|  |  |
| --- | --- |
|  | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
| NGUYỄN VĂN THỊNH | **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC** |
| **NGÀNH CÔNG NGHỆ THÔNG TIN** |
| **XÂY DỰNG WEBSITE QUẢN LÝ MỐI QUAN HỆ KHÁCH HÀNG**  **SỬ DỤNG ANGULAR VÀ ASP.NET CORE** |
|  | **CBHD :**  **ThS. Vũ Duy Giang** |
|  | **Sinh viên : Nguyễn Văn Thịnh** |
|  | **Mã số sinh viên : 2020600156** |
| CÔNG NGHỆ THÔNG TIN |  |
|  | Hà Nội – 2024 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------**  **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**  **NGÀNH CÔNG NGHỆ THÔNG TIN**  **XÂY DỰNG WEBSITE QUẢN LÝ MỐI QUAN HỆ KHÁCH HÀNG**  **SỬ DỤNG ANGULAR VÀ ASP.NET CORE**   |  |  | | --- | --- | | **CBHD** | **: ThS. Vũ Duy Giang** | | **Sinh viên** | **: Nguyễn Văn Thịnh** | | **Mã số sinh viên** | **: 2020600156** |     Hà Nội – 2024 |

**Mục lục**

[**DANH MỤC CÁC TỪ VIẾT TẮT** 1](#_Toc167111386)

[**DANH MỤC CÁC BẢNG** 1](#_Toc167111387)

[**DANH MỤC CÁC HÌNH, BIỂU ĐỒ** 1](#_Toc167111388)

[**LỜI CẢM ƠN** 3](#_Toc167111389)

[**LỜI NÓI ĐẦU** 4](#_Toc167111390)

[**MỞ ĐẦU** 5](#_Toc167111391)

[**1.** **GIỚI THIỆU CHUNG VỀ ĐỀ TÀI** 5](#_Toc167111392)

[**1.1.** **Tên đề tài** 5](#_Toc167111393)

[**1.2.** **Lý do chọn đề tài** 5](#_Toc167111394)

[**1.3.** **Tính cấp thiết của đề tài** 5](#_Toc167111395)

[**2.** **MỤC TIÊU VÀ PHƯƠNG PHÁP NGHIÊN CỨU** 6](#_Toc167111396)

[**2.1.** **Mục tiêu của đề tài** 6](#_Toc167111397)

[**2.2.** **Nội dung nghiên cứu** 6](#_Toc167111398)

[**2.3.** **Phương pháp luận và phương pháp nghiên cứu** 7](#_Toc167111399)

[2.3.1. Phương pháp luận 7](#_Toc167111400)

[2.3.2. Phương pháp nghiên cứu 8](#_Toc167111401)

[**3.** **BỐ CỤC DỰ KIẾN CỦA ĐỒ ÁN** 8](#_Toc167111402)

[**CHƯƠNG 1. TỔNG QUAN VỀ WEBSITE BÁN ĐỒ GIA DỤNG** 9](#_Toc167111403)

[**1.1.** **Giới thiệu về website bán hàng trực tuyến** 9](#_Toc167111404)

[1.1.1. Khái niệm 9](#_Toc167111405)

[1.1.2. Các thành phần cần quản trị: 9](#_Toc167111406)

[1.1.3. Tại sao cần quản lý mối quan hệ giữa 2 đối tượng này trong lĩnh vực kinh doanh: 10](#_Toc167111407)

[1.1.4. Khái niệm về website và lý do chọn website thay vì nền tảng khác? 10](#_Toc167111408)

[**1.2.** **Khảo sát** 11](#_Toc167111409)

[**1.3.** **Đối tượng nghiên cứu** 11](#_Toc167111410)

[**1.4.** **Phạm vi nghiên cứu** 12](#_Toc167111411)

[**1.5.** **Xác định yêu cầu** 12](#_Toc167111412)

[1.5.1. Mô tả yêu cầu chức năng 12](#_Toc167111413)

[1.5.2. Mô tả yêu cầu phi chức năng 13](#_Toc167111414)

[**1.6.** **Kết quả mong muốn đạt được của đề tài** 13](#_Toc167111415)

[**1.7.** **Kết luận chương** 13](#_Toc167111416)

[**CHƯƠNG 2. TÍCH HỢP CÔNG NGHỆ XÂY DỰNG PHẦN MỀM** 14](#_Toc167111417)

[**2.1.** **Ngôn ngữ và Framework sử dụng** 14](#_Toc167111418)

[2.1.1. Framework Angular 14](#_Toc167111419)

[1. HTML, CSS, JS: 14](#_Toc167111420)

[2. Framework Angular: 15](#_Toc167111421)

[2.1.2. Khái niệm về API và ASP.NET Core 17](#_Toc167111422)

[2.3.1. Cơ sở dữ liệu 20](#_Toc167111423)

[2.3.2. MySQL là gì? 20](#_Toc167111424)

[**2.3.** **Các phần mềm và công cụ sử dụng** 21](#_Toc167111425)

[2.3.1. Visual Studio Code (VSCode) 21](#_Toc167111426)

[2.3.2. Microsoft Visual Studio 23](#_Toc167111427)

[2.3.3. MySQL WorkBench: 24](#_Toc167111428)

[2.3.4. Quản lý mã nguồn (GitHub) 26](#_Toc167111429)

[**2.4.** **Kết luận chương 2** 27](#_Toc167111430)

[**CHƯƠNG 3. XÂY DỰNG WEBSITE QUẢN LÝ MỐI QUAN HỆ KHÁCH HÀNG** 28](#_Toc167111431)

[**3.1.** **Phân tích hệ thống** 28](#_Toc167111432)

[**3.2.** **Xây dựng biểu đồ Use case (UC)** 28](#_Toc167111433)

[3.2.1. Xác định tác nhân của hệ thống 28](#_Toc167111434)

[3.2.2. Xác định tác nhân và các chức năng của tác nhân 28](#_Toc167111435)

[3.2.3. Biểu đồ use case tổng quan 29](#_Toc167111436)

[**3.3.** **Mô tả chi tiết Use case** 29](#_Toc167111437)

[**3.4.** **Giao diện Website quản lý:** 74](#_Toc167111438)

[**3.5.** **Kiểm thử** 80](#_Toc167111439)

[3.8.1 Kế hoạch kiểm thử 80](#_Toc167111440)

[3.8.2 Kiểm thử chức năng 81](#_Toc167111441)

[3.8.3 Kết quả kiểm thử 84](#_Toc167111442)

[**3.6.** **Kết luận chương 3** 84](#_Toc167111443)

[**KẾT LUẬN** 85](#_Toc167111444)

[**TÀI LIỆU THAM KHẢO** 89](#_Toc167111445)

# **DANH MỤC CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **Từ viết tắt** | **Ý nghĩa** |
| 1 | CSDL | Cơ sở dữ liệu |
| 2 | UC | Use case |
| 3 | VS Code | Visual Studio Code |

# **DANH MỤC CÁC BẢNG**

[Bảng 3.1: Bảng tác nhân và chức năng của tác nhân 28](#_Toc167017587)

[Bảng 3.2: Bảng kiểm thử chức năng 67](#_Toc167017588)

# **DANH MỤC CÁC HÌNH, BIỂU ĐỒ**

[Hình 2.1: VueJS 13](#_Toc167019106)

[Hình 2.2: Hệ quản trị cơ sở dữ liệu MariaDb 19](#_Toc167019107)

[Hình 2.3: Hình ảnh phần mềm lập trình VSCode 21](#_Toc167019108)

[Hình 2.4: Hình ảnh phần mềm lập trình microsoft visual studio 23](#_Toc167019109)

[Hình 2.5: Hình ảnh phần mềm DbForge Studio for MySQL 24](#_Toc167019110)

[Hình 2.6: Hình ảnh công cụ quản lý mã nguồn GitHub 26](#_Toc167019111)

[Hình 3.1: Biểu đồ use case tổng quan 30](#_Toc167019112)

[Hình 3.2: Kiến trúc cơ sở dữ liệu 51](#_Toc167019113)

[Hình 3.3: Giao diện trang chủ 52](#_Toc167019114)

[Hình 3.4: Giao diện đăng ký 53](#_Toc167019115)

[Hình 3.5: Giao diện đăng nhập 53](#_Toc167019116)

[Hình 3.6: Giao diện xem sản phẩm theo danh mục 54](#_Toc167019117)

[Hình 3.7: Giao diện chi tiết sản phẩm 55](#_Toc167019118)

[Hình 3.8: Giao diện giỏ hàng 56](#_Toc167019119)

[Hình 3.9: Giao diện đặt hàng 57](#_Toc167019120)

[Hình 3.10: Giao diện đơn hàng của tôi 58](#_Toc167019121)

[Hình 3.11: Giao diện thông tin chi tiết đơn hàng 59](#_Toc167019122)

[Hình 3.12: Giao diện thông tin tài khoản 60](#_Toc167019123)

[Hình 3.13: Giao diện danh sách địa chỉ 60](#_Toc167019124)

[Hình 3.14: Giao diện thêm mới địa chỉ 61](#_Toc167019125)

[Hình 3.15: Giao diện đăng nhập (phía Admin) 62](#_Toc167019126)

[Hình 3.16: Giao diện tổng quan 62](#_Toc167019127)

[Hình 3.17: Giao diện quản lý sản phẩm 63](#_Toc167019128)

[Hình 3.18: Giao diện quản lý danh mục 64](#_Toc167019129)

[Hình 3.19: Giao diện quản lý đơn hàng 64](#_Toc167019130)

[Hình 3.20: Giao diện thông tin chi tiết đơn hàng (phía admin) 65](#_Toc167019131)

[Hình 3.21: Giao diện quản lý khách hàng 65](#_Toc167019132)

[Hình 3.22: Giao diện thống kê doanh thu 66](#_Toc167019133)

[Hình 3.23: Giao diện quản lý khuyến mãi 66](#_Toc167019134)

# **LỜI CẢM ƠN**

Trước khi vào nội dung của báo cáo, em xin chân thành cảm ơn thầy Vũ Huy Giang là người đồng hành, hỗ trợ em trong quá trình thực tập và trong cả làm đồ án tốt nghiệp. Nhờ có sự hướng dẫn tận tình của thầy, mà nghiên cứu của em được rõ ràng và hoàn thiện hơn. Những giải pháp và ý tưởng được triển khai theo hướng đi tốt nhất.

Tiếp theo, em xin cảm ơn Nhà trường, đặc biệt là các thầy cô trong khoa đã tạo điều kiện hết sức để báo cáo được hoàn thành tốt nhất. Những bài giảng, lý thuyết của các thầy cô là nền tảng để em hoàn thành được đồ án này.

Cuối cùng em xin cảm ơn quý công ty Cổ phần Misa, nhờ có quá trình thực tập, cũng như làm việc trong môi trường kỷ luật, năng động. Hơn nữa được cọ sát với những quy trình, dự án thực tế, giúp cho em phát triển những kỹ năng chuyên ngành và kỹ năng mềm.

Lời cuối cùng em xin chúc thầy cô luôn dồi dào sức khỏe, luôn vui vẻ và thành công trong cuộc sống!

Em xin chân thành cảm ơn!

Sinh viên thực hiện

Nguyễn Văn Thịnh

# **LỜI NÓI ĐẦU**

Hiện nay, công việc kinh doanh luôn được đẩy mạnh và một công ty luôn hướng tới việc mở rộng tệp khách hàng, tìm kiểm được nhiều tiềm năng nhất có thể. Với sự mở rộng như vậy, việc quản lý những khối dữ liệu là điều vô cùng khó khắn, nếu chỉ sử dụng giấy tờ, hay thậm chí là excel. Việc liên kết, móc nối chúng lại để sử dụng một cách hiệu quả nhất càng khó khăn hơn. Giấy tờ đôi khi có thể mất và tìm kiếm lâu hơn. Nên việc lưu trữ, quản lý những khách hàng này, chăm sóc để tìm ra cơ hội bàn hàng là điều hết sức cần thiết. Từ đó, em muốn tạo ra một phần mềm có thể giúp họ quản lý được những khách hàng, tiềm năng họ đã tìm được. Quản lý được hàng hóa trong kho, chính sách giá cho các mặt hàng khác nhau. Và mục đích cuối cùng là tạo được ra đơn hàng từ những khách hàng trên và quản lý tình trạng của những đơn hàng đó.

Đồ án của em tập trung vào việc xây dựng một hệ thống website quản lý chặt chẽ, cung cấp thông tin chính xác và dễ dàng thao tác với các chức năng đơn giản, nhanh chóng.

Em đã tiến hành nghiên cứu sâu rộng về nhu cầu của người dùng, xây dựng cấu trúc hệ thống và triển khai các công nghệ tiên tiến để đảm bảo tính linh hoạt, an toàn và hiệu quả cho website.

Trong báo cáo này, em sẽ trình bày về quá trình phát triển, kiến trúc hệ thống, chi tiết và cách thức kiểm thử, đánh giá hiệu suất của sản phẩm.

Để hoàn thành được đồ án tốt nghiệp này, em xin được gửi lời cảm ơn chân thành đến các thầy cô trong khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội đã tận tình giảng dạy và trang bị kiến thức cho em trong suốt thời gian em học tập tại trường. Giáo viên hướng dẫn đề tài – ThS. Vũ Duy Giang, giảng viên Khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội – đã tận tụy hết lòng giúp đỡ, hướng dẫn, chỉ dẫn tận tình để giúp em hoàn thành được đồ án.

# **MỞ ĐẦU**

# **GIỚI THIỆU CHUNG VỀ ĐỀ TÀI**

## **Tên đề tài**

Đề tài: “**Xây dựng website quản lý mối quan hệ khách hàng sử dụng Angular và ASP.NET Core**”

## **Lý do chọn đề tài**

Với sự phát triển không ngừng của các ngành kinh doanh nói chung và bán hàng nói riêng. Giờ đây, mọi người không cần phải đi trực tiếp đến các cửa hàng để mua sắm tiết kiệm rất nhiều thời gian và công sức. Cùng sự phát triển vượt bậc của internet, ngành kinh doanh online đang trở thành xu thế của thế giới. Các công ty không chỉ tập trung vào quảng bá sản phẩm, mà còn chủ động trong việc tìm kiếm, chăm sóc khách hàng. Đơn hàng được mua online dẫn đến việc phải kiểm soát trạng thái, số lượng của các đơn hàng. Từ đó, cho quản lý nắm rõ được tình hình hoạt động kinh doanh của công ty và đưa ra chiến lược hợp lý. Nếu chỉ lưu trên giấy tờ, hay thậm chí là excel thì việc quản lý vẫn sẽ gặp khó khăn khi khó tra cứu hoặc không dễ dàng với những người mới.

Do vậy, cần có một ứng dụng để quản lý từ lúc khách hàng chỉ là tiềm năng cho đến khi khách hàng đã mua hàng và sau đó là chăm sóc khách hàng. Mục đích chính của kinh doanh là bán được hàng, do vậy cũng cần quản lý cả trạng thái đơn hàng mà khách hàng mua. Giúp nhân viên kinh doanh dễ dàng quản lý khách hàng của mình và cấp trên quản lý được tình hình kinh doanh của công ty.

## **Tính cấp thiết của đề tài**

Không thiếu những ứng dụng được xây dựng để quản công việc này. Nhưng những ứng dụng này hoặc là bằng tiếng anh hoặc là dùng cho các doanh nghiệp có lượng khách hàng tương đối lớn và đa dạng. Nhiều khi đó là không cần thiết cho những doanh nghiệp hay startup nhỏ hoặc một nhân viên cần đảm nhiệm nhiều vị trí khác nhau.

Do đó, cần một ứng dụng gọn nhẹ, dễ thao tác và làm quen là điều cần thiết. Ứng dụng chỉ cần đủ những nghiệp vụ cần thiết cho kinh doanh, không cần kết nối đến bên thứ 3, marketing, email, ….

# **MỤC TIÊU VÀ PHƯƠNG PHÁP NGHIÊN CỨU**

## **Mục tiêu của đề tài**

Xây dựng và triển khai một trang web hiện đại, giàu chức năng quản lý, thao tác dễ dàng, thuận tiện. Người dùng lưu trữ và phân loại được dữ liệu dễ dàng nhất.

- Tạo ra một giao diện người dùng thân thiện và dễ sử dụng, giúp người dùng tìm kiếm, xem thông tin chi tiết của các phân hệ khách hàng, tiềm năng, đơn hàng.

- Xây dựng hệ thống quản lý dữ liệu với độ tin cậy và chặt chẽ với nhau, Xem được các số liệu liên quan đến dữ liệu một cách nhau chóng và tổng quan.

- Tích hợp các chức năng cập nhật dữ liệu nhanh chóng và tối ưu thao tác, và ràng buộc các dữ liệu liên quan với nhau.

- Tích hợp hệ thống phân quyền: Phân quyền chức năng dựa trên các quyền mà người dùng được phép truy cập, thay đổi dữ liệu các phân hệ.

## **Nội dung nghiên cứu**

- Phân tích thị trường: Nghiên cứu và phân tích thị trường đồ gia dụng trực tuyến để hiểu rõ về xu hướng tiêu dùng, nhu cầu của người dùng, và các đối thủ cạnh tranh trong ngành.

- Phân tích yêu cầu và tính năng: Thu thập và phân tích yêu cầu từ khách hàng để xác định các tính năng cần thiết cho trang web bao gồm chức năng tìm kiếm, thao tác dữ liệu khách hàng, sửa đổi, cập nhật thì tự động thay đổi dữ liệu tương ứng. Chuyển đổi từ tiềm năng sang khách hàng. Nhanh chóng thêm được đơn hàng từ khách hàng mà không phải nhập lại thông tin. Cho phép quản lý xem được số liệu báo cáo.

- Thiết kế giao diện người dùng (UI/UX): Phát triển bản thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX) dựa trên yêu cầu và phản hồi từ khách hàng để chức năng thao tác nhanh và thân thiện nhất.

- Phát triển và kiểm thử: Xây dựng và phát triển trang web dựa trên bản thiết kế đã được xác nhận, bao gồm phát triển hệ thống backend và frontend. Sau đó, thực hiện các bước kiểm thử chất lượng để đảm bảo tính ổn định và hiệu suất của trang web.

- Quản lý dữ liệu và bảo mật: Xây dựng hệ thống quản lý cơ sở dữ liệu cho phép dữ liệu được bảo mật đối với người dùng trong cùng 1 công ty.

- Triển khai và duy trì: Triển khai trang web vào môi trường sản xuất và duy trì hoạt động của trang web sau khi hoàn thành, bao gồm cập nhật nội dung, quản lý dịch vụ và hỗ trợ kỹ thuật cho người dùng.

Toàn bộ những nội dung nghiên cứu trên sẽ tập trung vào việc phát triển một trang web quản lý quan hệ khách hàng toàn diện và chất lượng cao, từ việc phân tích thị trường, thiết kế giao diện, phát triển tính năng, đến triển khai và duy trì sau này.

## **Phương pháp luận và phương pháp nghiên cứu**

### Phương pháp luận

* Nghiên Cứu Phát Triển (Development Research): Sử dụng phương pháp này để phát triển trang web từ đầu, từ việc thiết kế giao diện đến triển khai và duy trì. Quá trình này bao gồm việc tiến hành các bước lập trình, kiểm thử và tối ưu hóa để đảm bảo tính chính xác và hiệu quả của trang web.
* Phân Tích Yêu Cầu (Requirements Analysis): Áp dụng phương pháp này để xác định và hiểu rõ các yêu cầu của khách hàng và người dùng cuối. Quá trình này giúp xác định tính năng cần có của trang web và định hình quy trình phát triển.
* Thiết Kế Giao Diện Người Dùng (User Interface Design): Sử dụng phương pháp này để tạo ra giao diện người dùng (UI) hấp dẫn và dễ sử dụng cho trang web. Quá trình này bao gồm việc thiết kế các trang, bố cục, màu sắc và các yếu tố tương tác.

### Phương pháp nghiên cứu

* + Nghiên Cứu Thực Nghiệm: Thực hiện nghiên cứu thực nghiệm để phát triển và kiểm thử các tính năng của trang web bán đồ gia dụng. Quy trình này bao gồm việc thiết kế các thí nghiệm, thu thập dữ liệu và phân tích kết quả để đánh giá hiệu suất và hiệu quả của trang web.
  + Nghiên Cứu Thực Địa: Tiến hành nghiên cứu thực địa để thu thập thông tin về thị trường đồ gia dụng trực tuyến, nhu cầu của người dùng và các đối thủ cạnh tranh. Quy trình này giúp hiểu rõ về môi trường kinh doanh và yêu cầu của thị trường.
  + Phân Tích Dữ Liệu: Sử dụng phương pháp phân tích dữ liệu để đánh giá và phân tích các thông tin thu thập được từ trang web, bao gồm hành vi người dùng, giao dịch bán hàng và phản hồi từ khách hàng. Quy trình này giúp hiểu rõ hơn về cách người dùng tương tác với trang web và cải thiện trải nghiệm của họ.

# **BỐ CỤC DỰ KIẾN CỦA ĐỒ ÁN**

Bố cục dự kiến của đồ án bao gồm 3 chương

Chương 1: Tổng quan về website quản lý mỗi quan hệ khách hàng

Chương 2: Công nghệ sử dụng Angular và Asp.Net Core

Chương 3: Xây dựng website bán đồ gia dụng

# **CHƯƠNG 1. TỔNG QUAN VỀ WEBSITE BÁN ĐỒ GIA DỤNG**

## **Giới thiệu về website bán hàng trực tuyến**

### Khái niệm

Quản trị quan hệ khách hàng là công việc tiếp cận và giao tiếp với khách hàng, quản lý các thông tin của khách hàng từ đó phục vụ khách hàng tốt hơn và thiết lập mối quan hệ bền vững với họ. Hay nói cách khác, dựa trên dữ liệu và thông tin thu được từ khách hàng, cửa hàng sẽ đề ra được chiến lược chăm sóc khách hàng hợp lý và hiệu quả.

### Các thành phần cần quản trị:

* Quản lý/chủ cửa hàng: Đây sẽ là những người xây dựng quy trình quản lý khách hàng cho cửa hàng. Việc xây dựng quy trình không chỉ giúp nhân viên có định hướng làm việc tốt hơn mà còn giúp quản lý dễ dàng hơn trong việc theo dõi và đánh giá hiệu quả.
* Bộ phận bán hàng: Đây là những người trực tiếp nói chuyện với khách hàng. Nhân viên bán hàng sẽ xử lý các yêu cầu của khách hàng, tư vấn cho họ những sản phẩm phù hợp nhất với nhu cầu, mong muốn. Bên cạnh đó bộ phận này còn có nhiệm vụ ghi lại thông tin về khách hàng để phục vụ cho công việc chăm sóc khách hàng về sau.
* Bộ phận chăm sóc khách hàng (nếu có): Các cửa hàng còn có thể có nhân viên chăm sóc khách hàng nhằm đưa đến dịch vụ sau bán tốt nhất. Họ sẽ là những người phải lắng nghe các vấn đề của khách hàng và đưa ra các phương hướng xử lý các vướng mắc đó một cách nhanh chóng và hiệu quả. Hơn nữa, bạn còn phải đảm nhiệm các công việc như gọi điện, nhắn tin hay gửi email để thể hiện sự quan tâm tốt nhất đến khách hàng.
* Kết hợp với các bộ phận khác (nếu có): Nếu các cửa hàng nhỏ không đáp ứng đủ yêu cầu về nhân lực, bộ phận bán hàng và bộ phận chăm sóc khách hàng có thể kết hợp lại với nhau. Nhân viên tại cửa hàng vừa đảm nhiệm việc bán hàng vừa có trách nhiệm chăm sóc khách hàng.

### Tại sao cần quản lý mối quan hệ giữa 2 đối tượng này trong lĩnh vực kinh doanh:

* Ích lợi đối với nhà quản lý: Việc theo dõi thông tin khách hàng và có được những dữ liệu về thói quen, hành vi của khách sẽ giúp chủ cửa hàng đưa ra những quyết định kinh doanh phù hợp hơn.
* Ích lợi đối đối với nhân viên: Công việc quản lý quan hệ khách hàng sẽ hỗ trợ cho nhân viên trong việc thực hiện đơn hàng cũng như hỗ trợ khách hàng. Bạn sẽ có thể từ thông tin khách hàng để đơn giản hóa quá trình bán hàng, từ đó việc bán hàng trở nên dễ dàng hơn rất nhiều.
* Ích lợi đối với cửa hàng: Quản trị quan hệ khách hàng hiệu quả sẽ giúp cửa hàng duy trì được lượng khách hàng nhất định cũng như giành được niềm tin của những khách hàng mới. Nhờ đó chi phí tiếp thị được giảm đi một cách đáng kể và các công việc khác cũng diễn ra suôn sẻ hơn.

### Khái niệm về website và lý do chọn website thay vì nền tảng khác?

Website là một tập hợp các trang tin có chưa nội dung dạng văn bản, chữ số, âm thanh, hình ảnh, … được lưu trữ trên máy chủ (web server) và có thể truy cập qua internet. Đặc biệt là browser.

Ngày nay, website ngày càng phổ biến vì những tiện lợi mà nó mang lại:

* Nếu so sánh với 1 ứng dụng trên điện thoại, hay desktop thì trang website không cần cập nhật ứng dụng khi nhà phát triển phát triển sản phẩm mới.
* Dễ dàng truy cập qua cho cả điện thoại và desktop mà chỉ cần có brower.
* Không cần lo lắng nhiều về version vì tất cả đều là mới nhất.

Nhưng cũng tiềm tàng một số rủi ro và khó khăn:

* Nếu lượng người truy cập lớn, trang web dễ bị sập hơn so với ứng dụng trên nền tảng khác.
* Mức độ bảo mật của website cũng kém hơn ứng dụng của desktop và điện thoại vì dữ liệu dễ dàng bị nhìn thấy thông qua các request.
* Nếu một ứng dụng xử lý nhiều giao diện, nhà phát triển không có kinh nghiệm cũng dễ dàng gây ra tình trạng giật lag.

Chương trình tạo ra phải nhỏ gọn, không cần xử lý nhiều về mặt giao diện. Nên dựng ứng dụng trên website là lựa chọn phù hợp hơn cả.

## **Khảo sát**

Hiện nay, những trang website quản lý quan hệ khách ngày càng phát triển rộng rãi để đáp ứng được nhu cầu của nhân viên kinh doanh. Chính vì thế, em có thể tham khảo được rất nhiều các chức năng để hình thành lên một website quản lý mỗi quan hệ khách hàng. Dưới đây là một số chức năng chính của website:

* Quản lý tiềm năng
* Quản lý khách hàng
* Quản lý liên hệ
* Quản lý chính sách giá
* Quản lý đơn hàng
* Quản lý hàng hóa
* Thống kê báo cáo
* Phân quyền người dùng

## **Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài này bao gồm:

* Quản lý kinh doanh cần quản lý tình hình kinh doanh của nhân viên.
* Nhân viên kinh doanh cần quản lý khách hàng, tiềm năng và đơn hàng mà khách hàng đã đặt.
* Nhân viên chăm sóc khách hàng cần quản lý khách hàng để theo dõi, chăm sóc cho họ.
* Các website quản lý mỗi quan hệ khách hàng trong và ngoài nước.

## **Phạm vi nghiên cứu**

Phạm vi nghiên cứu sẽ tập trung vào việc tìm hiểu những nghiệp vụ của kinh doanh với khách hàng, từ đó tạo ra các tính năng phù hợp với yêu cầu nghiệp vụ.

## **Xác định yêu cầu**

### Mô tả yêu cầu chức năng

Cần phân quyền đối với các nhân viên có quyền xem, sửa, xóa, tùy chỉnh.

* Chức năng đăng nhập, đăng xuất: cho phép người dùng đăng nhập, đăng xuất để xác thực chính xác vị trí, nhân viên đang nắm giữ.
* Chức năng cập nhật quyền cho từng nhân viên, từng vị trí: cho phép cấp độ cao nhất phân quyền cho các cấp độ dưới.

Nhân viên cần quản lý khách hàng tiềm năng và hàng hóa họ mua.

* Chức năng quản lý tiềm năng: cho phép người dùng xem, sửa, xóa thông tin các khách hàng tiềm năng.
* Chức năng sinh khách hàng: cho phép người dùng mapping những thông tin của tiềm năng và chuyển đổi sang khách hàng. Không cần phải nhập lại những thông tin đã có.

Nhân viên cần quản lý khách hàng của mình và đơn hàng của họ.

* Chức năng quản lý khách hàng: cho phép người dùng xem, sửa, xóa thông tin khách hàng và liên kết đến đơn hàng của họ.
* Chức năng sinh đơn hàng: Cho phép khách hàng tạo đơn hàng với thông tin của họ. Không cần nhập lại những thông tin đã có.

Vì các khách hàng sẽ có thông tin các người liên quan để giao tiếp, liên hệ.

* Chức năng quản lý liên hệ: cho phép người dùng xem, thêm, sửa, xóa thông tin liên hệ.

Nhân viên cần quản lý đơn hàng của khách hàng.

* Chức năng quản lý đơn hàng: cho phép người dùng xem, thêm, sửa, xóa thông tin liên hệ. Các hàng hóa mà khách hàng đặt.

Nhân viên cần quản lý hàng hóa trong kho. Các chính sách giá áp dụng cho từng đợt sự kiện hay cho khách hàng riêng biệt.

* Chức năng quản lý hàng hóa: cho phép người dùng xem, thêm, sửa, xóa thông tin hàng hóa.
* Chức năng chính sách giá: cho phép người dùng xem, thêm, sửa, xóa chính sách giá và hiển thị giá của hàng hóa đã được áp chính sách giá.

### Mô tả yêu cầu phi chức năng

* Có hiệu năng cao, thời gian phản hồi nhanh, request dưới 1s cho những chức năng lấy về, dưới 2s cho những chức năng lưu.
* Giao diện người dùng hiện đại, thân thiện và dễ dàng thao tác tiện ích, chú trọng UX.
* Có khả năng đáp ứng 100 người truy cập cùng lúc.
* Bảo mật chống các cuộc tấn công.
* Backup dữ liệu 1 tuần/lần tránh mất mát dữ liệu.
* Tuân thủ quy định của pháp luật.

## **Kết quả mong muốn đạt được của đề tài**

* Phát triển và triển khai một trang web chuyên nghiệp, có hiệu quả đáp ứng được nghiệp vụ quản lý dữ liệu khách hàng của kinh doanh, đáp ứng được nghiệp vụ theo dõi tiến độ, quản lý tình hình kinh doanh của quản lý.
* Tạo ra một giao diện người dùng thân thiện và dễ sử dụng, giúp người dùng tìm kiếm, xem thông tin chi tiết nhanh chóng, dễ dàng tạo ra, linh hoạt trong việc sửa đổi dữ liệu.
* Tích hợp các tính năng an toàn và bảo mật tránh việc thất thoát dữ liệu, dữ liệu ràng buộc với nhau.
* Tạo ra một cơ sở hạ tầng kỹ thuật mạnh mẽ và linh hoạt, có thể mở rộng và phát triển trong tương lai để đáp ứng nhu cầu mở rộng kinh doanh tìm kiếm cơ hội và tạo đơn hàng.

## **1.7. Kết luận chương**

Trong chương 1, em đã trình bày về các khái niệm và lợi ích của website quản lý mối quan hệ khách hàng, khảo sát về đề tài, nghiên cứu về phạm vi và đối tượng nghiên cứu, xác định các yêu cầu thiết và kết quả mong muốn của đề tài.

# **CHƯƠNG 2. TÍCH HỢP CÔNG NGHỆ XÂY DỰNG PHẦN MỀM**

## **Ngôn ngữ và Framework sử dụng**

### Framework Angular

#### **Khái niệm về Framework Angular**



Hình 2.1: Angular

Trước khi đi vào framework Angular, chúng ta cần phải biết khái niệm về những kỹ thuật cấu tạo lên website và framework này:

1. HTML, CSS, JS:

* HTML là chữ viết tắt cho “HyperText Markup Language”, hay còn được gọi là "Ngôn ngữ Đánh dấu Siêu văn bản". Nó là một ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web với các mẩu thông tin được trình bày trên World Wide Web. Cùng với CSS và JavaScript, HTML tạo ra bộ ba nền tảng kỹ thuật cho World Wide Web. HTML được định nghĩa như là một ứng dụng đơn giản của SGML và được sử dụng trong các tổ chức cần đến các yêu cầu xuất bản phức tạp. HTML đã trở thành một chuẩn Internet do tổ chức World Wide Web Consortium (W3C) duy trì.
* CSS là chữ viết tắt cho “Cascading Style Sheets”, được sử dụng để mô tả giao diện và định dạng của một tài liệu viết bằng ngôn ngữ đánh dấu. Nó cung cấp một tính năng bổ sung cho HTML. Nó thường được sử dụng với HTML để thay đổi style của trang web và giao diện người dùng. Nó cũng có thể được sử dụng với bất kỳ loại tài liệu XML nào bao gồm cả XML đơn giản, SVG và XUL.
* JavaScript là một ngôn ngữ lập trình kịch bản đa nền tảng (Cross-platform). Nó chủ yếu được sử dụng để tăng cường cung cấp trải nghiệm thân thiện hơn với người dùng website như là bao gồm các trang web cập nhật tự động, cải tiến giao diện người dùng như menu và hộp thoại, hình động, đồ họa 2D và 3D, bản đồ tương tác, trình phát video... JavaScript là một ngôn ngữ nhỏ và nhẹ, chạy trong môi trường máy chủ lưu trữ. Nó có thể được kết nối với các đối tượng của môi trường để cung cấp kiểm soát chương trình đối với chúng.

Với bộ 3 ngôn ngữ trên. Chúng ta hoàn toàn có thể tạo nên giao diện người dùng tương tác với trang web của chúng ta nhưng tại sao lại sử dụng Angular. Hãy tìm hiểu ngay sau đây.

1. Framework Angular:

Angular là một mã nguồn mở viết bằng TypeScript và được sử dụng để thiết kế giao diện web (front – end). Angular được xây dựng, phát triển từ những năm 2009 và đang duy trì cho đến nay bởi Google. Angular thực sự được phát triển mạnh mẽ từ năm 2026, khi ra đời Angular 2. Coi như 1 phiên bản đập đi xây lại từ đầu. Được duy trì bởi GG nên có một cộng đồng lớn mạnh. Đến nay đã là Angular 18, cho thấy sự phát triển không ngừng nghỉ, ra đời những tính năng mới.

* Ưu điểm:
  + Kiến trúc component giống như lập trình hướng đối tượng nếu coi component như 1 class giúp cho lập trình viên dễ dàng sử dụng lại, dễ đọc, viết unit test, dễ dàng bảo trì phát triển.
  + Viết bằng ngôn ngữ TypeScrip, TS là một ngôn ngữ thay thế JS để lập trình viên dễ dàng bảo trì, kiểm soát lỗi. Nó là một ngôn ngữ cần thiết cho Angular để có tính thống nhất, scale dễ dàng.
  + RxJS là một thư viện dùng cho Angular, để bắt sự kiện và thực thi hành động mỗi biến thay đổi giá trị.
  + Hiệu suất cao vì trình tự động dọn rác và tự động render của Angular.
  + Được GG hỗ trợ và cộng đồng lớn.
  + Thư viện dựng sẵn cho các component giao diện Material giảm thiểu thời gian tạo các control.
  + Angular CLI: host server ngay trên máy của lập trình viên, và tự động bắt những dòng code thay đổi để chỉ build lại những file liên quan. Giảm thời gian build.
  + Đóng gói component và sử dụng cho các framework như vuejs, reactjs, jquery, …
* Nhược điểm:
  + Nếu dự án bắt đầu từu AngularJS thì việc chuyển đổi sang Angular sẵn rất khó khăn, đôi khi là không thể.
  + Angular dài dòng và phức tạp, 1 component có 3 file chưa kể nếu nó là module thì sẽ tăng lên thành 5 file. Tất nhiên, với dự án nhỉ thì điều này sẽ ảnh hưởng đôi chút về chi phí thời gian.
  + Angular đôi khi là khó khăn so với người mới nếu so sánh với React và Vue. Vì rất nhiều khái niệm và tính năng có thể có những dự án không cần dùng đến. Thêm nữa là TS cũng là ngôn ngữ cần phải học khi muốn làm việc với Angular.
  + Angular ra phiên bản liên tục và thay đổi rất nhiều về kiến trúc nên muốn nâng cấp phiên bản cũng tốn nhiều nỗ lực.

Tài liệu tham khảo: [Pros and Cons of Angular Development Framework (altexsoft.com)](https://www.altexsoft.com/blog/the-good-and-the-bad-of-angular-development/).

#### **Cài đặt và sử dụng Angular**

Để cài đặt và chạy một project Angular, cần làm theo các bước sau:

**Bước 1:** Cài đặt Node.js và npm tương thích với phiên bản Angular muốn cài.

Có thể tải xuống và cài đặt từ trang chủ của Node.js: nodejs.org

**Bước 2:** Cài đặt Angular CLI

Bạn chạy lệnh sau trong terminal hoặc command prompt:

npm install -g @angular/cli

Ý nghĩa của câu lệnh này là đang cài đặt angular trên toàn bộ máy tính của bạn. Sau khi cài xong bạn có thể gọi và chạy Angular CLI ở bất cứ đâu trên máy tính của bạn.

Nếu trên window đưa ra cảnh báo hãy sử dụng câu lệnh để cho phép cài đặt:

Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned

**Bước 3:** Tạo project Angular

Bạn có thể tạo một project Angular mới bằng lệnh sau:

ng new my-app

Trong đó, "my-app" là tên bạn muốn đặt cho project của mình.

**Bước 4:** Chạy project Angular

Sau khi tạo project, di chuyển vào thư mục bằng lệnh:

cd my-app

ng serve --open

Câu lệnh đầu tiên là để đi đến thư mục chưa project Angular.

Câu lệnh thứ 2 để nói cho Angular là tạo một host trên máy tính của bạn.

Host này mặc định là <http://localhost:4200/>. (Bạn có thể sửa lại host trên file angular.config.json trong thư mục project)

Bạn có thể mở đường dẫn trên và truy cập ứng dụng mặc định của Angular.

### Khái niệm về API và ASP.NET Core

#### ***Khái niệm về API***

Theo định nghĩa từ IBM:

“An API, or application programming interface, is a set of rules or protocols that let software applications communicate with each other to exchange data, features and functionality.”

Nghĩa là một bộ luật lệ hoặc giao thức để ứng dụng giao tiếp với nhau nhằm gửi dữ liệu đặt điểm và chức năng.

* Tức là API là một chuẩn để giao tiếp giữa các ứng dụng với nhau, mỗi ứng dụng có thểgiao tiếp theo một chuẩn khác nhau dựa theo yêu cầu ứng dụng.

Cách thức API làm việc:

Ví dụ:

* Khi khách hàng click vào nút mua hàng trên webpage, một API sẽ được gọi hay bước này còn gọi là request.
* Request sẽ bao gồm URI (Uniform Resource Identifier), headers, và cũng có thể chứa cả body.
* Sau khi webserver đón được request sẽ xử lý và trả về 1 phản hồi cho API, API lại trả về webpag. Được gọi là response.

Lợi ích khi dùng API:

* Phát triển sự cộng tác. Tức là mỗi ứng dụng có thể phát triển riêng lẻ và API giúp kết nối tới bên thứ 3. Giúp giảm chi phí đối với những dự án không có nhiều kinh phí.
* Tăng tốc độ đổi mới: Khi mà ứng dụng được tách riêng rẽ, ít phụ thuộc thì việc nâng cấp bảo trì sẽ tốn ít chi phí hơn.
* Kiếm tiền từ dữ liệu: bán API cũng mang lại là một nguồn thu nhập khổng lồ. Vì bên mua giảm được chi phí xây dựng API. Bên bán có thể bán cho nhiều bên khác nhau để kiếm lợi nhuận. Tất nhiên, đổi lại là độ bảo mật không cao.
* Bảo mật hệ thống: Xây dựng API là tách riêng các hệ thống với nhau. Mỗi hệ thống có một lớp bảo vệ nghiêm ngặt, từ đó khó có thể tấn công cũng lúc vào hệ thống chung.
* Bảo mật về quyền riêng tư: Mỗi nền tảng có một chính sách bảo mật riêng, do vậy API giúp cho việc bảo mật trở nên dễ dàng.

#### **Khái niệm về ASP.NET Core**

ASP.NET Core là một open-source mới và framework đa nền tảng (cross-platform) cho việc xây dựng những ứng dụng hiện tại dựa trên kết nối đám mây, giống như web apps, IoT và backend cho mobile.

Ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên phiên bản đầy đủ của .NET Framework. Nó được thiết kế để cung cấp và tối ưu development framework cho những dụng cái mà được triển khai trên đám mây (clound) hoặc chạy on-promise.

Nó bao gồm các thành phần theo hướng module nhằm tối thiểu tài nguyên và chi phí phát triển, như vậy bạn giữ lại được sự mềm giẻo trong việc xây dựng giải pháp của bạn. Bạn có thể phát triển và chạy những ứng dụng ASP.NET Core đa nền tảng trên Windows, Mac và Linux.

Đồng thời nó đã trở thành một mã nguồn mở. Đây là một thay đổi rất lớn và theo mình là quan trọng nhất của ASP.NET Core. Điều mà trước đây khó có một lập trình viên nào có thể nghĩ đến. Có lẽ đó cũng là một xu thế mà các ngôn ngữ lập trình hiện nay đang hướng tới.

C#, hay còn được gọi là C sharp là một ngôn ngữ lập trình cơ bản, được các lập trình viên đánh giá là một loại ngôn ngữ tuy đơn giản nhưng rất hiện đại và hiệu quả. C# lần đầu ra mắt vào năm 2000, do đội ngũ develop của Microsoft phát triển. C Sharp chủ yếu được xây dựng và phát triển dựa hai “cây đại thụ” trong đội ngũ ngôn ngữ lập trình là C++ và Java.

#### **Các đặc điểm và tính năng chính của ASP.NET Core**

1. Định nghĩa:

ASP.NET Core là một open-source mới và framework đa nền tảng (cross-platform) cho việc xây dựng những ứng dụng hiện tại dựa trên kết nối đám mây, giống như web apps, IoT và backend cho mobile. Ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên phiên bản đầy đủ của .NET Framework. Nó được thiết kế để cung cấp và tối ưu development framework cho những dụng cái mà được triển khai trên đám mây (clound) hoặc chạy on-promise.

Đồng thời nó đã trở thành một mã nguồn mở. Đây là một thay đổi rất lớn và theo mình là quan trọng nhất của ASP.NET Core. Điều mà trước đây khó có một lập trình viên nào có thể nghĩ đến. Có lẽ đó cũng là một xu thế mà các ngôn ngữ lập trình hiện nay đang hướng tới.

1. Lợi ích khi sử dụng:

* Hỗ trợ đa nền tảng: giúp việc hosting không phụ thuộc vào nền tảng đang chạy. Chạy được trên Windows, Linux, and Mac.
* Nhanh: ASP.NET core cho phép thêm các package vào ứng dụng, giảm thiểu request giúp cải thiện hiệu năng.
* Tích hợp được với các framework UI hiện đại.
* Hosting: Hỗ trợ hosting với nhiều webserver khác nhau. Không giống ASP.NET framework chỉ dựa vào IIS.
* Có thể share code giữa các frameword khác nhau vì chạy trên .NET core (>= .NET 4).
* Kích thước triển khai nhỏ hơn: vì chạy trên .NET core.

Từ những lợi ích trên việc sử dụng ASP.Net vào triển khai API là hoàn toàn hợp lý. Giúp triển khai trên các môi trường khác nhau, giảm thiểu chi phí và hơn nữa là hiệu năng cũng đảm bảo.

### Cơ sở dữ liệu

### MySQL là gì?



Hình 2.2: Hệ quản trị cơ sở dữ liệu MySQL

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở phổ biến, được phát triển bởi Oracle Corporation. MySQL được sử dụng rộng rãi trong các ứng dụng web và là một phần quan trọng của công nghệ LAMP (Linux, Apache, MySQL, PHP/Python/Perl).

MySQL được thiết kế để xử lý các cơ sở dữ liệu lớn và có thể đáp ứng các yêu cầu về tốc độ truy xuất và xử lý dữ liệu. MySQL hỗ trợ các tính năng quản trị cơ sở dữ liệu như tạo, sửa đổi, xóa và truy xuất dữ liệu, cùng với các tính năng bảo mật và sao lưu dữ liệu.

MySQL có nhiều phiên bản và cung cấp nhiều tùy chọn để tùy chỉnh và cấu hình hệ thống theo nhu cầu sử dụng của người dùng. MySQL cũng hỗ trợ các ngôn ngữ lập trình phổ biến như PHP, Java và C++, cho phép người dùng tạo ra các ứng dụng tùy chỉnh và tích hợp với các ứng dụng web.

Một số ưu điểm của MySQL bao gồm:

* Hiệu suất cao: Được thiết kế để xử lý các cơ sở dữ liệu lớn và có khả năng xử lý hàng nghìn truy vấn mỗi giây.
* Độ tin cậy cao: Được sử dụng rộng rãi và được đánh giá cao về độ tin cậy và sẵn sàng của hệ thống.
* Tính linh hoạt: Có nhiều phiên bản và tùy chọn cấu hình để phù hợp với các nhu cầu sử dụng khác nhau.
* Bảo mật: Cung cấp các tính năng bảo mật để bảo vệ dữ liệu khỏi các mối đe dọa bên ngoài.
* Tính mở rộng: Có khả năng mở rộng để đáp ứng các nhu cầu mở rộng của hệ thống và dữ liệu.

## **Các phần mềm và công cụ sử dụng**

### Visual Studio Code (VSCode)

Visual Studio Code là một trình mã biên tập được phát triển bởi Microsoft, là một trình soạn thảo mã nguồn mở, nhẹ và mạnh mẽ Và được viết tắt là VS Code. Một số chức năng tiêu biểu của VS Code mà chúng ta có thể kể đến như: Git, Debug, Syntax Highlighting, Snippets.

A blue logo with a blue background

Description automatically generated

Hình 2.3: Hình ảnh phần mềm lập trình VSCode

**Ưu Điểm:**

* **Nhẹ và nhanh**: So với nhiều IDE khác, VS Code rất nhẹ và khởi động nhanh, giúp tiết kiệm tài nguyên hệ thống
* **Tùy chỉnh cao**: Người dùng có thể tùy chỉnh gần như mọi khía cạnh của VS Code từ giao diện, phím tắt, đến các thiết lập công cụ
* **Cộng đồng lớn**: VS Code có một cộng đồng người dùng và phát triển mạnh mẽ, cung cấp nhiều tài nguyên học tập và hỗ trợ
* **Cập nhật thường xuyên**: Microsoft liên tục cập nhật VS Code với các tính năng mới, cải tiến hiệu năng và sửa lỗi

**Nhược Điểm:**

* **Không phải IDE hoàn chỉnh**: Mặc dù rất mạnh mẽ, VS Code vẫn là một trình soạn thảo mã nguồn và không có đầy đủ các tính năng của một IDE hoàn chỉnh như Visual Studio
* **Cần cấu hình nhiều**: Để tận dụng tối đa các tính năng của VS Code, người dùng có thể cần phải cài đặt và cấu hình nhiều tiện ích mở rộng và thiết lập.
* **Hiệu năng có thể giảm khi cài nhiều extensions**: Quá nhiều tiện ích mở rộng có thể làm giảm hiệu năng của VS Code.

### Microsoft Visual Studio

Microsoft Visual Studio là môi trường phát triển tích hợp (IDE) được thiết kế dành cho giới lập trình viên và các nhà phát triển ứng dụng. Đây là công cụ hỗ trợ phát triển phần mềm mạnh mẽ của Microsoft, cho phép người dùng viết, dịch mã và gỡ lỗi các ứng dụng dựa trên nhiều ngôn ngữ lập trình khác nhau như C++, C#, Visual Basic, Python, JavaScript... Visual Studio bao gồm một trình biên tập mã nguồn, các công cụ gỡ lỗi và xây dựng ứng dụng đa nền tảng. Nó giúp tăng năng suất và hiệu quả công việc cho các lập trình viên

A purple and white logo

Description automatically generated

Hình 2.4: Hình ảnh phần mềm lập trình microsoft visual studio

**Ưu Điểm:**

* **Hỗ trợ nhiều ngôn ngữ lập trình**: Visual Studio hỗ trợ nhiều ngôn ngữ lập trình bao gồm C#, C++, Visual Basic .NET, F#, Python, JavaScript, và nhiều ngôn ngữ khác. Điều này giúp nó trở nên đa năng cho nhiều loại dự án phát triển khác nhau
* **Hỗ trợ lập trình đa nền tảng**: Visual Studio có thể được sử dụng để phát triển ứng dụng cho nhiều nền tảng khác nhau bao gồm Windows, Android, iOS, và các ứng dụng web, nhờ vào các công cụ như Xamarin và .NET Core
* **Hệ sinh thái mở rộng phong phú**: Có một hệ sinh thái rộng lớn các tiện ích mở rộng có sẵn thông qua Visual Studio Marketplace. Các tiện ích mở rộng này có thể thêm các chức năng, công cụ và tùy chỉnh vào IDE
* **Khả năng gỡ lỗi mạnh mẽ**: Visual Studio cung cấp các công cụ gỡ lỗi mạnh mẽ cho phép nhà phát triển đặt điểm dừng, kiểm tra biến và từng bước thực hiện mã. Điều này giúp nhanh chóng xác định và sửa lỗi

**Nhược điểm:**

* **Yêu cầu tài nguyên hệ thống cao**: Visual Studio nổi tiếng vì tiêu tốn nhiều tài nguyên, yêu cầu một lượng lớn bộ nhớ và sức mạnh xử lý. Điều này có thể làm chậm hệ thống, đặc biệt là trên các máy cũ hoặc kém mạnh mẽ
* **Chi phí cao**: Mặc dù có phiên bản miễn phí (Visual Studio Community), nhưng các phiên bản nhiều tính năng hơn như Professional và Enterprise có thể khá đắt đỏ. Điều này có thể không khả thi cho các nhà phát triển cá nhân hoặc các công ty nhỏ
* **Quá trình cái đặt và cập nhật phức tạp**: Quá trình cài đặt có thể dài và phức tạp, và các bản cập nhật đôi khi có thể gây ra vấn đề hoặc yêu cầu thời gian dừng hệ thống đáng kể
* **Độ phức tạp cho người mới bắt đầu** : Mặc dù giao diện người dùng thân thiện, số lượng lớn các tính năng và tùy chọn có sẵn có thể làm cho người mới bắt đầu cảm thấy choáng ngợp, dẫn đến đường cong học tập dốc

### MySQL WorkBench:

MySQL Workbench là một công cụ quản lý cơ sở dữ liệu MySQL được phát triển và cung cấp miễn phí bởi MySQL, một phần của Oracle Corporation. Nó cung cấp một giao diện đồ họa cho người dùng để quản lý cơ sở dữ liệu MySQL một cách trực quan và thuận tiện. MySQL Workbench cung cấp nhiều tính năng như tạo, sửa, xóa cơ sở dữ liệu và bảng, thực thi truy vấn SQL, thiết kế cơ sở dữ liệu, đồ họa quan hệ, và nhiều tính năng khác để hỗ trợ việc quản lý cơ sở dữ liệu MySQL



Hình 2.5: Hình ảnh phần mềm MySQL Workbench

**Ưu điểm:**

* **Giao diện đồ họa thân thiện:** Workbench cung cấp một giao diện đồ họa dễ sử dụng, giúp người dùng tạo và quản lý cơ sở dữ liệu MySQL một cách trực quan và thuận tiện.
* **Các tính năng quản lý đa dạng:** Workbench cung cấp nhiều tính năng quản lý cơ sở dữ liệu như tạo, sửa, xóa cơ sở dữ liệu, bảng, truy vấn SQL, và các đối tượng khác, giúp người dùng dễ dàng thực hiện các thao tác quản lý.
* **Hỗ trợ đa nền tảng:** Phần mềm này có sẵn trên nhiều hệ điều hành khác nhau như Windows, macOS và Linux, giúp người dùng có thể sử dụng trên nhiều môi trường khác nhau.
* **Tích hợp với các công cụ khác:** Workbench tích hợp tốt với các công cụ khác như MySQL Shell, giúp người dùng thực hiện các thao tác quản lý và phát triển cơ sở dữ liệu một cách hiệu quả.

**Nhược điểm:**

* **Tính đồng bộ:** Một số người dùng gặp phải vấn đề về đồng bộ hóa khi làm việc trên nhiều phiên bản của MySQL và Workbench, đặc biệt là khi làm việc với các phiên bản MySQL khác nhau.
* **Yêu cầu tài nguyên hệ thống:** Workbench có thể yêu cầu một số tài nguyên hệ thống khá lớn, đặc biệt là khi làm việc với các cơ sở dữ liệu lớn và phức tạp.
* **Độ tin cậy:** Một số người dùng báo cáo về vấn đề về độ tin cậy của Workbench, bao gồm việc gặp phải lỗi và sự chậm trễ trong quá trình làm việc.

### Quản lý mã nguồn (GitHub)

GitHub là một nền tảng dựa trên web cho việc lưu trữ mã nguồn và quản lý dự án phần mềm, được xây dựng trên nền tảng Git. Nó cung cấp các tính năng như version control (sử dụng Git), issue tracking, pull requests và collaboration tools để hỗ trợ việc làm việc nhóm và quản lý dự án.



Hình 2.6: Hình ảnh công cụ quản lý mã nguồn GitHub

**Ưu Điểm:**

* **Quản lý mã nguồn hiệu quả:** GitHub cung cấp một hệ thống quản lý mã nguồn mạnh mẽ dựa trên Git, cho phép dễ dàng theo dõi các thay đổi, quản lý phiên bản, và cộng tác trong các dự án phần mềm.
* **Kho lưu trữ mã nguồn mở:** GitHub là nơi lưu trữ hàng triệu dự án mã nguồn mở, cho phép các nhà phát triển dễ dàng chia sẻ mã nguồn và đóng góp vào các dự án của người khác.
* **Tích hợp nhiều công cụ:** GitHub tích hợp tốt với nhiều công cụ và dịch vụ khác như Continuous Integration (CI), Continuous Deployment (CD), và các IDE phổ biến, giúp tự động hóa quy trình phát triển và triển khai
* **Bảo mật:** GitHub cung cấp nhiều tính năng bảo mật như bảo vệ nhánh (branch protection), quét bảo mật mã nguồn (code scanning), và quản lý các mã thông báo (token management).

**Nhược Điểm:**

* **Giới hạn trong việc quản lý dự án phức tạp:** Đối với các dự án rất lớn và phức tạp, GitHub có thể thiếu một số tính năng quản lý dự án nâng cao so với các công cụ chuyên biệt khác.
* **Phụ thuộc vào internet:** GitHub là một dịch vụ dựa trên web, do đó yêu cầu kết nối internet để truy cập và làm việc với các kho lưu trữ, điều này có thể là một trở ngại trong các tình huống không có internet.

## **Kết luận chương 2**

Trong chương 2 này, em đã giới thiệu về ngôn ngữ, các công nghệ sử dụng trong đề tài bao gồm Framework Angular, ASP.Net Core, cơ sở dữ liệu MySQL và các công cụ hỗ trợ triển khai các framework để có thể xây dựng hệ thống. Ngoài ra, trong qua trình phát triển có phát sinh thêm các thư viện để giảm thiểu thời gian coding.

# **CHƯƠNG 3. XÂY DỰNG WEBSITE QUẢN LÝ MỐI QUAN HỆ KHÁCH HÀNG**

## **Phân tích hệ thống**

* Phát triển và triển khai một trang web chuyên nghiệp, có hiệu quả đáp ứng được nghiệp vụ quản lý dữ liệu khách hàng của kinh doanh, đáp ứng được nghiệp vụ theo dõi tiến độ, quản lý tình hình kinh doanh của quản lý.
* Tạo ra một giao diện người dùng thân thiện và dễ sử dụng, giúp người dùng tìm kiếm, xem thông tin chi tiết nhanh chóng, dễ dàng tạo ra, linh hoạt trong việc sửa đổi dữ liệu.
* Tích hợp các tính năng an toàn và bảo mật tránh việc thất thoát dữ liệu, dữ liệu ràng buộc với nhau.
* Tạo ra một cơ sở hạ tầng kỹ thuật mạnh mẽ và linh hoạt, có thể mở rộng và phát triển trong tương lai để đáp ứng nhu cầu mở rộng kinh doanh tìm kiếm cơ hội và tạo đơn hàng.

## **Xây dựng biểu đồ Use case (UC)**

### Xác định tác nhân của hệ thống

Người dùng: Người sử dụng phần mềm để thực hiện các chức năng thêm sửa xóa các phân hệ. Chia ra dựa trên các vị trí khác nhau:

* Quản trị hệ thống: có đầy đủ các quyền.
* Nhân viên kinh doanh: có các quyền dựa theo nghiệp vụ riêng.
* Nhân viên chăm sóc: có các quyền dựa theo nghiệp vụ riêng.

### Xác định tác nhân và các chức năng của tác nhân

Bảng 3.1: Bảng tác nhân và chức năng của tác nhân

|  |  |
| --- | --- |
| Tác nhân | Use case |
| Người dùng | * Quản lý quyền truy cập * Đăng ký * Đăng nhập * Quản lý tiềm năng * Quản lý khách hàng * Quản lý đơn hàng * Quản lý liên hệ * Quản lý hàng hóa * Quản lý chính sách giá * Dashboard |

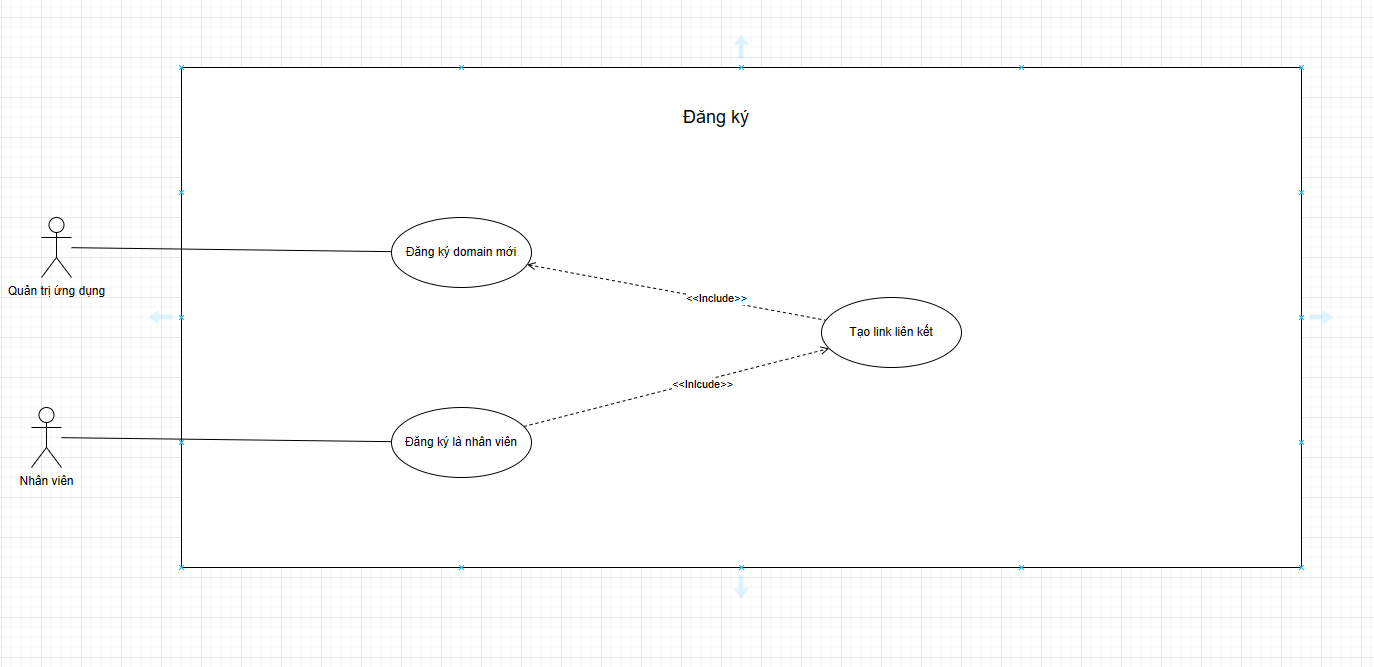
### Biểu đồ use case tổng quan



Hình 3.1: Biểu đồ use case tổng quan

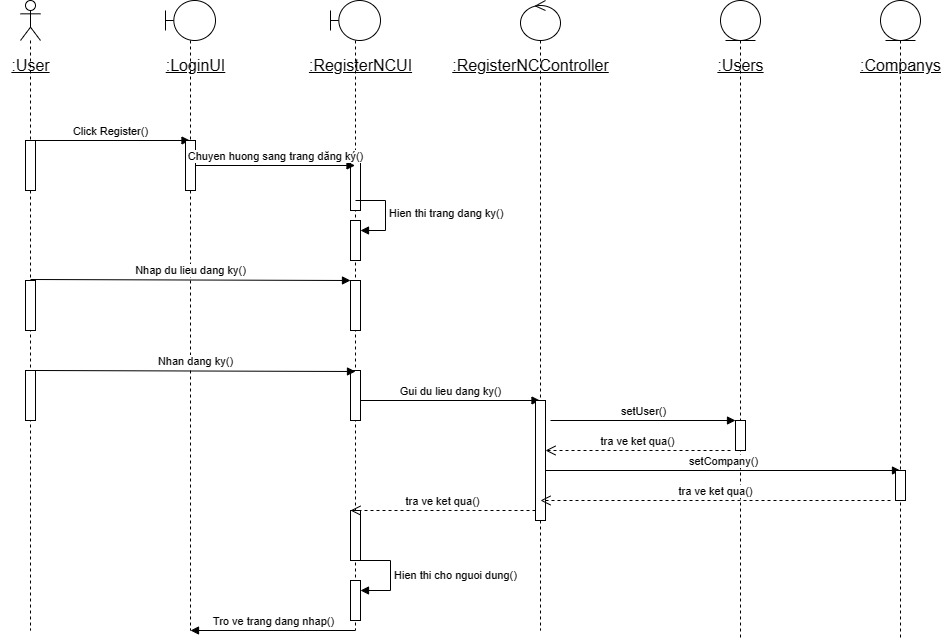
## **Mô tả chi tiết Use case**

* + - 1. Use case Đăng ký:
         1. Biểu đồ use case chi tiết:

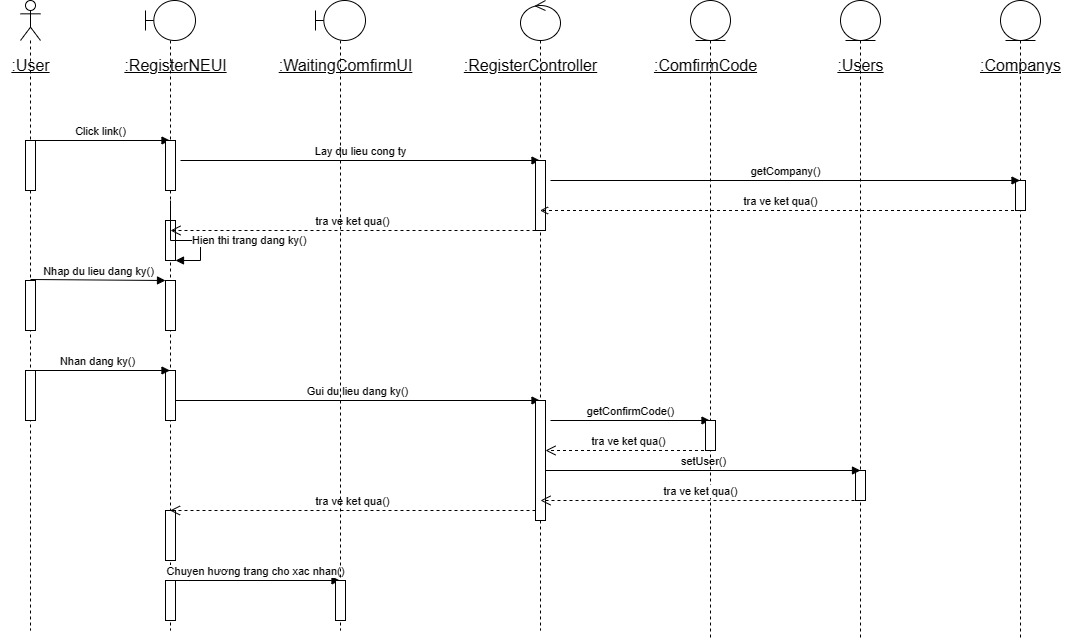


* + - * 1. Biểu đồ trình tự:

Use case đăng ký domain mới:



Use case đăng ký là nhân viên:



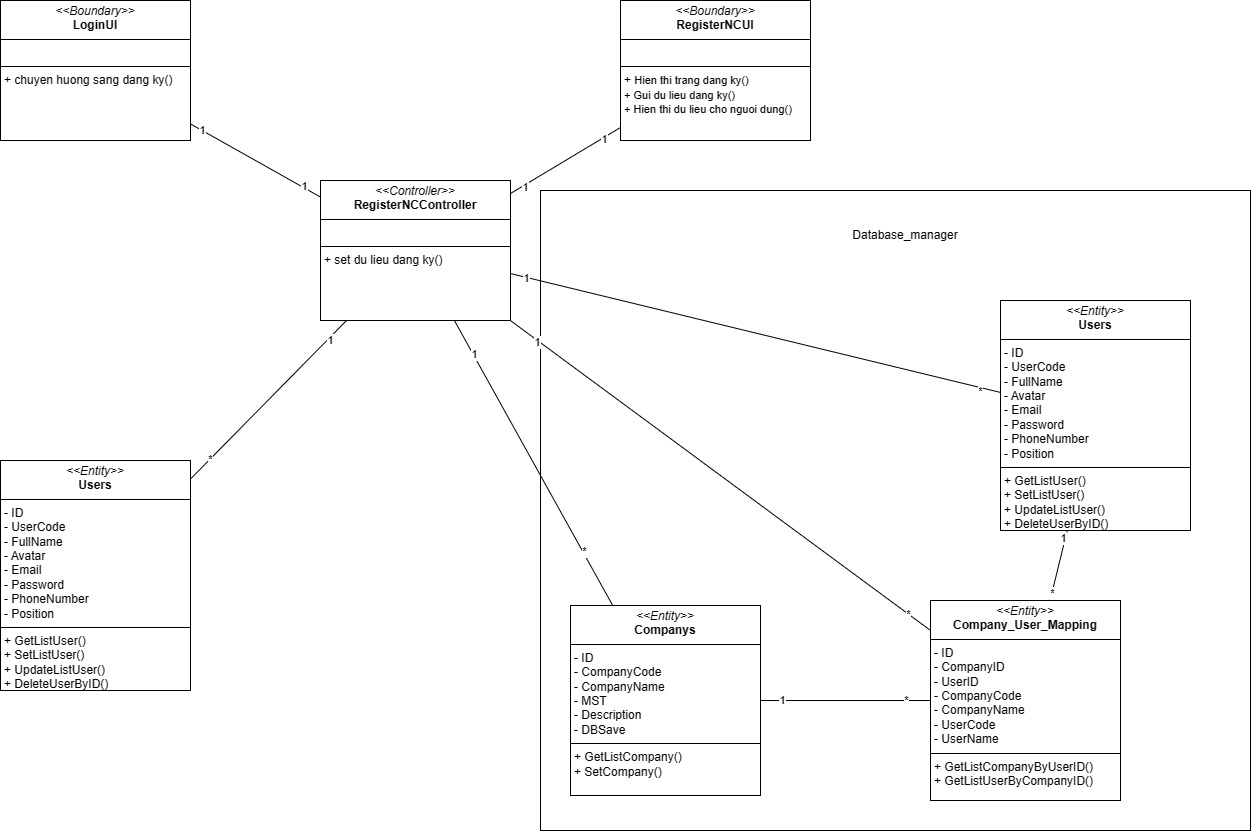
* + - * 1. Đặc tả chi tiết:

Use case đăng ký domain mới:

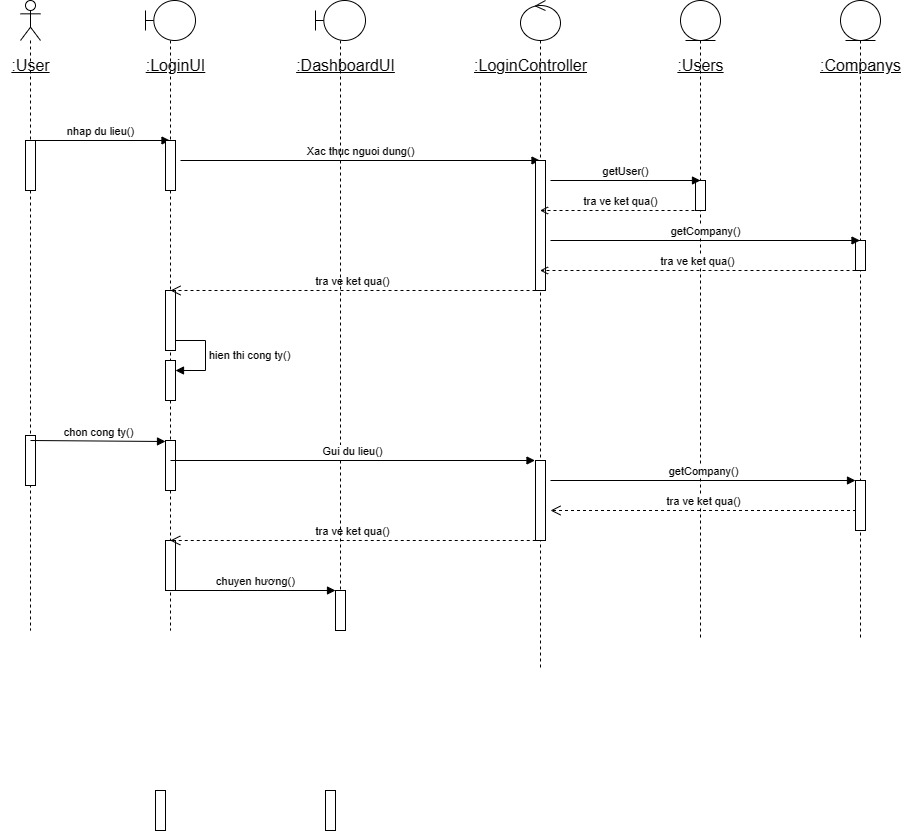
* **Mô tả vắn tắt:** Usecase này cho phép người dùng tạo tài khoản và domain mới.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Usecase này bắt đầu khi người dùng kích vào nút “Đăng ký” trong trang đăng nhập. Hệ thống sẽ yêu cầu nhập các thông tin (Họ và tên, số điện thoại, email, tên tài khoản, mật khẩu, tên công ty, vị trí trong công ty) để tạo tài khoản.
    - Sau khi nhập các thông tin và kích nút “Đăng ký”. Hệ thống kiểm tra email đã đăng ký chưa.
    - Hệ thống hiển thị thông báo đăng ký thành công, chuyển hướng đến trang chọn công ty.
    - Usecase kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 trong luồng cơ bản, nếu hệ thống kiểm tra email đã đăng ký tài khoản và tên công ty hợp lệ, hệ thống hiển thị thông báo xác nhận muốn tọa domain mới không.
    - Người dùng nhấn xác nhận, hệ thống tạo mới domain và chuyển đến trang tạo domain.
* **Các yêu cầu đặc biệt:** Người dùng cần nhập đúng và đầy đủ các trường bắt buộc để tạo tài khoản.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Đăng ký thành công sẽ tạo domain mới trong database.
* **Điểm mở rộng:** Không có.

Use case đăng ký là nhân viên:

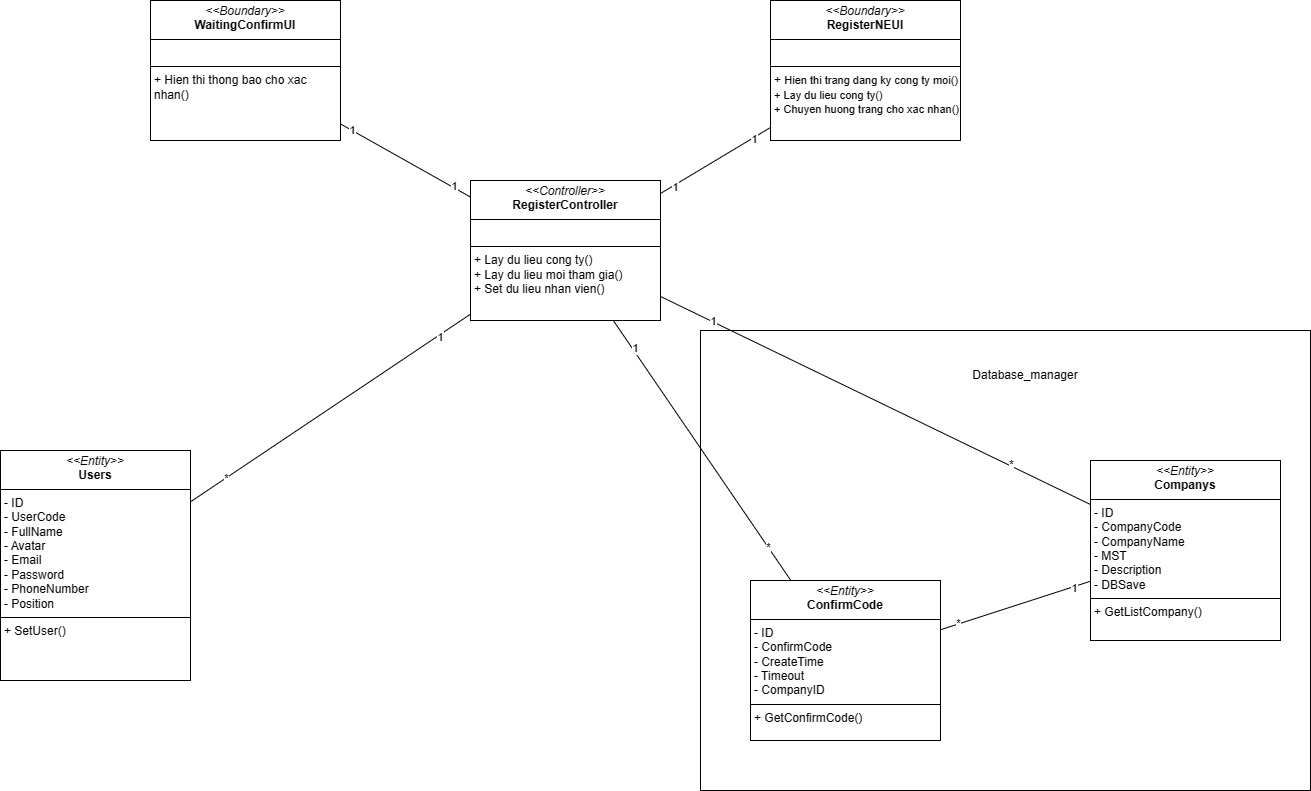
* **Mô tả vắn tắt:** Usecase này cho phép người dùng đăng ký là nhân viên mới trong 1 công ty.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Usecase này bắt đầu khi người dùng truy cập đường dẫn mà quản trị hệ thống đưa cho. Hệ thống sẽ yêu cầu nhập thông tin đăng nhập.
    - Sau khi nhập các thông tin và kích nút “Đăng nhập”. Hệ thống kiểm tra email đã đăng ký chưa.
    - Hệ thống hiển thị thông báo đăng ký thành công, chuyển hướng đến trang dashboard.
    - Usecase kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 trong luồng cơ bản, nếu hệ thống kiểm tra email chưa đăng ký, hệ thống chuyển hướng sang trang đăng ký.
    - Người dùng nhập thông tin nhấn xác nhận, hệ thống kiểm tra xác thực. Nếu thành công thì chuyển sang dashboard.
* **Các yêu cầu đặc biệt:** Người dùng cần nhập đúng và đầy đủ các trường bắt buộc để tạo tài khoản.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Đăng ký thành công sẽ insert thêm 1 dòng vào database.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



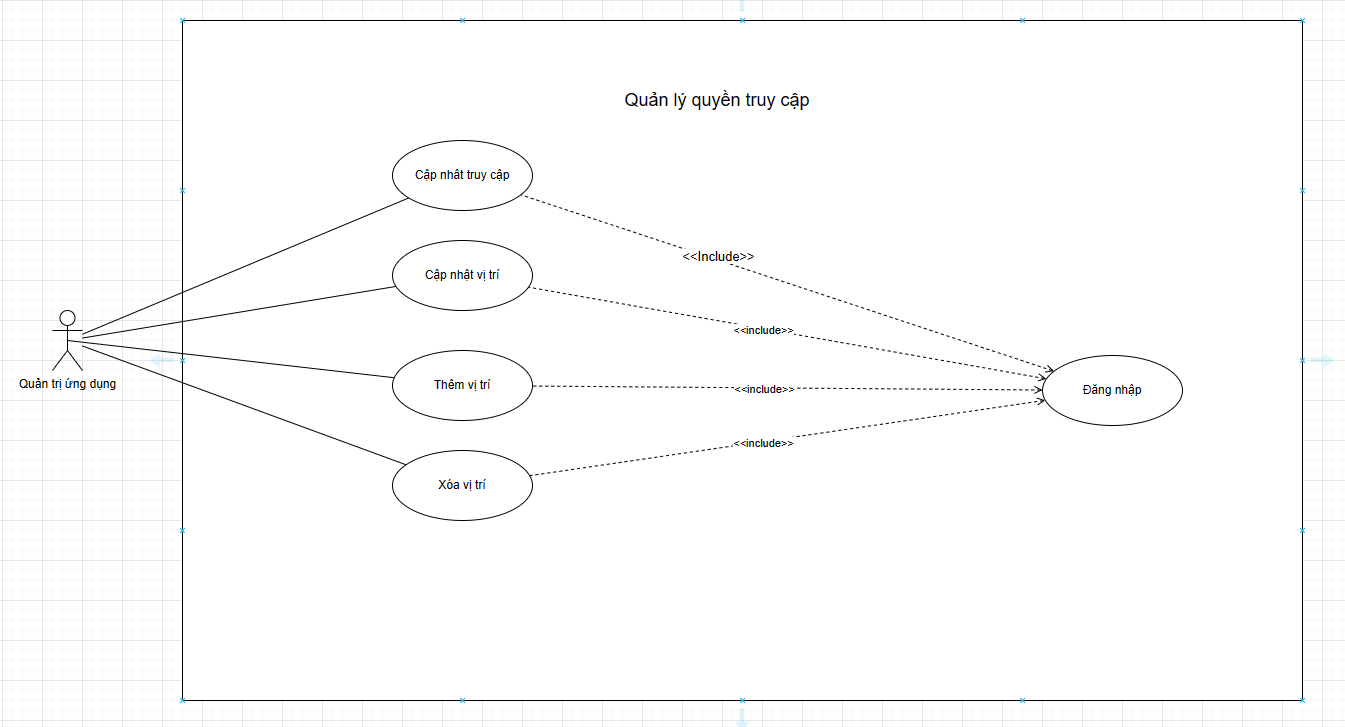
* + - 1. Đặc tả Use Case Đăng nhập:
         1. Biểu đồ Use case chi tiết:
         2. Biểu đồ trình tự:



* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng đăng nhập vào ứng dụng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Usecase này bắt đầu khi người dùng truy cập trang đăng nhập. Hệ thống hiển thị trang đăng nhập.
    - Sau khi nhập các thông tin và kích nút “Đăng nhập”. Hệ thống kiểm tra email đã đăng ký chưa.
    - Hệ thống hiển thị thông báo đăng nhập thành công, chuyển hướng đến trang chọn công ty.
    - Người dùng chọn công ty và nhận “Chọn”, hệ thống chuyển hướng đến trang dashboard.
    - Usecase kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 trong luồng cơ bản, nếu hệ thống kiểm tra email chưa đăng ký, hệ thống hiển thị lỗi.
* **Các yêu cầu đặc biệt:** Người dùng cần nhập đúng và đầy đủ các trường bắt buộc để tạo tài khoản.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Không có.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:

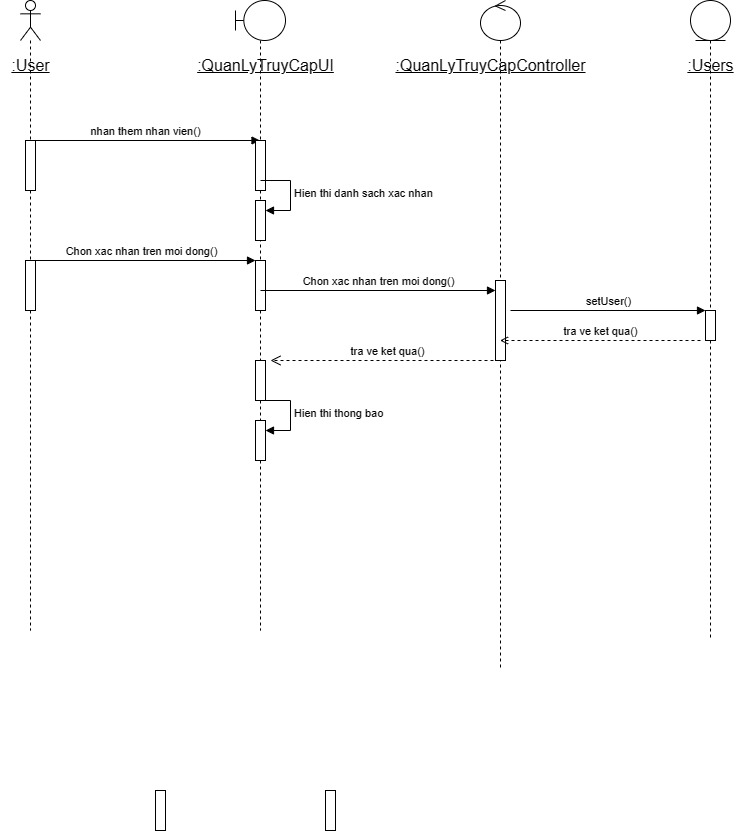


* + - 1. Đặc tả Use case Quản lý truy cập:
         1. Biểu đồ Use case chi tiết:

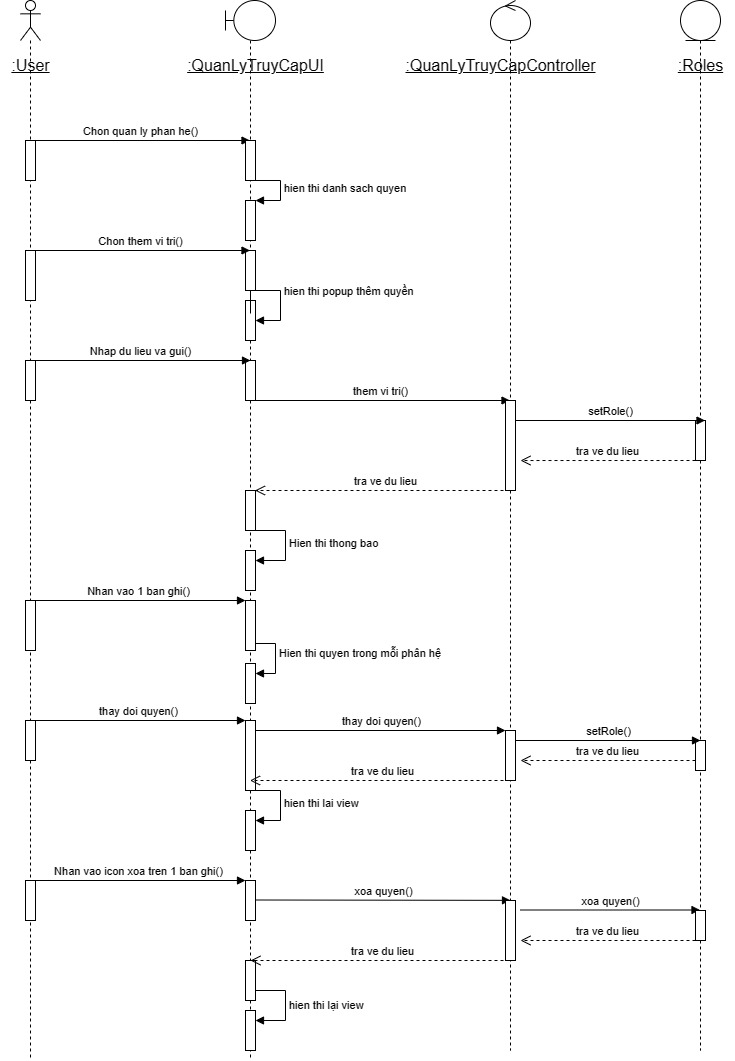


* + - * 1. Biểu đồ trình tự:

Use case quản lý truy cập ứng dụng:



Use case quản lý truy cập phân hệ:



* + - * 1. Đặc tả chi tiết:

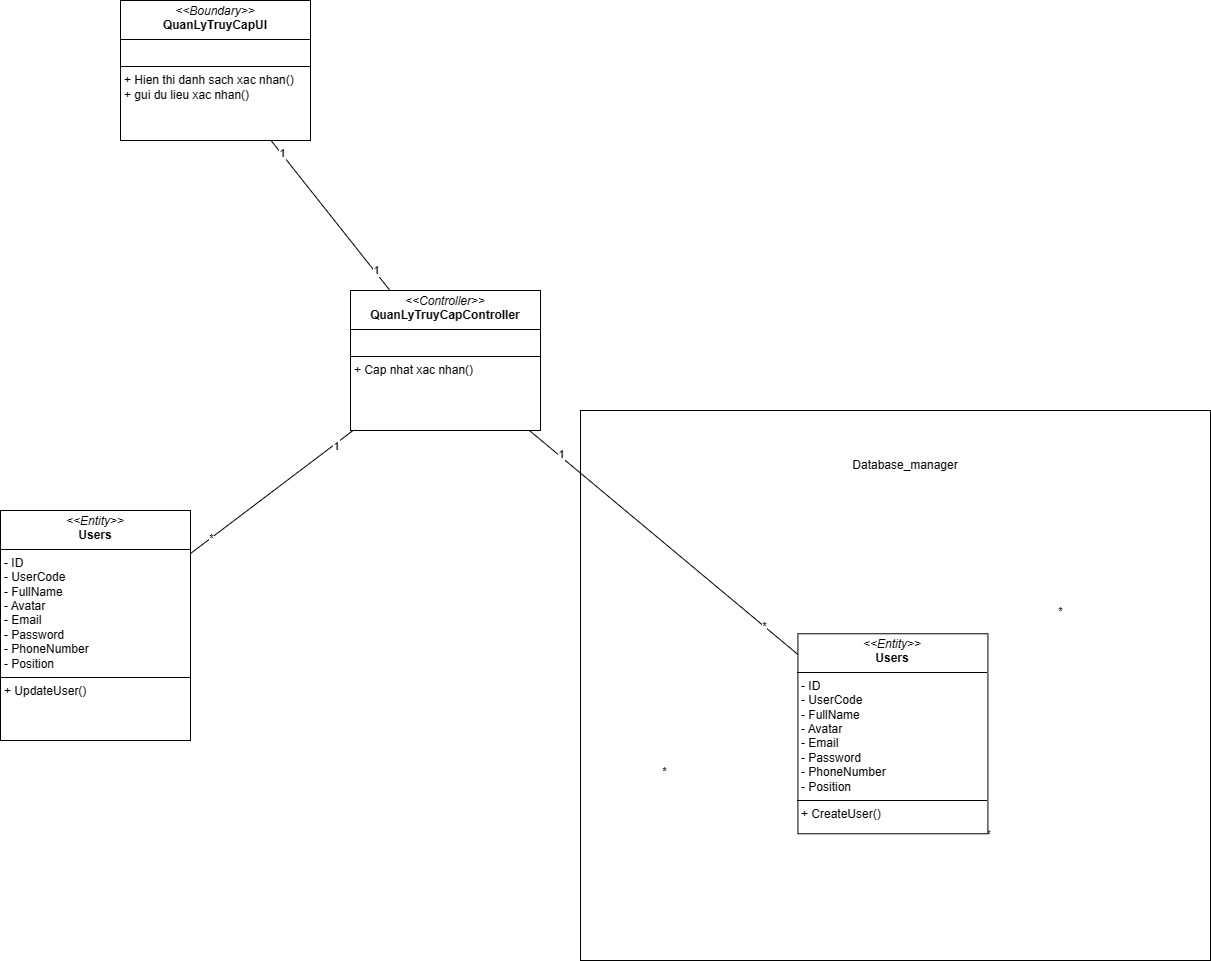
Quản lý truy cập ứng dụng:

* **Mô tả vắn tắt:** Usecase này cho phép người dùng bật tắt khả năng truy cập công ty của các nhân viên.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Usecase này bắt đầu khi người dùng nhấn vào “Quản lý truy cập”. Hệ thống chuyển hướng sang trang quản lý truy cập và hiển thị danh sách các user đang có trong công ty.
    - Người dùng nhấn vào nút bật, hệ thống cho phép người dùng truy cập hệ thống và hiển thị thông báo thành công.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 trong luồng cơ bản, nếu nút đang trong trạng thái bật, hệ thống sẽ dừng người dùng được chọn ra khỏi công ty.
* **Các yêu cầu đặc biệt:** người dùng là có vai trò là quản trị hệ thống.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Cập nhật bản ghi User trong domain.
* **Điểm mở rộng:** Không có.

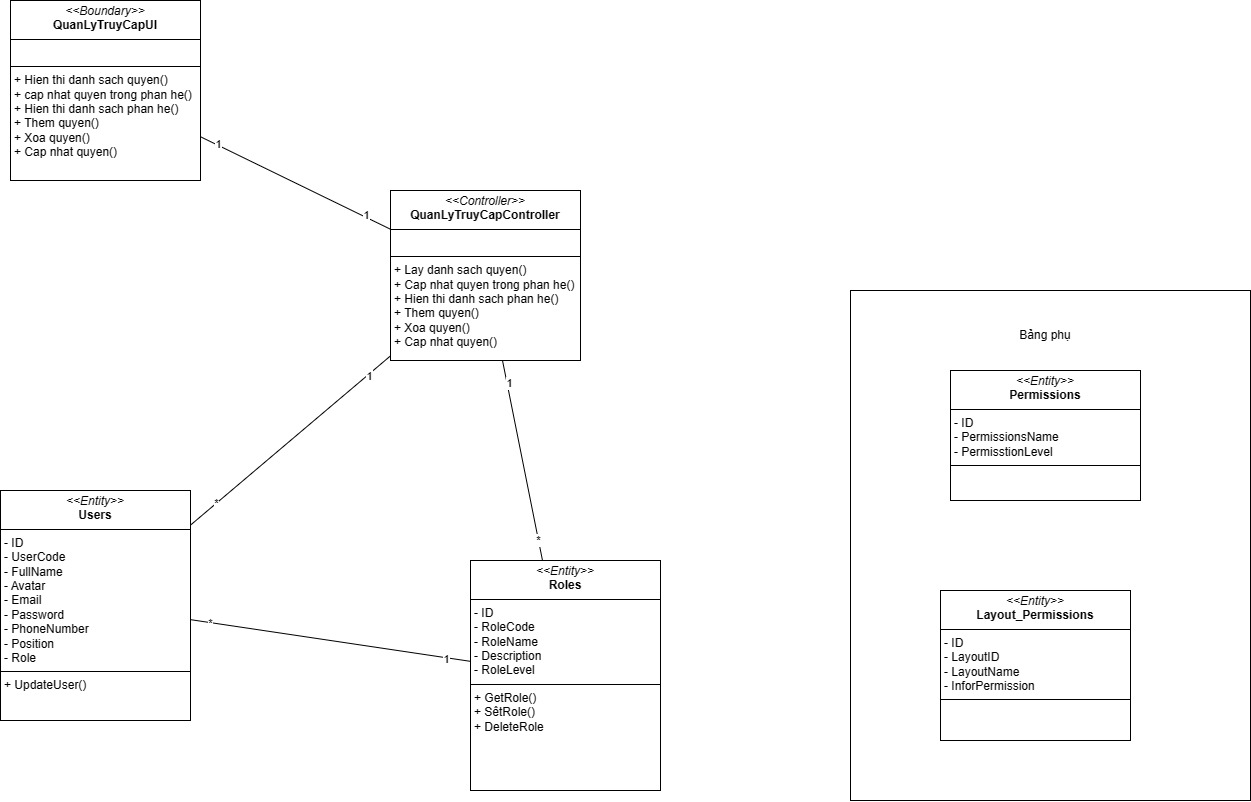
Quản lý truy cập phân hệ:

* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa quyền cho các phân hệ.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào tab phân hệ, hệ thống hiển thị danh sách các quyền.
    - 4 chức năng cho từng quyền:
      1. Xem chi tiết quyền:
         1. Nhấn vào 1 bản ghi, hệ thống chuyển hướng sang trang chi tiết quyền.
      2. Cập nhật quyền:
         1. Khi trong trang chi tiết quyền, người dùng nhấn vào bất kỳ 1 phân hệ, hệ thống hiển thị popup các quyền trong từng phân
         2. Người dùng cập nhật quyền và tắt popup, hệ thống cập nhật lại quyền.
      3. Thêm quyền:
         1. Khi trong trang danh sách quyền, người dùng nhấn vào “Thêm”, hệ thống hiển thị danh sách phân hệ quyền.
         2. Người dùng cập nhật quyền, tên vị trí và nhấn “lưu”, hệ thống thêm quyền mới và hiển thị lại view.
      4. Xóa quyền:
         1. Khi trong trang danh sách quyền, người dùng nhấn vào icon thùng rác nút xóa bên cạnh bản ghi muốn xóa, hệ thống hiển thị popup thông báo xác nhận xóa .
         2. Người dùng tiếp tục nhấn xác nhận, hệ thống xóa quyền khỏi ứng dụng.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước vào trang quyền truy cập, nếu người dùng không có quyền vào, hệ thống hiển thị lỗi.
    - Tại bước xóa tiềm năng, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về danh sách quyền.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng quyền truy cập trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:

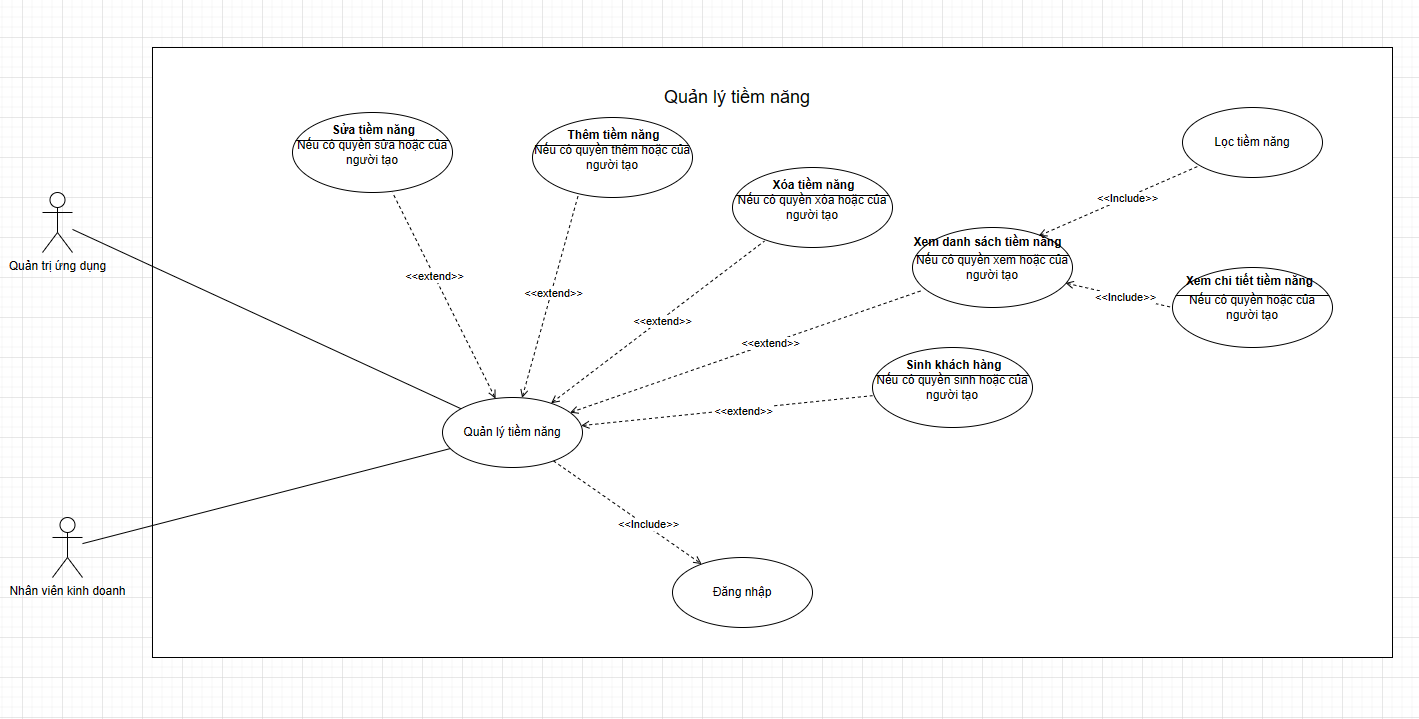
Quản lý truy cập ứng dụng:



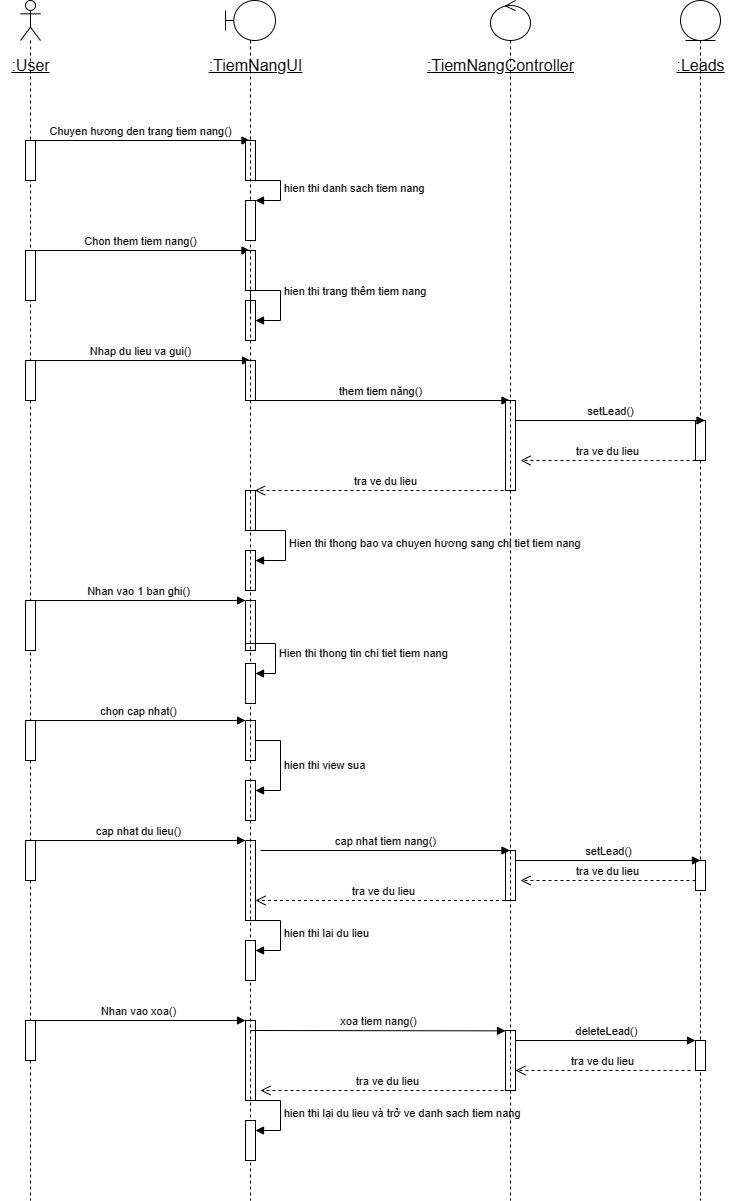
Quản lý truy cập phân hệ:



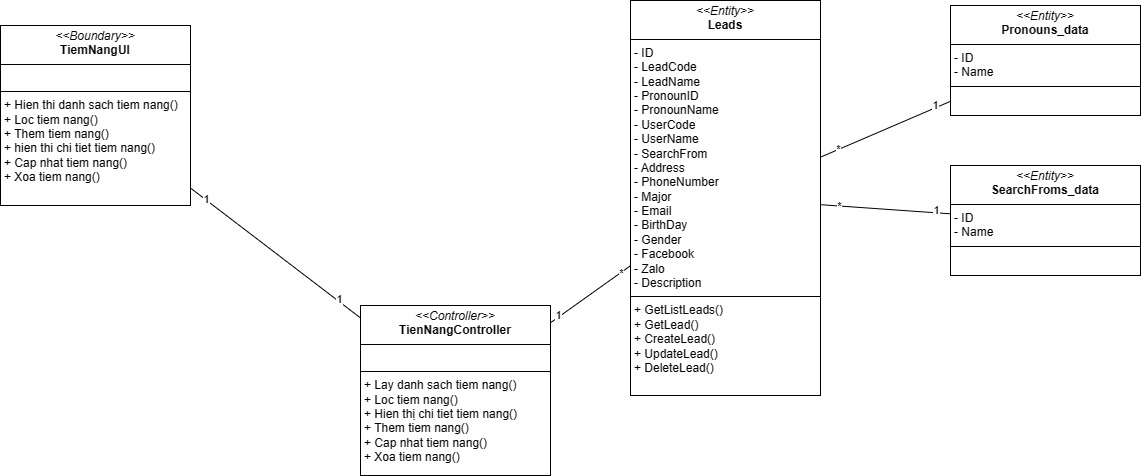
* + - 1. Đặc tả Use case Quản lý tiềm năng:
         1. Biểu đồ Use case:



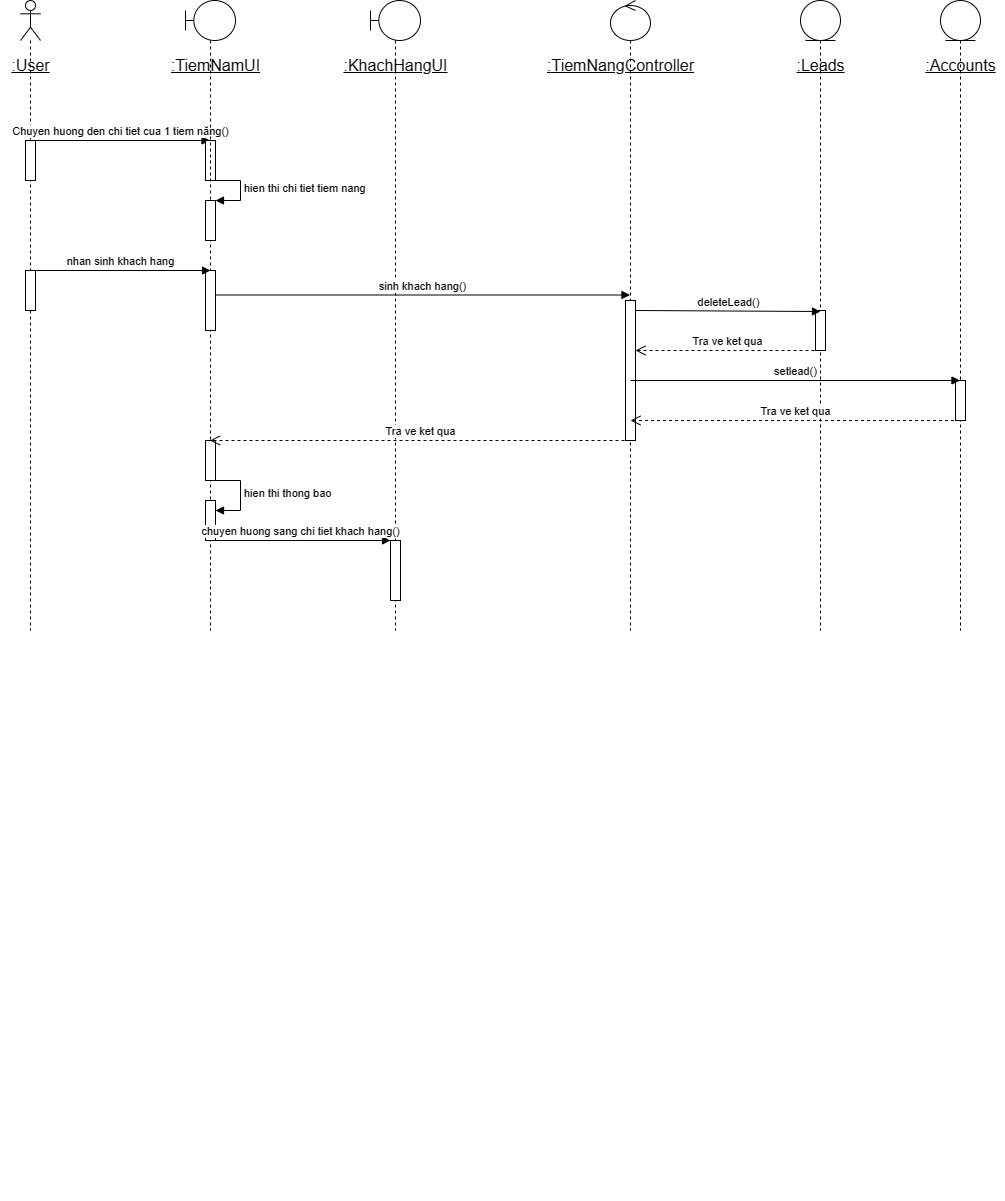
* + - * 1. Biểu đồ trình tự:



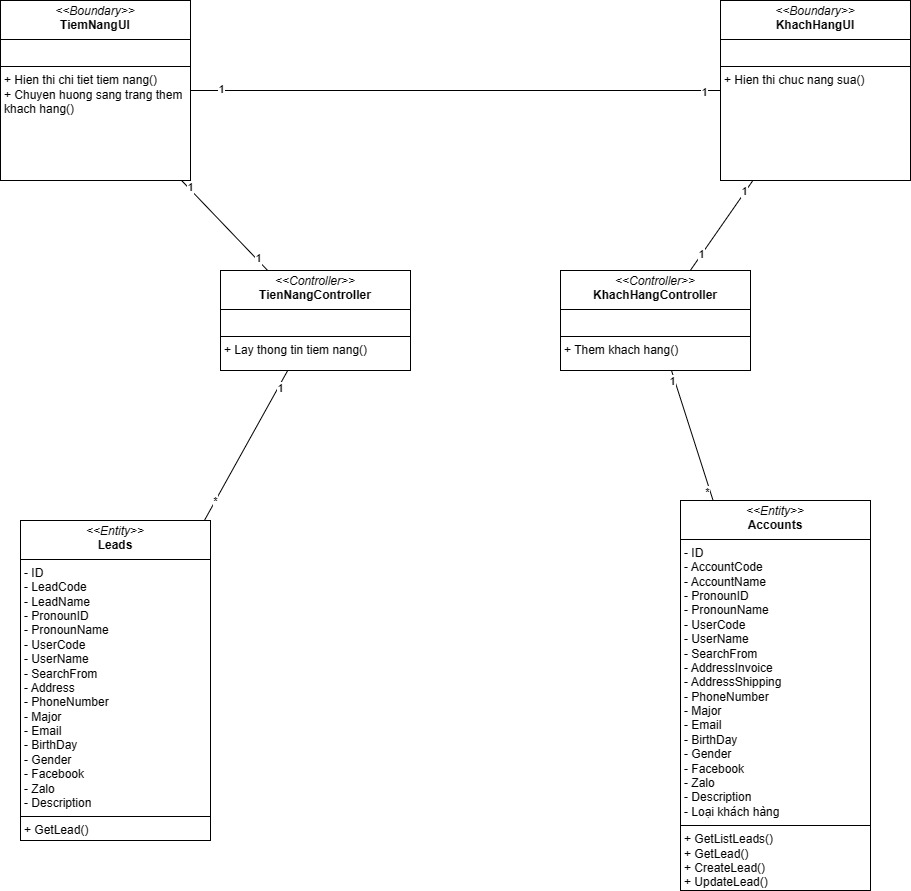
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ tiềm năng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Tiềm năng” trên thanh navbar, hệ thống hiển thị danh sách tiềm năng.
    - 4 chức năng cho Tiềm năng:
      1. Xem danh sách tiềm năng:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách tiềm năng.
      2. Lọc tiềm năng:
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách tiềm năng theo các điều kiện đã thêm.
      3. Xem chi tiết tiềm năng:
         1. Người dùng chọn 1 bản ghi trong danh sách tiềm năng, hệ thống hiển thị chi tiết tiềm năng.
      4. Cập nhật tiềm năng:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm tiềm năng:
         1. Tại màn hình danh sách tiềm năng, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm tiềm năng.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa tiềm năng:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách tiềm năng.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết tiềm năng.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Lead trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



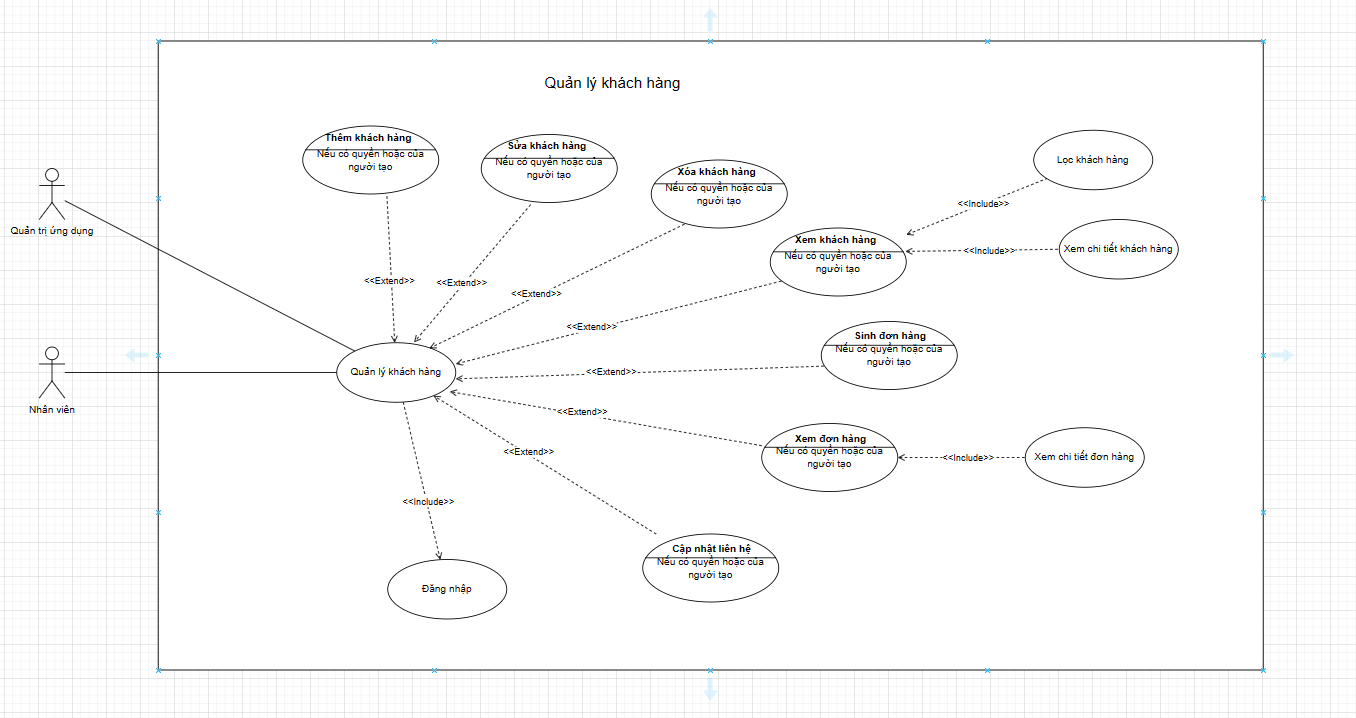
* + - 1. Đặc tả Use case Sinh khách hàng:
         1. Biểu đồ Use case:
         2. Biểu đồ trình tự:



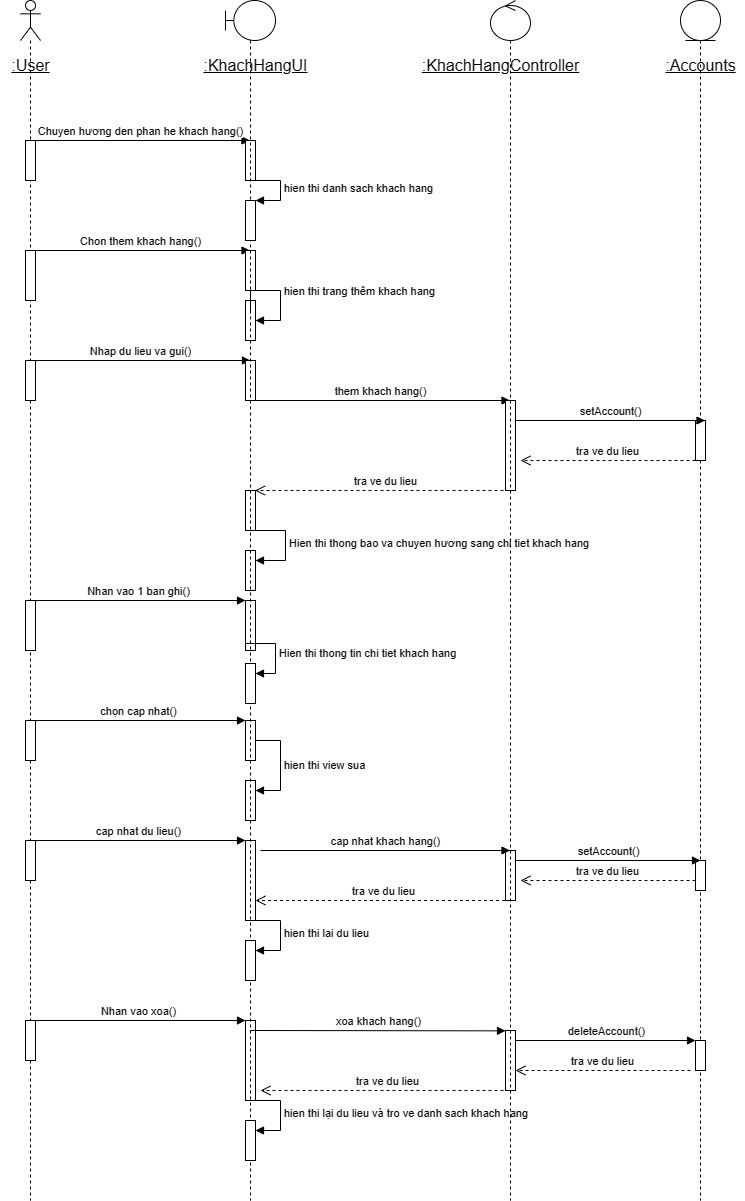
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng tạo khách hàng từ tiềm năng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Sinh khách hàng” trong màn hình chi tiết tiềm năng, hệ thống chuyển hướng đến trang thêm khách hàng và tự động fill các thông tin đã được lấy từ tiềm năng.
    - Người dùng chỉnh sửa dữ liệu và nhấn lưu, hệ thống lưu trữ khách hàng mới, xóa tiềm năng cũ và chuyển hướng đến màn hình chi tiết khách hàng
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 nếu người dùng nhấn cancel, hệ thống chuyển hướng người dùng về trang khách hàng.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Xóa bản ghi trong bảng Lead và thêm bản ghi trong bảng Account trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



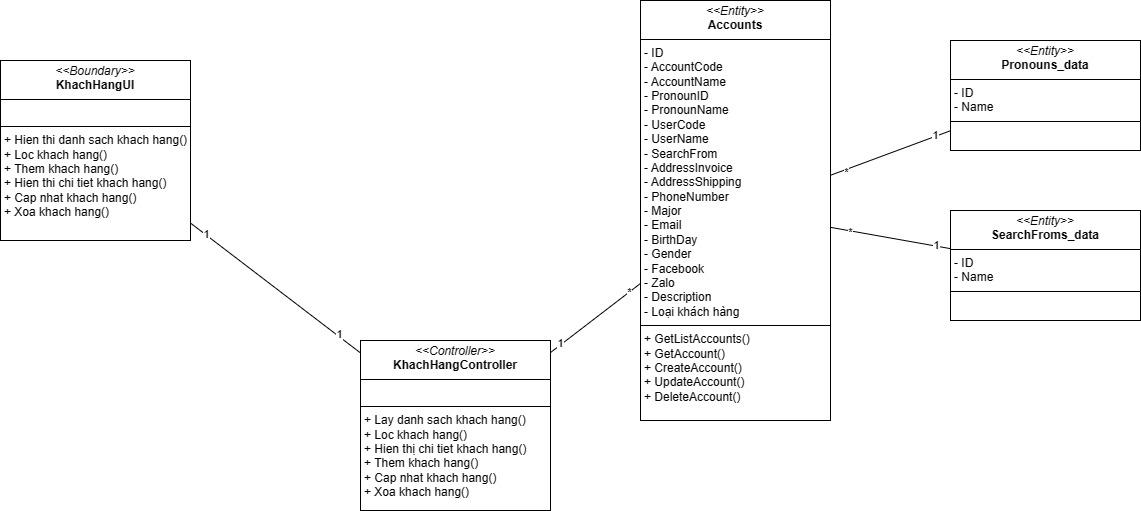
* + - 1. Đặc tả Use case CRUD khách hàng:
         1. Biểu đồ Use case:



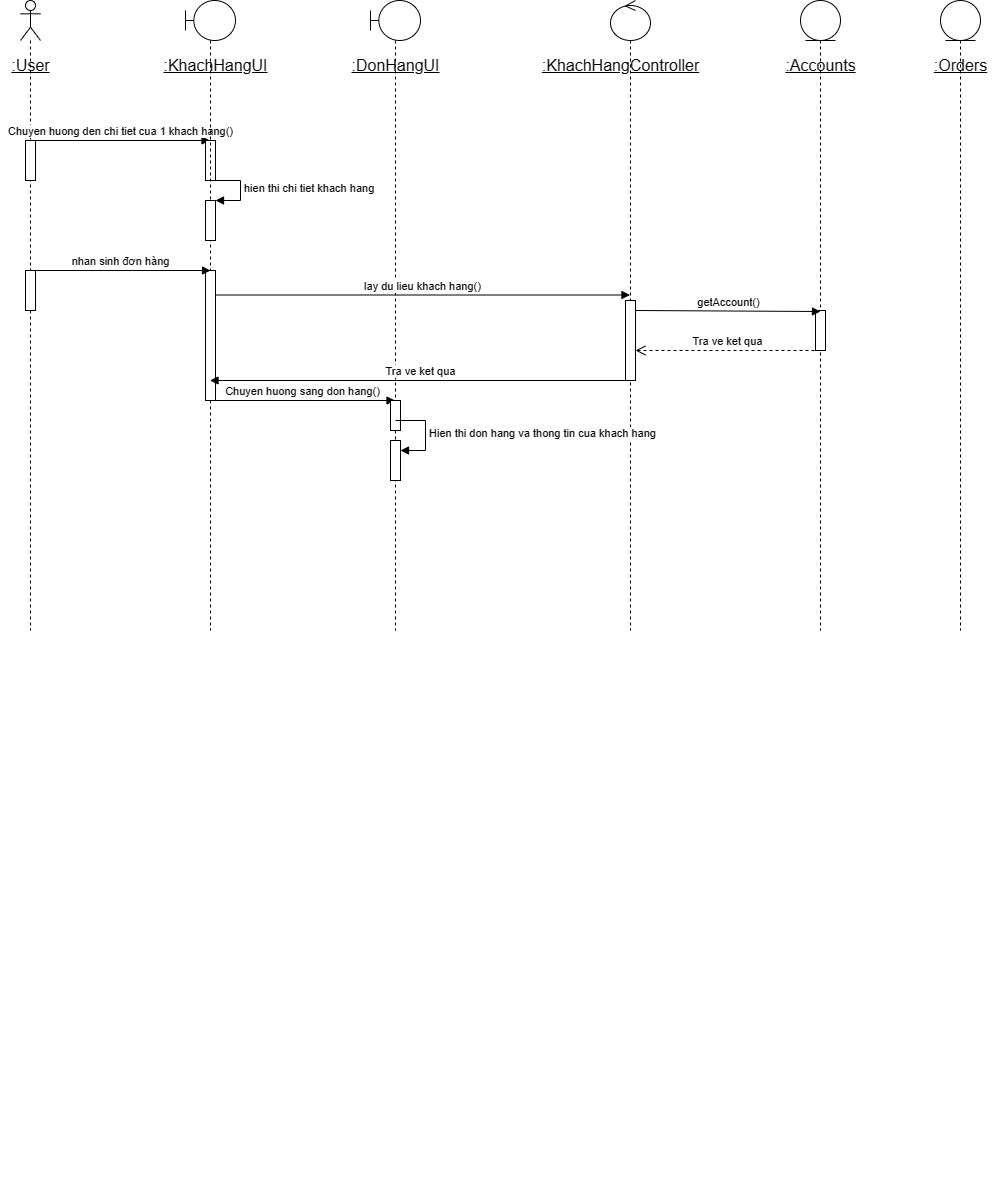
* + - * 1. Biểu đồ trình tự:



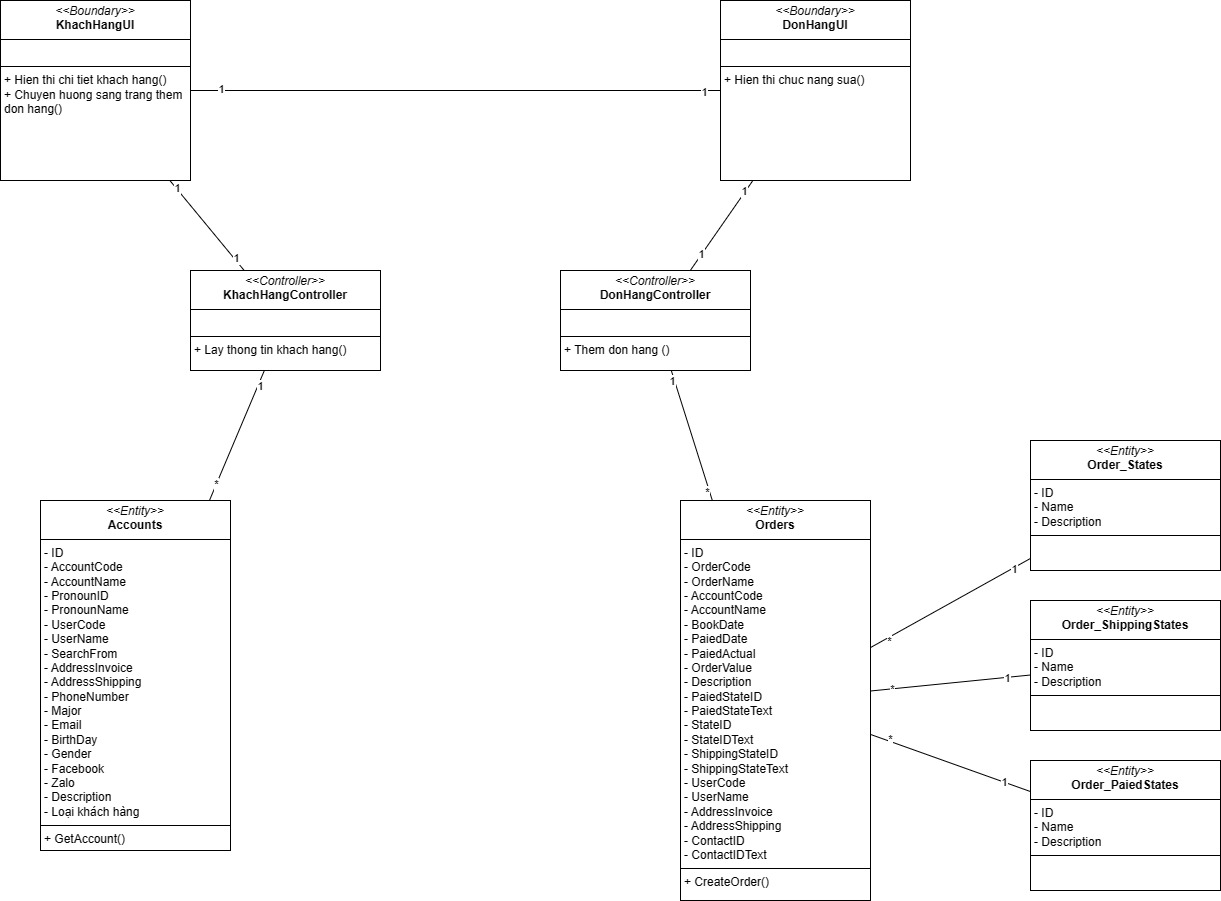
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ khách hàng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Khách hàng” trên thanh navbar, hệ thống hiển thị danh sách khách hàng.
    - 4 chức năng cho khách hàng:
      1. Xem danh sách khách hàng:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách khách hàng.
      2. Lọc khách hàng:
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách khách hàng theo các điều kiện đã thêm.
      3. Xem chi tiết khách hàng:
         1. Người dùng chọn 1 bản ghi trong danh sách khách hàng, hệ thống hiển thị chi tiết khách hàng.
      4. Cập nhật khách hàng:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm khách hàng:
         1. Tại màn hình danh sách khách hàng, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm khách hàng.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa khách hàng:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách khách hàng.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết khách hàng.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Account trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



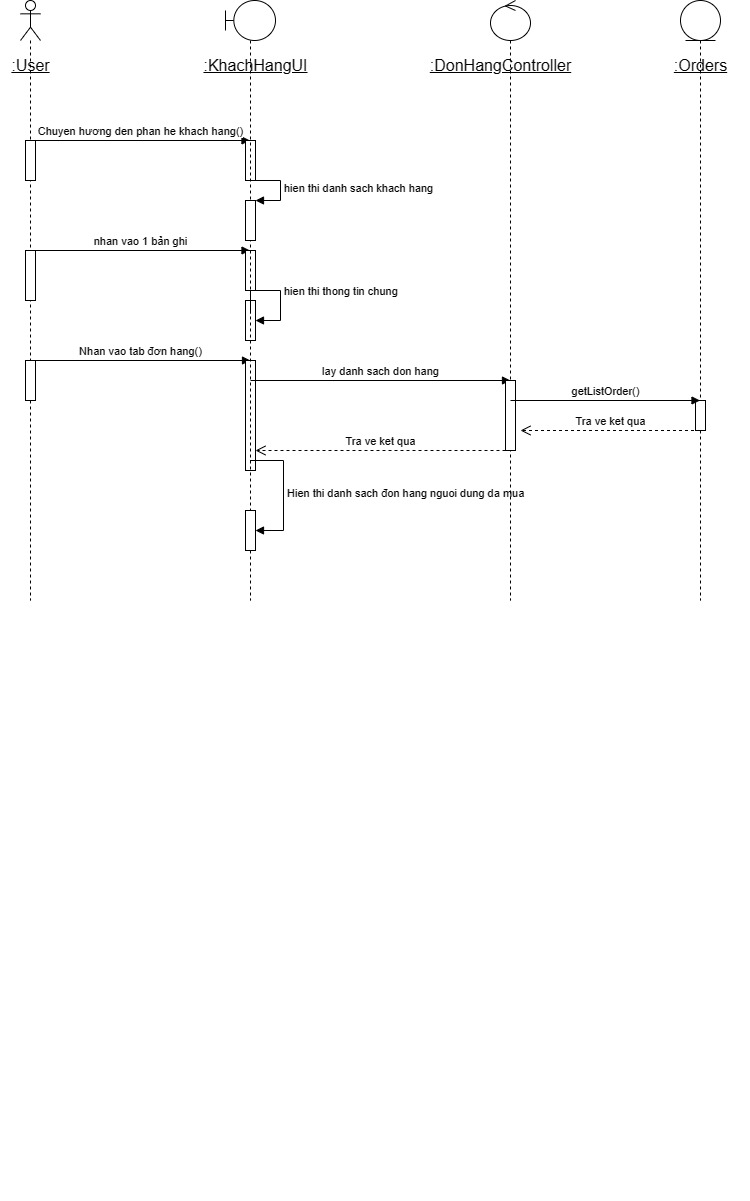
* + - 1. Đặc tả Use case Sinh đơn hàng:
         1. Biểu đồ Use case:
         2. Biểu đồ trình tự:



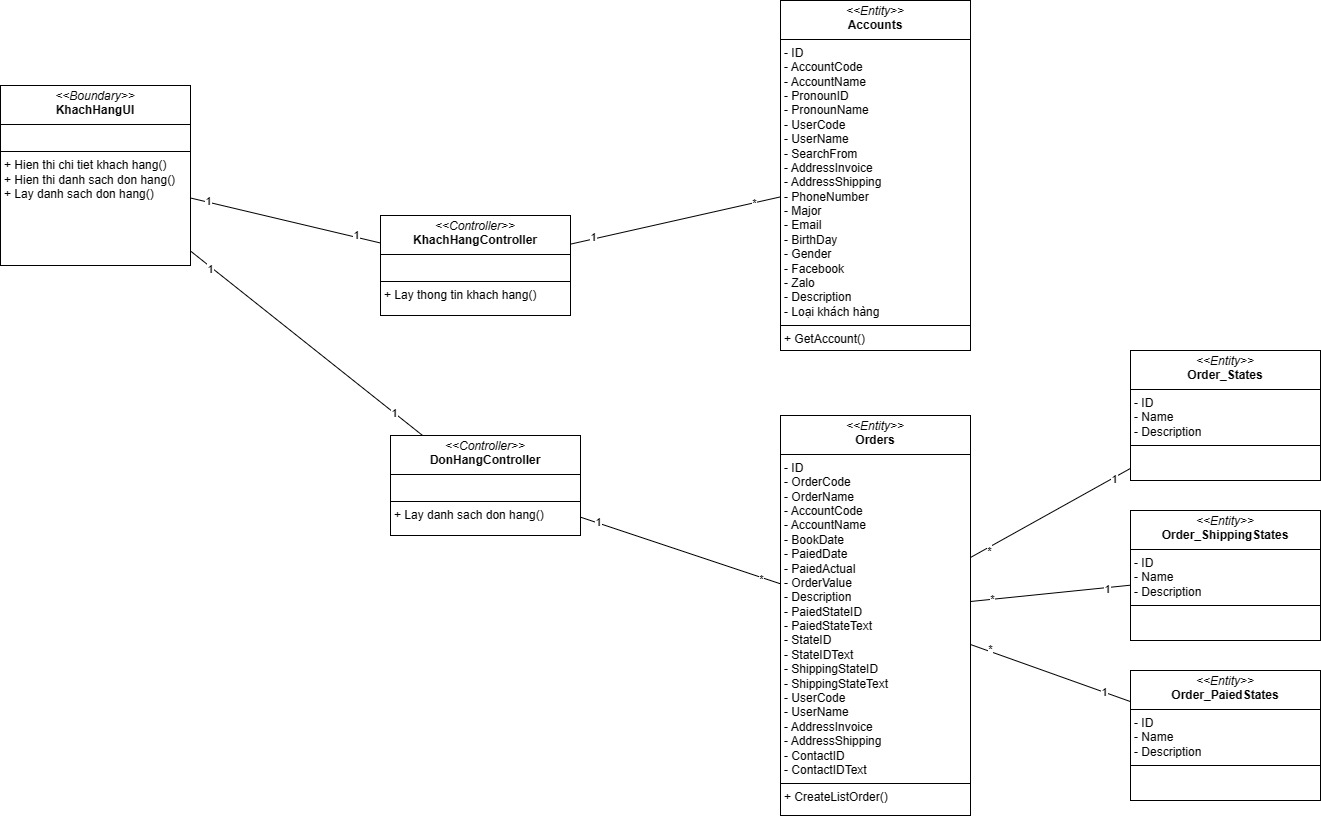
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng tạo đơn hàng từ khách hàng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Sinh đơn hàng” trong màn hình chi tiết khách hàng, hệ thống chuyển hướng đến trang thêm đơn hàng và tự động fill các thông tin đã được lấy từ khách hàng.
    - Người dùng chỉnh sửa dữ liệu và nhấn lưu, hệ thống lưu trữ đơn hàng mới và chuyển hướng đến màn hình chi tiết đơn hàng.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước 2 nếu người dùng nhấn cancel, hệ thống chuyển hướng người dùng về trang chi tiết khách hàng.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Nếu thành công thêm bản ghi trong bảng Orders trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



* + - 1. Đặc tả Use case Xem danh sách đơn hàng:
         1. Biểu đồ Use case:
         2. Biểu đồ trình tự:



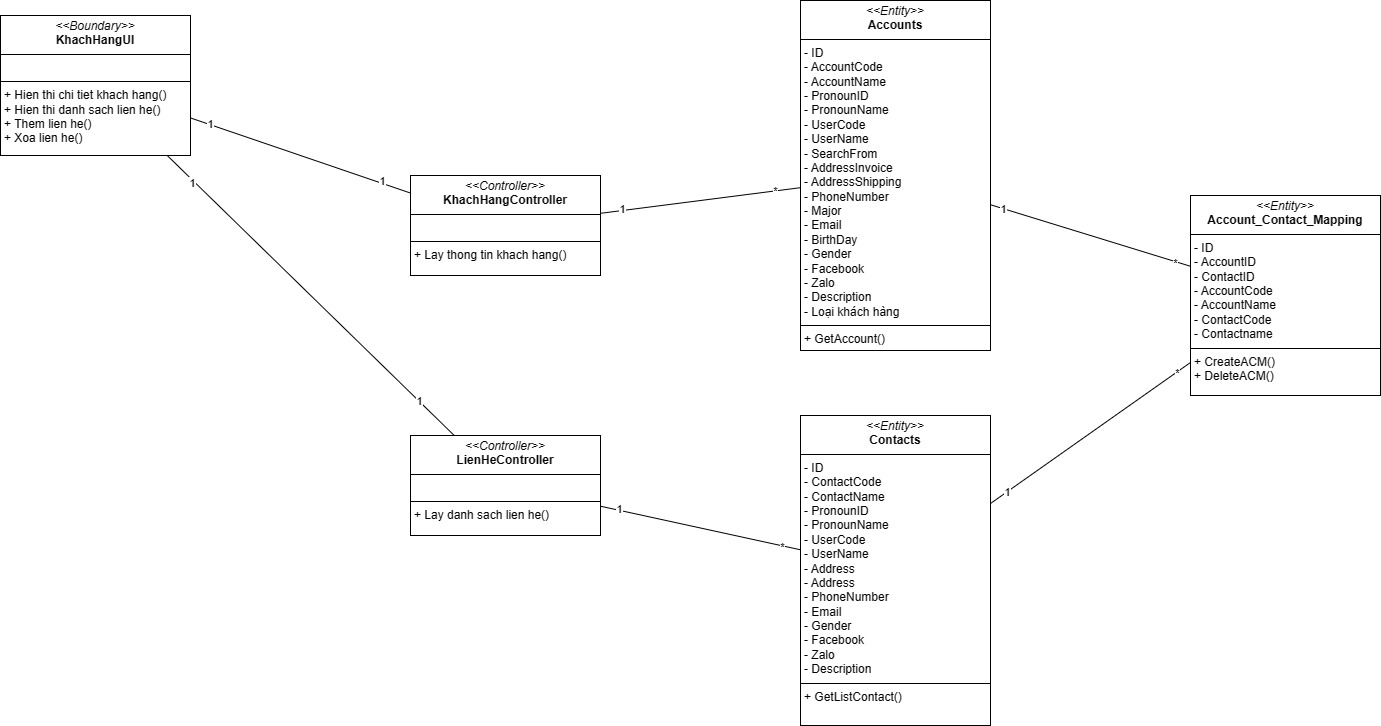
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem tất cả đơn hàng của khách hàng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn sang tab “Đơn hàng” trong màn hình chi tiết khách hàng, hệ thống chuyển sang tab đơn hàng và hiển thị danh sách đơn hàng của khách hàng.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Không có.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



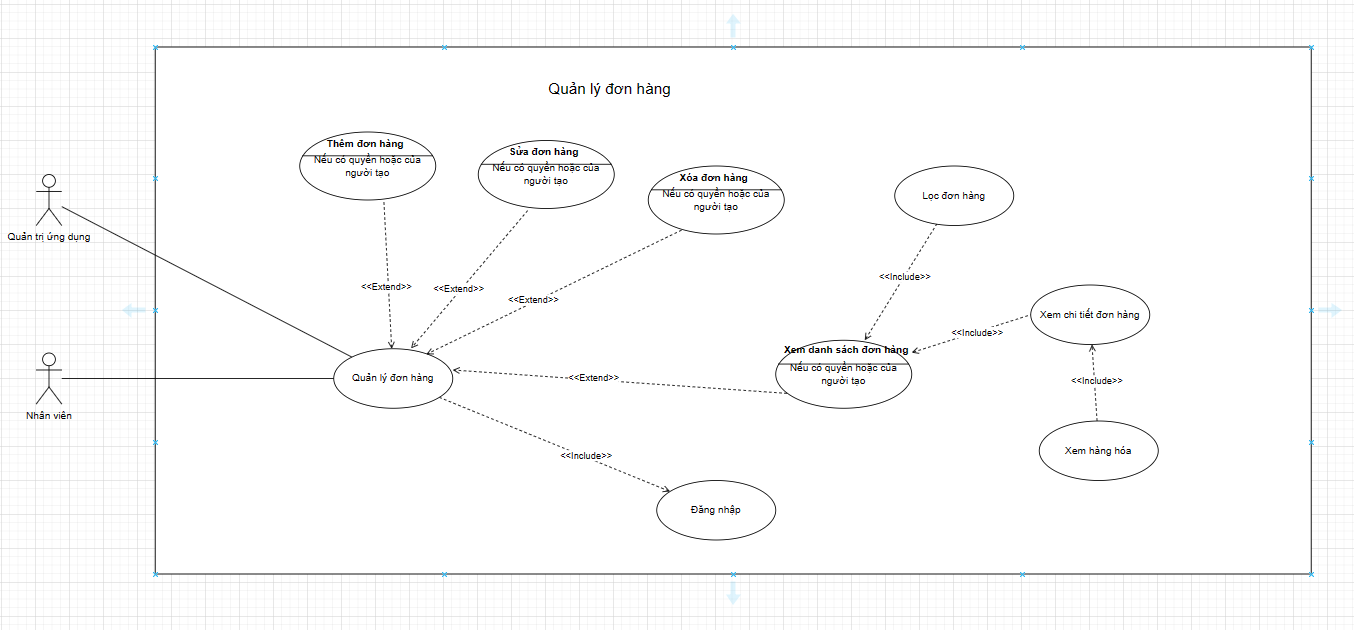
* + - 1. Đặc tả Use case Cập nhật liên hệ:
         1. Biểu đồ Use case:
         2. Biểu đồ trình tự:



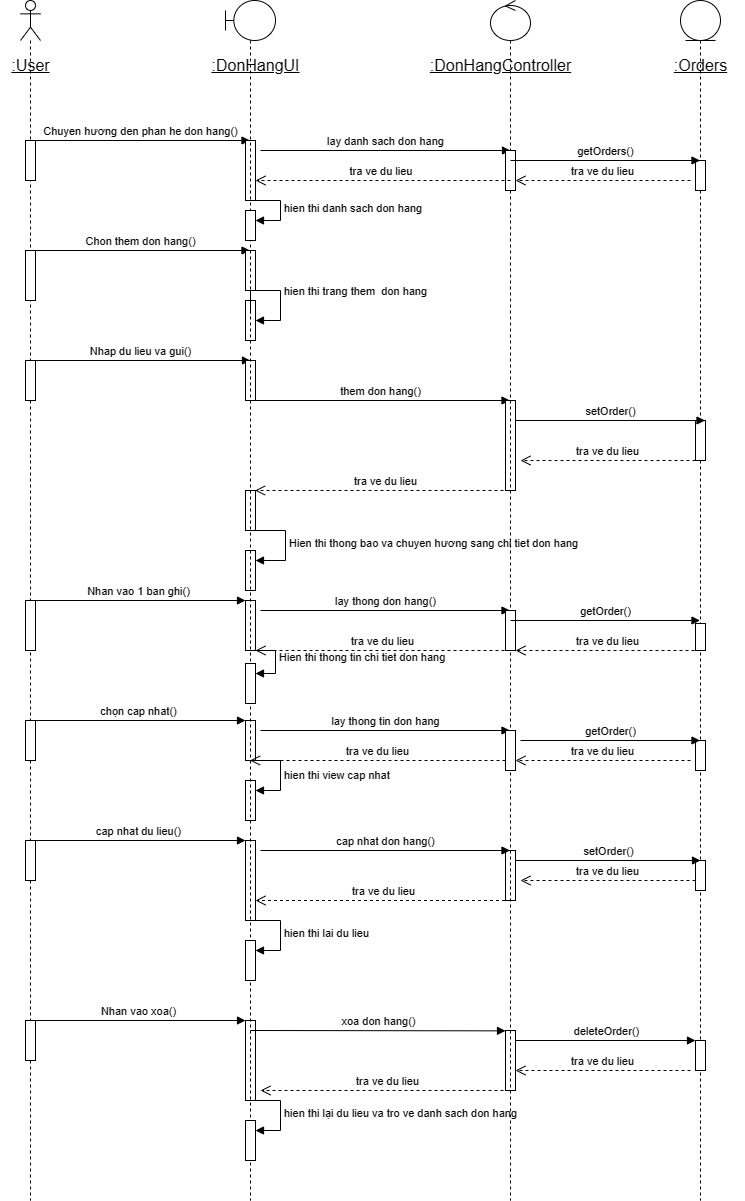
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng cập nhật lại các liên hệ của khách hàng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn sang tab “Liên hệ” trong màn hình chi tiết khách hàng, hệ thống chuyển sang tab liên hệ và hiển thị danh sách liên hệ của khách hàng.
    - 2 chức năng trong đây:
      1. Thêm:
         1. Người dùng chọn “thêm liên hệ”, hệ thống hiển thị danh sách liên hệ.
         2. Người dùng chọn liên hệ và xác nhận, hệ thống cập nhật thêm liên hệ vào danh sách liên hệ của khách hàng.
      2. Xóa:
         1. Người dùng nhấn xóa 1 bản ghi, hệ thống xóa liên hệ đó khỏi danh sách liên hệ của khách hàng
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Nếu thành công thì cập nhật lại bảng Contacts trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



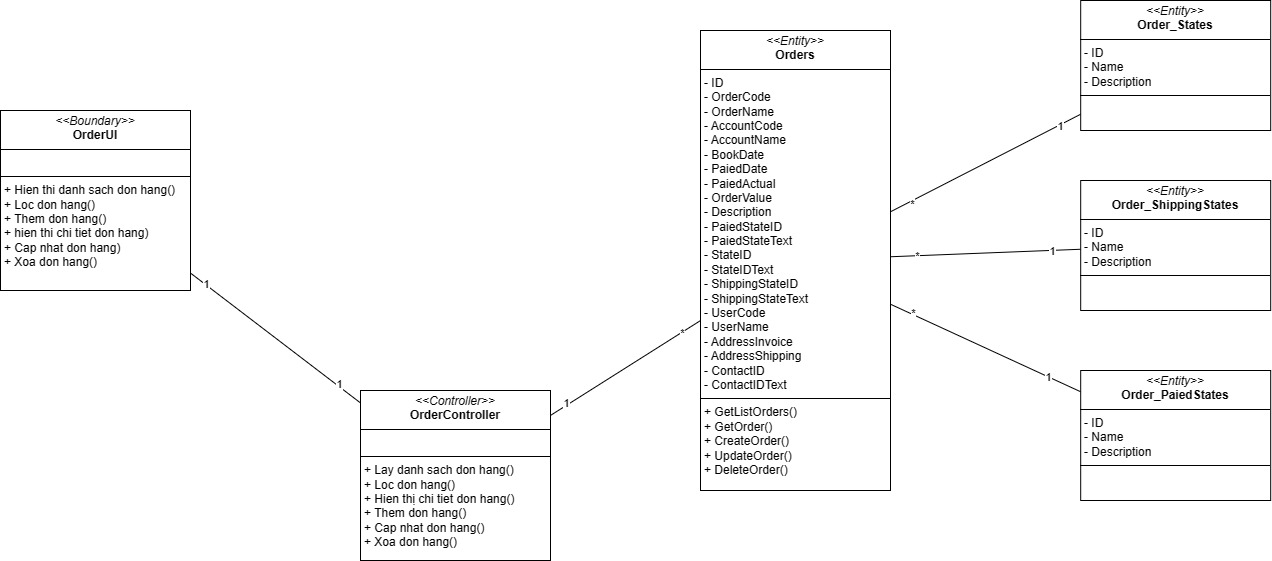
* + - 1. Đặc tả Use case CRUD đơn hàng:
         1. Biểu đồ Use case:



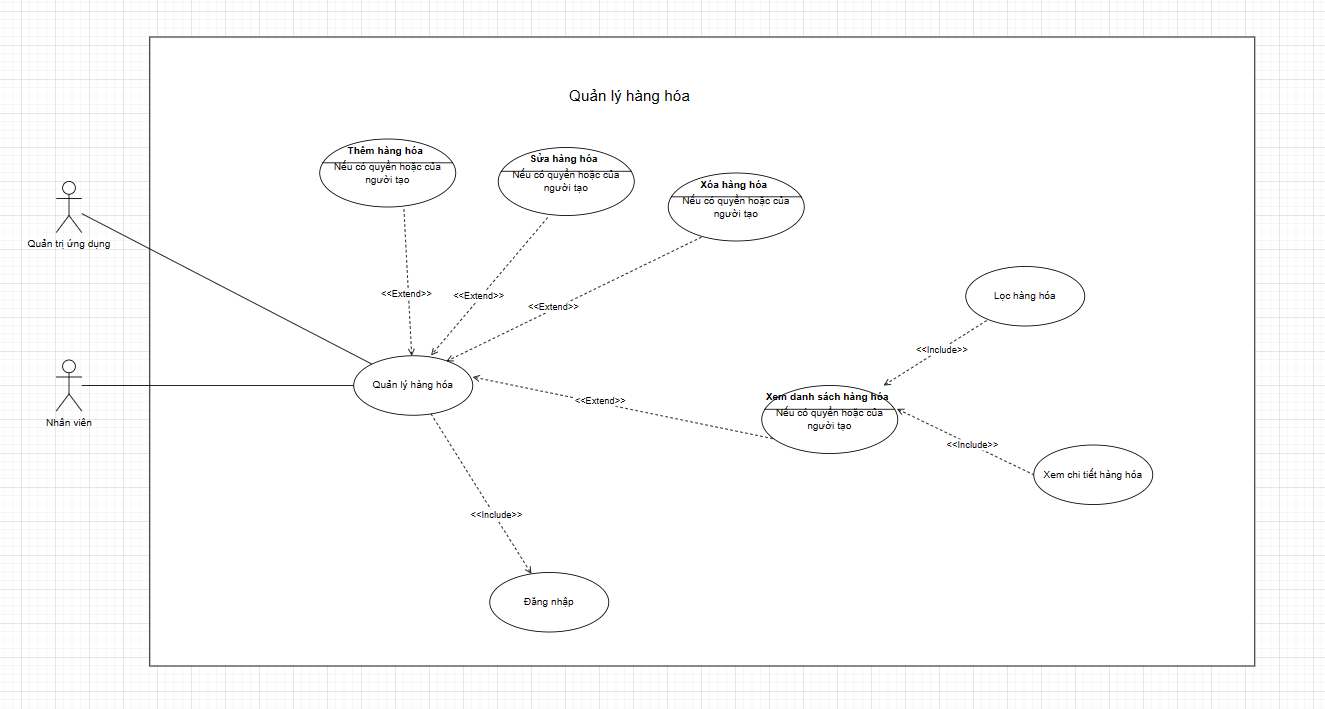
* + - * 1. Biểu đồ trình tự:



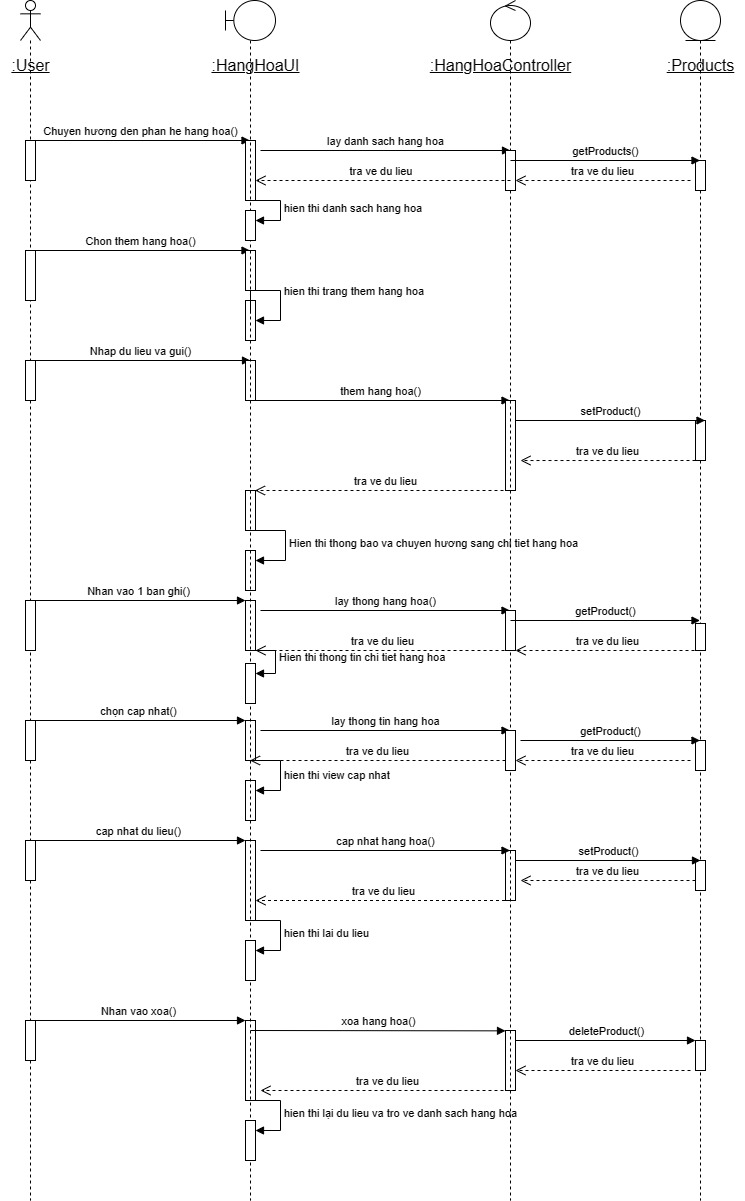
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ đơn hàng.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Đơn hàng” trên thanh navbar, hệ thống hiển thị danh sách đơn hàng.
    - 4 chức năng cho đơn hàng:
      1. Xem danh sách đơn hàng:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách đơn hàng.
      2. Lọc đơn hàng:
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách đơn hàng theo các điều kiện đã thêm.
      3. Xem chi tiết đơn hàng:
         1. Người dùng chọn 1 bản ghi trong danh sách đơn hàng, hệ thống hiển thị chi tiết đơn hàng.
      4. Cập nhật đơn hàng:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm đơn hàng:
         1. Tại màn hình danh sách đơn hàng, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm đơn hàng.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa đơn hàng:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách đơn hàng.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết khách hàng.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Order trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



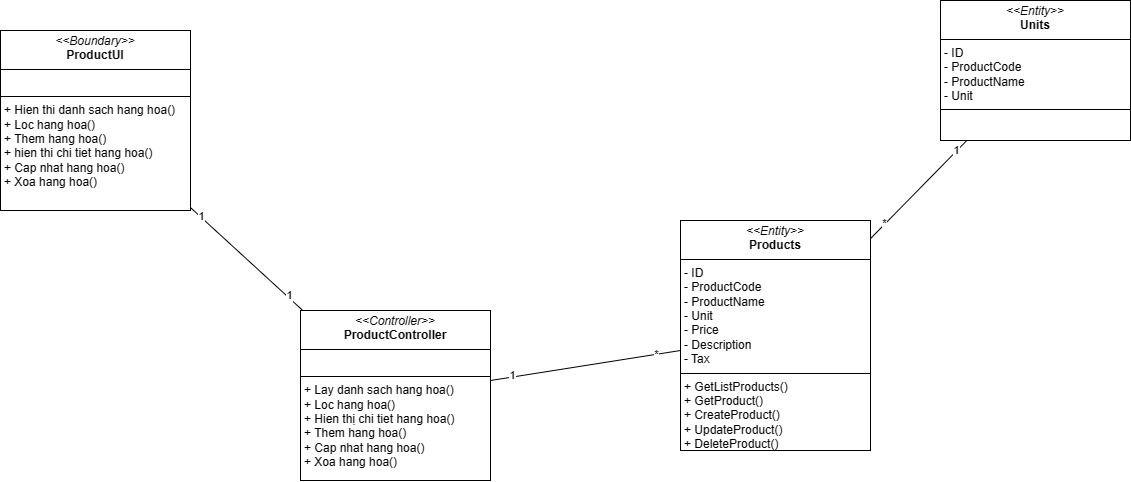
* + - 1. Đặc tả Use case CRUD hàng hóa:
         1. Biểu đồ Use case:



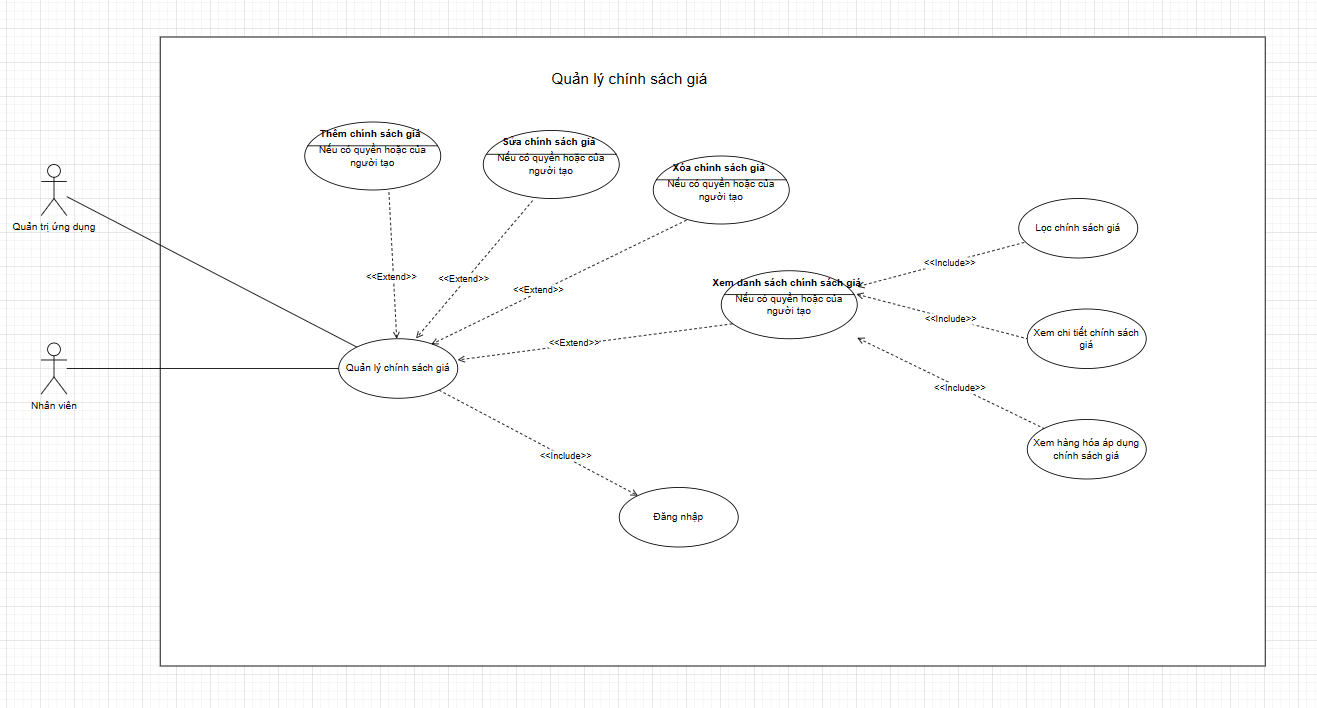
* + - * 1. Biểu đồ trình tự:



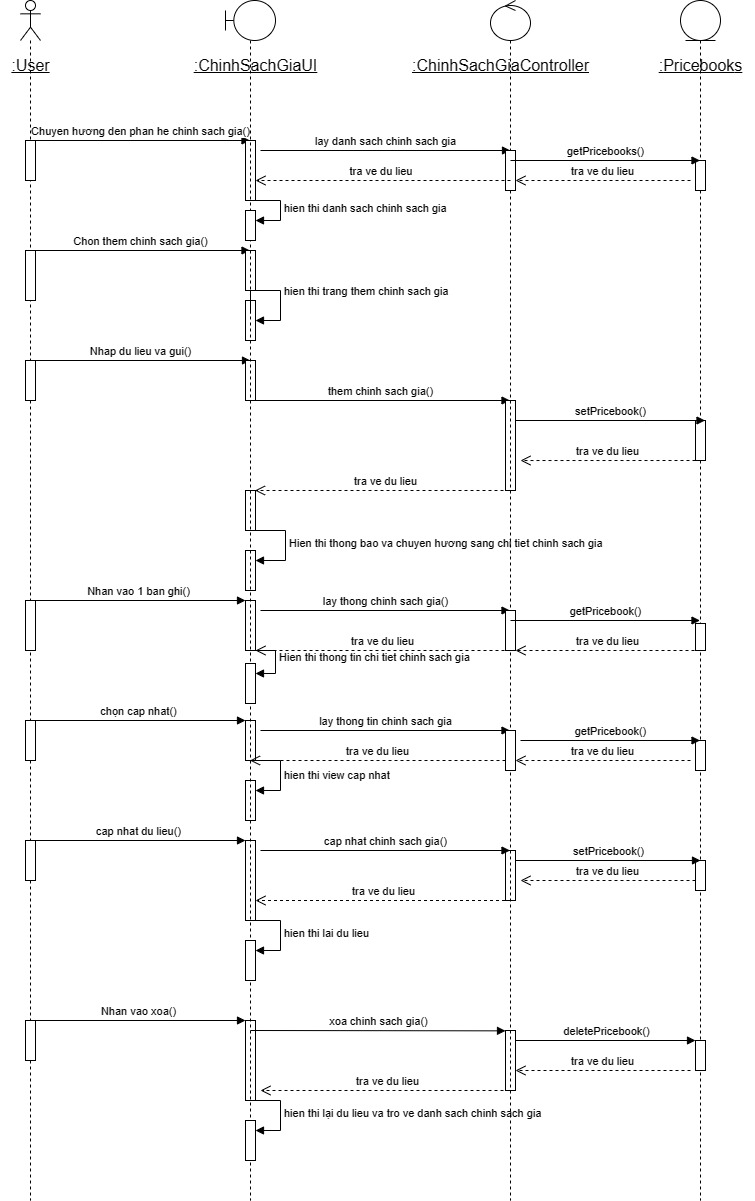
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ hàng hóa.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Hàng hóa” trên thanh navbar, hệ thống hiển thị danh sách hàng hóa.
    - 4 chức năng cho hàng hóa:
      1. Xem danh sách hàng hóa:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách hàng hóa.
      2. Lọc hàng hóa:
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách hàng hóa theo các điều kiện đã thêm.
      3. Xem chi tiết hàng hóa:
         1. Người dùng chọn 1 bản ghi trong danh sách hàng hóa, hệ thống hiển thị chi tiết hàng hóa.
      4. Cập nhật hàng hóa:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm hàng hóa:
         1. Tại màn hình danh sách hàng hóa, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm hàng hóa.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa hàng hóa:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách hàng hóa.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết hàng hóa.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Product trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



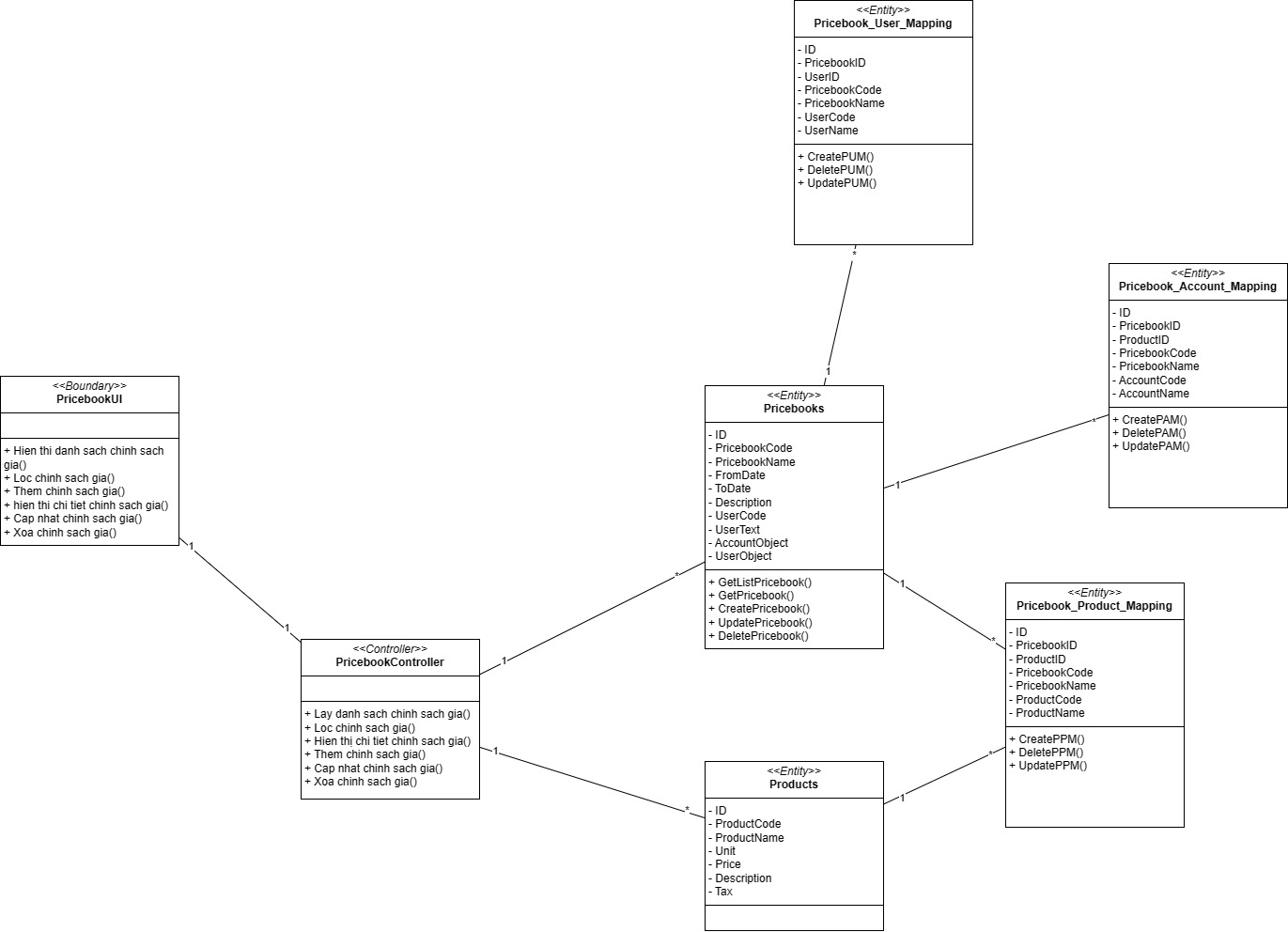
* + - 1. Đặc tả Use case CRUD chính sách giá:
         1. Biểu đồ Use case:



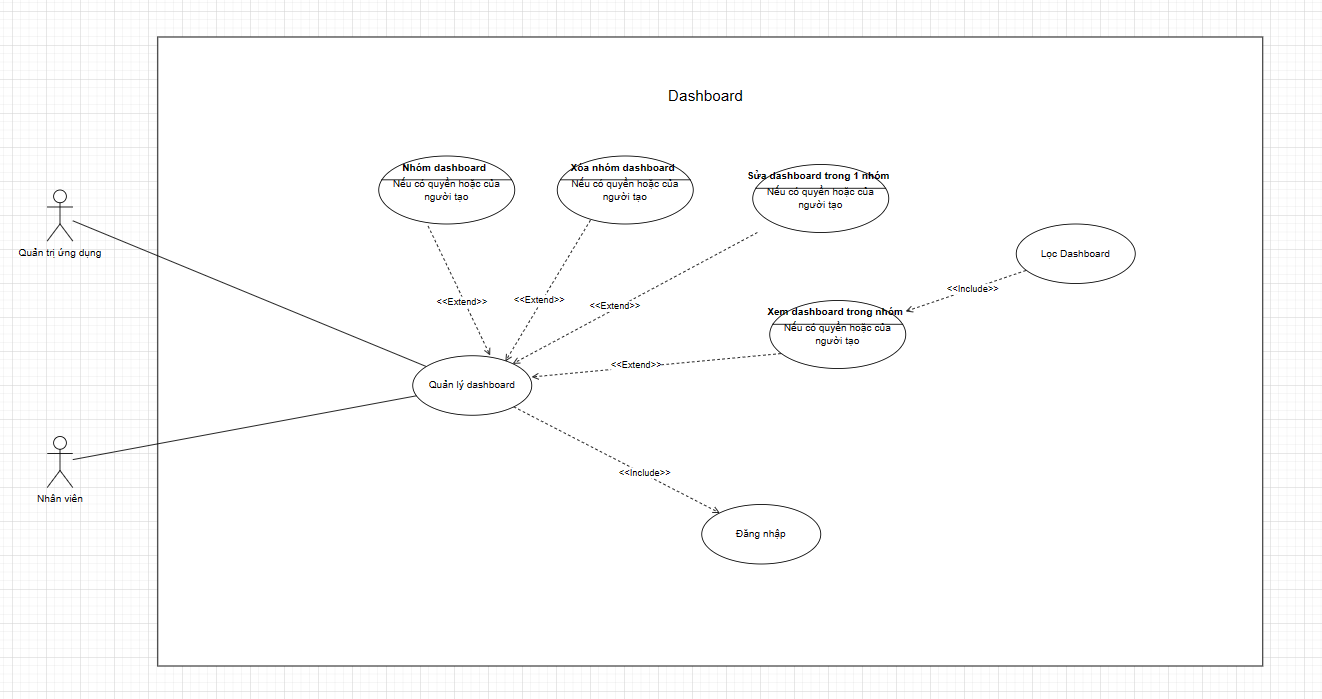
* + - * 1. Biểu đồ trình tự:



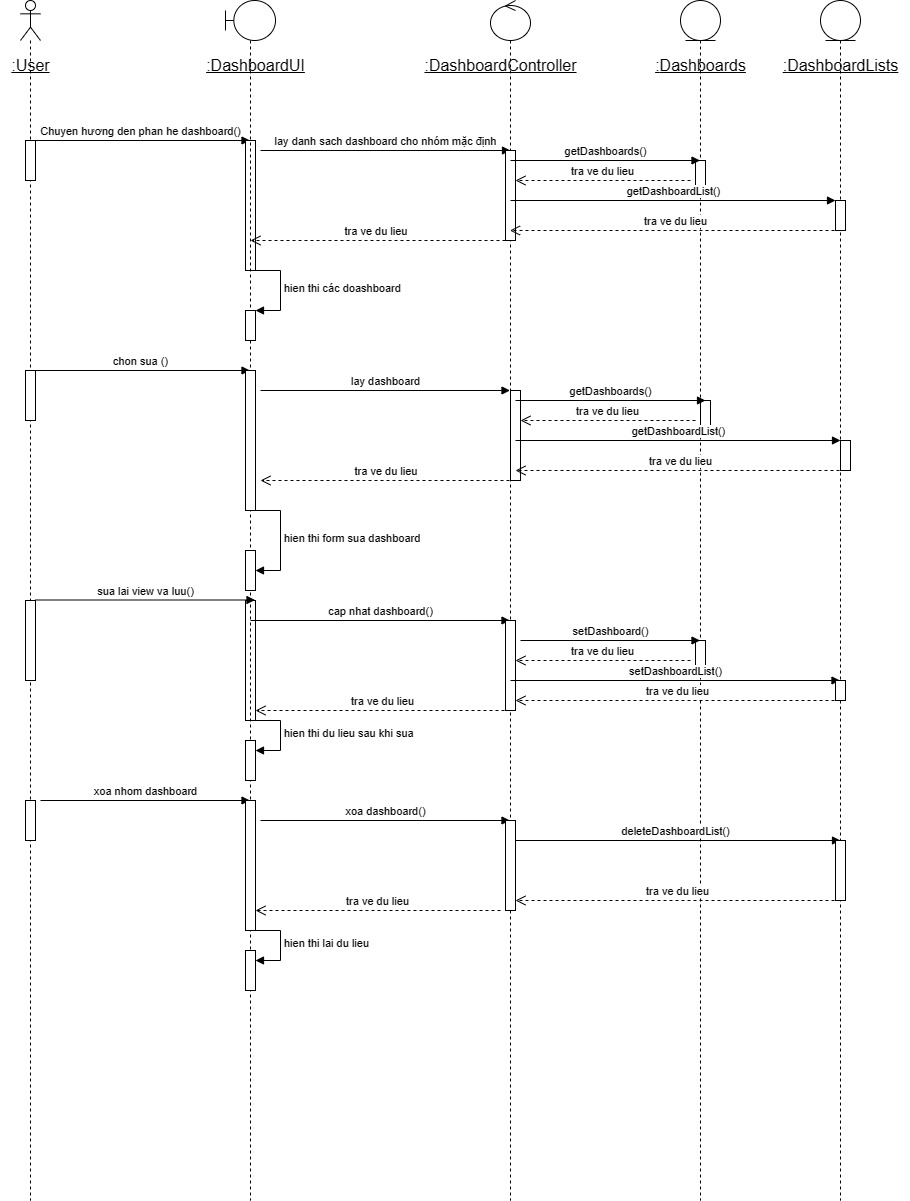
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ chính sách giá.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Chính sách giá” trên thanh navbar, hệ thống hiển thị danh sách chính sách giá.
    - 4 chức năng cho chính sách giá:
      1. Xem danh sách chính sách giá:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách chính sách giá.
      2. Lọc chính sách giá
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách chính sách giá theo các điều kiện đã thêm.
      3. Xem chi tiết chính sách giá:
         1. Người dùng chọn 1 bản ghi trong danh sách chính sách giá, hệ thống hiển thị chi tiết chính sách giá.
      4. Cập nhật chính sách giá:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm chính sách giá:
         1. Tại màn hình danh sách chính sách giá, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm chính sách giá.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa chính sách giá:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách chính sách giá.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết chính sách giá.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Pricebooks trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



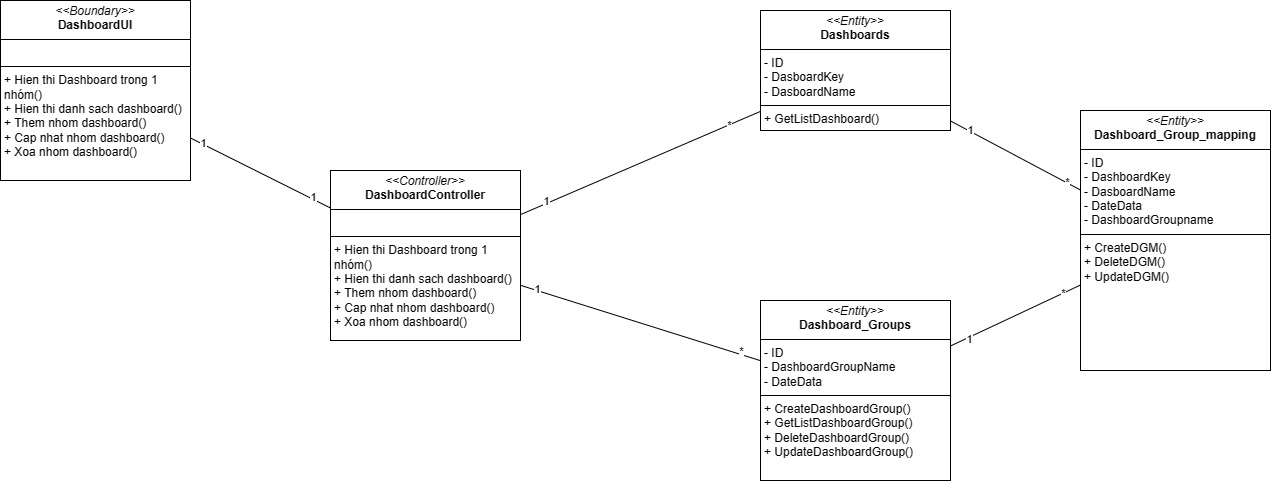
* + - 1. Đặc tả Use case Dashboard:
         1. Biểu đồ Use case:



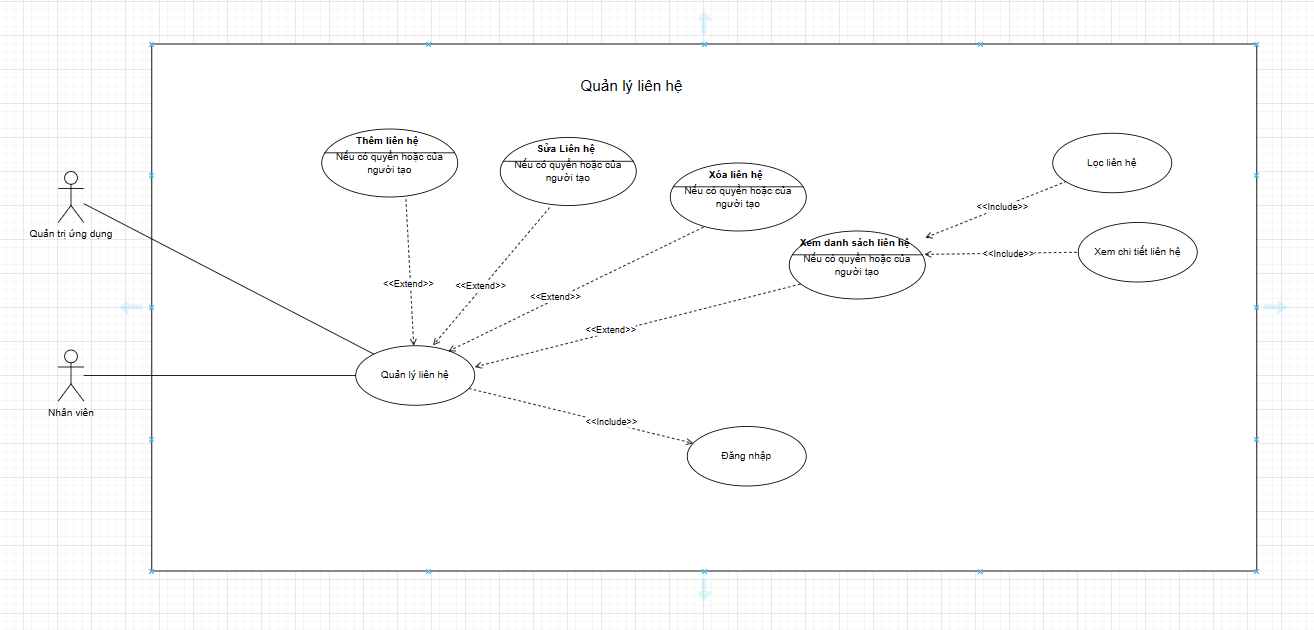
* + - * 1. Biểu đồ trình tự:



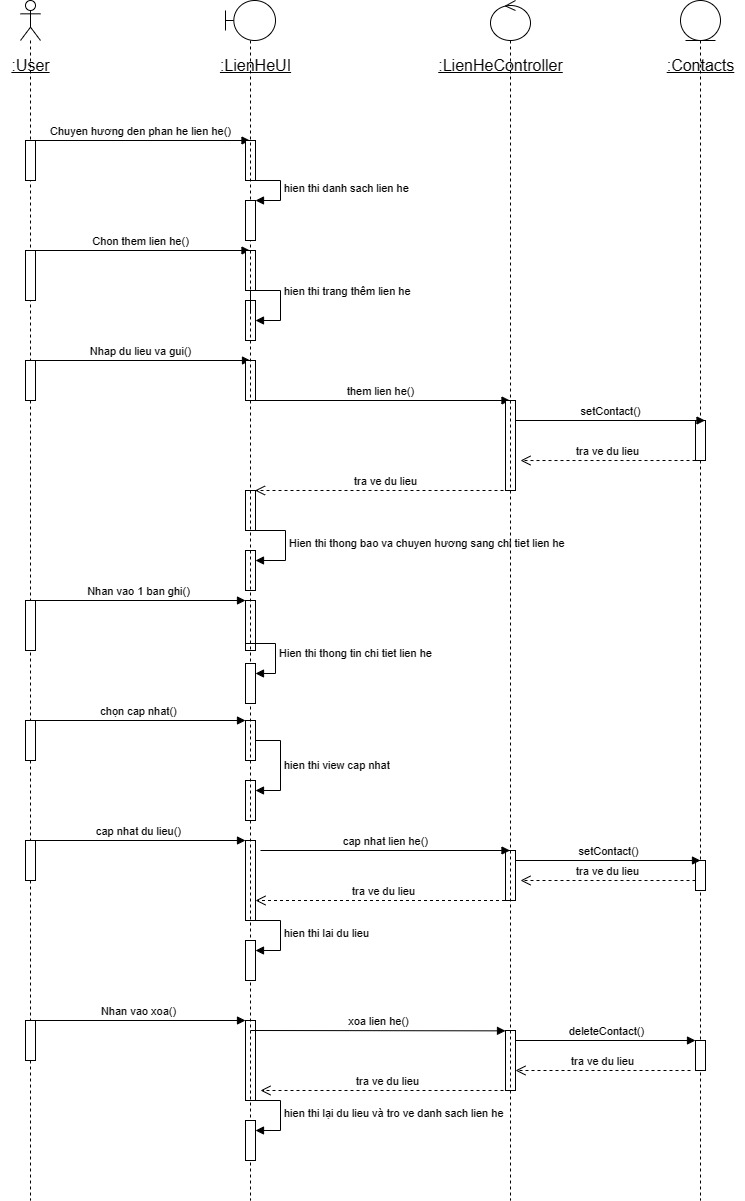
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ dashboard.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Dashboard” trên thanh navbar, hệ thống hiển thị danh sách nhóm dashboard và các dashboard có trong nhóm đó.
    - 4 chức năng cho dashboard:
      1. Xem dashboard trong nhóm dashboard:
         1. Người dùng chọn 1 nhóm dashboard trong danh sách dashboard, hệ thống hiển thị tất cả dashboard trong nhóm đó.
      2. Cập nhật dashboard trong nhóm:
         1. Trong bước 1.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các dashboard có trong nhóm và danh sách dashboard có trong hệ thống.
         2. Người dùng chỉnh sửa và nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      3. Thêm nhóm dashboard:
         1. Người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm nhóm dashboard.
         2. Người dùng chọn các dashboard cho nhóm và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      4. Xóa nhóm dashboard:
         1. Tại màn hình 1.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về nhóm dashboard đầu tiên.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 4.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống ẩn popup khỏi trang và giữ nguyên dữ liệu.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Dashboard và Dashboard\_Group trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



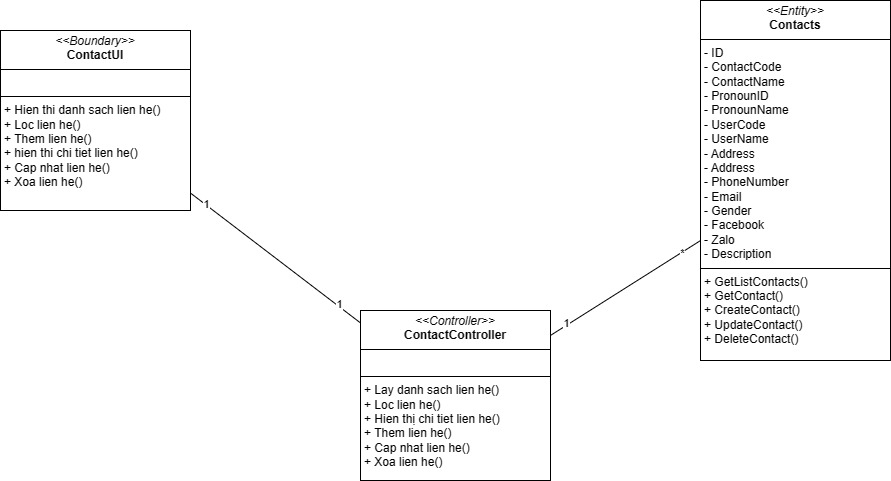
* + - 1. Đặc tả Use case CRUD Liên hệ:
         1. Biểu đồ Use case:



* + - * 1. Biểu đồ trình tự:



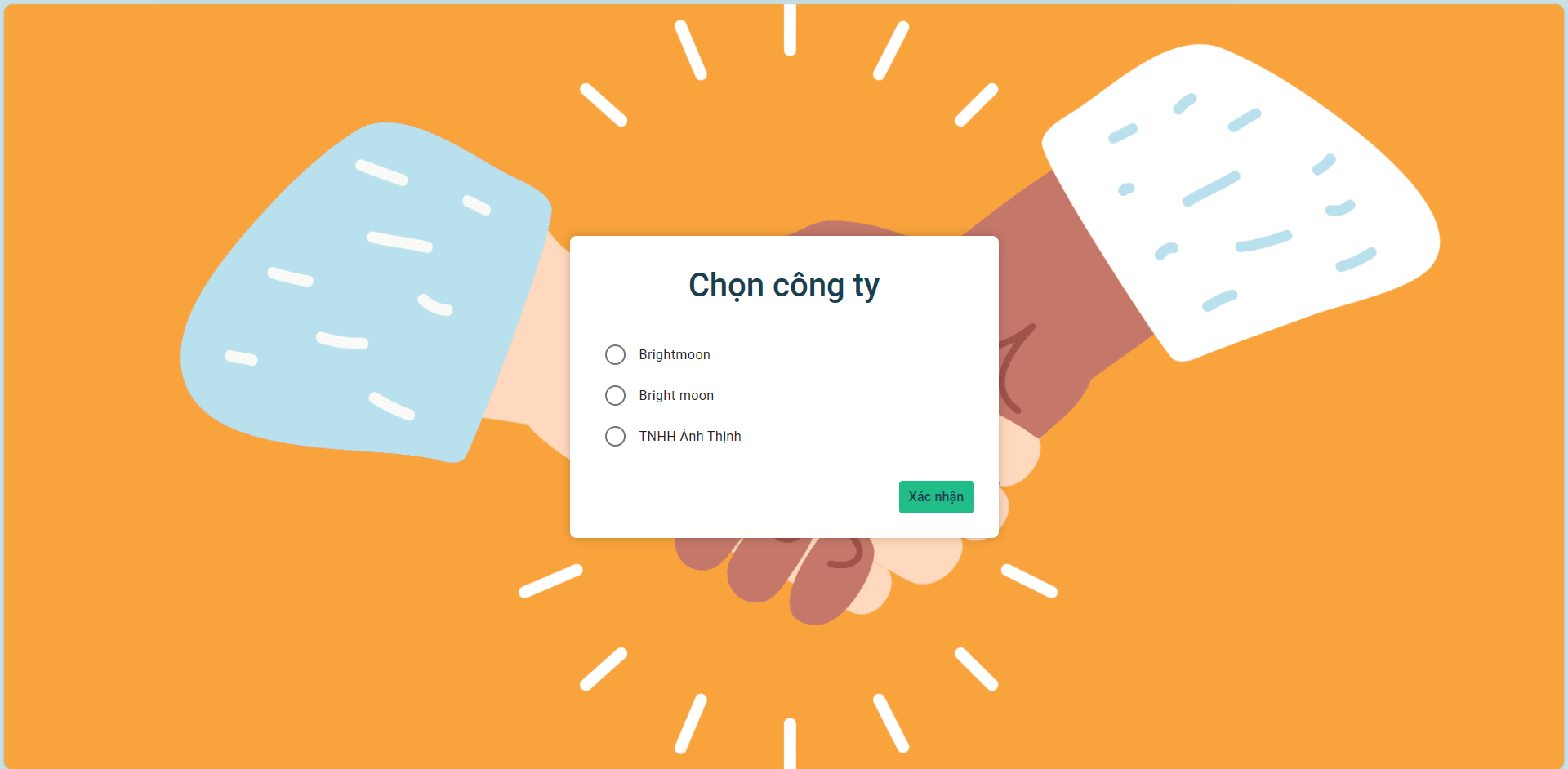
* + - * 1. Đặc tả chi tiết:
* **Mô tả vắn tắt:** Usecase này cho phép người dùng xem thêm sửa xóa phân hệ liên hệ.
* **Luồng sự kiện:**
  + **Luồng cơ bản:**
    - Use case này bắt đầu khi người dùng nhấn vào nút “Liên hệ” trên thanh navbar, hệ thống hiển thị danh sách liên hệ.
    - 4 chức năng cho liên hệ:
      1. Xem danh sách liên hệ:
         1. Khi hệ thống chuyển hướng người dùng đến trang, mặc định load phân trang danh sách liên hệ.
      2. Lọc liên hệ:
         1. Người dùng chọn các trường cần lọc và nhấn lưu, hệ thống hiển thị danh sách chính sách giá theo các điều kiện đã thêm.
      3. Xem chi tiết liên hệ:
         1. Người dùng chọn 1 bản ghi trong danh sách liên hệ, hệ thống hiển thị chi tiết liên hệ.
      4. Cập nhật liên hệ:
         1. Trong bước 3.a, người dùng nhấn sửa, hệ thống chuyển hướng đến trang sửa và hiển thị các thông tin cũ của bản ghi.
         2. Người dùng nhấn lưu, hệ thống lưu trữ lại dữ liệu và hiển thị dữ liệu đã được sửa.
      5. Thêm liên hệ:
         1. Tại màn hình danh sách chính sách giá, người dùng nhấn “Thêm”, hệ thống chuyển hướng đến trang thêm liên hệ.
         2. Người dùng điền đầy đủ thông tin và nhấn lưu, hệ thống lưu trữ dữ liệu và hiển thị lại dữ liệu đã lưu.
      6. Xóa liên hệ:
         1. Tại màn hình 3.a, người dùng nhấn nút xóa, hệ thống hiển thị thông báo xác nhận xóa.
         2. Người dùng nhấn xác nhận, hệ thống xóa bản ghi và chuyển hướng người dùng về trang danh sách liên hệ.
    - Người dùng nhấn vào phân hệ khác hoặc cài đặt, hệ thống chuyển hướng người dùng đến trang khác.
    - Use case kết thúc.
  + **Luồng rẽ nhánh:**
    - Tại bước xóa 6.a, người dùng nhấn vào “Hủy” trong popup xác nhận, hệ thống quay về trang chi tiết liên hệ.
* **Các yêu cầu đặc biệt:** Người dùng là có vai trò là quản trị hệ thống hoặc người vị trí có từng quyền tương ứng.
* **Tiền điều kiện**: Không có.
* **Hậu điều kiện:** Thêm, Cập nhật, xóa bản ghi trong bảng Contacts trong domain.
* **Điểm mở rộng:** Không có.
  + - * 1. Biểu đồ lớp:



## **Giao diện Website quản lý:**



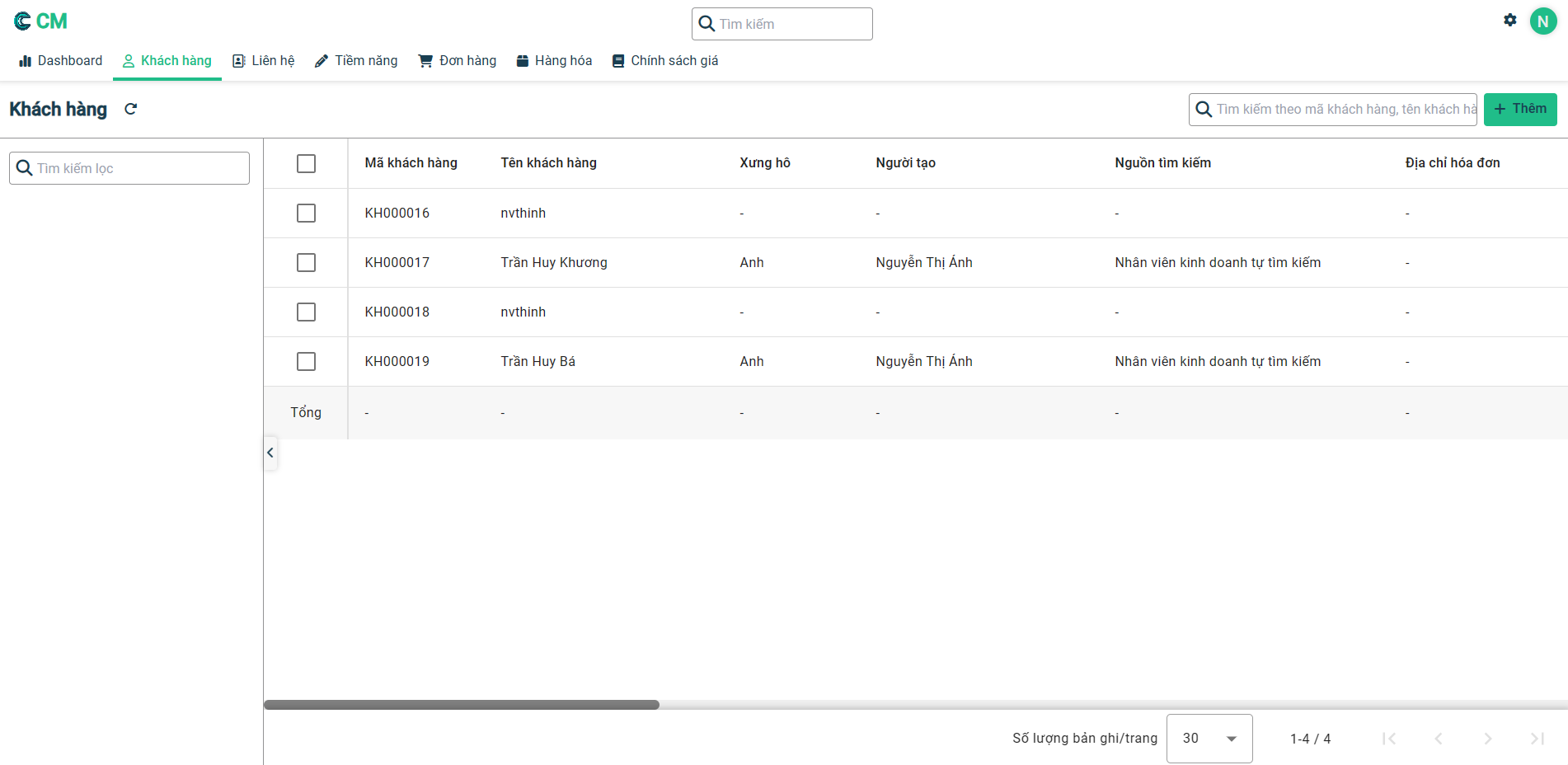
Giao diện đăng nhập: Cho phép người dùng đăng nhập vào ứng dụng



Giao diện chọn công ty: Người dùng có nhiều công ty khách nhau sẽ được chọn công ty đăng nhập làm việc trên công ty đó.



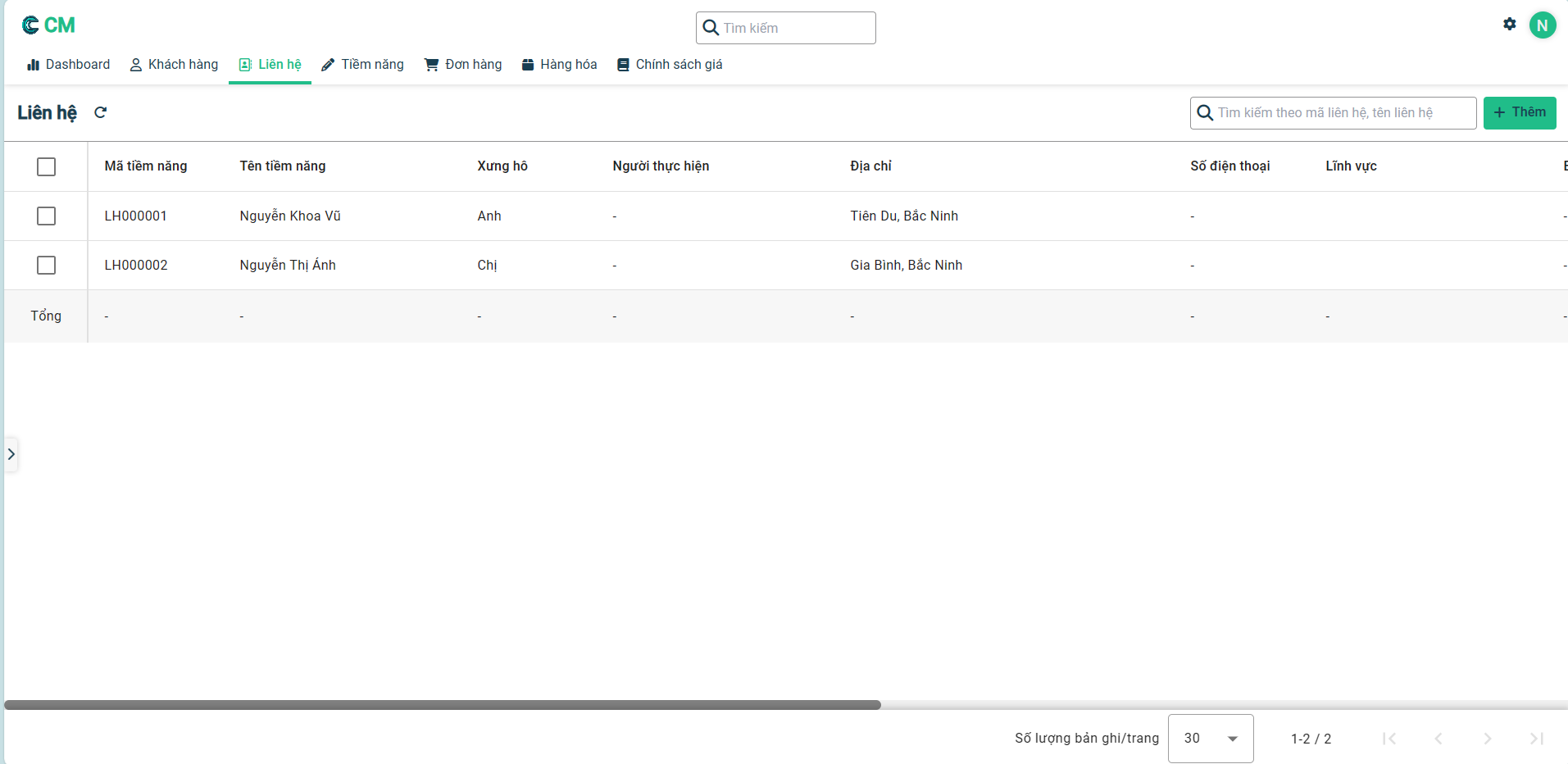
Màn hình đăng ký: Người dùng đăng ký công ty mới hay đăng ký tài khoản tại màn hình này. Khi đã có tài khoản, hệ thống sẽ hiển thị thông báo xác nhận tạo công ty mới.



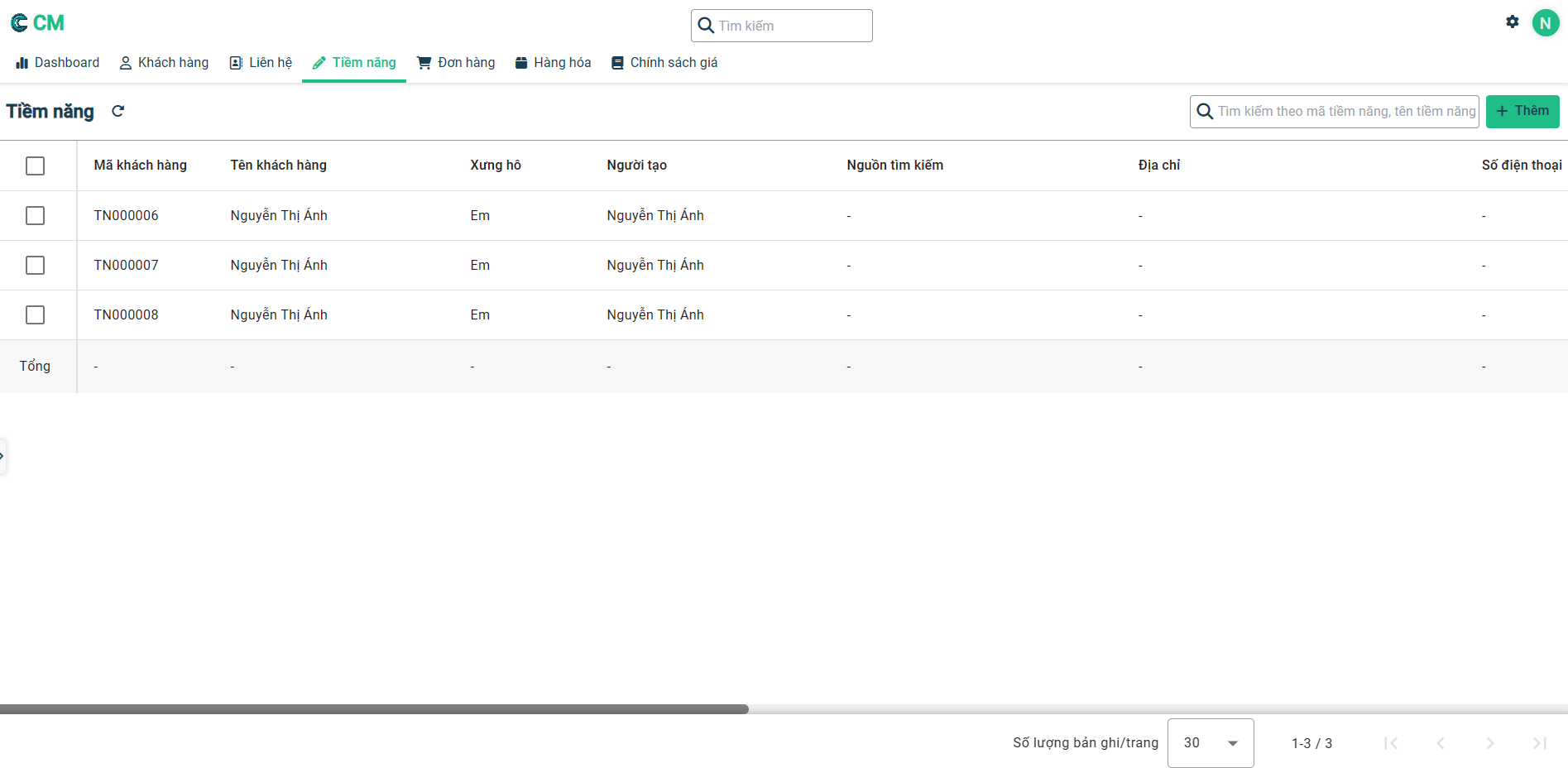
Màn hình quản lý Khách hàng: Tạo mới khách hàng, cập nhật và xóa dữ liệu khách hàng. Liên hệ sẽ được liên kết đến đây, mục đích là khách hàng là có thể là 1 công ty và liên hệ là 1 người trong công ty đó để trao đổi thông tin qua liên hệ.



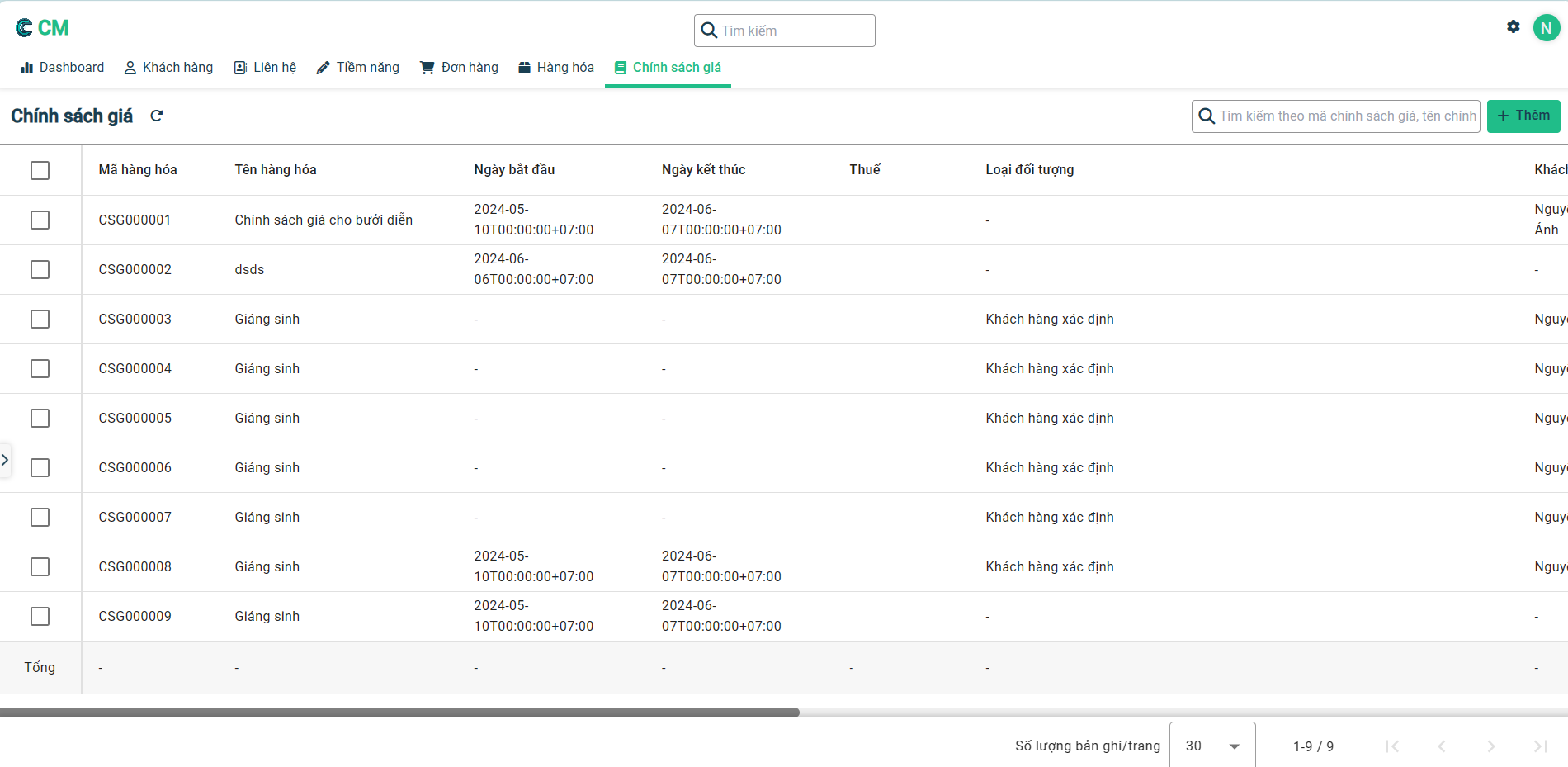
Màn hình thiết lập truy cập: Để quản trị viên ứng dụng (người tạo ra công ty) cho phép nhân viên đăng nhập ứng dụng hay không. Nghiệp vụ là khi nhân viên vào/ ra khỏi công ty thì sẽ ngắt kết nối không cho họ truy cập ứng dụng.



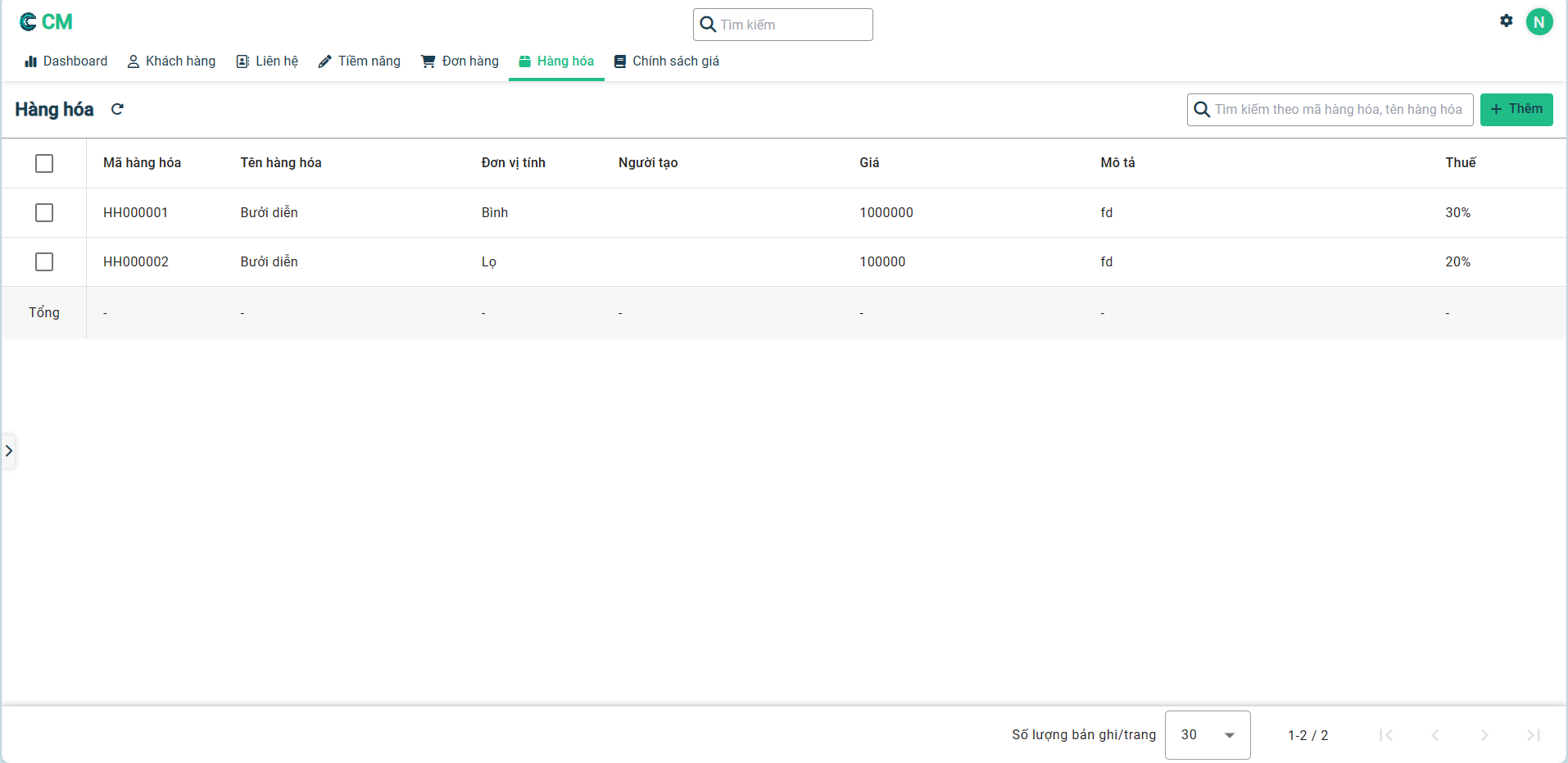
Màn hình quản lý liên hệ: Là nơi lưu trữ thông tin liên hệ của khách hàng. Mỗi khách hàng sẽ có nhiều liên hệ khác nhau tùy thời điểm. Do đó, cần lưu trữ lại các liên hệ của khách hàng để tiện trao đổi chăm sóc khách hàng.



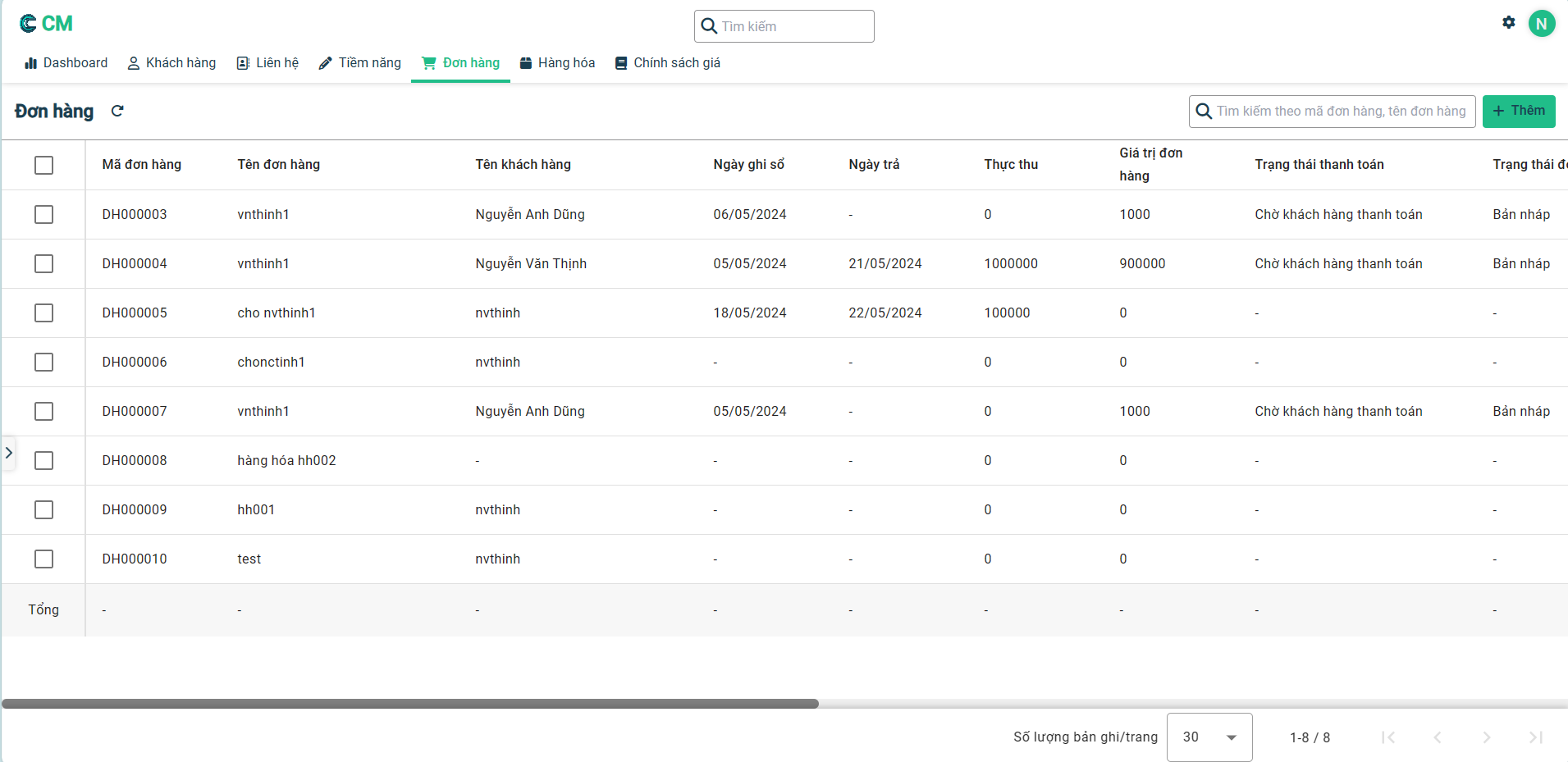
Màn hình tiềm năng: Là nơi lưu trữ khách hàng tiềm năng của công ty, những khách hàng này là người có ý định mua hàng, đang xem những sản phẩm của công ty. Họ cần được chăm sóc và chuyển hóa thành khách hàng khi mua hàng của công ty. Trong chức năng quản lý tiềm năng, có chức năng chuyển đổi thành khách hàng, giúp tiết kiệm thời gian nhập lại thông tin khách hàng.



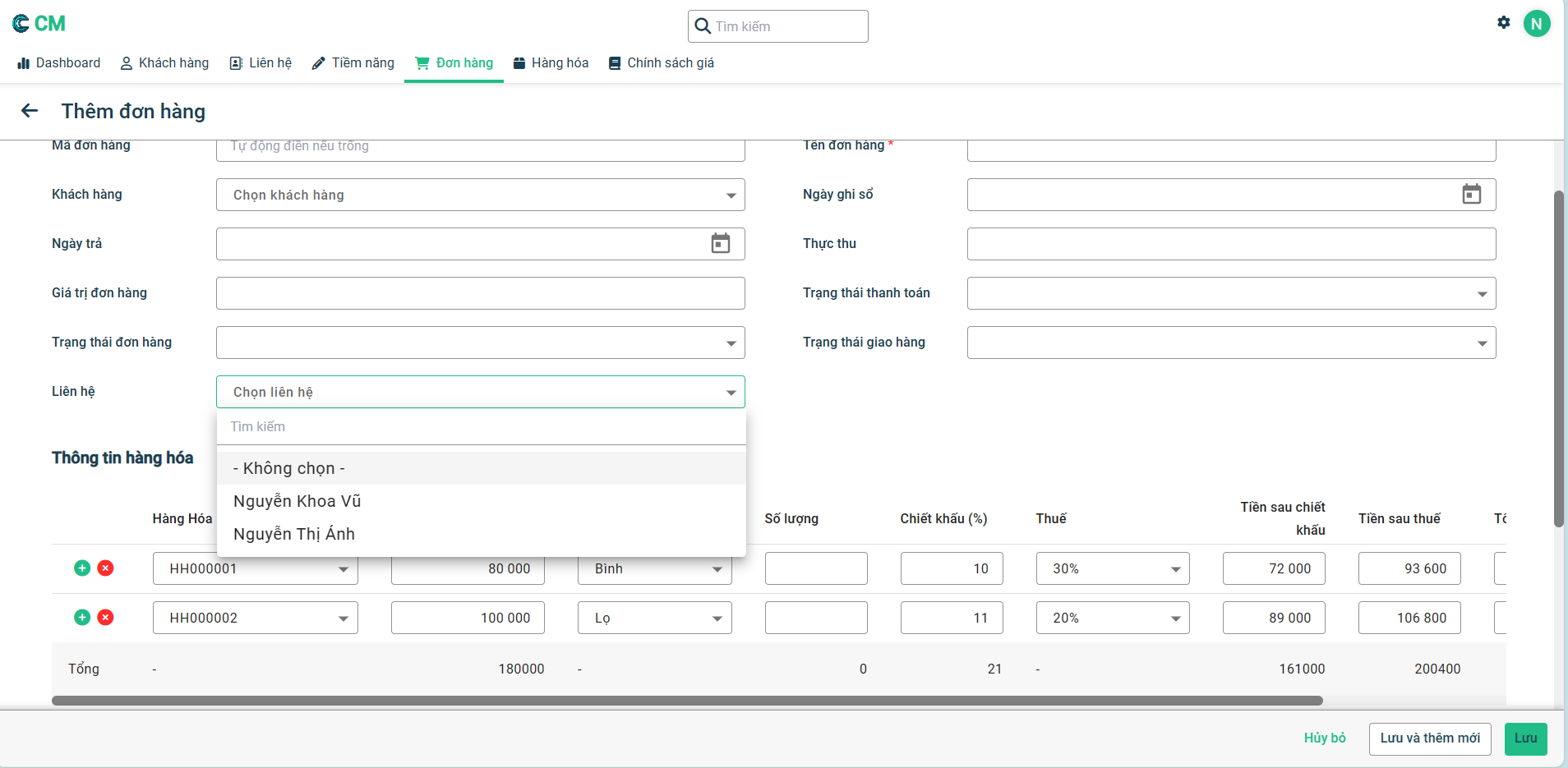
Màn hình quản lý chính sách giá: Là nơi quản lý những chính sách giá ưu đãi cho nhân viên trong công ty hay cho khách hàng riêng biệt trong từng thời điểm. Những hàng hóa khi bán cho những khách hàng đó sẽ được áp dụng những chính sách giá này.



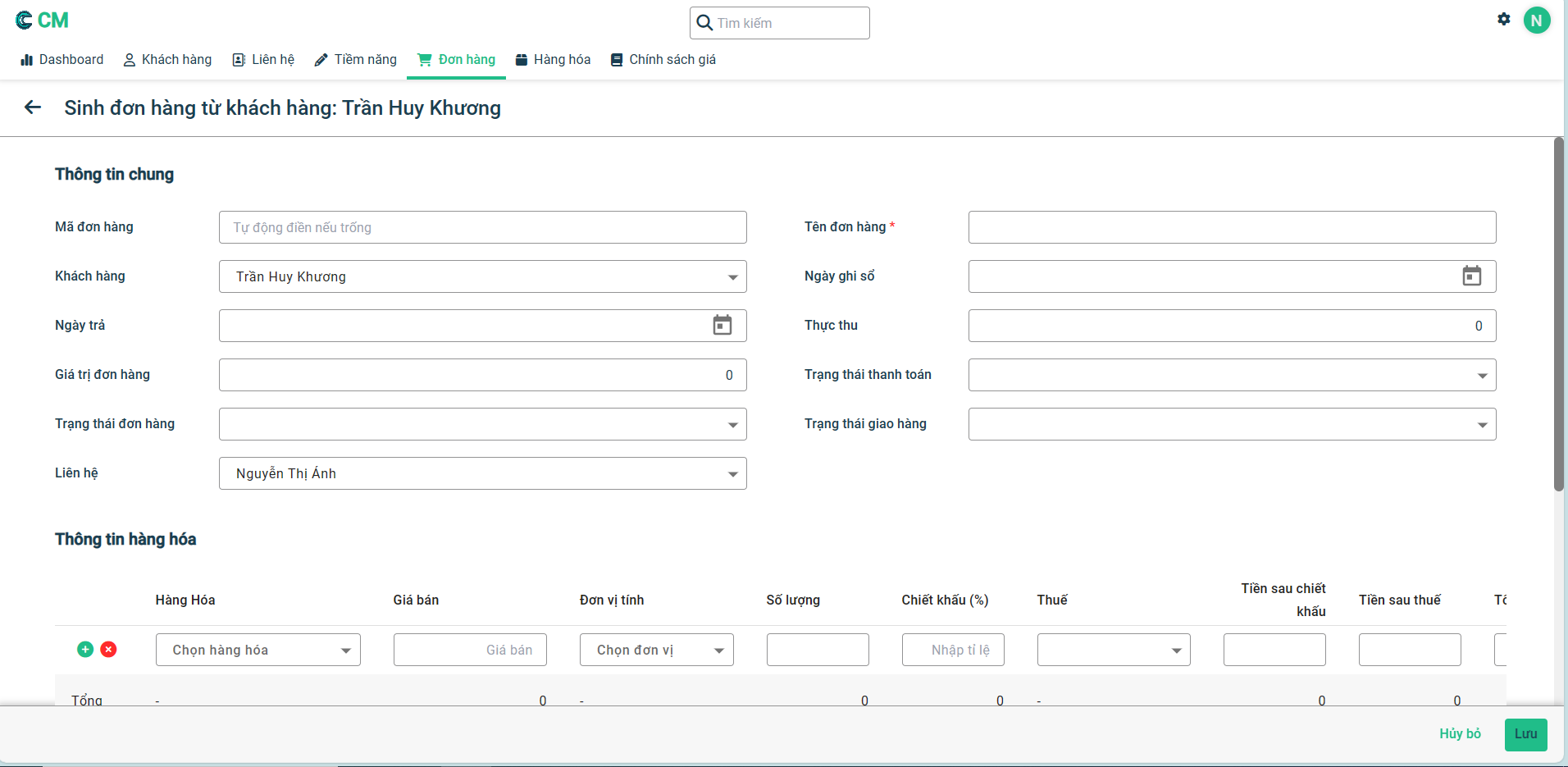
Màn hình quản lý hàng hóa: Dữ liệu hàng hóa được lưu trữ và quản lý tại đây. Kinh doanh có thể xem được hàng hóa có trong công ty và tồn kho của chúng. Từ đó tư vấn được cho khách hàng về những sản phẩm này.



Màn hình đơn hàng: là nơi quản lý các đơn hàng của khách hàng. Gồm những thông tin: Ngày trả tiền, khách hàng, liên hệ hóa đơn, liên hệ nhận hàng, trạng thái đơn hàng như nào, thực thu bao nhiêu, …



Màn hình thêm đơn hàng: Cho phép người dùng thêm mới đơn hàng gồm các thông tin của hàng hóa được tự động áp dụng chính sách giá (vẫn có thể sửa đổi). Các thông tin liên hệ, thực thu, trạng thái đơn hàng, …



Màn hình Sinh đơn hàng: Người dùng không cần phải mất công nhập lại thông tin của khách hàng mà chỉ tại màn hình chi tiết khách hàng chọn sinh đơn hàng. Các thông tin của khách hàng sẽ được tự động fill vào đơn hàng. Tiết kiệm thời gian nhập đơn hàng.

## **Kiểm thử**

### Kế hoạch kiểm thử

Những chức năng được kiểm thử:

* Đăng nhập: Người dùng đăng nhập vào hệ thống
* Đăng ký: Người dùng đăng ký tài khoản
* Quản lý Liên hệ: Người dùng tạo, thêm, cập nhật liên hệ
* Quản lý khách hàng: Người dùng tạo, thêm, cập nhật tiềm năng. Chuyển đổi tiềm năng sang liên hệ.
* Quản lý khách hàng: Người dùng tạo, thêm, cập nhật khách hàng.
* Quản lý hàng hóa: Người dùng tạo, thêm, cập nhật hàng hóa.
* Quản lý chính sách giá: Người dùng tạo chính sách giá ứng với các sản phẩm.
* Quản lý đơn hàng: Đơn hàng được tạo, thêm, cập nhật với các dữ liệu khách hàng, liên hệ, hàng hóa được tính theo chính sách giá tương ứng.

### Kiểm thử chức năng

Bảng 3.2: Bảng kiểm thử chức năng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Trường hợp** | **Đầu vào** | **Đầu ra** | **Kết quả** |
| 1 | Kiểm thử chức năng đăng nhập | Nhập tài khoản và mật khẩu hợp lệ. | Đăng nhập thành công điều hướng chọn công ty. | Đạt |
| Nhập tài khoản và mật khẩu không hợp lệ. | Đăng nhập không thành công và có thông báo ra màn hình. | Đạt |
| 2 | Kiểm thử chức năng đăng ký | Nhập thông tin đăng ký tài khoản hợp lệ. | Đăng ký thành công hiển thị thông báo và điều hướng về trang đăng nhập. | Đạt |
| Nhập thông tin đăng ký đã tồn tại. | Đăng ký không thành công và hiển thị thông báo. | Đạt |
| 3 | Kiểm thử chức năng quản lý liên hệ | Thêm liên hệ, cập nhật và xóa. | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| 4 | Kiểm thử chức năng quản lý tiềm năng | Thêm liên hệ, cập nhật và xóa | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| Chuyển đổi sang khách hàng | Hiển thị thông báo đã chuyển đổi và chuyển hướng sang trang khách hàng | Đạt |
| 5 | Kiểm thử chức năng quản lý khách hàng | Thêm liên hệ, cập nhật và xóa | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| Chọn liên hệ từ những liên hệ đã có. | Liên hệ hiển thị những liên hệ đã được nhập | Đạt |
| Chuyển đổi đơn hàng | Chuyển hướng sang trang thêm đơn hàng và tự động fill những thông tin của khách hàng vào dữ liệu đơn hàng | Đạt |
| 6 | Kiểm thử chức năng quản lý hàng hóa | Thêm hàng hóa, cập nhật và xóa | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| 7 | Kiểm thử chức năng chính sách giá | Thêm chính sách giá, cập nhật và xóa | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| Chọn hàng hóa đã thêm | Hiển thị thông tin hàng hóa và cho phép cập nhật lại | Đạt |
| 8 | Kiểm thử chức năng Quản lý đơn hàng | Thêm chính sách giá, cập nhật và xóa | Mỗi thao tác khi thành công hay thất bại đều hiển thị thông báo.  Dữ liệu lưu trữ đúng theo dữ liệu đã nhập | Đạt |
| Chọn khách hàng, liên hệ từ những dữ liệu đã thêm trong phân hệ khách hàng, liên hệ | Hiển thị tất cả khách hàng và liên hệ đã thêm để chọn | Đạt |
| Chọn hàng hóa từ các hàng hóa đã thêm | Tự động fill thông tin hàng hóa và các chính sách giá sao cho giá trị hàng hóa là nhỏ nhất. | Đạt |

### Kết quả kiểm thử

Tỉ lệ test case đạt: 100%

Tỉ lệ test case thất bại: 0%

Hệ thống chạy ổn định trên các trình duyệt web khác nhau như Firefox, Google Chrome, Microsoft Edge.

## **Kết luận chương 3**

Trong chương 3, em đã trình bày về kết quả đạt được sau khi hoàn thành dự án như: giao diện người dùng và các tính năng cùng với phần kiểm thử hệ thống.

# **KẾT LUẬN**

Bài báo cáo thực hiện nhằm mục đích giúp cho em được tiếp cận và hiểu rõ hơn về một hệ thống bán hàng được xây dựng bằng Angular, ASP.NET Core, MySQL. Trong quá trình thực hiện đề tài, em đã đạt được những kết quả như sau:

1. Về mặt kiến thức

* Em đã nắm được kiến thức cơ bản về Lập trình web bằng Angular, ASP.NET Core Web API, MySQL.
* Giúp em biết cách vận dụng các công cụ để thực hiện như Angular, ASP.NET Core Web API, MySQL, Draw.io, … và các kiến thức đã học được trên lớp vào bài tập lớn.
* Được ôn tập lại các kiến thức đã học ở kỳ trước như: phân tích và đặc tả yêu cầu phần mềm, cách thiết kế phần mềm, HTML, CSS…
* Đồ án giúp áp dụng và ôn lại các kiến thức về thiết kế giao diện người dùng và các tiện ích hỗ trợ người dùng.
* Giúp em hiểu hơn về hệ thống phần mềm quản lý khách hàng thông qua đặc tả yêu cầu phần mềm của bài toán.

2. Về mặt kỹ năng

* Kỹ năng làm việc độc lập: cho phép em hoàn thành công việc của họ một cách độc lập và không cần đến sự giám sát, hỗ trợ trực tiếp của người khác.
* Kỹ năng xác định mục tiêu: Mục tiêu của đồ án là đưa ra được website quản lý mối quan hệ khách hàng bằng Angular, ASP.NET Core Web API, MySQL. Xác định mục tiêu giúp em dễ dàng vạch ra những bược để thực hiện nó.
* Kỹ năng học hỏi: em cần phải học được cách học hỏi từ google, youtube. Tập trung lắng nghe và lắng nghe có chọn lọc giúp em tìm ra được những ý kiến phù hợp nhất để hoàn thiện bài tập lớn một cách tốt nhất.
* Kỹ năng thiết kế: em được đóng vai là một lập trình viên, phải phân tích và tìm hiểu những nội dung liên quan đến bài tập lớn. Em phải phân tích một website bán hàng bao gồm những gì, việc thiết kế giao diện cũng phải thân thiện và dễ dàng sử dụng cho người mua nó.
* Kỹ năng tìm kiếm thông tin: trong thời đại thông tin phát triển của internet, việc tìm kiếm cũng trở nên dễ dàng hơn.
* Kỹ năng phân tích: phân tích và đặc tả yêu cầu của phần mềm là một kỹ năng không thể thiếu trong một bài tập lớn về phần mềm. Nó giúp em đưa ra được các chức năng hiệu năng của phần mềm, giao diện của phần mềm để làm thỏa mãn được mong muốn của người mua.

3. Bài học kinh nghiệm

Kinh nghiệm khi làm bài tập lớn:

* Cần phải trau dồi, học hỏi thêm kiến thức về Lập trình web Angular, ASP.NET Core Web API, MySQL.
* Nghiên cứu sâu và nhiều hơn về các phần mềm bán hàng nên có để hoàn thiện bài tập lớn tốt hơn nữa.
* Cần mở rộng thêm các hướng giải quyết vấn đề cho bài toán và tham khảo những nguồn tài liệu tiếng anh để việc nghiên cứu bài tập lớn được tốt hơn.

Bài học rút ra:

Để làm tốt công việc của đồ án điều cần không chỉ là kiến thức lý thuyết mà cần phải có kinh nghiệm thực tế. Kinh nghiệm thực tế ở đây không chỉ đơn giản là kinh nghiệm về ngành em theo học. Mà cần phải có thêm kinh nghiệm về các kỹ năng mềm như: kỹ năng giao tiếp, kỹ năng làm việc độc lập… Đối chiếu giữa lý thuyết được học khi ở trên ghế nhà trường và thực tế bên ngoài, giúp em củng cố kiến thức của bản thân, có cái nhìn sâu rộng hơn về chuyên ngành em đang theo học.

Khi làm đồ án em nhận thấy việc làm sao vận dụng được các kiến thức đã học vào đồ án để đem lại giá trị thực tế mới là quan trọng. Chính vì vậy em cần phải nỗ lực học tập tốt hơn nữa những kiến thức khi vẫn còn ở trên ghế nhà trường. Không ngừng nâng cao kiến thức về chuyên ngành công nghệ thông tin mà còn phải bổ sung thêm các kinh nghiệm thực tế để thành công hơn trong tương lai.

Tinh thần chủ động và tự tin chính là bài học kinh nghiệm mà em cần học hỏi trong khi làm bài tập lớn. Luôn phải chủ động tìm hiểu các thông tin và giải quyết bài toán độc lập.

Sau khi thực hiện xong đồ án em đã rút ra cho mình nhiều kinh nghiệm hơn về việc phân tích và đặc tả yêu cầu phần mềm, từ đó tạo cơ sở dữ liệu, hiển thị những thông tin cần thiết lên màn hình.

Qua đồ án cũng giúp em bổ sung được cho bản thân mình những kinh nghiệm quý báu về việc tìm kiếm thông tin, tài liệu trên mạng để hoàn thành đề tài này.

4. Thuận lợi và khó khăn

\* Thuận lợi:

* Đã cố gắng trình bày bài tập lớn một cách khoa học và có hệ thống những kiến thức hiểu biết của bản thân, có tham khảo các tài liệu về các vấn đề có liên quan đến nội dung tìm hiểu, nghiên cứu.
* Cố gắng bám sát yêu cầu của bài tập lớn và làm theo sự hướng dẫn của thầy Vũ Duy Giang nhưng đồ án được làm trong thời gian ngắn nên không thể tránh khỏi sự thiếu sót, rất mong sự đóng góp ý kiến của thầy để bài tập lớn được hoàn thiện tốt hơn.

\* Khó khăn:

Mặc dù đã cố gắng trong quá trình thực hiện đề tài nhưng do công nghệ còn mới đối với em và ứng dụng vẫn còn một vài chức năng mà bản thân em chưa tìm hiểu hết trong thời gian thực hiện đề tài. Và với kinh nghiệm còn hạn chế nên đề tài của em không thể tránh khỏi những thiếu sót, em rất mong các thầy cô có thể góp ý cho em.

5. Hướng phát triển đề tài

* Để chương trình được phát triển hơn trong tương lai em cần tiếp tục cố gắng trau dồi kiến thức của bản thân và hoàn thiện bài tập lớn tốt hơn.
* Xây dựng website quản lý khách hàng là một bài toán tiềm năng ứng dụng thực tiễn cao, đem lại lợi ích cho xã hội, những để phát triển thành một hệ thống hoàn chỉnh và có thể đưa ứng dụng vào thực tế một cách rộng rãi thì cần cải tiến nhiều nghiệp vụ và chức năng hỗ trợ người dùng hơn nữa.
* Trong thời gian tới em sẽ tiếp tục tìm hiểu sâu hơn về website, ứng dụng trên điện thoại theo hướng giải pháp cho lĩnh vực kinh doanh, nâng cấp cả về bảo mật để khách hàng tin cậy sử dụng sản phẩm.

Và cuối cùng em xin chân thành cảm ơn thầy giáo Ths. Vũ Duy Giang đã giúp em hoàn thành được đề tài

# **TÀI LIỆU THAM KHẢO**

[1] <https://www.w3schools.com>

[2] https://angular.io/

[4] <https://learn.microsoft.com/vi-vn/aspnet/core/tutorials/web-api-help-pages-using-swagger?view=aspnetcore-6.0>

[5] <https://www.learndapper.com>

[6] [What Is an API (Application Programming Interface)? | IBM](https://www.ibm.com/topics/api)