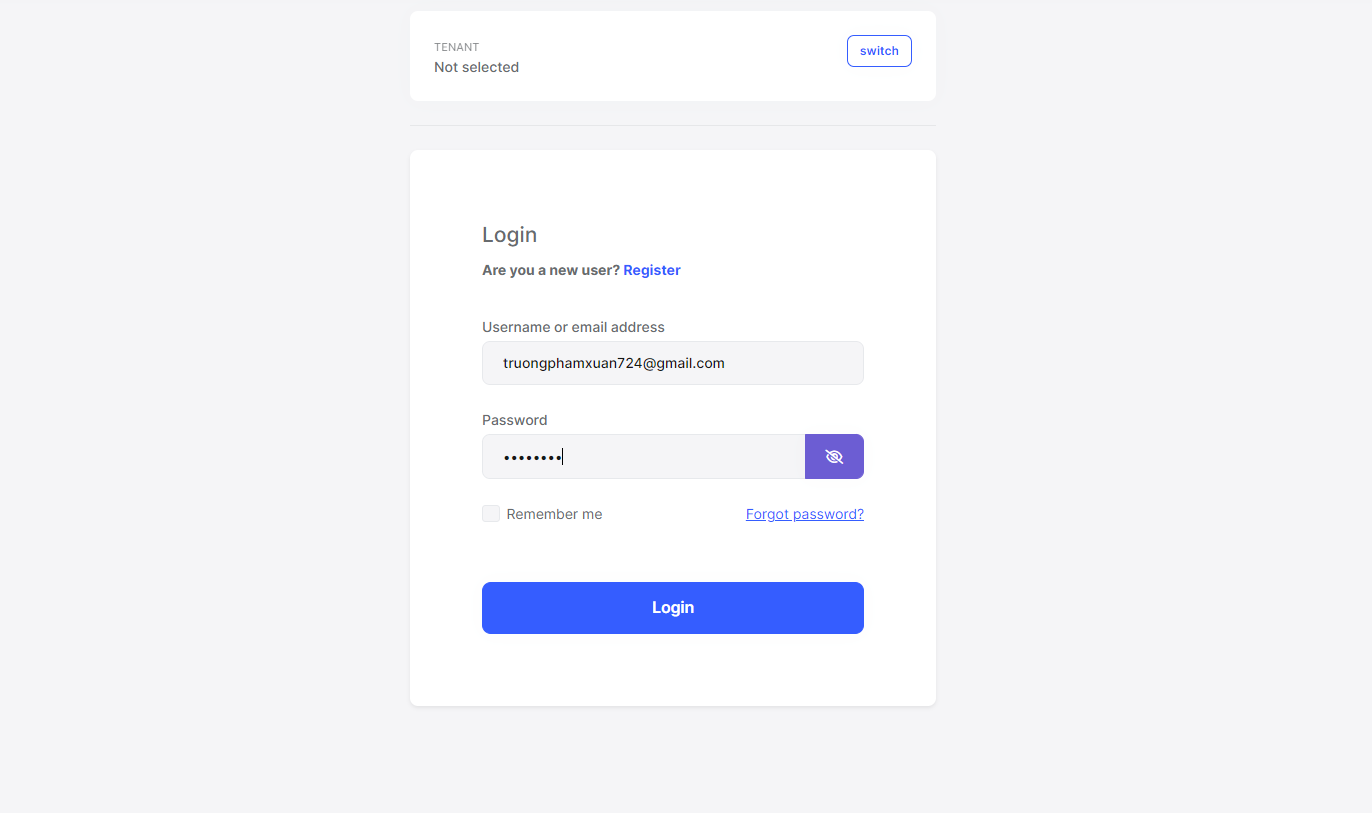
**3.7.2. Giao diện trang Public**



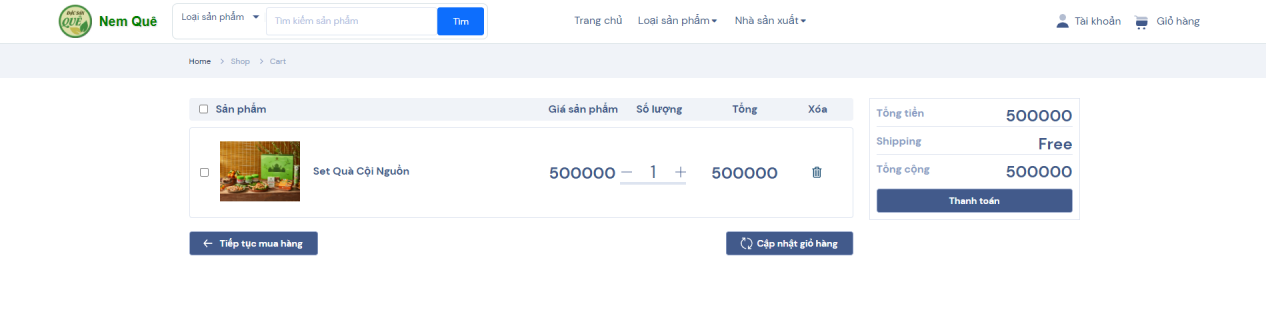
**Hình 3.29****. Giao diện trang chủ**



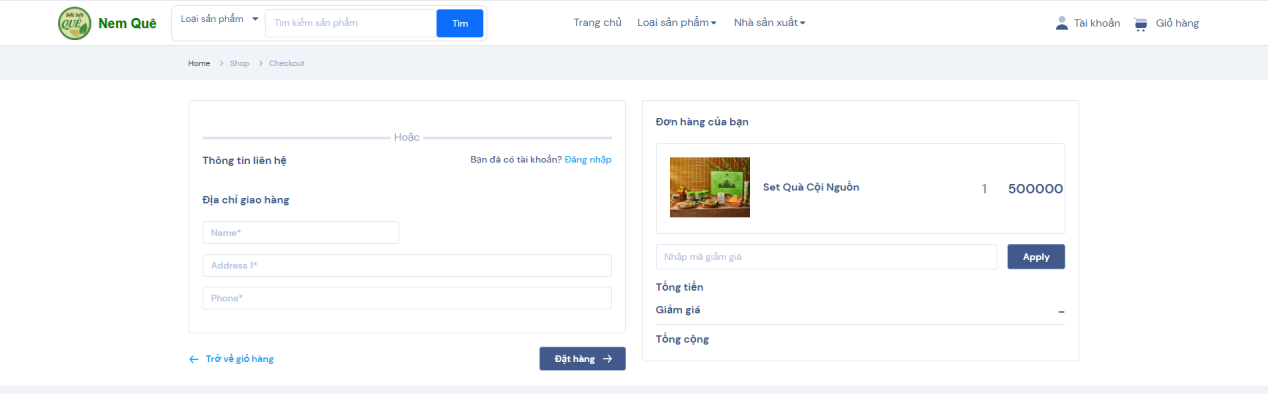
**Hình 3.30****. Giao diện xem chi tiết sản phẩm**



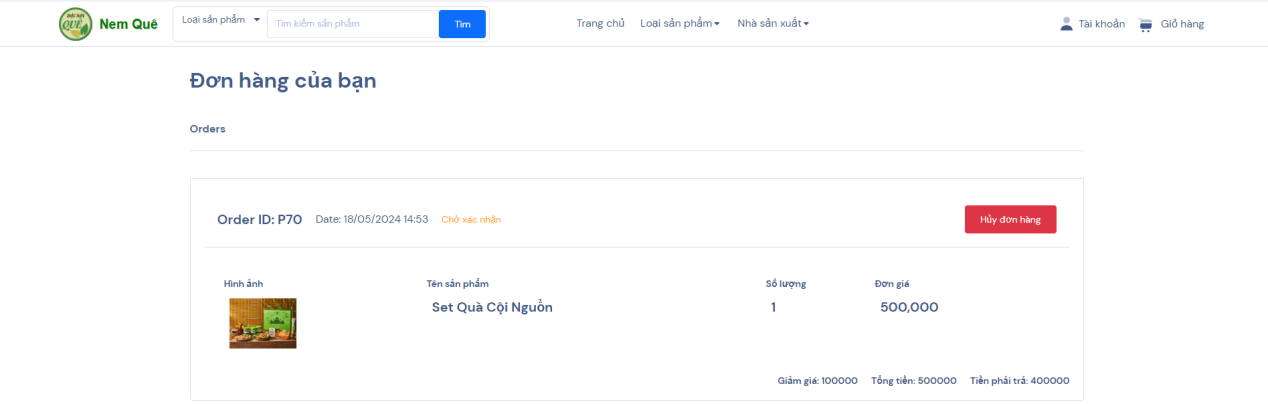
**Hình 3.31****. Giao diện trang Login**



**Hình 3.32****. Giao diện Giỏ hàng**



**Hình 3.33****. Giao diện thanh toán**



**Hình 3.34****. Giao diện xem đơn hàng**

**KẾT LUẬN**

**Kiến thức, kỹ năng và kinh nghiệm đã học được**

- Kỹ năng phân chia công việc và lên kế hoạch: Trong quá trình làm dự án, em đã phát triển khả năng phân chia công việc và lập kế hoạch một cách hiệu quả, giúp tối ưu hóa thời gian và tài nguyên đồng thời đảm bảo tiến độ công việc được thực hiện đúng hạn.

- Hiểu biết về mã nguồn mở: em đã nắm vững khái niệm về mã nguồn mở, nhận thức được các ưu nhược điểm và ứng dụng của phần mềm mã nguồn mở, giúp em nhận thấy sức mạnh của cộng đồng mã nguồn mở và lợi ích mà nó mang lại trong phát triển phần mềm.

- Ứng dụng phần mềm mã nguồn mở: em đã áp dụng các phần mềm mã nguồn mở để xây dựng một ứng dụng hoàn chỉnh, tiết kiệm thời gian và khai thác các công cụ và tài nguyên sẵn có từ cộng đồng lập trình mã nguồn mở.

- Kỹ năng đọc và tìm kiếm tài liệu bằng tiếng Anh: Kỹ năng này giúp em tiếp cận được thông tin mới nhất và nắm vững các công nghệ tiên tiến trong lĩnh vực phát triển phần mềm.

- Kỹ năng tư duy logic và phân tích: Đây là kỹ năng quan trọng để giải quyết các vấn đề phức tạp trong quá trình phát triển phần mềm.

- Hiểu biết về kiến trúc phần mềm: em đã học và áp dụng các nguyên lý thiết kế phần mềm để xây dựng hệ thống ổn định và linh hoạt.

- Kỹ năng sử dụng Git và quản lý mã nguồn: em đã hiểu rõ về quy trình làm việc với Git, giúp theo dõi và quản lý các thay đổi trong mã nguồn một cách hiệu quả.

- Kiến thức về bảo mật thông tin: em đã nắm vững các nguyên tắc cơ bản về bảo mật thông tin và áp dụng chúng vào quá trình phát triển phần mềm.

- Kỹ năng giải quyết vấn đề và thích ứng với công nghệ mới: em đã đối mặt với nhiều thách thức và học được cách tìm ra các giải pháp hiệu quả.

- Kỹ năng tự học và nâng cao kiến thức\*: em đã phát triển khả năng tự học và tìm hiểu về các công nghệ mới thông qua việc đọc sách, tham gia các khóa học trực tuyến và thực hành trên dự án.

**Kết quả đạt được**

- Xây dựng và triển khai ứng dụng web hoàn chỉnh: Với kiến thức về Angular và ASP.Net Core, em đã phát triển và triển khai thành công một ứng dụng web đáp ứng nhanh chóng và hiệu quả.

- Phát triển giao diện người dùng tương tác: Sử dụng Angular, em đã tạo ra giao diện người dùng tương tác (UI) trực quan và dễ sử dụng, nâng cao trải nghiệm người dùng và tăng tính tương tác của ứng dụng.

- Phân tích và xử lý dữ liệu: Sử dụng các thư viện và công cụ phân tích dữ liệu, em đã phân tích và xử lý dữ liệu từ nhiều nguồn khác nhau một cách chính xác và linh hoạt.

- Tích hợp công nghệ mới: em luôn cập nhật và tích hợp các công nghệ mới vào dự án, giúp ứng dụng luôn cạnh tranh và tiên tiến.

- Nhận diện và khắc phục lỗi nhanh chóng: Với kỹ năng phân tích và debugging, em đã nhận diện và khắc phục lỗi nhanh chóng, đảm bảo ứng dụng luôn hoạt động ổn định.

- Quản lý các thư mục, thư viện chặt chẽ: em đã rèn luyện được kỹ năng quản lý thư mục và thư viện một cách chặt chẽ, giúp dễ dàng bảo trì và mở rộng mã nguồn.

- Thao tác với các API và nền tảng hiệu quả: em đã thực hiện các thao tác liên quan đến API và nền tảng một cách hiệu quả, từ kết nối và gửi yêu cầu đến xử lý dữ liệu trả về.

**Hướng phát triển**

- Xây dựng Website với nhiều chức năng hơn, hoàn thiện hơn về mặt giao diện.

- Tiếp tục trau dồi kiến thức để nâng cao trình độ của bản thân.

-Tiếp tục hoàn thiện các phân tích website bán nem.

**TÀI LIỆU THAM KHẢO**

Tiếng Việt

[1] Hoàng Quang Huy (2016), “*Giáo trình kiểm thử phần mềm*”, Nhà xuất bản Thống kê.

[2] Nguyễn Trường Sinh (2006), “*Tự học Javascript*”, Nhà xuất bản LĐXH.

Tiếng Anh

1. Charles P. Pfleeger, Shari Lawrence Pfleeger, Jonathan Margulies (2015), “*Security in Computing (Fifth edition)*”, Nhà xuất bản Prentice Hall.

Các nguồn tham khảo:

<https://angular.io/docs>

<https://docs.abp.io/en/abp/latest>

<https://primeng.org/>

<https://github.com/abpframework/abp>

<https://docs.abp.io/en/abp/latest/Startup-Templates/Application>

<https://viblo.asia/p/tim-hieu-ve-entity-attribute-value-pattern-eav-structural-pattern-Eb85okV452G>

<https://docs.abp.io/en/abp/latest/Best-Practices/Index>

<https://github.com/primefaces/sakai-ng>

https://dialogflow.cloud.google.com/#/agent/bottest-hcbj/integrations