

## **ĐỒ ÁN CHUYÊN NGÀNH**

# **WEBSITE QUẢN LÝ CHUỖI CỬA HÀNG BÁCH HÓA**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành: **CÔNG NGHỆ PHẦN MỀM**

Giảng viên hướng dẫn : ThS. Bùi Mạnh Toàn

Sinh viên thực hiện :

Nguyễn Hữu Quý

MSSV: 2180607948

Lớp: 21DTHD4

TP. Hồ Chí Minh, 2024

## **ĐỒ ÁN CHUYÊN NGÀNH**

# **WEBSITE QUẢN LÝ CHUỖI CỬA HÀNG BÁCH HÓA**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành: **CÔNG NGHỆ PHẦN MỀM**

Giảng viên hướng dẫn : Ths. Bùi Mạnh Toàn

Sinh viên thực hiện :

Nguyễn Hữu Quý

MSSV: 2180607948

Lớp: 21DTHD4

TP. Hồ Chí Minh, 2024

## LỜI NÓI ĐẦU

Sau đại dịch COVID-19, thói quen tiêu dùng của con người đã thay đổi đáng kể. Người tiêu dùng ngày càng thắt chặt chi tiêu, chỉ tập trung mua sắm những sản phẩm đáp ứng trực tiếp nhu cầu thiết yếu và có giá trị thiết thực. Đối mặt với bối cảnh này, các chuỗi cửa hàng bách hóa cần tối ưu hóa hoạt động quản lý, đảm bảo cung ứng hàng hóa hiệu quả, kiểm soát chi phí, và đồng thời duy trì chất lượng dịch vụ để giữ chân khách hàng.

Đồ án "Website quản lý chuỗi cửa hàng bách hóa" được thực hiện nhằm xây dựng một hệ thống quản lý hiện đại, đáp ứng yêu cầu quản lý chặt chẽ về hàng hóa, nhân sự, và kinh doanh trong tình hình mới. Hệ thống này sẽ hỗ trợ các cửa hàng trong chuỗi hoạt động đồng bộ, từ khâu nhập hàng, bán hàng đến phân tích dữ liệu kinh doanh. Qua đó, giúp các nhà quản lý đưa ra những quyết định đúng đắn, kịp thời để tối ưu hóa nguồn lực, tăng cường hiệu quả vận hành và đáp ứng nhu cầu ngày càng khắt khe của khách hàng. Báo cáo sẽ trình bày quá trình phân tích và phát triển hệ thống, bao gồm việc khảo sát nhu cầu thực tế trong lĩnh vực bán lẻ, thiết kế giao diện, cơ sở dữ liệu, chức năng hệ thống, cũng như các bước kiểm thử để đảm bảo tính ổn định và hiệu quả. Ngoài ra, nhóm thực hiện sẽ đề xuất các giải pháp cải tiến nhằm nâng cao tính ứng dụng của sản phẩm trong thực tiễn.

Thông qua đồ án này, tôi hy vọng sẽ không chỉ hoàn thành các yêu cầu học thuật mà còn mang lại một công cụ hữu ích, góp phần giúp các doanh nghiệp trong lĩnh vực bán lẻ vượt qua thách thức sau đại dịch, đồng thời phát triển bền vững trong tương lai.

## **LỜI CẢM ƠN**

Để hoàn thành tốt đồ án này, em xin chân thành cảm ơn thầy Bùi Mạnh Toàn đã hướng dẫn em trong suốt quá trình học và quá trình làm báo cáo đồ án này. Bên cạnh đó để hoàn thành tốt đồ án này em cũng đã nhận được nhiều sự giúp đỡ của bạn bè và quý thầy cô. Một lần nữa xin chân thành cảm ơn.

Tuy nhiên do thời gian hạn hẹp, mặc dù đã nỗ lực hết sức mình nhưng đồ án của em khó tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự thông cảm và góp ý của thầy để đề tài của em được hoàn thiện hơn.

Em xin chân thành cảm ơn!

**Sinh viên thực hiện**

Nguyễn Hữu Quý

## **LỜI CAM ĐOAN**

Em xin cam đoan đề tài “Website quản lý chuỗi cửa hàng bách hóa” được tiến hành một cách minh bạch, công khai. Toàn bộ nội dung và kết quả dựa trên sự cố gắng cũng sự nỗ lực của cả nhóm cùng với sự giúp đỡ không nhỏ từ thầy và bạn bè.

Em xin cam đoan kết quả của phần mềm được đưa ra trong bài báo cáo là trung thực và không sao chép hay sử dụng kết quả của bất kỳ đề tài nghiên cứu nào tương tự.

Em sẵn sàng chịu toàn bộ trách nhiệm nếu phát hiện rằng có bất kỳ sự sao chép kết quả nghiên cứu nào trong bài đồ án này.

[illegible]

# MỤC LỤC

LỜI NÓI ĐẦU .....	i
LỜI CAM ĐOAN .....	iii
NHẬN XÉT CỦA GIẢNG VIÊN .....	iv
CHƯƠNG 1: TỔNG QUAN .....	1
1.1. Giới thiệu đề tài .....	1
1.2. Nhiệm vụ đồ án .....	1
1.3. Đối tượng và phạm vi của phẩm mềm.....	1
1.3.1. Đối tượng.....	1
1.3.2. Phạm vi .....	1
1.4. Cấu trúc đồ án.....	2
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT .....	3
2.1. Các khái niệm .....	3
2.1.1. Giới thiệu về kiến trúc Microservices.....	3
2.1.2. API Gateway trong hệ thống Microservices [3] .....	5
2.1.3. .NET Core MVC và Web API [5] .....	6
2.1.4. OTP.NET và xác thực hai yếu tố (2FA) [6] .....	8
2.1.5. SMTP và gửi Email trong .NET Core [7] .....	10
2.1.6. Hệ thống Xác thực và Phân quyền [8] .....	10
2.1.7. Tổng quan về giao tiếp giữa các Microservices .....	12
2.1.8. Bảo mật.....	15
2.1.9. Ứng dụng Google Maps API trong hệ thống [10].....	16
2.1.10. AJAX (Asynchronous JavaScript and XML) [11] .....	18
2.1.11. FetchAPI [12].....	19
2.1.12 .Rate Limiting[13] .....	19
2.2. Tổng quan về các công cụ .....	20
2.2.1. Ngôn ngữ lập trình C# [14].....	20
2.2.2. Hệ quản trị cơ sở dữ liệu SQL Server [15] .....	21
2.2.3. Môi trường lập trình Visual Studio 2022 [16] .....	22
2.2.4. Phần mềm StarUML [17] .....	22
2.2.5. Công cụ CASE Studio 2 .....	23
2.2.6. Phần mềm GitHub [18] .....	24
2.2.7. Bootstrap [19] .....	25
2.3. Xác định yêu cầu .....	26

2.3.1. Yêu cầu chung.....	26
2.3.2. Giao diện .....	26
2.3.3. Các tác vụ cơ bản .....	27
2.3.4 Các công nghệ được sử dụng .....	28
2.4. Yêu cầu hệ thống .....	29
2.5. Yêu cầu hệ thống .....	30
2.6. Sơ đồ phân rã chức năng Use-case .....	30
2.6.1. Use-case tổng quát .....	30
2.6.2. Use-case đăng ký.....	31
2.6.4. Use-case quản lý sản phẩm .....	32
2.6.5. Use-case quản lý nhập kho .....	32
2.6.6. Use-case quản lý xuất kho .....	33
2.6.7. Use-case quản lý chi nhánh.....	34
2.6.8. Use-case quản lý khách hàng .....	34
2.6.9. Use-case quản lý khuyến mãi .....	35
2.7. Activity Diagram .....	35
2.7.1. Chức năng đăng ký.....	35
2.7.2. Chức năng đăng nhập.....	36
2.7.3. Chức năng thêm sản phẩm .....	37
2.7.4. Chức năng nhập lô hàng .....	38
2.7.5. Chức năng phân phối hàng cho các chi nhánh .....	39
2.7.6. Chức năng quản lý khuyến mãi .....	40
2.7.7. Chức năng giao đơn hàng .....	41
2.7.8. Chức năng mua hàng.....	42
2.7.9. Chức năng xác nhận đơn hàng.....	43
2.8. Sơ đồ Database .....	44
2.8.1. Chi nhánh Database .....	44
2.8.2.Hóa đơn Database .....	45
2.8.3. Sản phẩm Database .....	46
2.8.4. Giảm giá Database.....	46
2.8.5. Người dùng Database.....	47
2.9. Mô hình Sequence Diagram .....	48
2.9.1. Sequence Diagram chức năng đăng ký .....	48
2.9.2. Sequence Diagram chức năng đăng nhập .....	48



2.9.3. Sequence Diagram chức năng thêm sản phẩm.....	49
2.9.4. Sequence Diagram chức năng thêm chi nhánh .....	49
2.9.5. Sequence Diagram chức năng mua hàng .....	50
2.10. Danh sách thực thể .....	50
CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM .....	59
3.1. Các thành phần chức năng của hệ thống .....	59
3.2. Thiết kế giao diện hệ thống .....	60
3.2.1. Giao diện đăng nhập .....	60
3.2.2. Giao diện đăng ký.....	61
3.2.3. Giao diện quản lý sản phẩm .....	62
3.2.4. Giao diện quản lý nhập kho.....	63
3.2.5. Giao diện quản lý xuất kho.....	63
3.2.6. Giao diện quản lý chi nhánh.....	64
3.2.7. Giao diện quản lý khách hàng.....	66
3.2.8. Giao diện quản lý khuyến mãi .....	67
3.2.9. Giao diện giao hàng cho khách.....	68
3.2.10. Giao diện chi tiết đơn hàng giao cho khách.....	69
3.2.11. Giao diện chọn hàng hóa .....	69
3.2.12. Giao diện chi tiết hàng hóa .....	70
3.2.13. Giao diện giỏ hàng .....	70
3.2.14. Giao diện thanh toán .....	71
3.2.15. Giao diện Google Maps.....	72
3.2.16. Giao diện Google Maps.....	72
3.2.17. Giao diện cửa hàng (POS) .....	73
3.2.18. Giao diện đơn hàng online (POS) .....	74
3.2.19. Giao diện chi tiết đơn hàng online (POS) .....	74
3.2.20. Giao diện tất cả đơn hàng của chi nhánh (POS) .....	75
CHƯƠNG 4: KẾT LUẬN.....	77
4.1. Các vấn đề đã giải quyết.....	77
4.2. Các vấn đề chưa giải quyết.....	77
TÀI LIỆU THAM KHẢO .....	78

## DANH MỤC HÌNH ẢNH

Hình 2.1. Kiến trúc Microservices áp dụng trong project .....	4
Hình 2.2. Mô hình tổng quát .....	30
Hình 2.3. Mô hình đăng ký .....	31
Hình 2.4. Mô hình đăng nhập .....	31
Hình 2.5. Mô hình quản lý sản phẩm .....	32
Hình 2.6. Mô hình quản lý nhập kho (1) .....	32
Hình 2.7. Mô hình quản lý nhập kho (2) .....	33
Hình 2.8. Mô hình quản lý xuất kho .....	33
Hình 2.9. Mô hình quản lý chi nhánh .....	34
Hình 2.10. Mô hình quản lý khách hàng .....	34
Hình 2.11. Quản lý khuyến mãi .....	35
Hình 2.12. Hệ thống đăng ký .....	35
Hình 2.13. Hệ thống đăng nhập .....	36
Hình 2.14. Hệ thống thêm sản phẩm .....	37
Hình 2.15. Hệ thống nhập lô hàng .....	38
Hình 2.16. Hệ thống phân phối hàng cho các chi nhánh .....	39
Hình 2.17. Hệ thống thêm mã khuyến mãi .....	40
Hình 2.18. Hệ thống giao đơn hàng .....	41
Hình 2.19. Hệ thống mua hàng .....	42
Hình 2.20. Hệ thống xác nhận mua hàng .....	43
Hình 2.21. Cơ sở dữ liệu “Chi nhánh” .....	44
Hình 2.22. Cơ sở dữ liệu “Hóa đơn” .....	45
Hình 2.23. Cơ sở dữ liệu “Sản phẩm” .....	46
Hình 2.24. Cơ sở dữ liệu “Giảm giá” .....	47
Hình 2.25. Cơ sở dữ liệu “Người dùng” .....	47
Hình 2.26. Mô hình Sequence Diagram chức năng đăng ký .....	48
Hình 2.27. Mô hình Sequence Diagram chức năng đăng nhập .....	49
Hình 2.28. Mô hình Sequence Diagram chức năng thêm sản phẩm .....	49
Hình 2.29. Mô hình Sequence Diagram chức năng thêm chi nhánh .....	49
Hình 2.30. Mô hình Sequence Diagram chức năng mua hàng .....	50
Bảng 2.31. Thực thể Branches .....	50
Hình 2.32. Thực thể Product_Branches .....	51

Hình 2.33. Thực thể ImportProductHistory .....	51
Hình 2.34. Thực thể Invoices .....	52
Hình 2.35. Thực thể Invoice_Products .....	53
Hình 2.36. Thực thể PaymentMethods .....	53
Hình 2.37. Thực thể Status .....	54
Hình 2.38. Thực thể Products .....	54
Hình 2.39. Thực thể Batches .....	55
Hình 2.40. Thực thể Images .....	55
Hình 2.41. Thực thể CategoryProducts .....	55
Hình 2.42. Thực thể Carts .....	56
Hình 2.43. Thực thể Promotions .....	57
Hình 2.44. Thực thể PromotionType .....	57
Hình 2.45. Thực thể Users .....	57
Hình 2.46. Thực thể UserOtherInfo .....	58
Hình 2.47. Thực thể Role .....	58

# CHƯƠNG 1: TỔNG QUAN

## 1.1. Giới thiệu đề tài

Trong thời đại công nghệ số hiện nay, việc áp dụng công nghệ vào quản lý và bán hàng đã trở thành yếu tố thiết yếu giúp các doanh nghiệp, đặc biệt là các chuỗi cửa hàng bách hóa, duy trì và phát triển trong môi trường cạnh tranh khốc liệt. Các hệ thống quản lý bán hàng trực tuyến hiện có trên thị trường như Sapo POS, KiotViet, Haravan hay Odoo ERP đều mang lại nhiều lợi ích trong việc quản lý sản phẩm, đơn hàng, kho hàng và hỗ trợ bán hàng đa kênh. Tuy nhiên, chúng vẫn tồn tại một số hạn chế như chi phí triển khai cao, khó sử dụng đối với các doanh nghiệp nhỏ, hoặc thiếu các tính năng nâng cao như phân tích hành vi khách hàng và quản lý khuyến mãi. Vì vậy, đề tài này tập trung phát triển một hệ thống quản lý bán hàng trực tuyến với chi phí hợp lý, tích hợp các tính năng tối ưu hóa quy trình quản lý sản phẩm, đơn hàng và khuyến mãi, đồng thời hỗ trợ bán hàng đa kênh phù hợp với nhu cầu của các chuỗi cửa hàng bách hóa vừa và nhỏ. Hệ thống sẽ góp phần cải thiện hiệu quả kinh doanh và nâng cao trải nghiệm người dùng.

## 1.2. Nhiệm vụ đề án

Phát triển một hệ thống quản lý bán hàng hoàn chỉnh, đáp ứng được các nhu cầu thực tế của một chuỗi cửa hàng, bao gồm quản lý sản phẩm, khách hàng, nhân viên, kho hàng và giao dịch mua bán. Hệ thống sẽ đảm bảo tính tiện dụng, hiệu quả và bảo mật trong hoạt động quản lý và giao dịch.

## 1.3. Đối tượng và phạm vi của phạm vi

### 1.3.1. Đối tượng

Phần mềm Website quản lý chuỗi cửa hàng bách hóa được áp dụng hỗ trợ các doanh nghiệp chuỗi cửa hàng bách hóa có thể quản lý chuỗi cửa hàng dễ dàng hơn.

### 1.3.2. Phạm vi

Phạm vi dự án tập trung vào việc xây dựng hệ thống quản lý bán hàng toàn diện nhằm đáp ứng nhu cầu quản lý và hoạt động của một cửa hàng hoặc

chuỗi cửa hàng bán lẻ. Hệ thống sẽ bao gồm các chức năng chính như quản lý sản phẩm, kho hàng, bán hàng, nhân sự, và báo cáo doanh thu.

#### **1.4. Cấu trúc đề án**

CHƯƠNG 1: TỔNG QUAN

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

CHƯƠNG 4: KẾT LUẬN

## CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

### 2.1. Các khái niệm

#### 2.1.1. Giới thiệu về kiến trúc Microservices

##### 2.1.1.1 Định nghĩa Microservices [1]

Microservices là một kỹ thuật phát triển phần mềm, một biến thể thuộc kiến trúc hướng dịch vụ (SOA), cấu trúc một ứng dụng như một tập hợp các dịch vụ được ghép lỏng lẻo. Microservices là một thiết kế kiến trúc dùng để xây dựng một ứng dụng phân tán thông qua các container. Mỗi chức năng của ứng dụng hoạt động như một dịch vụ độc lập, do đó kiến trúc này được đặt tên là micro-service. Thông qua kiến trúc này, mỗi dịch vụ có thể dễ dàng mở rộng quy mô và cập nhật mà không làm gián đoạn hay ảnh hưởng bất kỳ dịch vụ nào khác trong ứng dụng.

Martin Fowler (2014), một trong những người tiên phong trong định nghĩa về Microservices, đã mô tả chúng như một phong cách kiến trúc giúp cải thiện tính linh hoạt trong phát triển phần mềm. Theo ông, Microservices cho phép chia nhỏ hệ thống thành các module có thể được phát triển, thử nghiệm, triển khai và mở rộng độc lập mà không ảnh hưởng đến các module khác. [2]

Điểm nổi bật của Microservices là mỗi dịch vụ thường tự quản lý dữ liệu và tài nguyên của riêng nó, đồng thời giao tiếp với các dịch vụ khác thông qua giao diện API (Application Programming Interface) hoặc các cơ chế giao tiếp khác, chẳng hạn như message queue. Điều này tạo ra một môi trường

##### 2.1.1.2. Ưu và nhược điểm của kiến trúc Microservices

###### **\* Ưu điểm của Microservices**

- Cho phép dễ dàng continuous delivery và deployment các ứng dụng lớn, phức tạp nhờ cải thiện khả năng bảo trì - mỗi service tương đối nhỏ do đó dễ hiểu và thay đổi hơn, khả năng testing dễ dàng hơn - các services nhỏ hơn và nhanh hơn để test hoặc khả năng triển khai tốt hơn - các services có thể được triển khai độc lập.

- Cho phép các services được phát triển bởi những team khác nhau. Mỗi team có thể phát triển, thử nghiệm, triển khai và mở rộng quy mô dịch vụ của mình một cách độc lập với tất cả các team khác.

- Giảm thiểu các rủi ro như lỗi trong một service thì chỉ có service đó bị ảnh hưởng. Các services khác sẽ tiếp tục xử lý các yêu cầu.

- Khi triển khai các services bạn có thể lựa chọn nhiều công nghệ mới. Tương tự khi có thay đổi lớn đối với các services hiện có bạn có thể dễ dàng thay đổi công nghệ

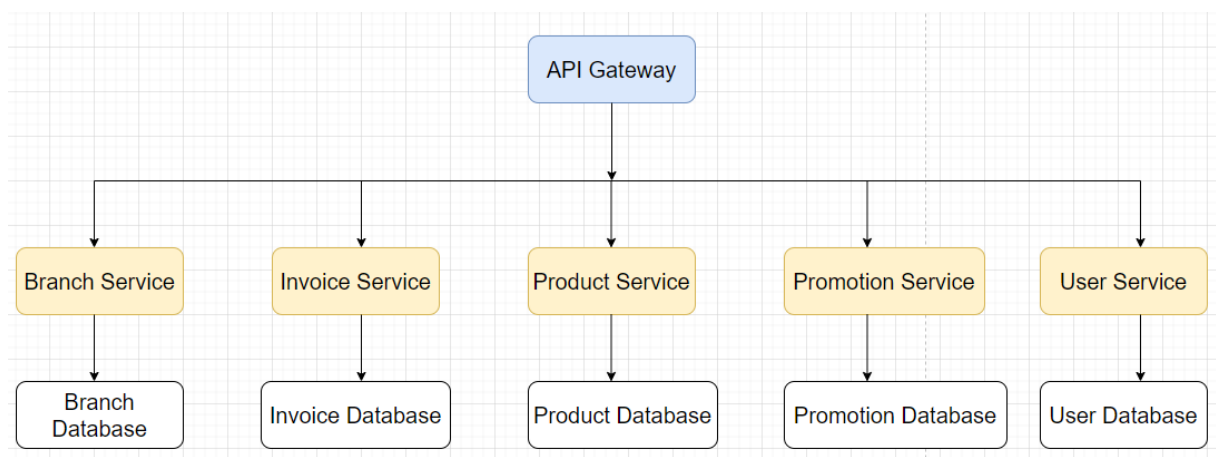
#### **\* Nhược điểm của Microservices**

- Khi triển khai hoặc quản lý microservices nếu làm thủ công theo cách đã làm với ứng dụng một khối thì sẽ phức tạp hơn rất nhiều.

- Phải xử lý sự cố khi kết nối chậm, lỗi khi thông điệp không gửi được hoặc thông điệp gửi đến nhiều đích đến vào các thời điểm khác nhau.

- Các nhà phát triển phải đối phó với sự phức tạp của việc tạo ra một hệ thống phân tán như: khó khăn trong việc đảm bảo toàn vẹn CSDL nếu triển khai theo kiến trúc cơ sở dữ liệu phân vùng, cần implement việc communication giữa các inter-services.

#### **2.1.1.3. Kiến trúc Microservices áp dụng trong project**



Hình 2.1. Kiến trúc Microservices áp dụng trong project

- Chi phí vận hành cao hơn: Vì mỗi dịch vụ là một ứng dụng riêng biệt, số lượng tài nguyên (máy chủ, cơ sở dữ liệu, công cụ giám sát) cần để vận hành hệ thống sẽ cao hơn so với kiến trúc nguyên khối.

## 2.1.2. API Gateway trong hệ thống Microservices [3]

### 2.1.2.1. Tổng quan về API Gateway

API Gateway là một thành phần quan trọng trong kiến trúc Microservices, đóng vai trò làm điểm trung gian giữa client (người dùng hoặc ứng dụng bên ngoài) và các dịch vụ bên trong hệ thống. Thay vì để client trực tiếp gọi các dịch vụ khác nhau thông qua các endpoint riêng lẻ, API Gateway cung cấp một giao diện duy nhất để tập trung hóa và quản lý tất cả các yêu cầu từ client.

Vai trò chính của API Gateway:

- Đơn giản hóa giao tiếp: Client chỉ cần biết về API Gateway mà không cần quan tâm đến cách các dịch vụ bên trong được tổ chức hoặc hoạt động. API Gateway xử lý việc định tuyến, giao tiếp và quản lý yêu cầu đến các dịch vụ tương ứng.

- Tăng tính bảo mật: Thông qua API Gateway, tất cả các yêu cầu được kiểm tra và xác thực trước khi đến các dịch vụ backend. Điều này giúp bảo vệ hệ thống khỏi các yêu cầu không hợp lệ hoặc độc hại.

- Cải thiện hiệu suất: API Gateway cung cấp các tính năng như caching và nén dữ liệu, giúp giảm tải cho các dịch vụ backend và tối ưu hóa thời gian phản hồi.

- Tích hợp dễ dàng: API Gateway hỗ trợ tích hợp nhiều dịch vụ Microservices với các cơ chế như định tuyến thông minh, cân bằng tải và chuyển đổi định dạng dữ liệu (JSON, XML).

### 2.1.2.2. Tính năng của API Gateway Ocelot [4]

Ocelot là một thư viện mã nguồn mở được thiết kế dành riêng cho việc triển khai API Gateway trong hệ thống Microservices dựa trên nền tảng .NET. Nó cung cấp một tập hợp các tính năng phong phú để đáp ứng các yêu cầu thường gặp khi triển khai API Gateway. Tính năng chính của Ocelot:



- Biến đổi dữ liệu (Data Transformation): Ocelot cho phép biến đổi định dạng dữ liệu giữa client và dịch vụ backend, giúp cải thiện tính tương thích và giảm yêu cầu sửa đổi ở backend, giúp đơn giản hóa việc tích hợp giữa các hệ thống với định dạng dữ liệu khác nhau và giảm thời gian phát triển và bảo trì backend.

### 2.1.3. .NET Core MVC và Web API [5]

#### 2.1.3.1. Tổng quan về .NET Core MVC

.NET Core MVC là một framework mạnh mẽ, cho phép phát triển ứng dụng web theo mô hình Model-View-Controller (MVC), giúp tách biệt các thành phần chính trong một ứng dụng web:

- Model:

+ Đại diện cho dữ liệu và logic nghiệp vụ.

+ Có thể tương tác trực tiếp với cơ sở dữ liệu thông qua ORM như Entity Framework Core.

+ Thực hiện các quy trình xử lý dữ liệu, xác minh đầu vào và các logic khác.

- View:

+ Hiển thị dữ liệu và giao diện người dùng.

+ Sử dụng Razor để tạo ra các trang HTML động, giúp render dữ liệu từ Model.

+ Kết hợp các thư viện giao diện như Bootstrap hoặc Tailwind CSS để tạo trải nghiệm người dùng hiện đại.

- Controller:

+ Là cầu nối giữa Model và View.

+ Xử lý các yêu cầu HTTP từ client, tương tác với Model để lấy dữ liệu, và trả về View hoặc dữ liệu JSON.

\* Ưu điểm khi sử dụng .NET Core MVC trong dự án:

- Kiến trúc rõ ràng: Tách biệt các trách nhiệm, giúp mã nguồn dễ bảo trì và mở rộng.

- Kiến trúc rõ ràng: Tách biệt các trách nhiệm, giúp mã nguồn dễ bảo trì và mở rộng.

- Hỗ trợ tốt cho frontend: Kết hợp Razor với các framework frontend hiện đại như React hoặc Angular.

### 2.1.3.2. Tổng quan về .NET Core Web API

.NET Core Web API là một công cụ được thiết kế để xây dựng các dịch vụ RESTful API với hiệu suất cao và dễ dàng tích hợp. Web API thường sử dụng các phương thức HTTP như GET, POST, PUT, DELETE để cung cấp khả năng giao tiếp dữ liệu giữa client và server.

- Các tính năng chính:

- + RESTful: Dựa trên kiến trúc REST, giúp xây dựng các API dễ sử dụng và thống nhất.

- + Hiệu suất cao: Được tối ưu hóa cho các ứng dụng web xử lý nhiều yêu cầu với tài nguyên thấp.

- + Bảo mật: Tích hợp dễ dàng các cơ chế bảo mật như JWT, OAuth 2.0, API Key.

- + Tính độc lập: Các API được thiết kế với tiêu chuẩn mở, có thể được sử dụng bởi mọi loại client (web, mobile, desktop).

- Ưu điểm của Web API trong hệ thống Microservices:

- + Tách biệt và độc lập: Các API có thể hoạt động độc lập, dễ dàng triển khai và mở rộng riêng lẻ.

- + Tương thích đa nền tảng: API RESTful không phụ thuộc vào ngôn ngữ hoặc nền tảng của client.

- + Khả năng tái sử dụng: Một API có thể phục vụ cho nhiều ứng dụng (web app, mobile app).

- Công nghệ hỗ trợ:

- + Swagger: Giúp tự động tạo tài liệu API, hỗ trợ lập trình viên hiểu và sử dụng API hiệu quả.

- + Entity Framework Core: Cung cấp ORM để dễ dàng thao tác với cơ sở dữ liệu.
- + Middleware: Xử lý các yêu cầu trước khi chúng đến API Controller, thường được dùng để kiểm tra bảo mật hoặc log.

### 2.1.3.3. Vai trò của Web API trong hệ thống

Trong một hệ thống Microservices, Web API đóng vai trò trung tâm, đặc biệt trong việc kết nối client và backend. Một số vai trò nổi bật bao gồm:

- Xử lý và cung cấp dữ liệu từ cơ sở dữ liệu: Web API chịu trách nhiệm xử lý yêu cầu dữ liệu từ client, tương tác với cơ sở dữ liệu để lấy dữ liệu, và trả về dưới dạng JSON hoặc XML.
- Tích hợp với API Gateway (Ocelot): Web API cung cấp các endpoint mà API Gateway sẽ sử dụng để định tuyến yêu cầu từ client đến dịch vụ phù hợp.
- Bảo mật và phân quyền: Web API đảm bảo rằng chỉ những người dùng được xác thực và có quyền phù hợp mới có thể truy cập dữ liệu hoặc thực hiện hành động.
- Quản lý trạng thái người dùng: Web API quản lý trạng thái người dùng như phiên làm việc, thông tin giỏ hàng, hoặc lịch sử giao dịch.
- Cung cấp dịch vụ cho nhiều loại client: Một Web API có thể phục vụ đồng thời nhiều loại client khác nhau, từ các ứng dụng web sử dụng React, Angular, đến các ứng dụng mobile như iOS, Android.

### *2.1.4. OTP.NET và xác thực hai yếu tố (2FA) [6]*

#### 2.1.4.1. Tổng quan về OTP.NET

OTP.NET là một thư viện mã nguồn mở được thiết kế để hỗ trợ tạo và xác thực mã OTP (One-Time Password) dựa trên các chuẩn phổ biến như:

- TOTP (Time-Based One-Time Password):
  - + Sử dụng thời gian hiện tại làm cơ sở để tạo mã OTP.
  - + Các mã này thường có thời hạn ngắn, ví dụ: 30 giây.
  - + Đảm bảo tính duy nhất và đồng bộ giữa server và client.

- HOTP (HMAC-Based One-Time Password):

+ Dựa trên bộ đếm (counter) thay vì thời gian.

+ Ít phổ biến hơn trong xác thực 2FA do không đồng bộ tự động như TOTP.

Tính năng chính của OTP.NET:

- Sinh mã OTP:

+ Sử dụng secret key và thuật toán mã hóa SHA-1 (hoặc SHA-256, SHA-512).

+ Mã OTP có thể tùy chỉnh độ dài (thường từ 6 đến 8 ký tự số).

- Xác thực mã OTP:

+ Kiểm tra mã OTP do người dùng nhập có hợp lệ hay không.

+ Có thể cho phép độ lệch thời gian để xử lý các trường hợp không đồng bộ giữa server và client.

- Tích hợp dễ dàng:

+ Hỗ trợ .NET Core và các ứng dụng .NET Framework.

+ Có thể sử dụng với các ứng dụng web, mobile, hoặc dịch vụ backend.

- Ứng dụng OTP.NET trong dự án:

+ Sinh mã OTP tạm thời: OTP.NET được dùng để tạo mã tạm thời khi người dùng đăng ký tài khoản, cần có bước xác thực OTP để đảm bảo chính người dùng đó đã đăng ký. Mã này sẽ được gửi qua Email người dùng đăng ký.

+ Xác thực mã OTP: Khi người dùng nhập mã OTP, server sử dụng OTP.NET để kiểm tra mã có hợp lệ hay không bằng cách kiểm tra tính chính xác (mã khớp với secret key), xác minh thời hạn (mã OTP vẫn còn hiệu lực trong khoảng thời gian cho phép).

#### 2.1.4.2. Lợi ích của xác thực hai yếu tố (2FA)

Xác thực hai yếu tố (2FA) là một biện pháp bảo mật bổ sung, yêu cầu người dùng cung cấp hai yếu tố khác nhau để xác minh danh tính. Trong dự án, 2FA được triển khai với sự kết hợp giữa mật khẩu truyền thống và mã OTP tạm thời.

- Ngăn chặn tấn công dò mật khẩu (Brute Force): Ngay cả khi mật khẩu bị đánh cắp, hacker vẫn không thể truy cập tài khoản nếu không có mã OTP hợp lệ.

- Giảm thiểu rủi ro bị đánh cắp thông tin: OTP chỉ có hiệu lực trong thời gian ngắn, làm giảm khả năng sử dụng mã OTP cũ hoặc bị rò rỉ.

- Bảo vệ khỏi tấn công phishing: Hacker khó có thể sử dụng mật khẩu thu thập được nếu không có quyền truy cập mã OTP kèm theo.

### 2.1.5. SMTP và gửi Email trong .NET Core [7]

#### 2.1.5.1. Khái niệm SMTP

SMTP (Simple Mail Transfer Protocol) là một giao thức tiêu chuẩn được sử dụng để gửi email qua Internet. Đây là một trong những giao thức quan trọng trong hệ thống email, giúp truyền tải thư từ client đến máy chủ email và giữa các máy chủ email với nhau.

Các tính năng chính của SMTP:

- Gửi email từ client đến server: SMTP đảm bảo email được truyền tải an toàn từ ứng dụng của bạn đến máy chủ email.

- Truyền email giữa các server: Máy chủ SMTP chịu trách nhiệm chuyển tiếp email đến đích cuối cùng.

- Bảo mật: Hỗ trợ mã hóa TLS/SSL để bảo vệ thông tin trong quá trình truyền tải.

### 2.1.6. Hệ thống Xác thực và Phân quyền [8]

#### 2.1.6.1. Xác thực với JWT

JWT (JSON Web Token) là một tiêu chuẩn mở (RFC 7519) cho phép truyền tải thông tin an toàn giữa các bên dưới dạng token. Trong hệ thống Microservices, JWT được sử dụng phổ biến để xác thực người dùng và phân phối quyền truy cập.

- JWT bao gồm 3 thành phần chính:

- + Header: Chứa thông tin về loại token và thuật toán mã hóa (ví dụ: HS256 hoặc RS256).

+ Payload: Chứa dữ liệu (claims) được mã hóa, như ID người dùng, vai trò, hoặc thời gian hết hạn.

+ Signature: Kết hợp giữa Header, Payload và một khóa bí mật để đảm bảo tính toàn vẹn của token.

- Quy trình hoạt động của JWT trong hệ thống:

+ Đăng nhập và tạo token: Người dùng cung cấp thông tin đăng nhập (username/password) thông qua API. Sau đó hệ thống xác thực thông tin và tạo một token JWT chứa các thông tin như: ID người dùng, vai trò (role), thời gian hết hạn, ... và có thể tùy biến thêm ví dụ như thêm trực tiếp access token vào luôn.

+ Gửi token trong yêu cầu: Người dùng đính kèm token JWT trong header của các yêu cầu API, thường dưới dạng: Authorization: Bearer <JWT>. Token JWT thường có kí tự “eyJ” làm tiền tố.

+ Xác thực token tại API Gateway: API Gateway sẽ kiểm tra Token có hợp lệ không (chữ ký đúng và chưa hết hạn)?, Payload chứa thông tin phù hợp với yêu cầu (ví dụ: quyền truy cập)? Nếu hợp lệ, API Gateway chuyển tiếp yêu cầu đến dịch vụ tương ứng.

+ Xác minh tại dịch vụ: Các dịch vụ Microservices có thể kiểm tra lại token hoặc dựa vào quyết định của API Gateway.

- Lợi ích khi sử dụng JWT

+ Tách biệt và độc lập: Người dùng chỉ cần đăng nhập một lần, sau đó có thể truy cập tất cả các dịch vụ mà không cần xác thực lại.

+ Hiệu quả: Không cần lưu trữ trạng thái phiên (session) trên server, giảm tải cho backend.

+ An toàn: Payload được mã hóa, và chữ ký đảm bảo rằng token không bị giả mạo.

+ Tương thích cao: JWT hoạt động độc lập với ngôn ngữ lập trình, dễ dàng tích hợp vào các hệ thống Microservices.

### 2.1.6.2. Phân quyền dựa trên vai trò

Phân quyền (Authorization) dựa trên vai trò được triển khai bằng cách gán vai trò (role) cho từng người dùng và sử dụng vai trò này để kiểm soát quyền truy cập.

- Vai trò (Role) trong dự án: Trong project của em thì em phân 5 role: admin, quản lý cửa hàng, nhân viên, cửa hàng (POS) và khách hàng.

- Quy trình phân quyền:

- + Token chứa thông tin vai trò: Khi tạo JWT, vai trò của người dùng được thêm vào payload.

- + Kiểm tra vai trò trong yêu cầu: Khi nhận được yêu cầu, API Gateway đọc thông tin vai trò từ token và kiểm tra quyền truy cập.

- + Quyết định chuyển tiếp: Nếu người dùng không có quyền, trả về mã lỗi 403 Forbidden, còn nếu hợp lệ, chuyển tiếp yêu cầu đến dịch vụ tương ứng.

- Ưu điểm của phân quyền dựa trên vai trò

- + Dễ quản lý: Chỉ cần thay đổi vai trò của người dùng để điều chỉnh quyền truy cập.

- + Linh hoạt: Có thể mở rộng để hỗ trợ các quyền chi tiết hơn dựa trên nhóm hoặc cấp bậc.

- + An toàn: Giảm rủi ro truy cập trái phép nhờ kiểm tra toàn diện từ Gateway đến dịch vụ.

### *2.1.7. Tổng quan về giao tiếp giữa các Microservices*

Trong kiến trúc Microservices, giao tiếp giữa các dịch vụ là yếu tố cốt lõi đảm bảo sự hoạt động trơn tru và hiệu quả của toàn hệ thống. Phương thức giao tiếp phải đáp ứng các yêu cầu về hiệu năng, độ tin cậy, và khả năng mở rộng.

#### 2.1.7.1. RESTful API [9]

RESTful API (Representational State Transfer) là một phong cách kiến trúc phổ biến để xây dựng các dịch vụ web. Các Microservices trong dự án giao tiếp với nhau chủ yếu thông qua RESTful API.

- Lý do chọn RESTful API:

+ Tương thích cao: RESTful API sử dụng HTTP và định dạng JSON, giúp các dịch vụ dễ dàng tương tác với nhau bất kể ngôn ngữ lập trình hay nền tảng.

+ Đơn giản và hiệu quả: HTTP Request/Response rất dễ triển khai và tối ưu hóa.

+ Khả năng mở rộng: RESTful API hỗ trợ tốt cho các hệ thống có quy mô lớn và phân tán.

- Đặc điểm của RESTful API trong dự án:

+ Các phương thức được sử dụng:

- GET: Lấy dữ liệu (ví dụ: danh sách người dùng, thông tin sản phẩm).
- POST: Tạo mới dữ liệu (ví dụ: thêm người dùng mới).
- PUT/PATCH: Cập nhật dữ liệu (ví dụ: chỉnh sửa thông tin người dùng).
- DELETE: Xóa dữ liệu (ví dụ: xóa tài khoản).

+ Cấu trúc URL rõ ràng và nhất quán: Ví dụ:

- GET /api/users/: Lấy danh sách người dùng.
- POST /api/orders: Tạo đơn hàng mới.

+ Sử dụng mã trạng thái HTTP để phản hồi:

- 200 OK: Yêu cầu thành công.
- 201 Created: Tạo mới thành công.
- 400 Bad Request: Yêu cầu không hợp lệ.
- 401 Unauthorized: Không được phép.
- 500 Internal Server Error: Lỗi từ hệ thống.

- Ưu điểm của RESTful API:

+ Tính độc lập: Mỗi Microservice có thể phát triển, triển khai và mở rộng riêng lẻ mà không ảnh hưởng đến các dịch vụ khác.



+ **Khả năng tích hợp:** RESTful API dễ dàng tích hợp với các ứng dụng web, mobile, hoặc các bên thứ ba.

+ **Dễ bảo trì:** Với cấu trúc rõ ràng, RESTful API dễ dàng kiểm tra, gỡ lỗi và cập nhật.

#### 2.1.7.2. Các ràng buộc giao tiếp

Trong hệ thống Microservices, giao tiếp giữa các dịch vụ có thể được thực hiện theo hai kiểu: đồng bộ và bất đồng bộ, tùy thuộc vào yêu cầu cụ thể.

- Giao tiếp đồng bộ:

+ **Định nghĩa:** Giao tiếp đồng bộ yêu cầu dịch vụ A gửi yêu cầu đến dịch vụ B và chờ phản hồi trước khi tiếp tục.

+ Ưu điểm:

- Đơn giản và dễ triển khai.
- Phù hợp cho các tác vụ yêu cầu phản hồi ngay lập tức (ví dụ: xác thực người dùng, kiểm tra tồn kho).

+ Nhược điểm:

- Gây chậm trễ nếu dịch vụ B có độ trễ cao hoặc gặp lỗi.
- Tăng phụ thuộc giữa các dịch vụ.

+ Triển khai trong dự án: Sử dụng HTTP Request/Response qua RESTful API.

- Giao tiếp bất đồng bộ:

+ **Định nghĩa:** Giao tiếp bất đồng bộ cho phép dịch vụ A gửi yêu cầu đến dịch vụ B mà không cần chờ phản hồi ngay lập tức. Dịch vụ B xử lý yêu cầu và phản hồi sau khi hoàn thành.

+ Ưu điểm:

- Tăng hiệu năng và khả năng xử lý lượng lớn yêu cầu.
- Giảm phụ thuộc trực tiếp giữa các dịch vụ.
- Phù hợp với các tác vụ tốn thời gian hoặc không cần phản hồi ngay lập tức (ví dụ: gửi email, xử lý đơn hàng).

+ Nhược điểm:

- Phức tạp hơn trong triển khai và theo dõi.

- Cần cơ chế đảm bảo tin nhắn không bị mất (Message Queue).
- Ứng dụng trong dự án
  - RESTful API (đồng bộ):
    - Xác thực người dùng qua JWT.
    - Xử lý yêu cầu thông thường từ client.

#### 2.1.8. Bảo mật

Bảo mật là yếu tố quan trọng trong các hệ thống Microservices, đặc biệt với các hệ thống quản lý thông tin nhạy cảm của người dùng. Để đảm bảo an toàn cho dữ liệu và dịch vụ, dự án đã triển khai các biện pháp bảo mật như mã hóa dữ liệu, giới hạn tốc độ API và sử dụng HTTPS.

##### 2.1.8.1. Mã hóa dữ liệu

- Khái niệm: Mã hóa là quá trình chuyển đổi dữ liệu từ dạng dễ đọc (plain text) sang dạng mã hóa (cipher text), chỉ những bên được cấp quyền mới có thể giải mã. Việc mã hóa giúp ngăn chặn truy cập trái phép vào dữ liệu khi bị đánh cắp hoặc tấn công.

##### - Cách triển khai trong dự án:

+ Dữ liệu nhạy cảm:

- Mật khẩu người dùng được mã hóa bằng thuật toán bcrypt trước khi lưu vào cơ sở dữ liệu.
- Các thông tin quan trọng khác như số thẻ tín dụng, mã OTP được mã hóa sử dụng AES (Advanced Encryption Standard).

+ JWT (JSON Web Token):

- Token JWT chứa thông tin xác thực người dùng được ký bằng thuật toán HMAC-SHA256 để đảm bảo tính toàn vẹn và tránh bị giả mạo.

+ Lưu trữ an toàn: Dữ liệu mã hóa được lưu kèm với một salt (chuỗi ngẫu nhiên) để tăng cường tính bảo mật.

##### - Lợi ích:

+ Ngăn chặn việc đọc trộm dữ liệu trong trường hợp bị rò rỉ.

+ Đảm bảo dữ liệu được bảo mật ngay cả khi hệ thống bị xâm nhập.

##### 2.1.8.2. Sử dụng HTTPS

- Khái niệm: HTTPS (HyperText Transfer Protocol Secure) là phiên bản an toàn của HTTP, sử dụng giao thức SSL/TLS để mã hóa dữ liệu trao đổi giữa client và server.

- Lợi ích:

- + Ngăn chặn tấn công Man-in-the-Middle (MITM).
- + Bảo vệ thông tin người dùng khi trao đổi qua mạng.

### *2.1.9. Ứng dụng Google Maps API trong hệ thống [10]*

#### 2.1.9.1. Tổng quan về Google Maps API

Google Maps API là một bộ công cụ mạnh mẽ do Google cung cấp, cho phép các ứng dụng tích hợp các tính năng liên quan đến bản đồ và vị trí địa lý. Các dịch vụ nổi bật bao gồm:

- Maps: Hiển thị bản đồ trực quan.
- Geocoding API: Chuyển đổi địa chỉ thành tọa độ (kinh độ, vĩ độ) và ngược lại.
- Distance Matrix API: Tính toán khoảng cách và thời gian di chuyển giữa các điểm.
- Directions API: Lập kế hoạch tuyến đường chi tiết, bao gồm chỉ đường và thời gian di chuyển.
- Places API: Cung cấp thông tin về các địa điểm cụ thể (ví dụ: cửa hàng, quán ăn).

Trong project của tôi, Distance Matrix API và Geocoding API là hai thành phần chính được sử dụng để tính toán khoảng cách và chi phí giao hàng.

#### 2.1.9.2. Geocoding API và ứng dụng trong hệ thống

Geocoding API giúp chuyển đổi địa chỉ cụ thể (ví dụ: địa chỉ cửa hàng, địa chỉ giao hàng) thành tọa độ địa lý (kinh độ và vĩ độ). Điều này là cần thiết để tính toán khoảng cách giữa các địa điểm.

Ứng dụng trong hệ thống:

- Người dùng nhập địa chỉ giao hàng khi đặt hàng.
- Hệ thống sử dụng Geocoding API để chuyển đổi địa chỉ thành tọa độ địa lý.
- Tọa độ này được sử dụng làm đầu vào cho Distance Matrix API để tính toán khoảng cách và thời gian giao hàng.

#### 2.1.9.3. Distance Matrix API và tính toán chi phí giao hàng

Distance Matrix API cung cấp khoảng cách và thời gian di chuyển giữa hai hoặc nhiều địa điểm dựa trên các phương thức di chuyển khác nhau (xe máy, ô tô, đi bộ). Đây là cơ sở để tính phí giao hàng.

Ứng dụng trong hệ thống:

- Hệ thống gửi yêu cầu đến Distance Matrix API với:- Tọa độ cửa hàng nơi hàng hóa được xuất phát.
- Tọa độ địa chỉ khách hàng.

API trả về:

- Khoảng cách giữa hai điểm (m).

Hệ thống tính phí giao hàng dựa trên khoảng cách và một bảng giá đã được định nghĩa trước.

#### 2.1.9.4. Tích hợp Google Maps API vào hệ thống

Việc tích hợp Google Maps API trong dự án được thực hiện thông qua:

- Khóa API: Mỗi ứng dụng sử dụng Google Maps API cần một khóa API để định danh và xác thực.
- Khóa API: Mỗi ứng dụng sử dụng Google Maps API cần một khóa API để định danh và xác thực.
- Khóa API: Mỗi ứng dụng sử dụng Google Maps API cần một khóa API để định danh và xác thực.

Luồng hoạt động trong hệ thống:

- Người dùng gửi yêu cầu đặt hàng (bao gồm địa chỉ giao hàng).
- Hệ thống:
  - Gửi yêu cầu đến Geocoding API để lấy tọa độ.
  - Gửi yêu cầu đến Distance Matrix API để tính khoảng cách
- Kết quả:
  - Hệ thống hiển thị chi phí giao hàng ước tính cho người dùng trước khi xác nhận đơn hàng.
  - Thông tin được lưu trữ trong cơ sở dữ liệu và hiển thị trong hóa đơn.

#### 2.1.9.5. Lợi ích của việc sử dụng Google Maps API

- Chính xác cao: Sử dụng dữ liệu bản đồ và thuật toán của Google, đảm bảo tính chính xác khi tính toán khoảng cách.
- Tăng trải nghiệm người dùng: Người dùng có thể thấy chi phí giao hàng ước tính ngay lập tức, minh bạch và tiện lợi.
- Tự động hóa: Giảm thiểu sai sót do nhập liệu thủ công nhờ tự động chuyển đổi địa chỉ và tính toán khoảng cách.
- Hiệu quả kinh tế: Tối ưu hóa tuyến đường, giúp tiết kiệm chi phí vận chuyển.

#### 2.1.9.6. Các ràng buộc và hạn chế

- Chi phí API: Google Maps API tính phí dựa trên số lượng yêu cầu. Với lượng lớn đơn hàng, chi phí có thể tăng đáng kể.
- Hạn chế API Key: Khóa API cần được bảo mật để tránh bị sử dụng trái phép.

#### **Tốc độ xử lý:**

- Một số yêu cầu phức tạp có thể mất thời gian xử lý, ảnh hưởng đến trải nghiệm người dùng.

#### 2.1.10. AJAX (*Asynchronous JavaScript and XML*) [11]

- Khái niệm: AJAX là một kỹ thuật được sử dụng trong phát triển web, cho phép các trang web gửi và nhận dữ liệu từ máy chủ mà không cần phải tải lại toàn bộ trang. AJAX là sự kết hợp của nhiều công nghệ, bao gồm HTML, CSS, JavaScript, và các yêu cầu HTTP.

- Đặc điểm chính:
  - + Hoạt động bất đồng bộ: Giúp cải thiện hiệu suất và trải nghiệm người dùng bằng cách chỉ làm mới các phần cụ thể của trang web.
  - + Sử dụng nhiều định dạng dữ liệu: XML, JSON, HTML, hoặc văn bản thuần túy.
- Ứng dụng:
  - + Làm mới nội dung trang mà không tải lại toàn bộ.
  - + Gửi và nhận dữ liệu trong các ứng dụng web (chẳng hạn như form hoặc bảng).
  - + Hiện thị thông báo thời gian thực.

### 2.1.11. *FetchAPI* [12]

- Khái niệm:

+ Fetch API là một giao diện lập trình ứng dụng được tích hợp trong JavaScript, dùng để thực hiện các yêu cầu HTTP một cách hiện đại và linh hoạt. Nó thay thế cách làm việc với XMLHttpRequest truyền thống, hỗ trợ cú pháp Promise để làm cho mã dễ đọc và quản lý hơn.

+ FetchAPI là công nghệ mới hơn AJAX và không cần thư viện JQuery như AJAX.

- Đặc điểm chính:

+ Hỗ trợ bất đồng bộ (Asynchronous): Fetch API sử dụng Promise, cho phép thực hiện các yêu cầu HTTP mà không làm gián đoạn luồng xử lý chính.

+ Cú pháp đơn giản: Sử dụng cú pháp fetch() để gửi các yêu cầu GET, POST, PUT, DELETE,...

+ Tích hợp trong trình duyệt: Không cần cài đặt thêm thư viện bên ngoài.

### 2.1.12. *Rate Limiting*[13]

- Khái niệm: Rate Limiting là một kỹ thuật được sử dụng để giới hạn số lượng yêu cầu mà một máy khách (client) có thể gửi đến máy chủ (server) trong một khoảng thời gian nhất định. Kỹ thuật này thường được áp dụng để bảo vệ máy chủ khỏi việc bị quá tải hoặc tấn công từ chối dịch vụ (DDoS).

- Đặc điểm chính:

+ Giới hạn số lượng yêu cầu

+ Bảo vệ máy chủ

+ Tăng cường bảo mật

- Ứng dụng:

+ API bảo mật: Hạn chế số lượng yêu cầu từ một địa chỉ IP cụ thể.

+ Quản lý tài nguyên: Đảm bảo tài nguyên máy chủ không bị quá tải.

+ Kiểm soát hành vi: Ngăn chặn spam từ các người dùng không hợp lệ.

## 2.2. Tổng quan về các công cụ

### 2.2.1. Ngôn ngữ lập trình C# [14]

C# là một ngôn ngữ lập trình đơn giản, hiện đại, mục đích tổng quát, hướng đối tượng được phát triển bởi Microsoft và được phê chuẩn bởi European Computer Manufacturers Association (ECMA) và International Standards Organization (ISO). C# được phát triển bởi Anders Hejlsberg và team của ông trong khi phát triển .NET Framework. C# được thiết kế cho `Common Language Infrastructure (CLI), mà gồm Executable Code và Runtime Environment, cho phép chúng ta sử dụng các ngôn ngữ high-level đa dạng trên các nền tảng và cấu trúc máy tính khác nhau.

Cấu trúc C# khá gần với các ngôn ngữ high-level truyền thống, C và C++ và là một ngôn ngữ lập trình hướng đối tượng. Nó có sự giống nhau mạnh mẽ với Java, nó có nhiều đặc điểm lập trình mạnh mẽ mà làm cho nó trở nên ưa thích với các lập trình viên trên toàn thế giới.

Ưu điểm:

- Là một trong số những ngôn ngữ thuần hướng đối tượng.
- Chuyên sử dụng để lập trình cho windows.
- Thiết kế trang web, đơn giản và dễ hiểu.
- Ngôn ngữ dễ học, dễ tiếp cận với Java
- Khả năng tương tác với Database dễ dàng hơn rất nhiều.
- Được window hỗ trợ đầy đủ các control.
- Thư viện .NET nhẹ, dễ cài đặt và được miễn phí.
- Ngôn ngữ mã nguồn mở.
- Code/Build trên Visual Studio, một IDE tiện lợi, mạnh mẽ của Microsoft.
- Có thể sử dụng để lập trình web thông qua C# thuần hoặc ASP.NET.
- IDE Visual Studio hỗ trợ debug, build cực khủng.

Nhược điểm:

- Khi muốn xài IDE bạn cần phải trả phí sử dụng.
- Chỉ đem lại hiệu quả tốt nhất trên Window.
- Lập trình Mobile cần phải thông qua trung gian Xamarin (có phí).
- Sử dụng Database tốt nhất với SQL Server.

### 2.2.2. Hệ quản trị cơ sở dữ liệu SQL Server [15]

SQL Server hay còn gọi là Microsoft SQL Server, viết tắt là **MS SQL Server**. Đây là một phần mềm được phát triển bởi Microsoft dùng để lưu trữ dữ liệu dựa trên chuẩn RDBMS, và nó cũng là một hệ quản trị cơ sở dữ liệu quan hệ đối tượng (ORDBMS). SQL Server là một trong 3 công nghệ dữ liệu dẫn đầu hiện nay cùng với Oracle Database và IBM's DB2.

SQL Server hoạt động là một máy chủ cơ sở dữ liệu. Chức năng chính của nó là lưu trữ và truy xuất dữ liệu theo yêu cầu của các ứng dụng phần mềm. Nó sử dụng câu lệnh SQL để trao đổi dữ liệu giữa máy Client và máy cài SQL Server.

SQL Server cung cấp đầy đủ công cụ để quản lý, từ giao diện GUI cho đến việc sử dụng ngôn ngữ truy vấn SQL. Ngoài ra điểm mạnh của nó là Microsoft nên có khá nhiều nền tảng kết hợp hoàn hảo với SQL Server như ASP.NET, C# xây dựng Winform, bởi vì nó hoạt động hoàn toàn độc lập.

Ưu điểm:

- Có thể cài nhiều phiên bản MS SQL khác nhau trên cùng một máy tính.
- Duy trì riêng biệt các môi trường sản xuất, phát triển, thử nghiệm.
- Giảm thiểu các vấn đề tạm thời trên cơ sở dữ liệu.
- Tách biệt các đặc quyền bảo mật.
- Duy trì máy chủ dự phòng.

Nhược điểm:

- SQL Server chỉ chạy trên hệ điều hành Windows.



- Cần thanh toán phí license để chạy nhiều CSDL (database).

### 2.2.3. Môi trường lập trình Visual Studio 2022 [16]

Microsoft Visual Studio là một môi trường phát triển tích hợp (IDE) từ Microsoft. Microsoft Visual Studio còn được gọi là "Trình soạn thảo mã nhiều người sử dụng nhất thế giới", được dùng để lập trình C++ và C# là chính.

Visual Studio bao gồm một trình soạn thảo mã hỗ trợ IntelliSense cũng như cải tiến mã nguồn. Công cụ tích hợp khác bao gồm một mẫu thiết kế các hình thức xây dựng giao diện ứng dụng, thiết kế web, thiết kế lớp và thiết kế giản đồ cơ sở dữ liệu. Nó chấp nhận các plug-in nâng cao các chức năng ở hầu hết các cấp bao gồm thêm hỗ trợ cho các hệ thống quản lý phiên bản (như Subversion) và bổ sung thêm bộ công cụ dành cho các khía cạnh trong quy trình phát triển website.

### 2.2.4. Phần mềm StarUML [17]

StarUML là một mô hình nền tảng, là phần mềm hỗ trợ UML (Unified Modeling Language). Nó hỗ trợ các phương pháp tiếp cận MDA (Model Driven Architecture) bằng cách hỗ trợ các khái niệm hồ sơ UML. Tức là StarUML hỗ trợ phân tích và thiết kế hệ thống một điều mà bất cứ dự án nào đều cần có. Ngoài ra dùng StarUML sẽ đảm bảo tối đa hóa năng suất và chất lượng của các dự án phần mềm của bạn. Vì nó cho phép mô hình hóa nên sẽ không phụ thuộc vào người code, ngôn ngữ code hay nền tảng sử dụng.

Ưu điểm:

- Miễn phí và mã mã nguồn mở: starUML có phiên bản mã nguồn mở, nghĩa là bạn có thể sử dụng nó miễn phí và thậm chí tùy chỉnh mã nguồn nếu cần.
- Hỗ trợ UML đầy đủ: StarUML hỗ trợ các biểu đồ và kỹ thuật UML phổ biến như lớp, sở hữu, hoạt động, tuần tự, trạng thái, và nhiều loại khác.
- Giao diện người dùng thân thiện: Giao diện của StarUML dễ sử dụng và thân thiện, giúp người dùng nhanh chóng làm quen với các tính năng của nó.

- Mở rộng tính năng: StarUML hỗ trợ cài đặt các plugin mở rộng, cho phép bạn tùy chỉnh và mở rộng chức năng theo nhu cầu cụ thể của dự án.

Nhược điểm:

- Hạn chế về hiệu suất: StarUML có thể trở nên chậm và tốn tài nguyên máy tính khi làm việc với các dự án lớn và phức tạp.
- Không cập nhật thường xuyên: Trong quá khứ, StarUML đã có vấn đề về việc cập nhật phần mềm đều đặn, tạo ra sự lo ngại về tính ổn định và tính năng mới. Tuy nhiên, điều này có thể đã thay đổi sau thời điểm tôi cắt đứt kiến thức vào năm 2022.
- Hỗ trợ yếu thế cho nhiều ngôn ngữ lập trình: Trong số các tính năng, hỗ trợ cho nhiều ngôn ngữ lập trình không phải là điểm mạnh của StarUML.
- Giao diện có thể cảm thấy lỗi thời: Giao diện người dùng của StarUML có thể cảm thấy lỗi thời so với một số công cụ thiết kế phần mềm khác có giao diện đẹp và hiện đại hơn.

#### 2.2.5. Công cụ CASE Studio 2

Case Studio – Phần mềm vẽ mô hình Entity Relationship Diagrams (ERD) và Data Flow Diagrams (DFD) cũng như tạo script SQL cho nhiều cơ sở dữ liệu một cách tự động. Phần mềm hỗ trợ đầy đủ cho 20 cơ sở dữ liệu như Oracle, DB2, MS SQL, Sybase, MySQL, Firebird, PostgreSQL vvv.

Chức năng chính của CASE Studio 2: tạo mô hình ERD từ mã script SQL, Đảo mã từ file HTML cụ thể hay file dữ liệu RTF, xuất mô hình Data Flow Diagrams thành file quản lý định dạng XML, Templater editor...

Phiên bản 2.23.1 hỗ trợ thêm cho: Informix 10, Oracle 9i/10g, MS SQL 2005/2000, DB2 version 8, PostgreSQL 8.1/8.0 , báo cáo định dạng HTML/RTF.

Ưu điểm:

- Mô hình hóa dữ liệu: CASE Studio 2 cho phép người dùng dễ dàng tạo mô hình cơ sở dữ liệu bằng cách sử dụng biểu đồ ER (Entity-Relationship) và chú thích bảng cơ sở dữ liệu, quan hệ, và các yếu tố khác.

- Tích hợp hệ quản lý cơ sở dữ liệu: Công cụ này có tích hợp với các hệ quản lý cơ sở dữ liệu phổ biến, cho phép bạn tự động tạo các tạo, cập nhật, và xóa truy vấn dựa trên mô hình cơ sở dữ liệu.
- Hỗ trợ đội làm việc: CASE Studio 2 cho phép nhiều người dùng làm việc cùng một dự án, điều này hữu ích trong môi trường phát triển phần mềm đa người tham gia.
- Xuất dự án: Công cụ này cho phép bạn xuất dự án dưới dạng tài liệu hoặc mã nguồn, giúp bạn chia sẻ thông tin với các thành viên khác trong dự án.

Nhược điểm:

- Hạn chế về nền tảng: CASE Studio 2 thường chạy trên nền tảng Windows, điều này có nghĩa rằng người dùng hệ điều hành khác (như macOS hoặc Linux) có thể gặp khó khăn trong việc sử dụng công cụ này.
- Không còn được phát triển: Một nhược điểm lớn là CASE Studio 2 đã ngừng phát triển từ một thời gian dài và không còn nhận bất kỳ cập nhật nào. Điều này có thể dẫn đến vấn đề về tính bảo mật và khả năng tương thích trong tương lai.
- Không phải là công cụ UML toàn diện: CASE Studio 2 chủ yếu tập trung vào thiết kế cơ sở dữ liệu và quản lý dự án, không cung cấp các tính năng mô hình hóa và thiết kế phần mềm UML phức tạp.

#### 2.2.6. Phần mềm GitHub [18]

GitHub là một dịch vụ nổi tiếng cung cấp kho lưu trữ mã nguồn Git cho các dự án phần mềm. Github có đầy đủ những tính năng của Git, ngoài ra nó còn bổ sung những tính năng về social để các developer tương tác với nhau. GitHub có 2 phiên bản: miễn phí và trả phí. Với phiên bản có phí thường được các doanh nghiệp sử dụng để tăng khả năng quản lý team cũng như phân quyền bảo mật dự án. Còn lại thì phần lớn chúng ta đều sử dụng Github với tài khoản miễn phí để lưu trữ source code.

Github cung cấp các tính năng social networking như feeds, followers, và network graph để các developer học hỏi kinh nghiệm của nhau thông qua lịch sử commit.

Ưu điểm:

- Dễ sử dụng và trực quan.
- Hỗ trợ đa dạng ngôn ngữ lập trình.
- Quản lý phiên bản hiệu quả.
- Có cộng đồng lớn mạnh.
- Hỗ trợ tích hợp các công cụ phát triển phần mềm khác nhau.

Nhược điểm:

- Giới hạn trên các dự án riêng tư.
- Tính riêng tư và bảo mật.
- Phụ thuộc vào kết nối internet.
- Cần hiểu biết về Git.

### 2.2.7. Bootstrap [19]

Bootstrap là một framework CSS và JavaScript mã nguồn mở được phát triển bởi nhóm Twitter. Nó cung cấp một bộ công cụ giao diện người dùng để tạo ra các trang web và ứng dụng web có giao diện thân thiện, linh hoạt và dễ dàng tương thích trên nhiều thiết bị và kích thước màn hình khác nhau.

Ưu điểm:

- Thiết kế đáp ứng (Responsive design): Bootstrap cung cấp lớp grid linh hoạt và các thành phần giao diện người dùng (UI components) được tối ưu hóa để tự động điều chỉnh và hiển thị đẹp trên các thiết bị khác nhau, từ máy tính đến điện thoại di động.
- Tiết kiệm thời gian: Bootstrap cung cấp một loạt các lớp CSS và các thành phần UI đã được thiết kế sẵn, giúp giảm thiểu thời gian phát triển và kiểm thử, bằng cách cho phép nhà phát triển sử dụng lại mã nguồn mà không cần phải viết lại từ đầu.
- Tương thích trên nhiều trình duyệt: Bootstrap được kiểm tra và tối ưu hóa để hoạt động trên hầu hết các trình duyệt web phổ biến như Chrome, Firefox, Safari, và Edge.
- Cộng đồng lớn và hỗ trợ đa dạng: Với một cộng đồng lớn và phát triển mạnh mẽ, Bootstrap có sẵn rất nhiều tài liệu, hướng dẫn và các theme mẫu cho nhà phát triển.

- Tích hợp dễ dàng: Bootstrap có thể dễ dàng tích hợp với các framework back-end như Laravel, Ruby on Rails, hoặc Django.

Nhược điểm:

- Giao diện trang web giống nhau: Do Bootstrap được sử dụng rộng rãi và cung cấp các mẫu giao diện mặc định, nên có nguy cơ cao rằng các trang web sử dụng Bootstrap có thể trông giống nhau và thiếu tính cá nhân hóa.
- Dung lượng lớn: Nếu chỉ sử dụng một phần nhỏ của Bootstrap cho một trang web nhỏ, việc tải xuống toàn bộ framework có thể làm tăng thời gian tải trang.
- Khả năng tùy chỉnh hạn chế: Bootstrap có thể bị hạn chế trong việc tùy chỉnh giao diện theo ý muốn của bạn, đặc biệt là khi cần phải thay đổi các đặc tính thiết kế cụ thể.

## 2.3. Xác định yêu cầu

### 2.3.1. Yêu cầu chung

- Kiểu bố cục gọn gàng, có thể chèn thêm quảng cáo.
- Phong cách thiết kế đơn giản.
- Không sử dụng các font chữ không chuẩn cho nội dung website. Nên sử dụng các font Unicode chuẩn như Arial, Verdana, Tahoma, Times News Roman.

### 2.3.2. Giao diện

- Giao diện thân thiện với người dùng, các yêu cầu thể hiện rõ ràng, dễ hiểu.
- Màu sắc hài hòa dễ nhìn.
- Nội dung giao diện:
  - + Admin:
    - \* SẢN PHẨM: THÊM, XÓA, SỬA
    - \* LÔ HÀNG: THÊM, XÓA, SỬA
    - \* XUẤT KHO: THÊM, XÓA, SỬA, TÌM KIẾM

\* CHI NHÁNH: THÊM, XÓA, SỬA, XEM CHI TIẾT (NHÂN SỰ, HÀNG HÓA, HÓA ĐƠN, NHẬP HÀNG), THỐNG KÊ.

\* KHÁCH HÀNG: XEM CHI TIẾT, CHẶN HOẶC BỎ CHẶN KHÁCH HÀNG.

\* KHUYẾN MÃI: THÊM, XÓA SỬA.

+ Quản lý:

\* SẢN PHẨM: XEM CHI TIẾT SẢN PHẨM

\* CHI NHÁNH: XEM CHI TIẾT HÀNG HÓA, SỬA, XEM NHÂN SỰ, HÓA ĐƠN, NHẬP HÀNG.

+ Nhân viên (Shipper)

\* SẢN PHẨM: CHỈNH SỬA TRẠNG THÁI ĐƠN HÀNG

+ Cửa hàng: THÊM SẢN PHẨM VÀO HÀNG ĐỢI, HOÀN TẤT ĐƠN ĐẶT ONLINE CỦA KHÁCH, XEM LỊCH SỬ ĐƠN HÀNG

+ Khách hàng:

\* SẢN PHẨM: THÊM SẢN PHẨM VÀO GIỎ HÀNG, GIỎ HÀNG, THANH TOÁN, XEM LỊCH SỬ MUA HÀNG.

### *2.3.3. Các tác vụ cơ bản*

- Giao diện của Admin: đăng nhập vào trang web; thêm xóa, sửa sản phẩm (Có thể thêm bằng cách import file excel); thêm xóa sửa hàng nhập kho, xuất kho; thêm xóa sửa chi nhánh và xem các chi tiết trong chi nhánh; xem thông tin khách hàng; thêm xóa sửa thêm voucher giảm giá

- Giao diện quản lý: đăng nhập vào trang web; xem chi tiết các sản phẩm; Xem thống kê của cửa hàng, chi tiết hàng hóa, nhân sự, hóa đơn, lịch sử nhập hàng cũng như chỉnh sửa thông tin chi nhánh.

- Giao diện nhân viên (shipper): đăng nhập vào trang web, xem chi tiết đơn hàng và xác nhận đơn hàng đã được giao thành công.

- Giao diện cửa hàng: Thêm các sản phẩm vào hàng đợi để thanh toán, xác nhận đơn hàng online và xem lịch sử đơn hàng.

- Giao diện khách hàng:

- + Đối với khách hàng không đăng nhập: tìm kiếm sản phẩm, thêm sản phẩm vào giỏ hàng (lưu vào session), chọn sản phẩm muốn mua, đặt và nhận hàng tại cửa hàng, sau đó sẽ có email xác nhận gửi cho khách hàng.

- + Đối với khách hàng đã đăng nhập: tìm kiếm sản phẩm, thêm sản phẩm vào giỏ hàng (lưu vào database), chọn sản phẩm muốn mua, nhập mã voucher (nếu có), chọn phương thức thanh toán (tại cửa hàng hoặc ship tận nơi, nếu chọn ship tận nơi sẽ hiển thị Google Map để xác định khoảng cách và tính tiền ship), nhập các thông tin cơ bản, sau đó sẽ có email xác nhận gửi cho khách hàng.

#### *2.3.4 Các công nghệ được sử dụng*

##### 2.3.4.1.Backend Development

- .NET Core Web API:

- + Xây dựng các dịch vụ backend độc lập trong kiến trúc Microservices.

- + Các dịch vụ bao gồm: BranchService, ProductService, InvoiceService, PromotionService, UserService.

- Entity Framework Core:

- + ORM (Object-Relational Mapping) để quản lý cơ sở dữ liệu.

- + Hỗ trợ truy vấn và thao tác trên MS SQL Server.

- Ocelot API Gateway:

- + Tích hợp và điều phối các dịch vụ backend thông qua một cổng API duy nhất.

- + Hỗ trợ cân bằng tải, quản lý truy cập, và caching.

##### 2.3.4.2. Frontend Development

- ASP.NET Core MVC:

- + Giao diện người dùng (UI) tương tác với các dịch vụ backend thông qua API Gateway.

- + Xây dựng các trang web động và phản hồi nhanh.

#### 2.3.4.3. Authentication & Security

- OTP.NET: Tạo mã OTP (One-Time Password) để xác thực người dùng khi đăng ký, đăng nhập, hoặc thực hiện giao dịch.
- SMTP (Simple Mail Transfer Protocol): Gửi email xác nhận và mã OTP đến người dùng.

#### 2.3.4.4. External APIs

- Google Maps API:
  - + Tính toán khoảng cách giữa chi nhánh và địa chỉ giao hàng của khách hàng.
  - + Ước tính chi phí vận chuyển dựa trên khoảng cách.

#### 2.3.4.5. Database:

- MS SQL Server:
  - + Lưu trữ dữ liệu cho từng microservice trong hệ thống.
  - + Sử dụng các bảng riêng cho từng dịch vụ để đảm bảo tính phân tán.

#### 2.3.4.6. Công cụ DevOps:

- PostMan: Test API trong quá trình phát triển và kiểm thử.

#### 2.3.4.7. Kiến trúc và Quản lý Dữ Liệu

- Kiến trúc Microservices:
  - + Chia nhỏ hệ thống thành các dịch vụ độc lập, dễ bảo trì và mở rộng.
  - + Các dịch vụ giao tiếp thông qua REST API.

#### 2.3.4.8. Công cụ hỗ trợ phát triển:

- Visual Studio 2022: IDE chính để phát triển ứng dụng .NET Core.
- GitHub: Quản lý mã nguồn và kiểm soát phiên bản.
- Swagger (OpenAPI): Tạo tài liệu tự động cho các API trong hệ thống.

#### 2.3.4.9. Giao tiếp giữa các Microservices:

- RESTful APIs: Giao thức chính để các dịch vụ giao tiếp với nhau thông qua HTTP.

### **2.4. Yêu cầu hệ thống**

- Máy chủ có khả năng xử lý chính xác, lưu trữ lâu dài, bảo mật.
- Thông tin có tính đồng bộ, phân quyền quản lý chặt chẽ.
- Có tính ứng dụng thực tế cao.

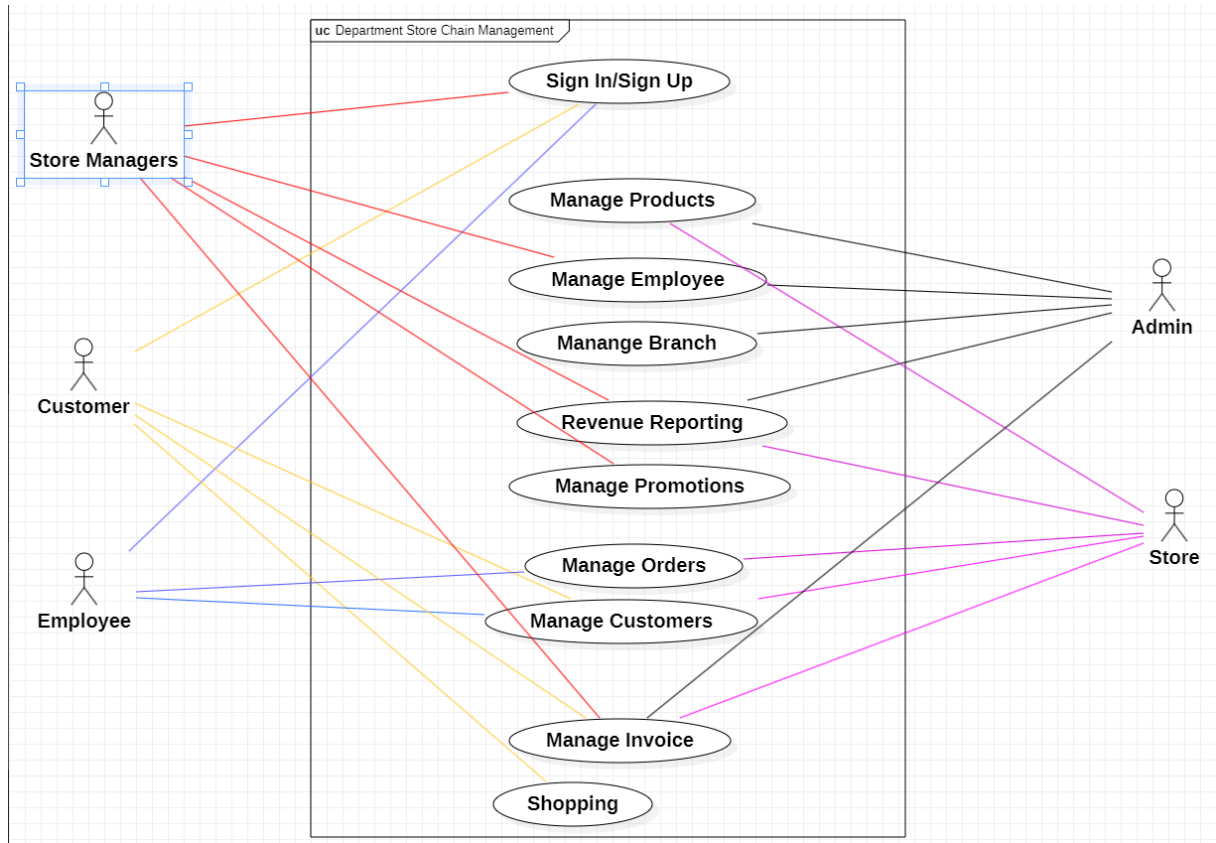


## 2.5. Yêu cầu hệ thống

- Máy chủ có khả năng xử lý chính xác, lưu trữ lâu dài, bảo mật.
- Thông tin có tính đồng bộ, phân quyền quản lý chặt chẽ.
- Có tính ứng dụng thực tế cao.

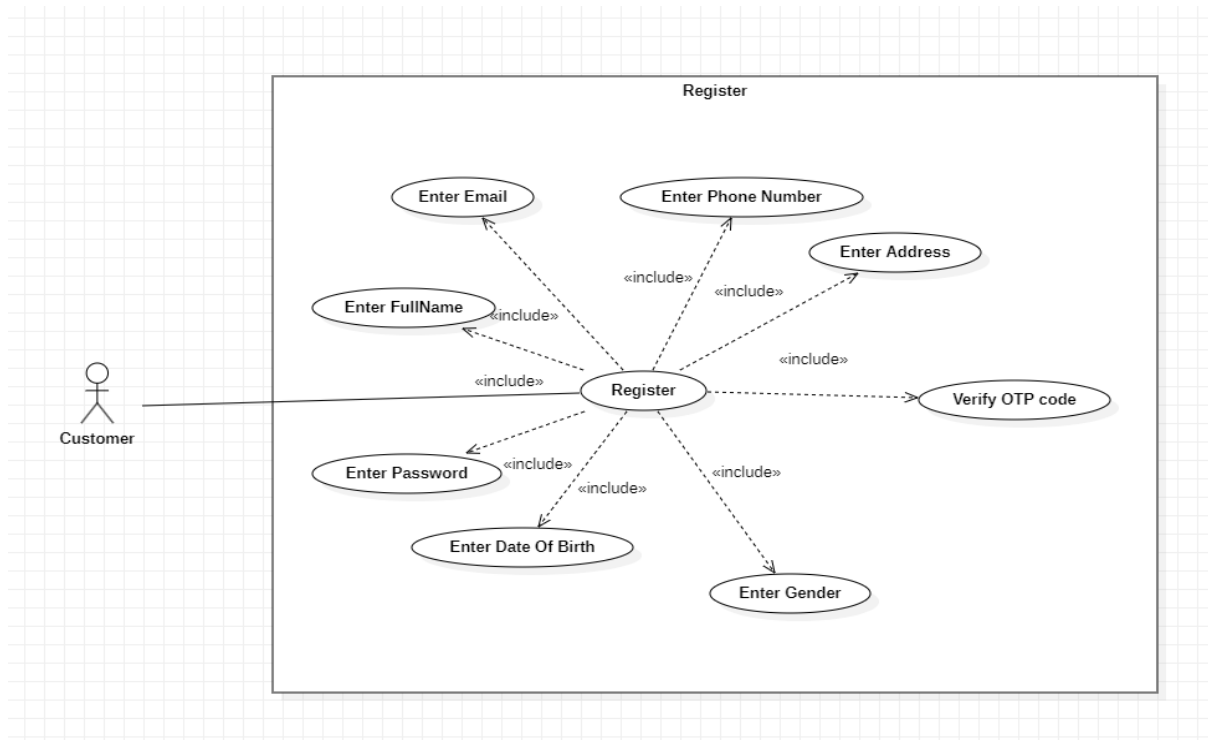
## 2.6. Sơ đồ phân rã chức năng Use-case

### 2.6.1. Use-case tổng quát



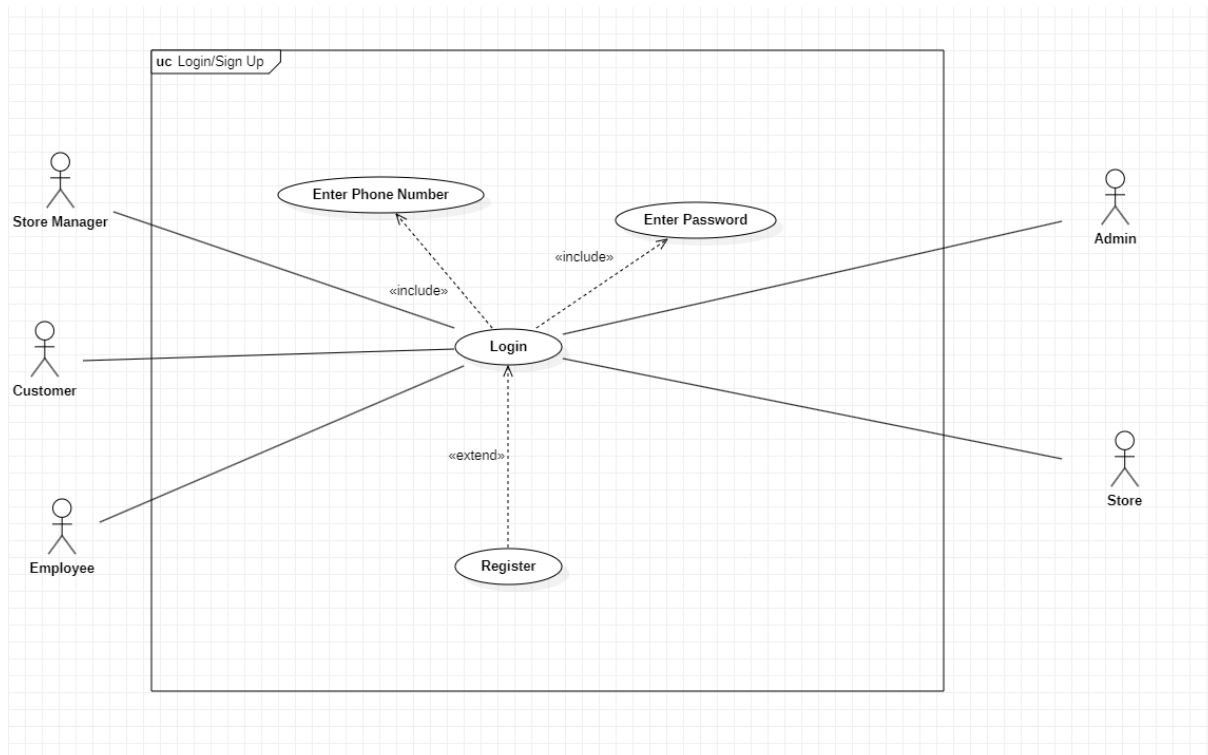
Hình 2.2. Mô hình tổng quát

### 2.6.2. Use-case đăng ký



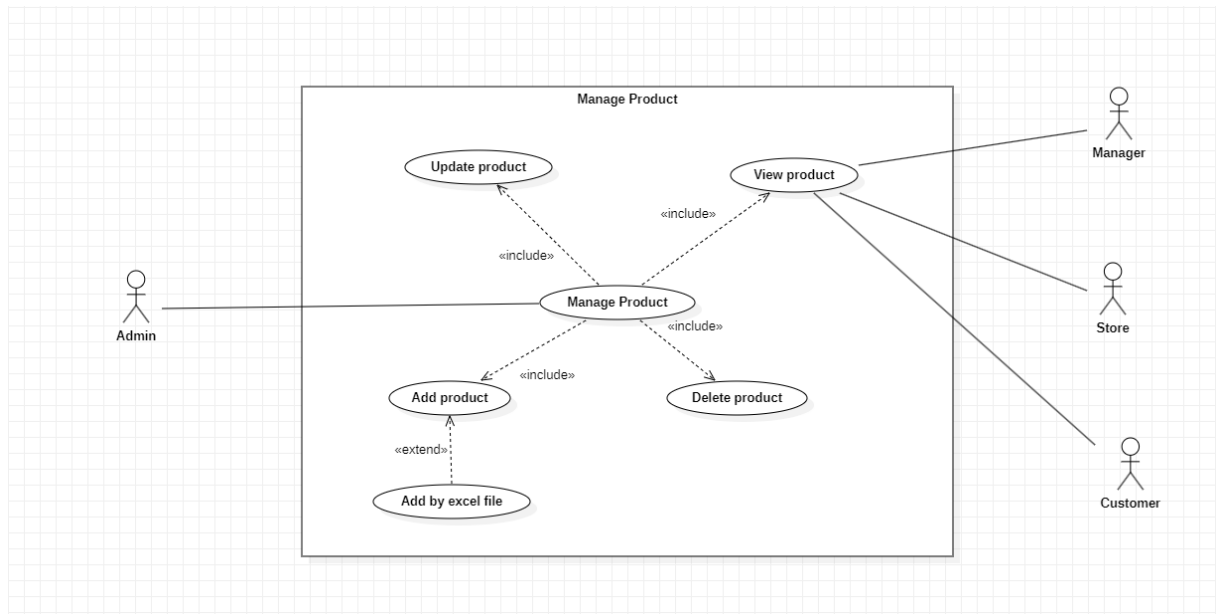
Hình 2.3. Mô hình đăng ký

### 2.6.3. Use-case đăng nhập



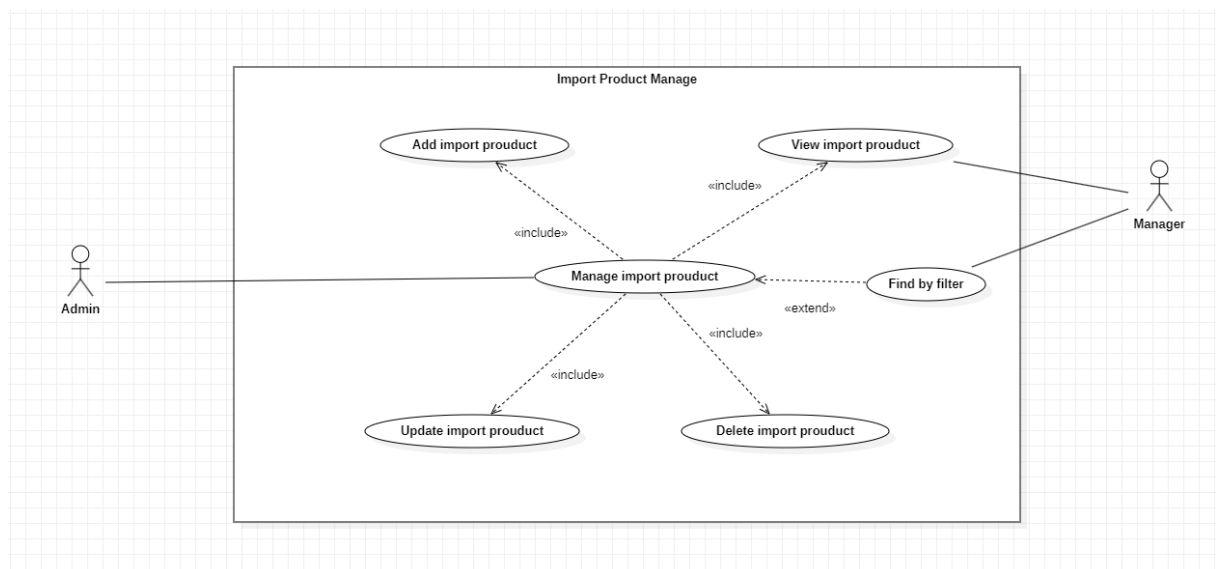
Hình 2.4. Mô hình đăng nhập

#### 2.6.4. Use-case quản lý sản phẩm

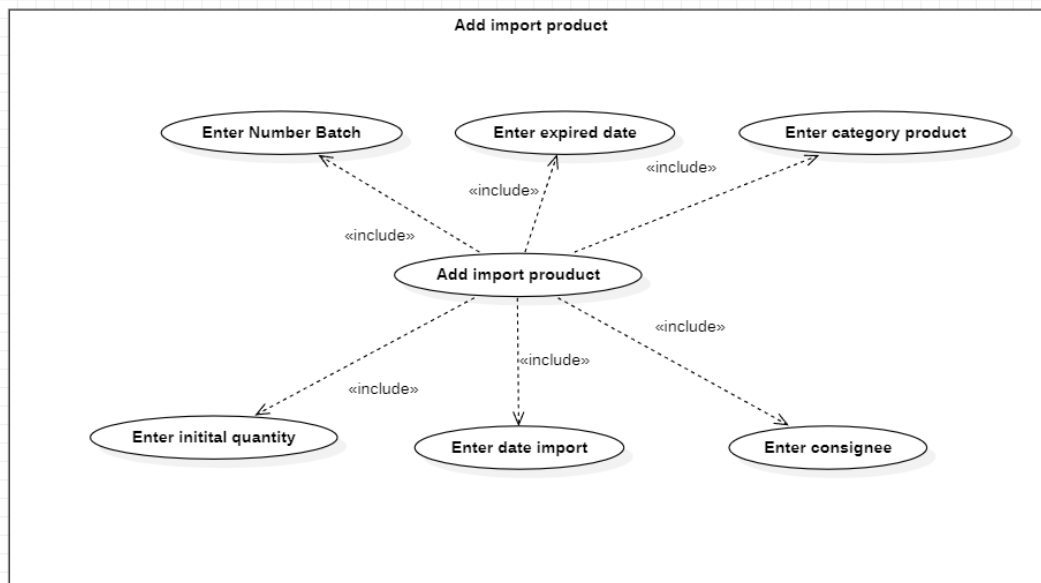


Hình 2.5. Mô hình quản lý sản phẩm

#### 2.6.5. Use-case quản lý nhập kho

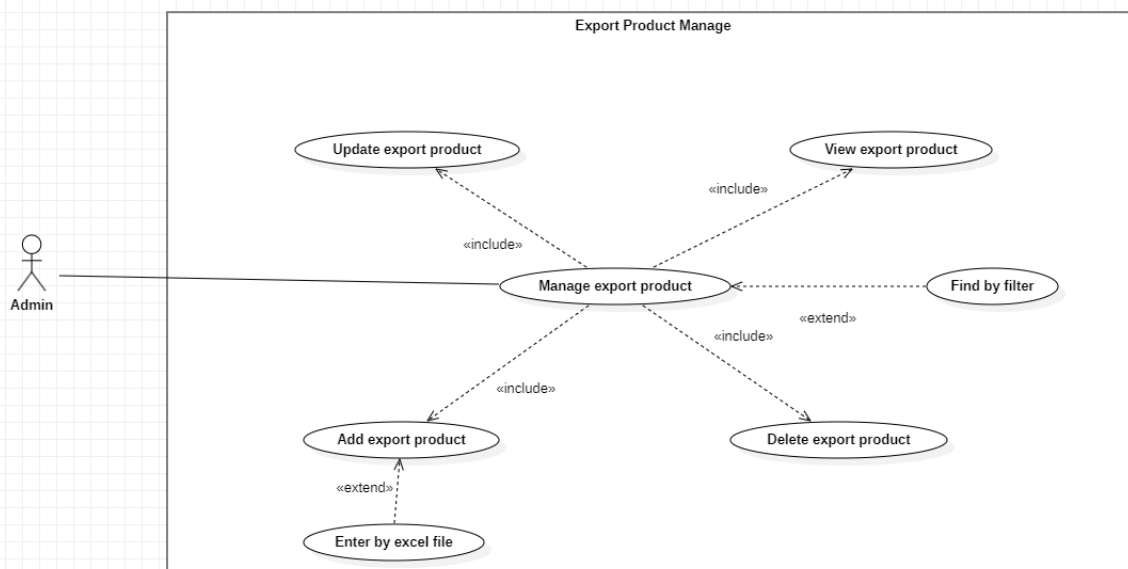


Hình 2.6. Mô hình quản lý nhập kho (1)



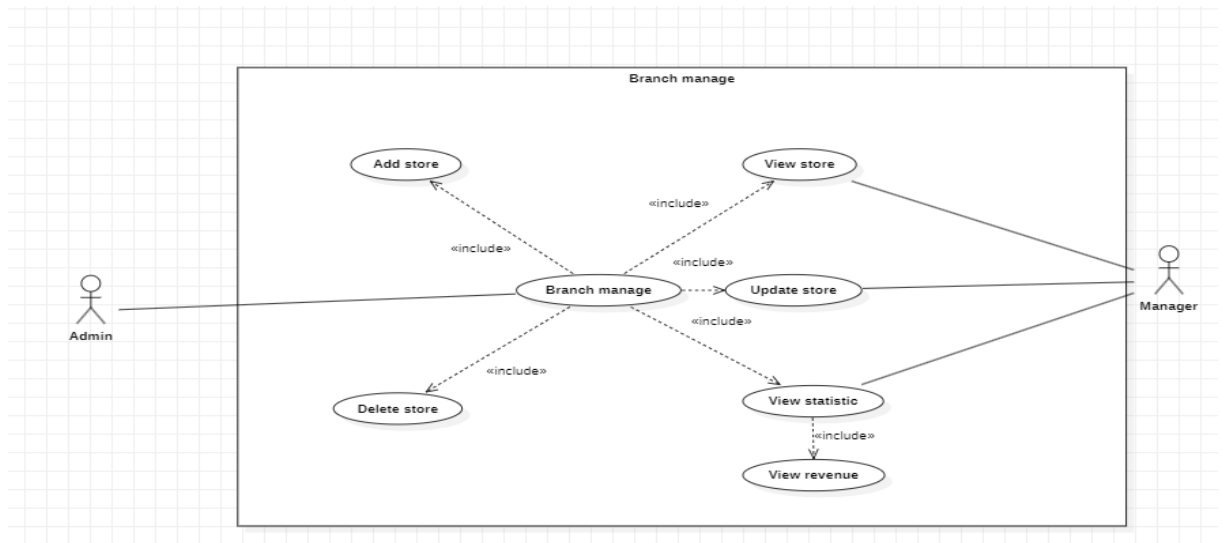
Hình 2.7. Mô hình quản lý nhập kho (2)

#### 2.6.6. Use-case quản lý xuất kho



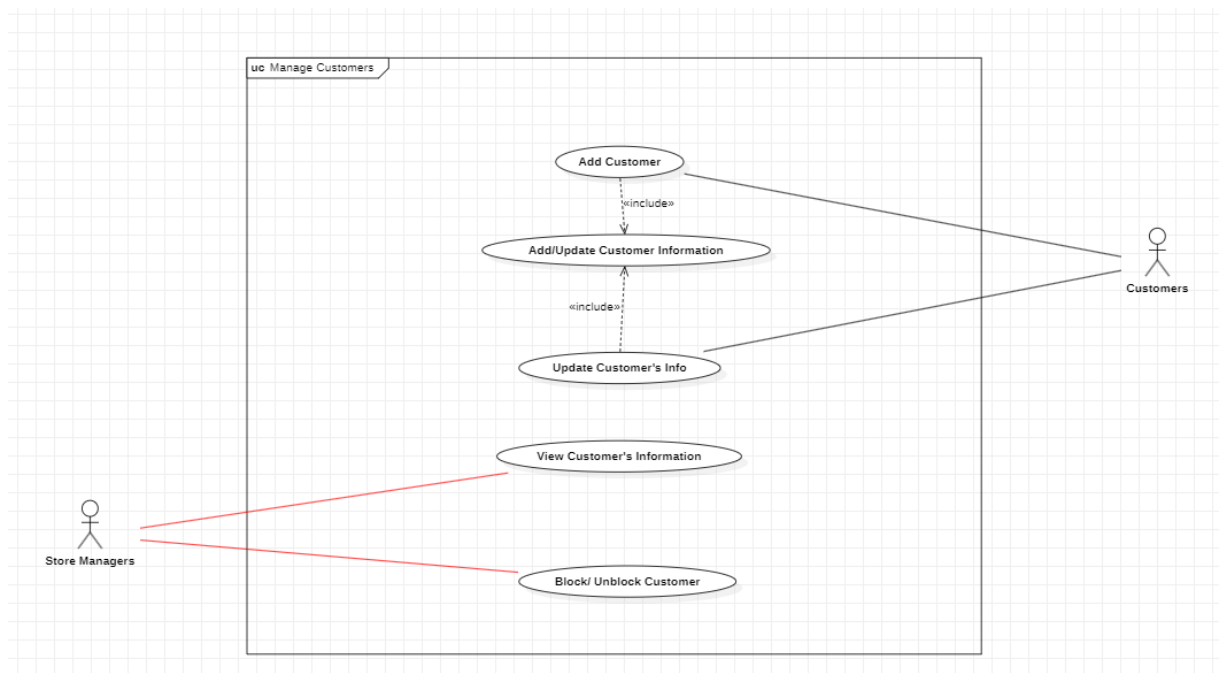
Hình 2.8. Mô hình quản lý xuất kho

### 2.6.7. Use-case quản lý chi nhánh



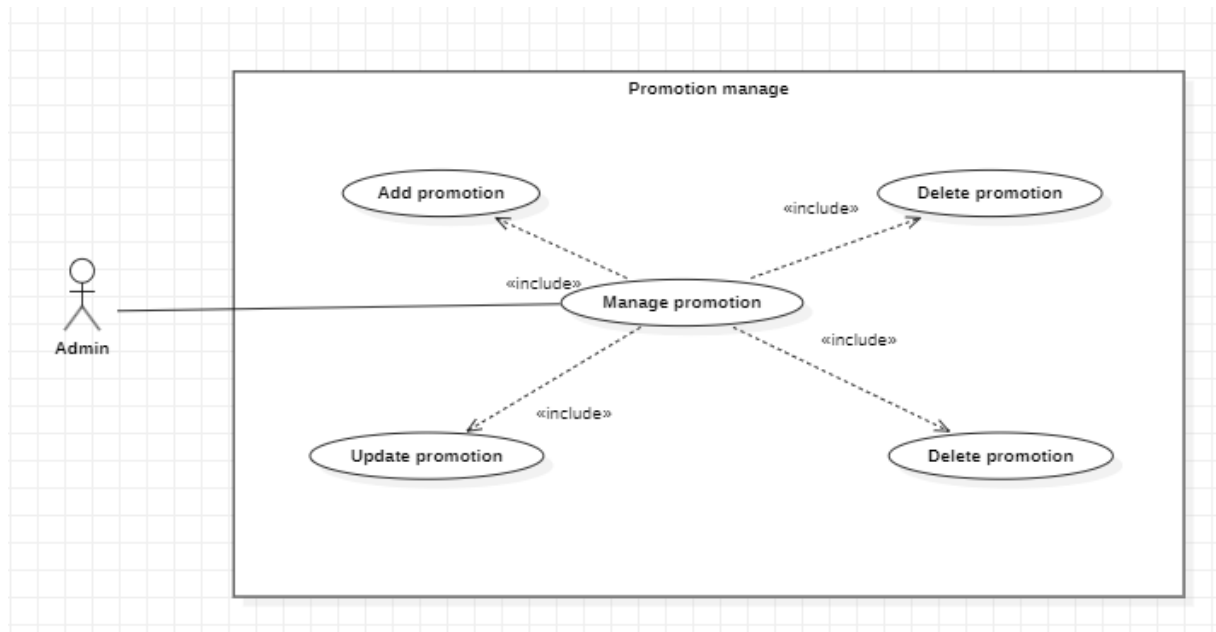
Hình 2.9. Mô hình quản lý chi nhánh

### 2.6.8. Use-case quản lý khách hàng



Hình 2.10. Mô hình quản lý khách hàng

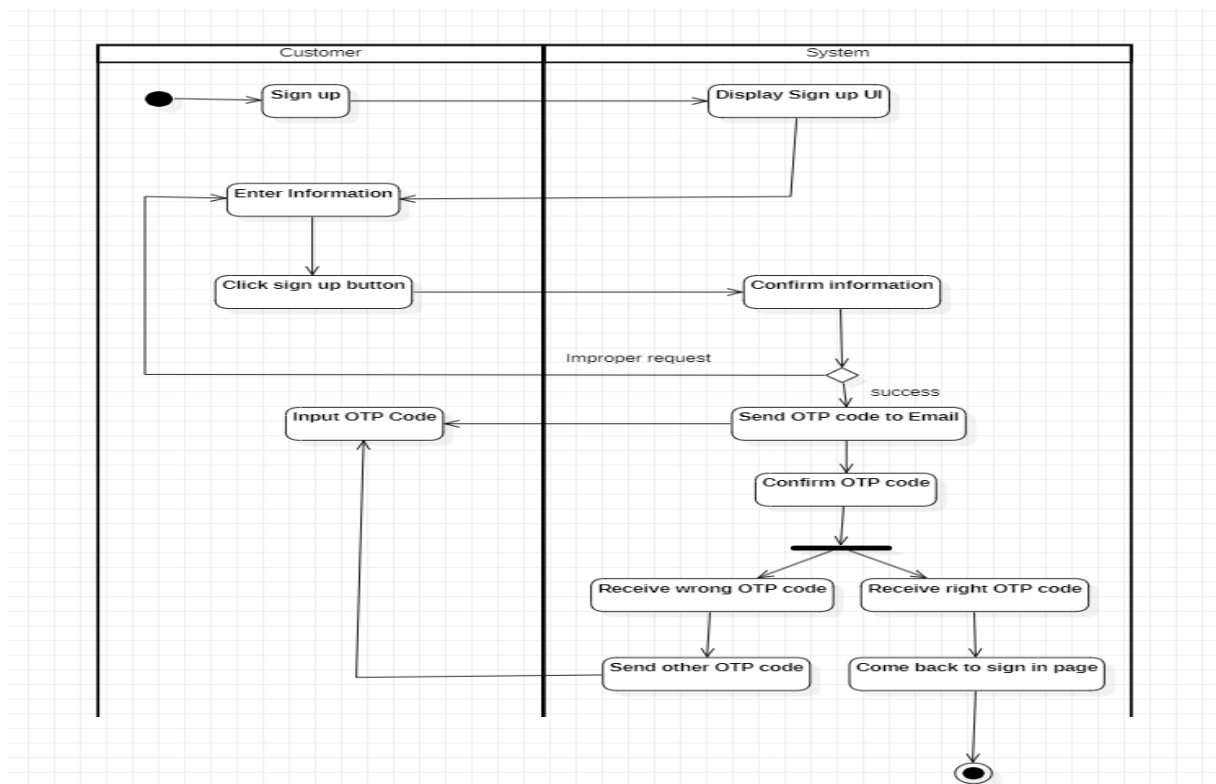
### 2.6.9. Use-case quản lý khuyến mãi



Hình 2.11. Quản lý khuyến mãi

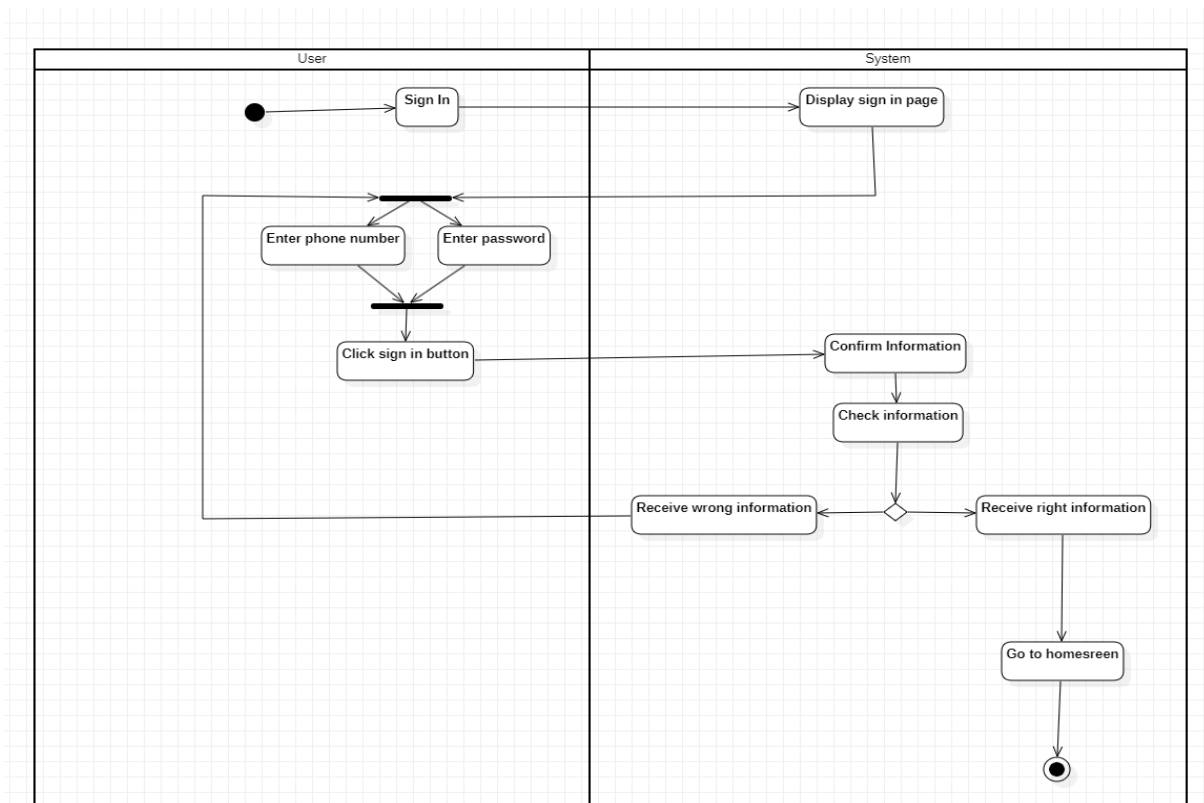
## 2.7. Activity Diagram

### 2.7.1. Chức năng đăng ký



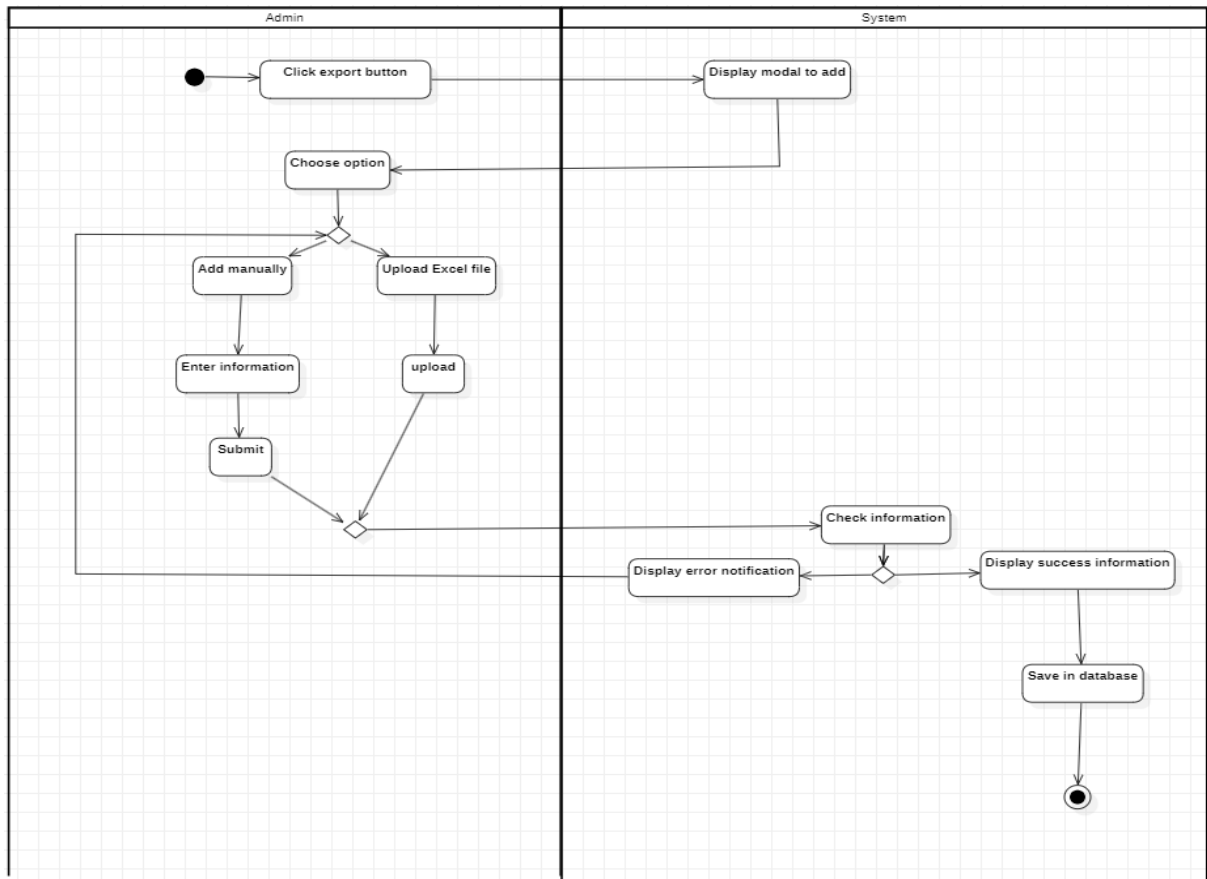
Hình 2.12. Hệ thống đăng ký

### 2.7.2. Chức năng đăng nhập



Hình 2.13. Hệ thống đăng nhập

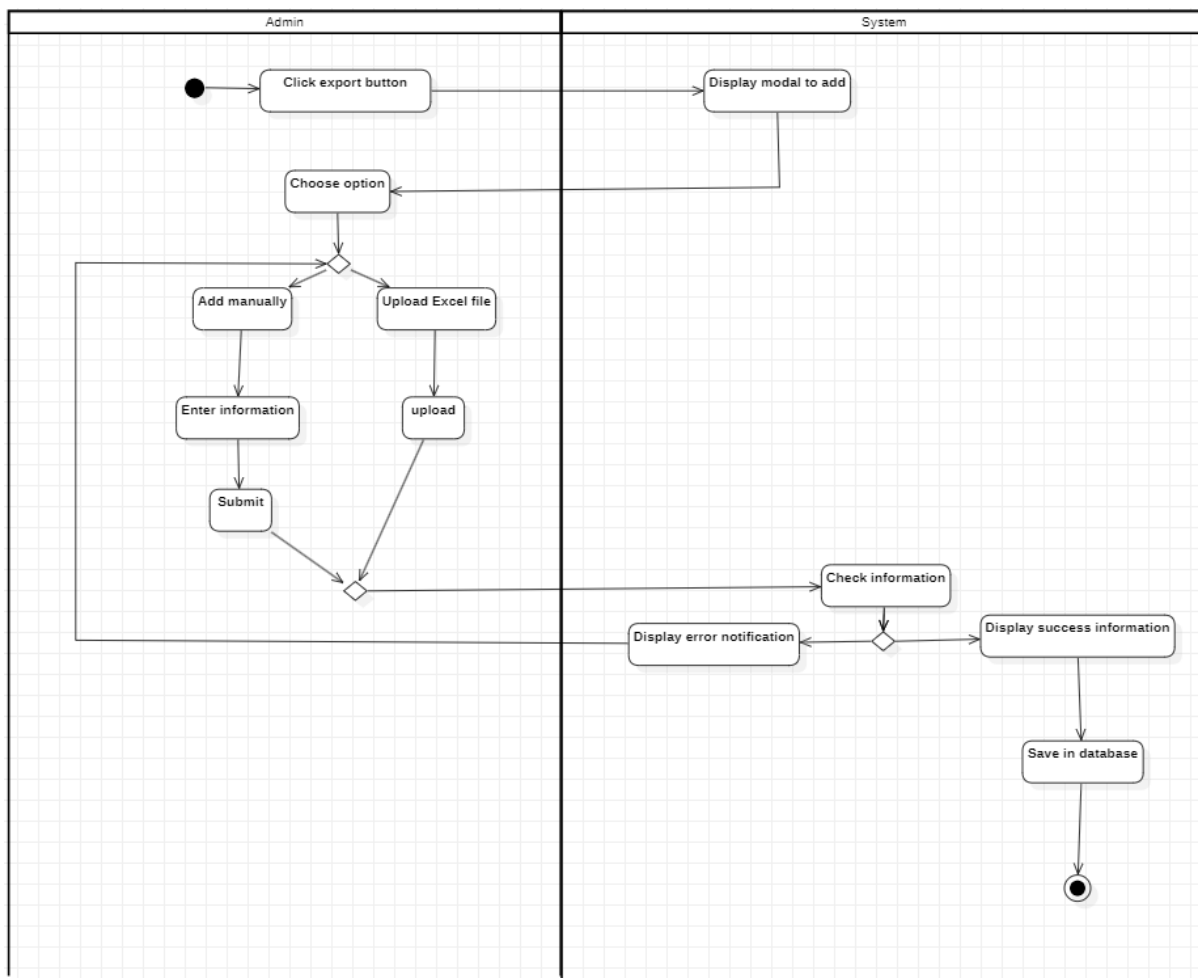
### 2.7.3. Chức năng thêm sản phẩm



Hình 2.14. Hệ thống thêm sản phẩm

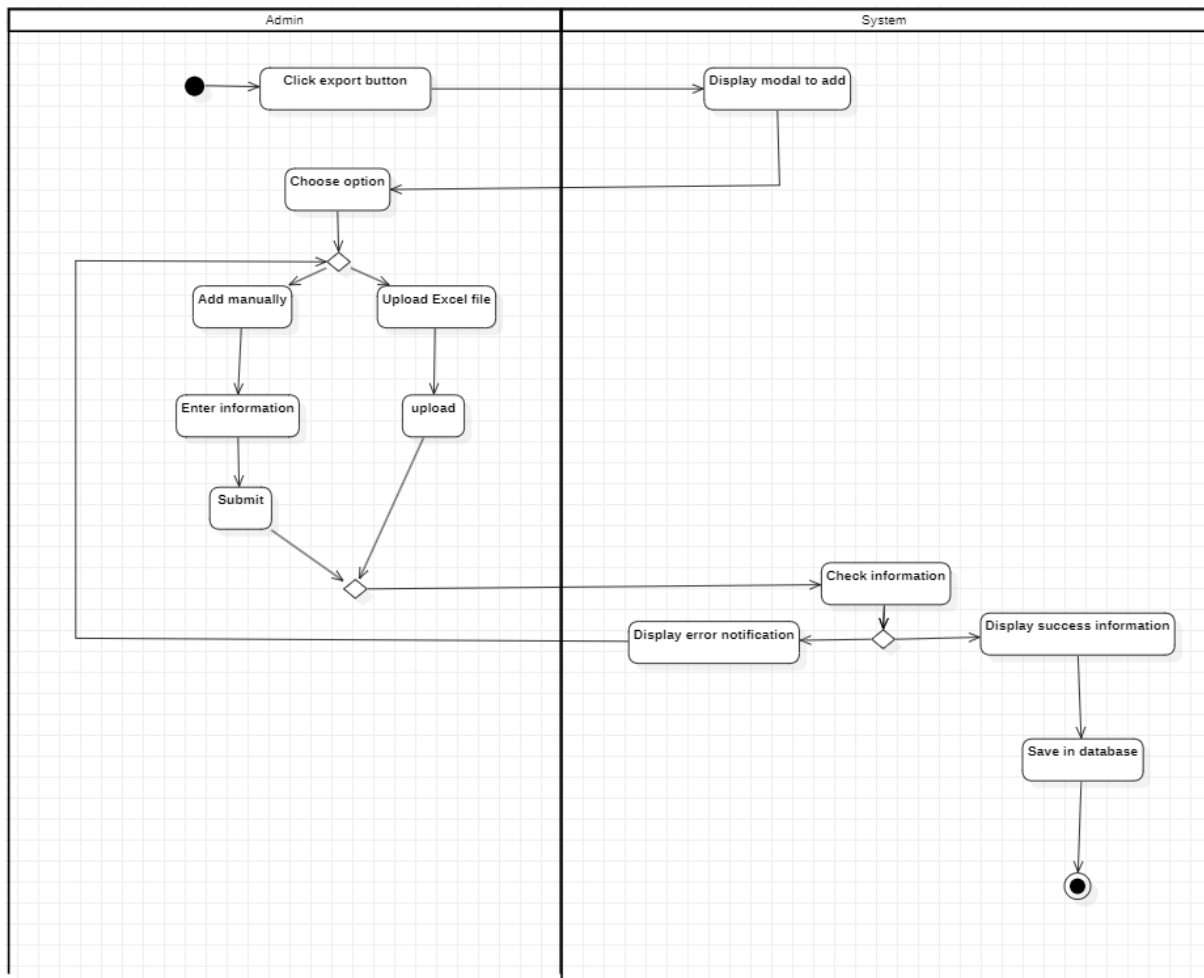


#### 2.7.4. Chức năng nhập lô hàng



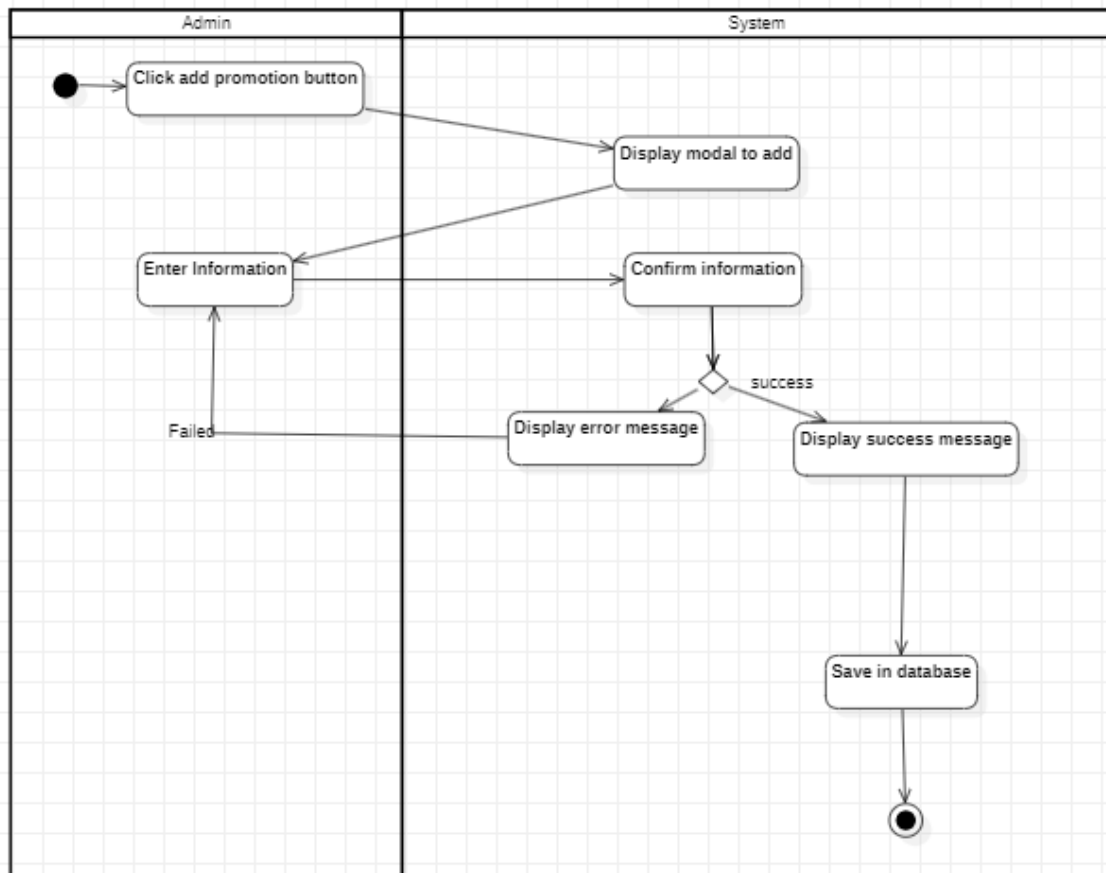
Hình 2.15. Hệ thống nhập lô hàng

### 2.7.5. Chức năng phân phối hàng cho các chi nhánh



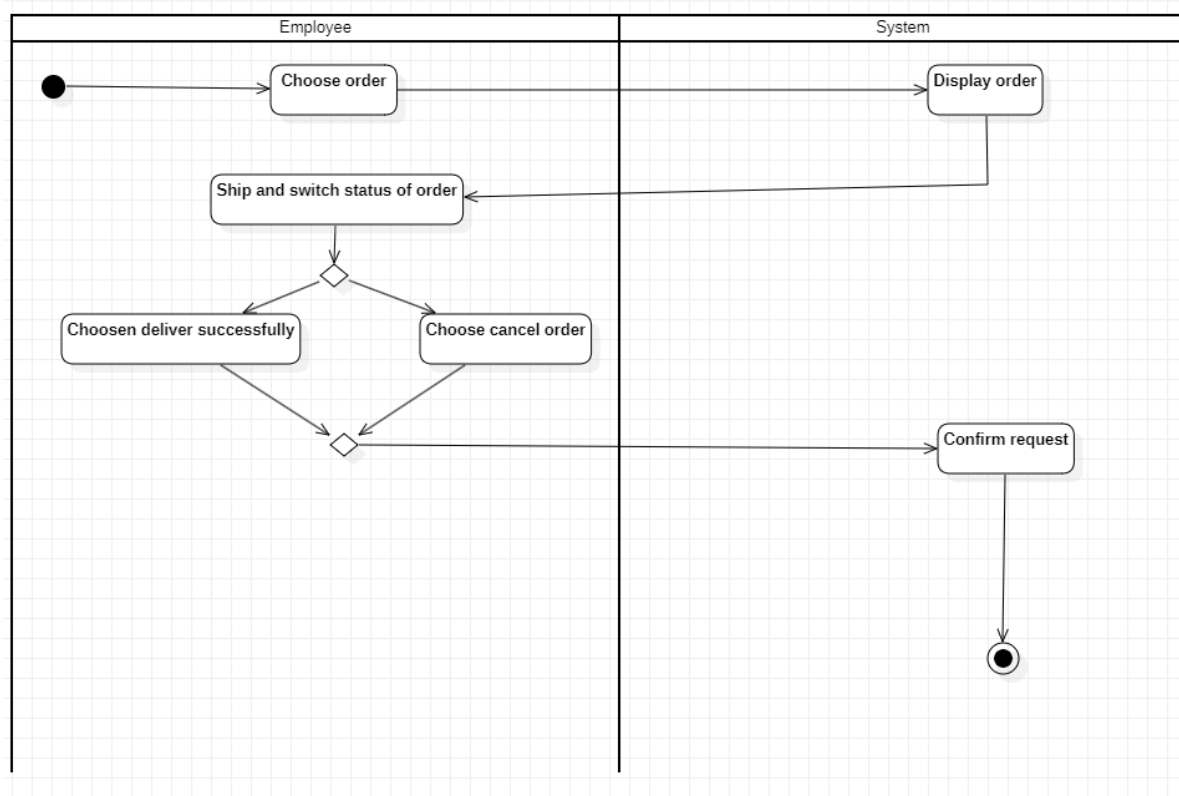
Hình 2.16. Hệ thống phân phối hàng cho các chi nhánh

### 2.7.6. Chức năng quản lý khuyến mãi



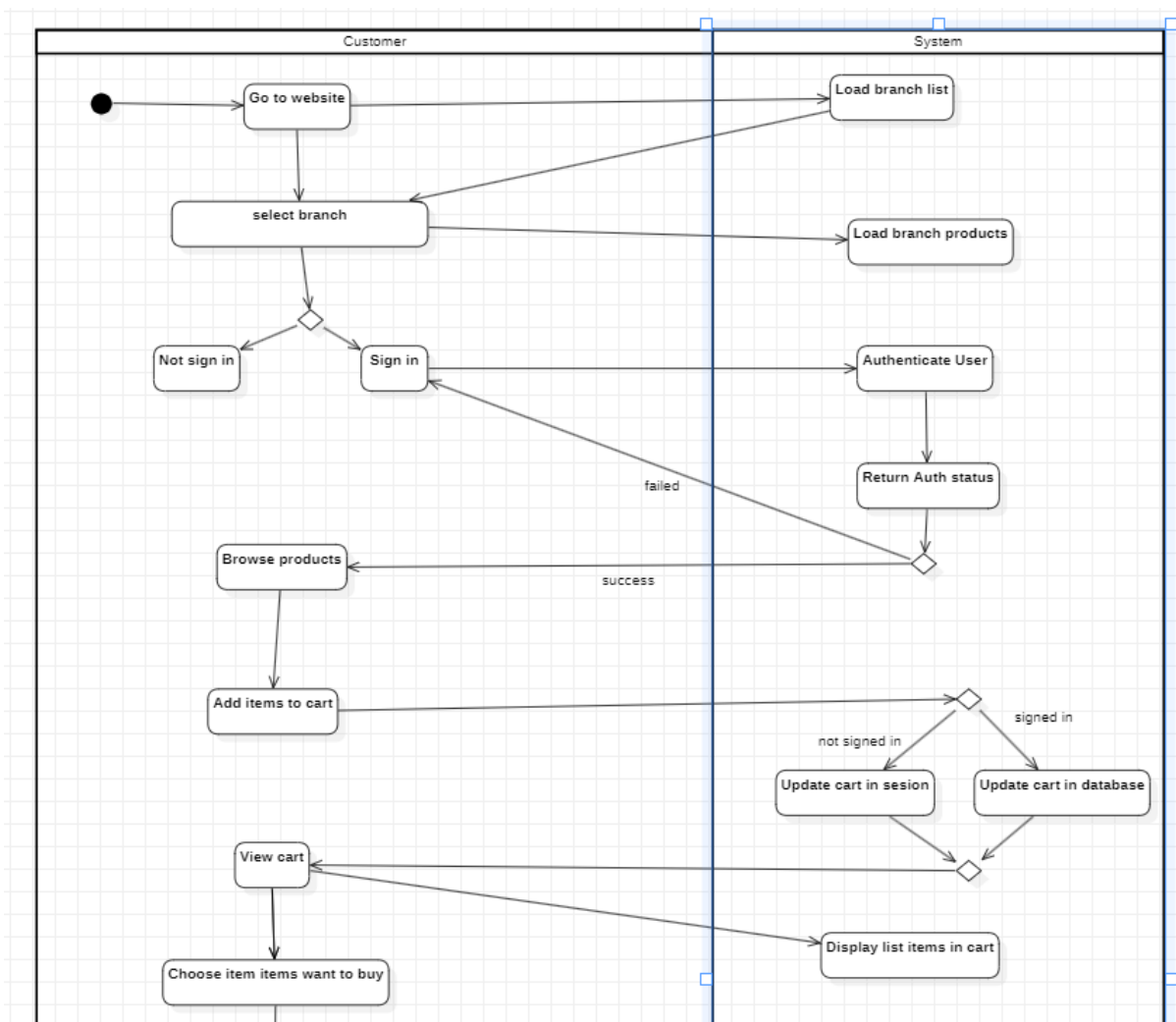
Hình 2.17. Hệ thống thêm mã khuyến mãi

### 2.7.7. Chức năng giao đơn hàng



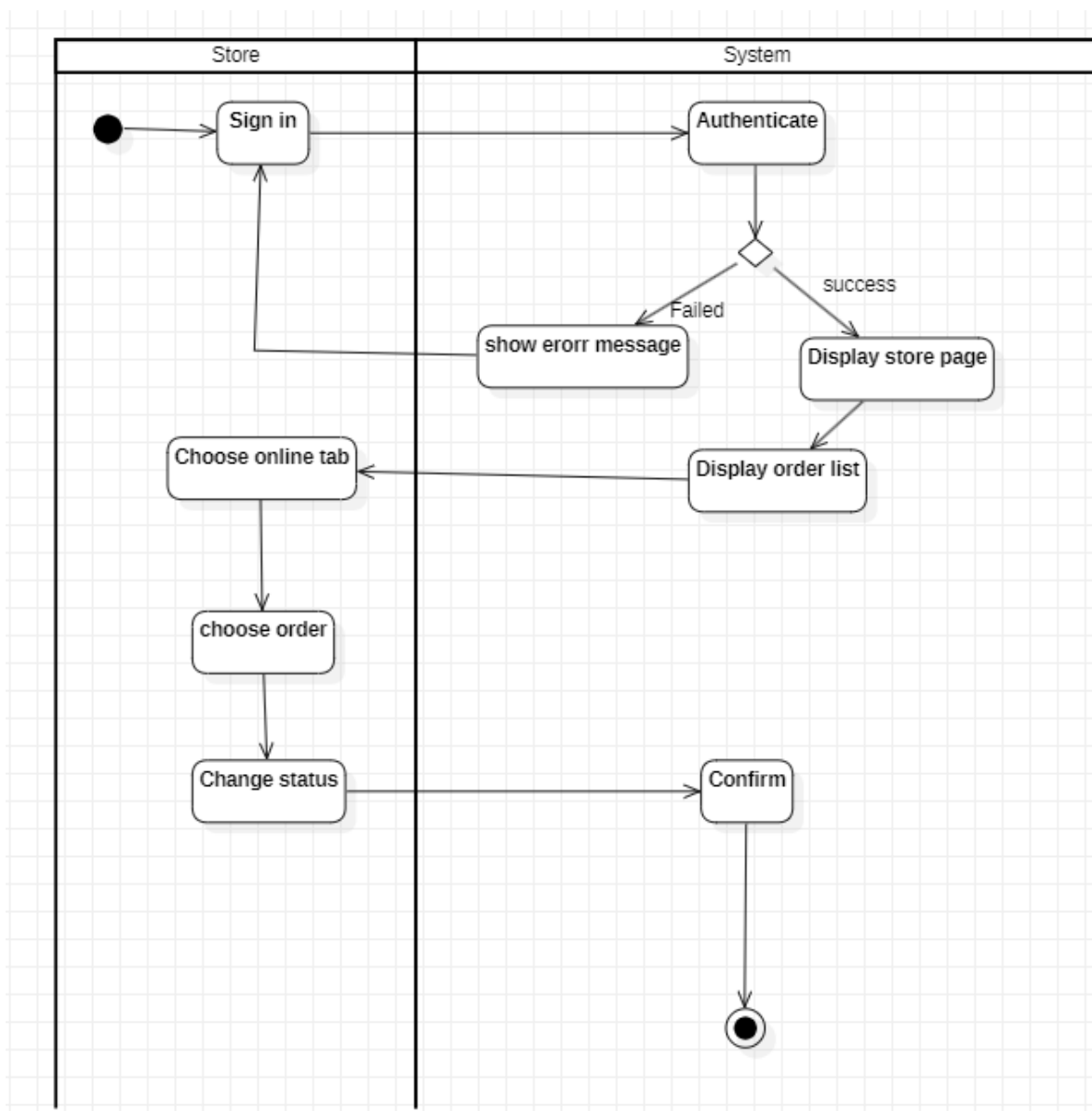
Hình 2.18. Hệ thống giao đơn hàng

### 2.7.8. Chức năng mua hàng



Hình 2.19. Hệ thống mua hàng

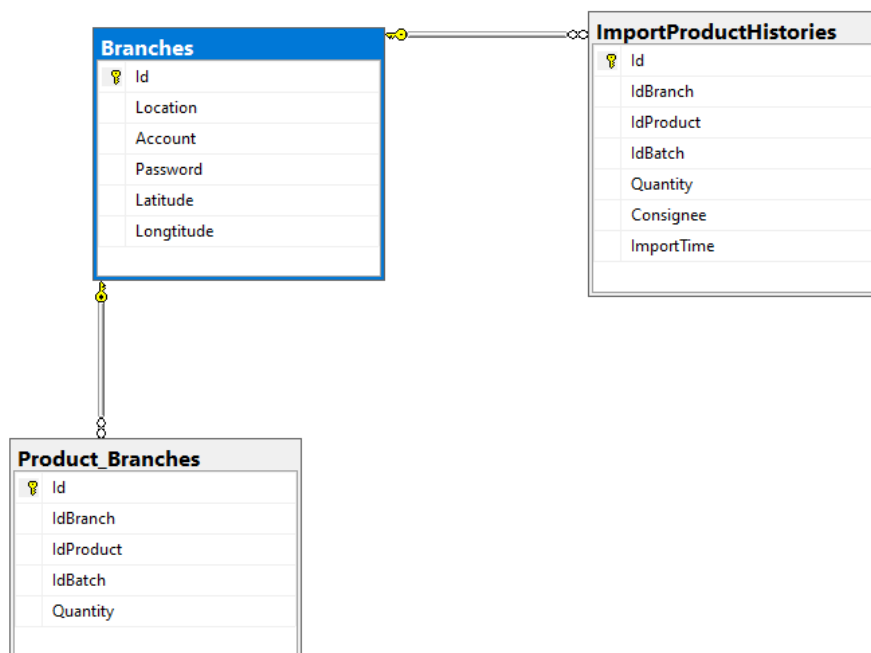
### 2.7.9. Chức năng xác nhận đơn hàng



Hình 2.20. Hệ thống xác nhận mua hàng

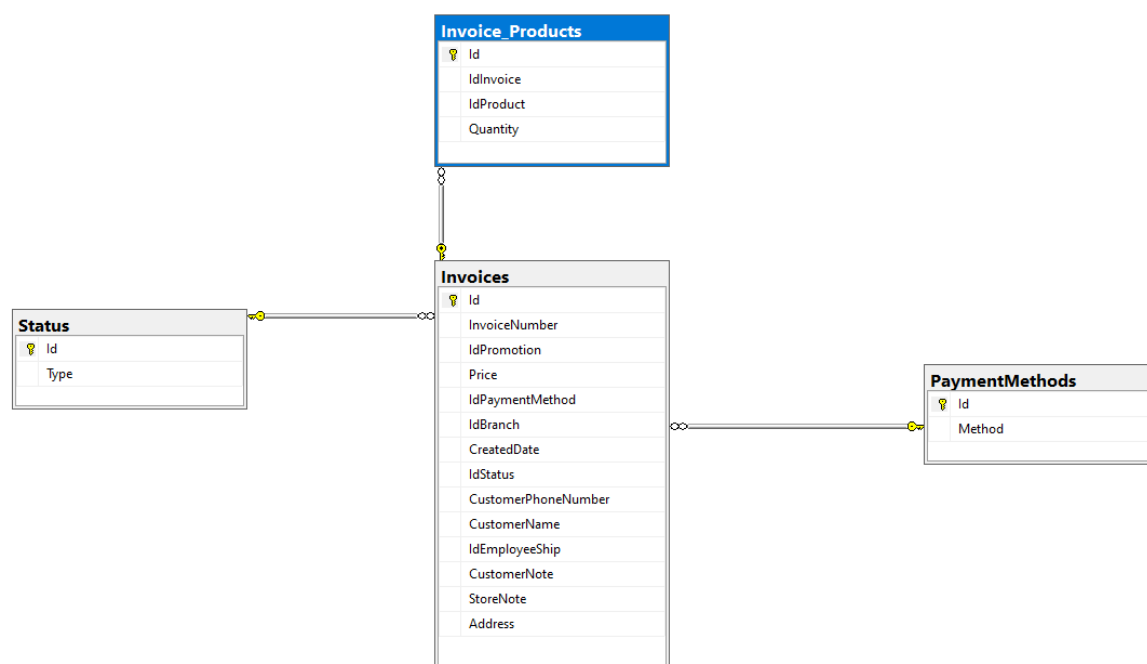
## 2.8. Sơ đồ Database

### 2.8.1. Chi nhánh Database



Hình 2.21. Cơ sở dữ liệu “Chi nhánh”

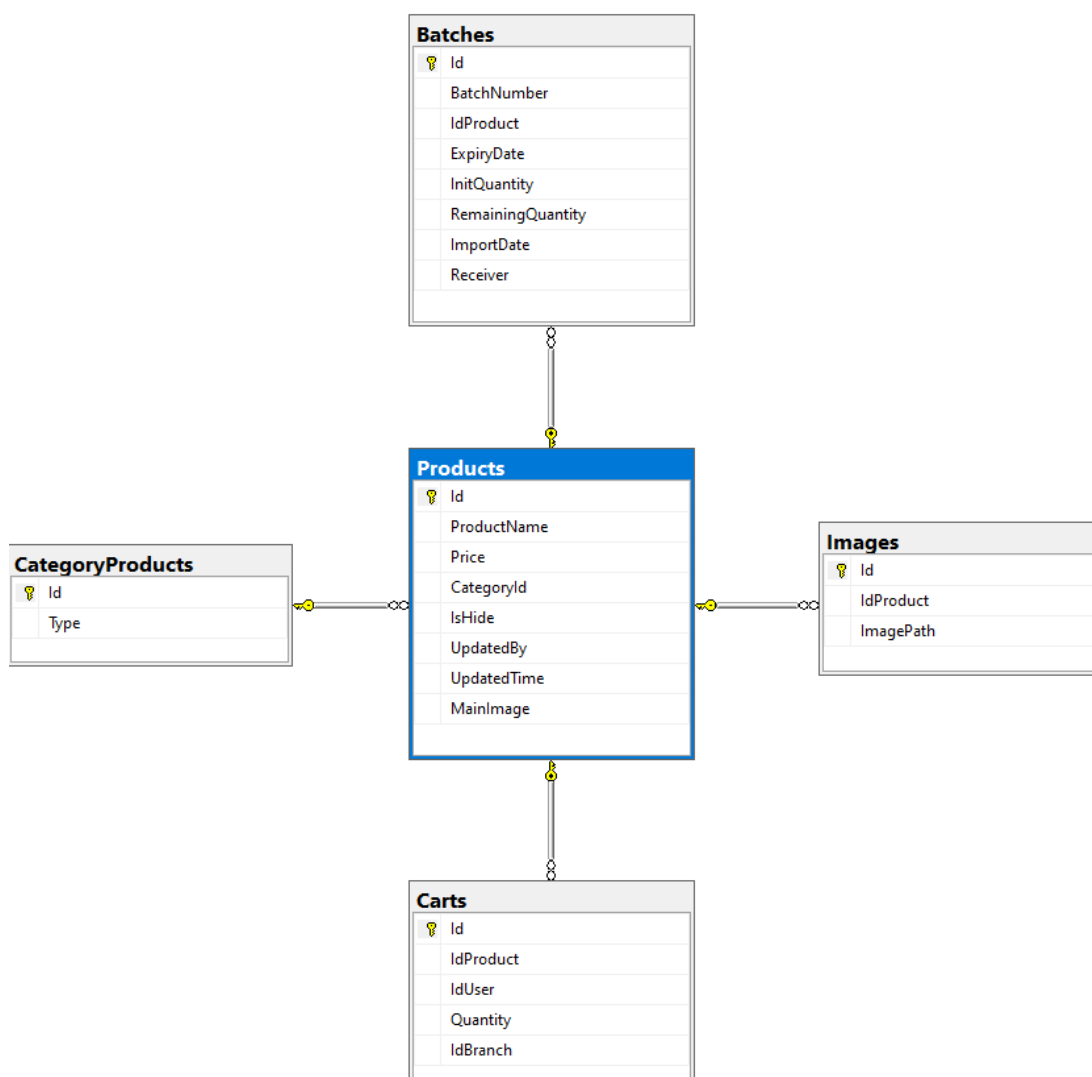
### 2.8.2. Hóa đơn Database



Hình 2.22. Cơ sở dữ liệu “Hóa đơn”

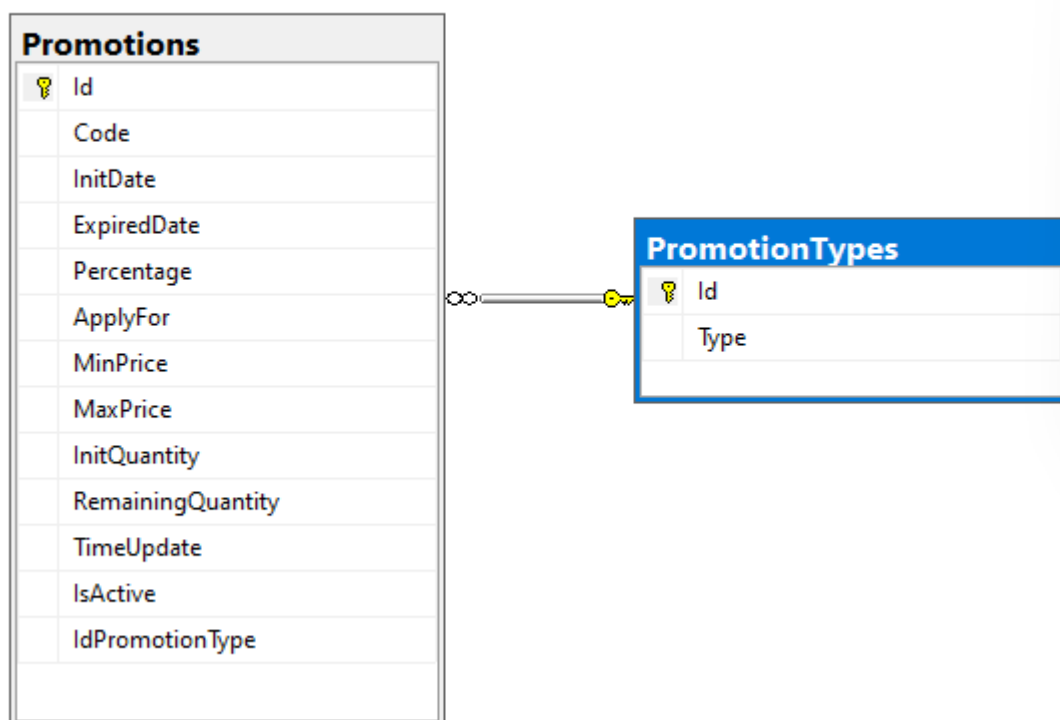


### 2.8.3. Sản phẩm Database



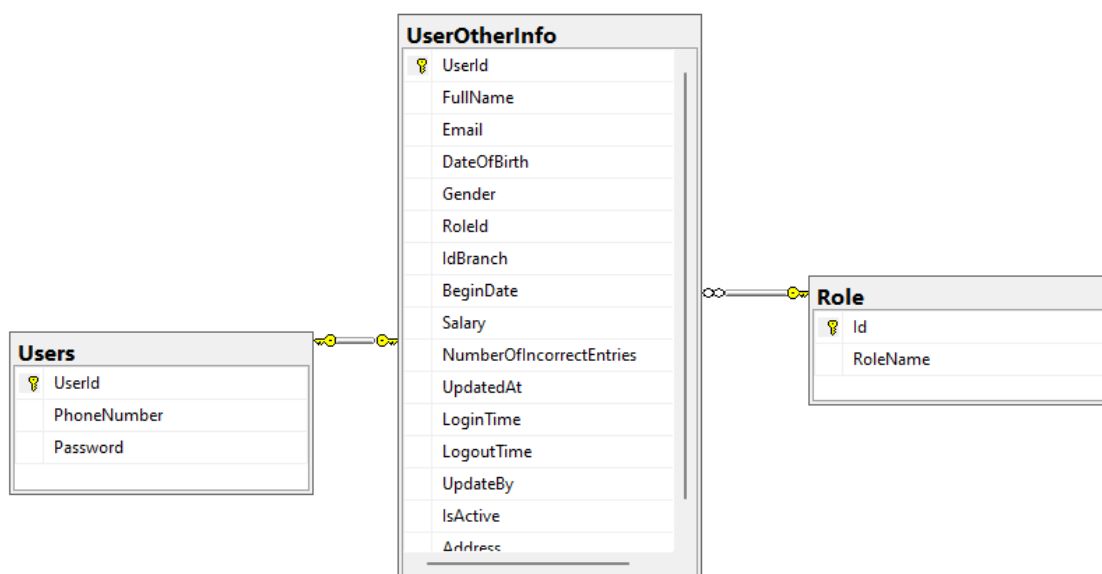
Hình 2.23. Cơ sở dữ liệu “Sản phẩm”

### 2.8.4. Giảm giá Database



Hình 2.24. Cơ sở dữ liệu “Giảm giá”

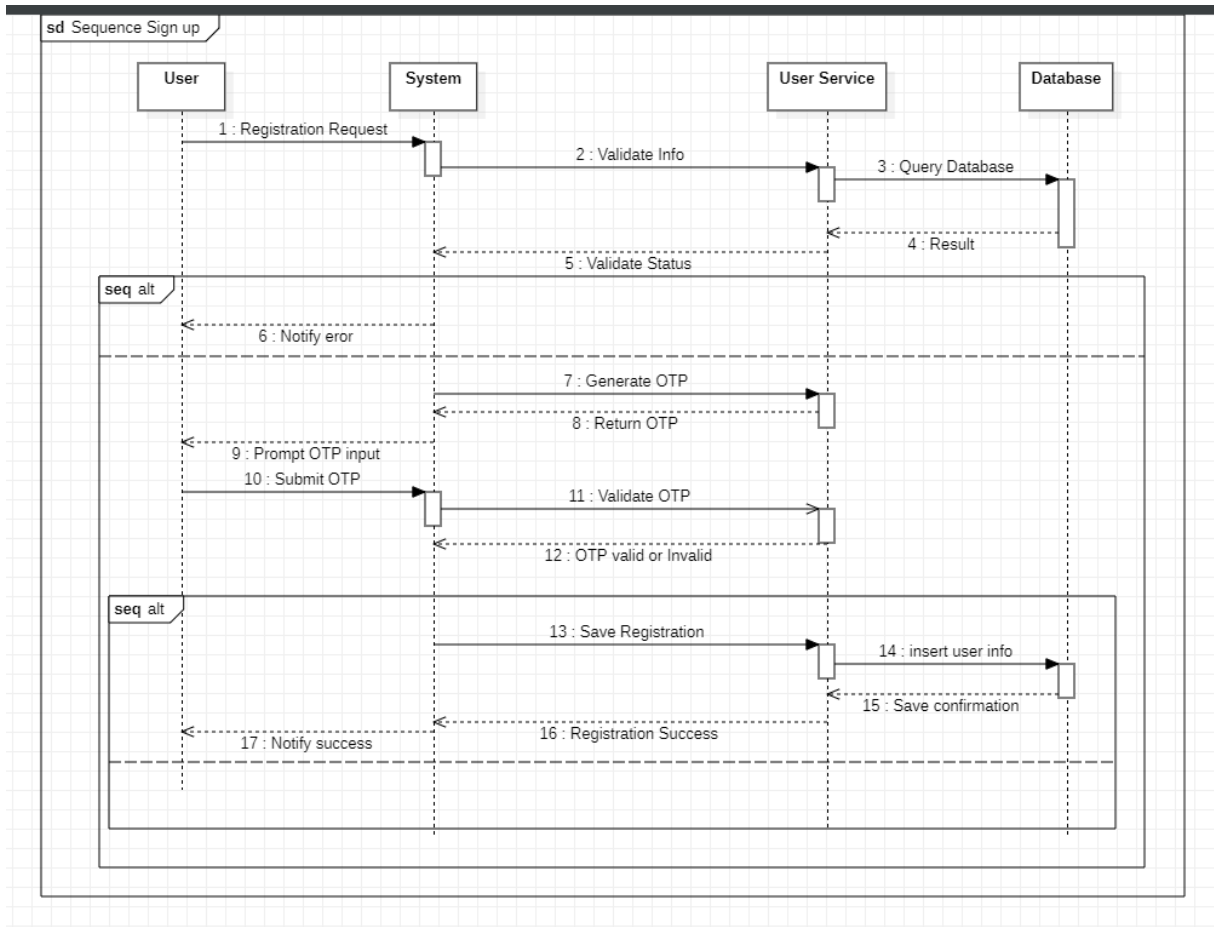
#### 2.8.5. Người dùng Database



Hình 2.25. Cơ sở dữ liệu “Người dùng”

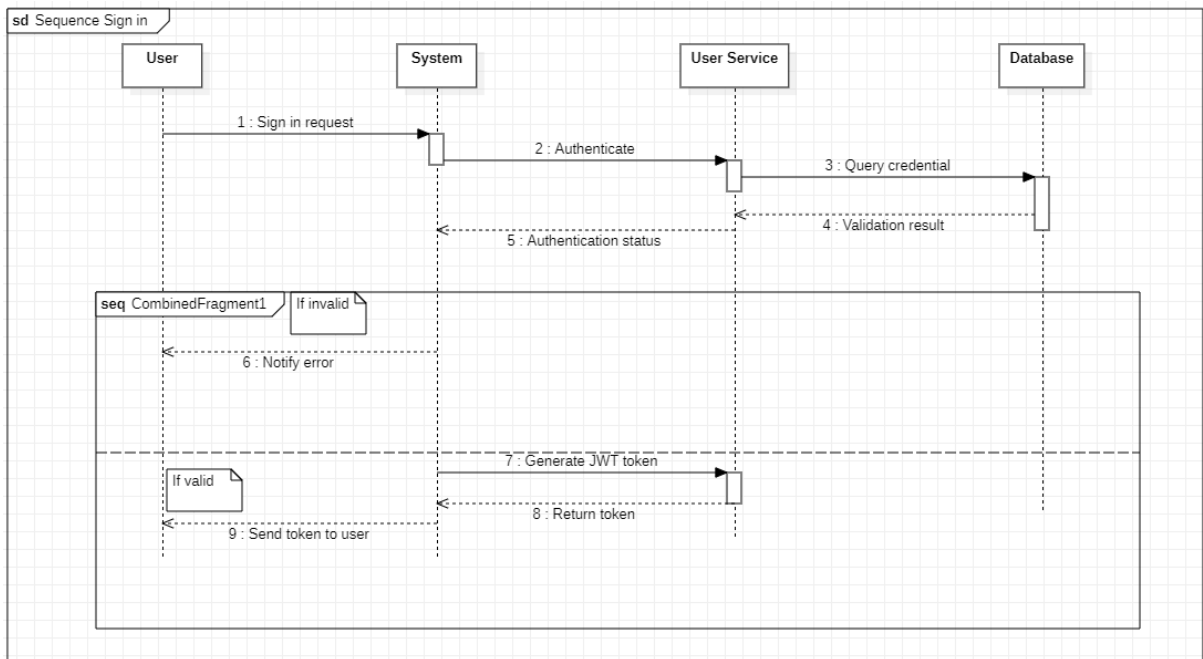
## 2.9. Mô hình Sequence Diagram

### 2.9.1. Sequence Diagram chức năng đăng ký



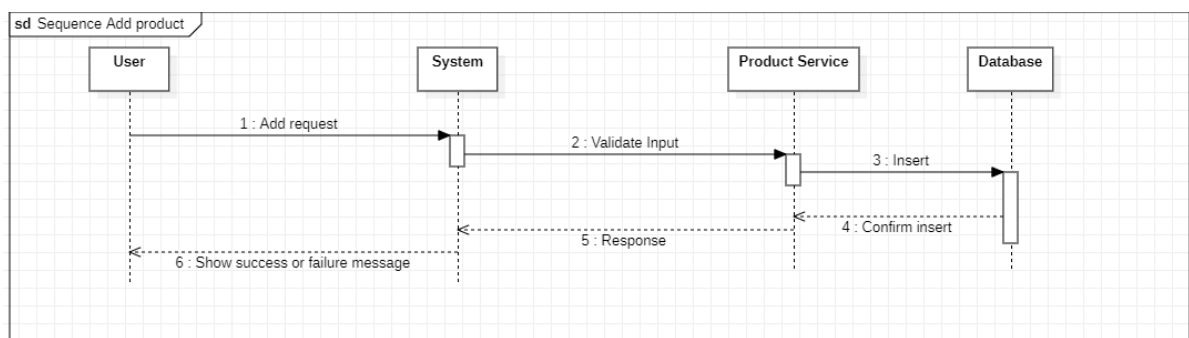
Hình 2.26. Mô hình Sequence Diagram chức năng đăng ký

### 2.9.2. Sequence Diagram chức năng đăng nhập



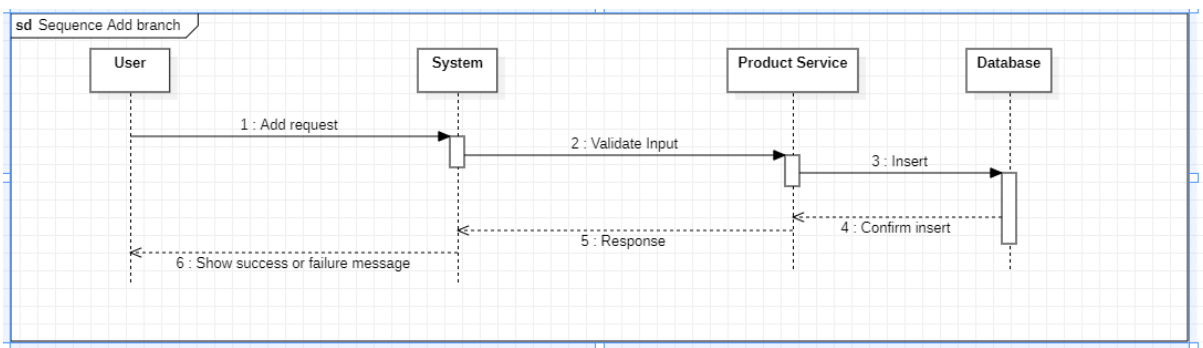
Hình 2.27. Mô hình Sequence Diagram chức năng đăng nhập

### 2.9.3. Sequence Diagram chức năng thêm sản phẩm



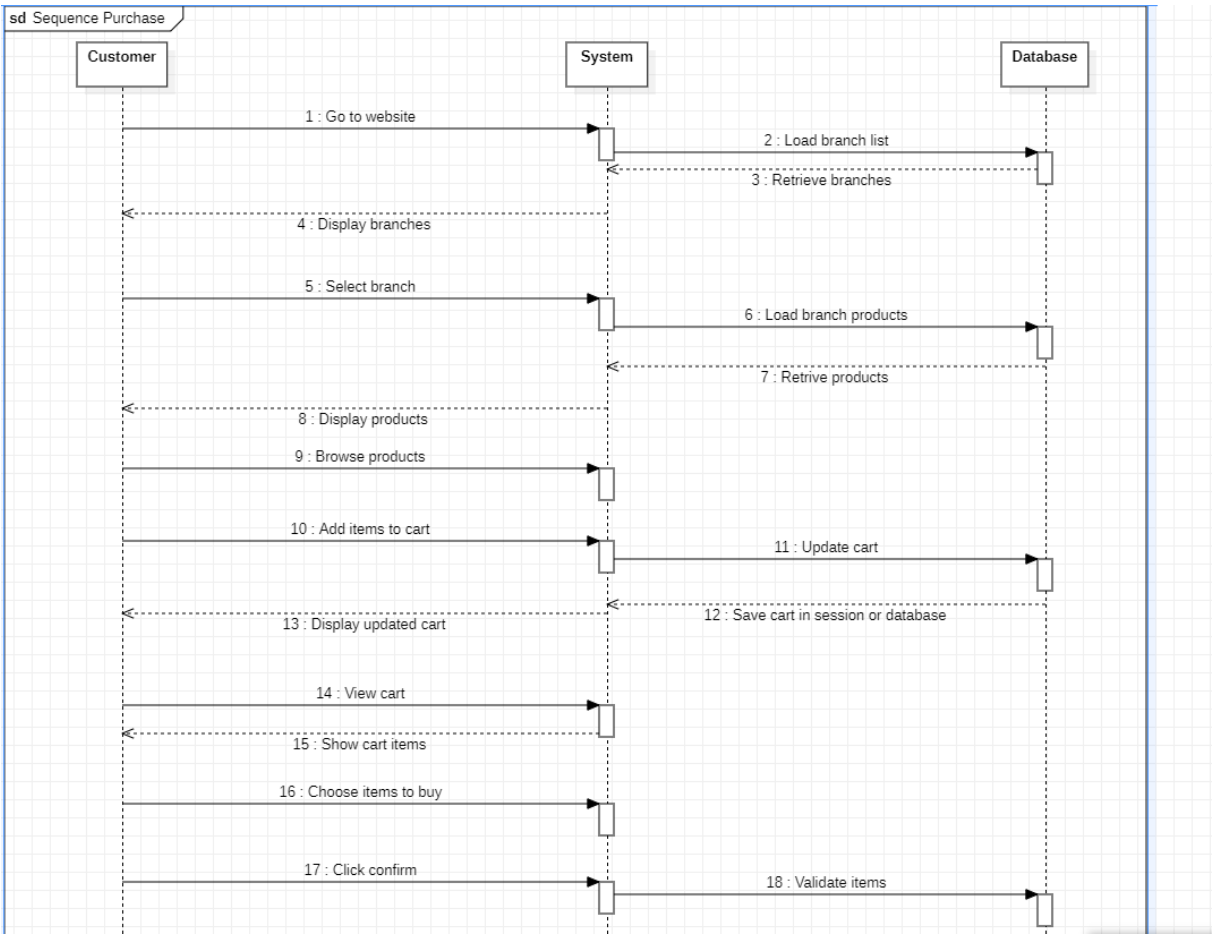
Hình 2.28. Mô hình Sequence Diagram chức năng thêm sản phẩm

### 2.9.4. Sequence Diagram chức năng thêm chi nhánh



Hình 2.29. Mô hình Sequence Diagram chức năng thêm chi nhánh

2.9.5. Sequence Diagram chức năng mua hàng



Hình 2.30. Mô hình Sequence Diagram chức năng mua hàng

2.10. Danh sách thực thể

- Thực thể Branches: chứa thông tin các chi nhánh

Bảng 2.31. Thực thể Branches

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
Account	Nvarchar(MAX)	<input type="checkbox"/>
Password	Nvarchar(61)	<input type="checkbox"/>
Latitude	Nvarchar(20)	<input type="checkbox"/>
Longtitude	Nvarchar(20)	<input type="checkbox"/>

- Thực thể Product\_Branches: chứa các sản phẩm ở từng các chi nhánh khác nhau.

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
IdBranch	int	<input type="checkbox"/>
IdProduct	int	<input type="checkbox"/>
IdBatch	int	<input type="checkbox"/>
Quantity	int	<input type="checkbox"/>

Hình 2.32. Thực thể Product\_Branches

- Thực thể ImportProductHistory: Chứa lịch sử nhập hàng của từng chi nhánh

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
IdBranch	int	<input type="checkbox"/>
IdProduct	int	<input type="checkbox"/>
IdBatch	int	<input type="checkbox"/>
Quantity	int	<input type="checkbox"/>
Consignee	Nvarchar(20)	<input type="checkbox"/>
ImportTime	Datetime(7)	<input type="checkbox"/>

Hình 2.33. Thực thể ImportProductHistory

- Thực thể Invoices: Chứa cách giao dịch hóa đơn

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
InvoiceNumber	Nvarchar(12)	<input type="checkbox"/>
IdPromotion	int	<input checked="" type="checkbox"/>
Price	int	<input type="checkbox"/>
IdPaymentMethod	int	<input checked="" type="checkbox"/>
IdBranch	int	<input type="checkbox"/>
CreatedDate	Datetime2(7)	<input type="checkbox"/>
IdStatus	smallint	<input type="checkbox"/>
CustomerPhoneNumber	Nvarchar(10)	<input checked="" type="checkbox"/>
CustomerName	Nvarchar(50)	<input checked="" type="checkbox"/>
IdEmployeeShip	int	<input checked="" type="checkbox"/>
CustomerNote	Nvarchar(100)	<input checked="" type="checkbox"/>
StoreNote	Nvarchar(100)	<input checked="" type="checkbox"/>
Address	Nvarchar(200)	<input checked="" type="checkbox"/>

Hình 2.34. Thực thể Invoices

- Thực thể Invoice\_Products: Chứa các chi tiết các món hàng cũng như số lượng trong hóa đơn đó.

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
IdInvoice	int	<input type="checkbox"/>
IdProduct	int	<input type="checkbox"/>
Quantity	int	<input type="checkbox"/>

Hình 2.35. Thực thể Invoice\_Products

- Thực thể PaymentMethods: chứa các cách thức mua hàng mua hàng.

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
Method	Nvarchar(20)	<input type="checkbox"/>

Hình 2.36. Thực thể PaymentMethods

- Thực thể Status: Hiện thị các trạng thái hiện tại của đơn hàng.

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>



Type	Nvarchar(20)	<input type="checkbox"/>
------	--------------	--------------------------

Hình 2.37. Thực thể Status

- Thực thể Products: Chứa thông tin các sản phẩm

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
ProductName	Nvarchar(30)	<input type="checkbox"/>
CategoryId	tinyInt	<input type="checkbox"/>
IsHide	bit	<input type="checkbox"/>
UpdatedBy	int	<input type="checkbox"/>
UpdateTime	Datetime2(7)	<input type="checkbox"/>
MainImage	Nvarchar(100)	<input checked="" type="checkbox"/>

Hình 2.38. Thực thể Products

- Thực thể Batches: Chứa các lô hàng của loại hàng hóa đó

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
BatchNumber	Nvarchar(10)	<input type="checkbox"/>
ExpiryDate	date	<input type="checkbox"/>
InitQuantity	int	<input type="checkbox"/>

RemainingQuantity	int	<input type="checkbox"/>
ImportDate	Datetime2(7)	<input type="checkbox"/>
Receiver	Nvarchar(30)	<input type="checkbox"/>

Hình 2.39. Thực thể Batches

- Thực thể Images: Chứa các hình ảnh của các sản phẩm

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
IdProduct	int	<input type="checkbox"/>
ImagePath	Nvarchar(100)	<input type="checkbox"/>

Hình 2.40. Thực thể Images

- Thực thể CategoryProducts: Chứa các phân loại của sản phẩm đó

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
Type	Nvarchar(10)	<input type="checkbox"/>

Hình 2.41. Thực thể CategoryProducts

- Thực thể Carts: Chứa các sản phẩm trong giỏ hàng

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
IdProduct	int	<input type="checkbox"/>
IdUser	int	<input type="checkbox"/>
Quantity	smallint	<input type="checkbox"/>
IdBranch	int	<input type="checkbox"/>

Hình 2.42. Thực thể Carts

- Thực thể Promotions: Chứa các thông tin về mã giảm giá

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
Code	Nvarchar(10)	<input type="checkbox"/>
InitDate	date	<input type="checkbox"/>
ExpiredDate	date	<input type="checkbox"/>
Percentage	tinyint	<input type="checkbox"/>
ApplyFor	int	<input checked="" type="checkbox"/>
MinPrice	int	<input type="checkbox"/>
MaxPrice	int	<input type="checkbox"/>
InitQuantity	int	<input type="checkbox"/>
RemainingQuantity	int	<input type="checkbox"/>
TimeUpdate	Datetime2	<input type="checkbox"/>

IsActive	bit	<input type="checkbox"/>
IdPromotion	tinyint	

Hình 2.43. Thực thể Promotions

- Thực thể PromotionType: Chứa phân loại mã giảm giá

Column Name	Data Type	Allow Nulls
Id	int	<input type="checkbox"/>
Type	Nvarchar(30)	<input type="checkbox"/>

Hình 2.44. Thực thể PromotionType

- Thực thể Users: Chứa thông tin đăng nhập của người dùng

Column Name	Data Type	Allow Nulls
UserId	int	<input type="checkbox"/>
PhoneNumber	Nvarchar(10)	<input type="checkbox"/>
Password	Nvarchar(61)	<input checked="" type="checkbox"/>

Hình 2.45. Thực thể Users

- Thực thể UserOtherInfo: Chứa thông tin cá nhân của người dùng

Column Name	Data Type	Allow Nulls
UserId	int	<input type="checkbox"/>
FullName	Nvarchar(60)	<input type="checkbox"/>

Email	Nvarchar(30)	<input type="checkbox"/>
DateOfBirth	date	<input type="checkbox"/>
Gender	tinyint	<input type="checkbox"/>
RoleId	tinyint	<input type="checkbox"/>
IdBranch	Nvarchar(4)	<input checked="" type="checkbox"/>
BeginDate	date	<input checked="" type="checkbox"/>
Salary	int	<input checked="" type="checkbox"/>
NumberOfIncorrectEntries	tinyint	<input type="checkbox"/>
UpdatedAt	Datetime2(7)	<input checked="" type="checkbox"/>
LoginTime	Datetime2(7)	<input checked="" type="checkbox"/>
LogoutTime	Datetime2(7)	<input checked="" type="checkbox"/>
UpdatedBy	int	<input type="checkbox"/>
IsActive	bit	<input type="checkbox"/>
Address	Nvarchar(max)	<input type="checkbox"/>

Hình 2.46. Thực thể UserOtherInfo

- Thực thể Role: Chứa vai trò của người dùng

Column Name	Data Type	Allow Nulls
Id	Tinyint	<input type="checkbox"/>
RoleName	Nvarchar(10)	<input type="checkbox"/>

Hình 2.47. Thực thể Role

## CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

### 3.1. Các thành phần chức năng của hệ thống

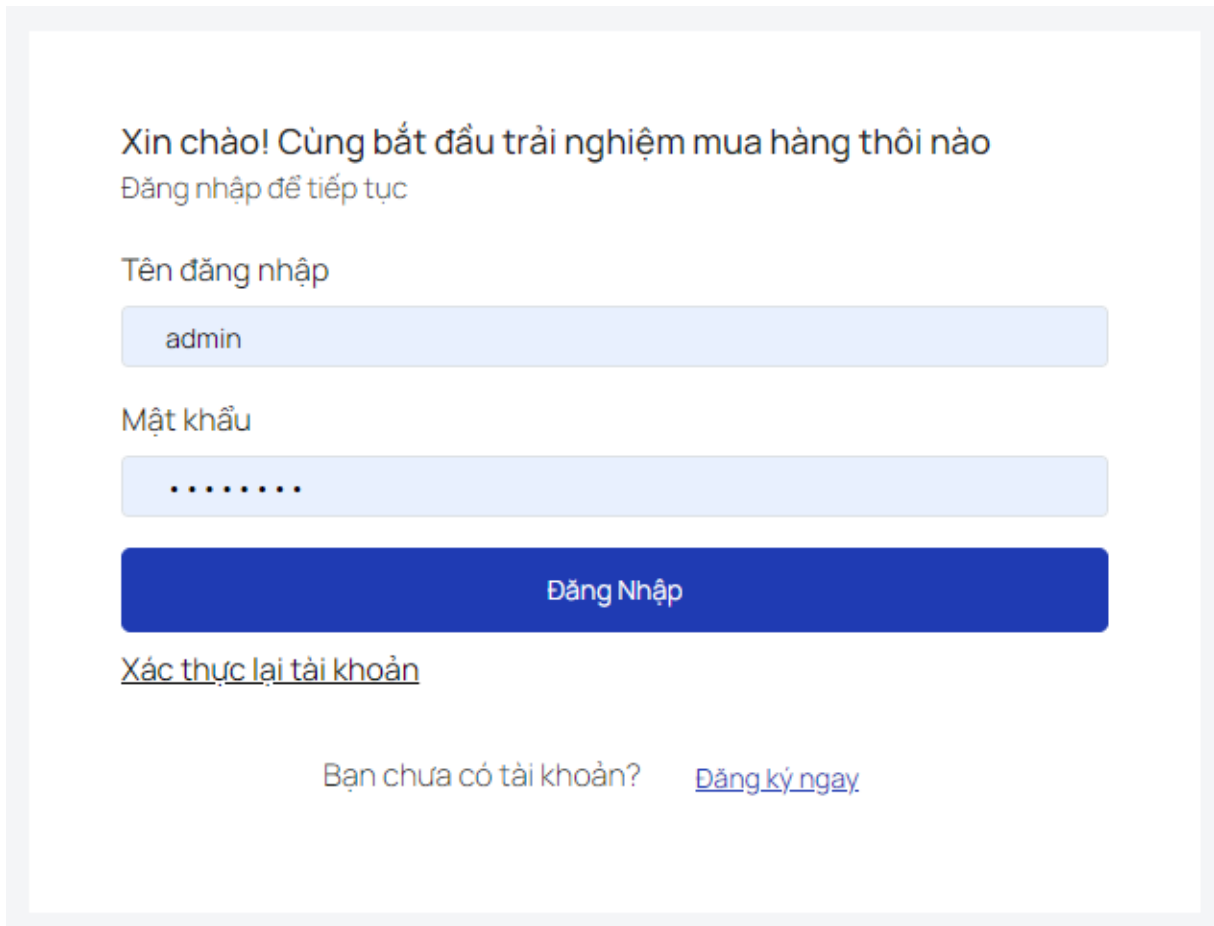
- Thành phần giao diện: Có người dùng chính là Admin, Manager, Employee, Customer, Store.
- Thành phần chức năng: đăng nhập, đăng xuất, thêm sản phẩm, nhập xuất lô hàng, thêm xóa sửa, thống kê doanh thu, bán hàng.
- Thành phần xử lý:
  - + Xử lý xác thực và phân quyền người dùng:
    - \* Đảm bảo người dùng (Admin, Manager, Employee, Customer, Store) được xác thực trước khi truy cập hệ thống.
    - \* Gán quyền truy cập theo vai trò (role-based access control).
  - + Xử lý quản lý sản phẩm:
    - \* Xử lý logic để thêm, xóa, sửa thông tin sản phẩm.
    - \* Kiểm tra dữ liệu nhập vào.
    - \* Tự động kiểm tra mã sản phẩm trùng lặp.
  - + Xử lý nhập xuất hàng hóa:
    - \* Xử lý nhập hàng từ nhà cung cấp (cập nhật tồn kho).
    - \* Xử lý xuất hàng theo yêu cầu (cập nhật thông tin vận chuyển, hóa đơn xuất).
  - + Xử lý bán hàng:
    - \* Tính toán hóa đơn tự động khi khách hàng thêm sản phẩm vào giỏ.
    - \* Kiểm tra tình trạng kho hàng trước khi bán.
    - \* Áp dụng mã giảm giá hoặc ưu đãi khi cần.
  - + Xử lý quản lý giỏ hàng:
    - \* Thêm sản phẩm vào giỏ hàng.
    - \* Xóa sản phẩm khỏi giỏ hàng.
    - \* Cập nhật số lượng trong giỏ hàng theo yêu cầu.
  - + Xử lý giao dịch thanh toán:
    - \* Xác thực phương thức thanh toán
  - + Xử lý thống kê và báo cáo:

- \* Tạo báo cáo doanh thu
- \* Báo cáo lợi nhuận theo chi nhánh
- + Xử lý quản lý khách hàng:
  - \* Lưu thông tin khách hàng (thông tin cá nhân, lịch sử mua hàng).
  - \* Cập nhật thông tin tài khoản khách hàng.
- + Xử lý thông báo:
  - \* Gửi thông báo qua email
- + Xử lý tích hợp với hệ thống kho:
  - \* Đồng bộ thông tin kho hàng với các chi nhánh khác.
  - \* Kiểm tra tồn kho khi thực hiện các giao dịch bán hàng.
- + Xử lý quản lý chi nhánh:
  - \* Quản lý thông tin từng chi nhánh (địa chỉ, thông tin liên lạc, tồn kho).
  - \* Cập nhật số lượng hàng hóa tại các chi nhánh.
- + Xử lý quản lý nhân viên:
  - \* Thêm, xóa, chỉnh sửa thông tin nhân viên.
  - \* Gán quyền hạn cho nhân viên theo vai trò (Manager, Employee).
- Môi trường thực nghiệm: Hệ điều hành Windows 11.
- Cấu hình Laptop thực hiện: + AMD Ryzen 5 5600H 16GB Ram

## **3.2.Thiết kế giao diện hệ thống**

### *3.2.1. Giao diện đăng nhập*

Chức năng đăng nhập hoạt động bình thường

A login form with a light gray border. At the top, it says "Xin chào! Cùng bắt đầu trải nghiệm mua hàng thôi nào" followed by "Đăng nhập để tiếp tục". Below this are two input fields: "Tên đăng nhập" with the text "admin" and "Mật khẩu" with masked characters ".....". A blue button labeled "Đăng Nhập" is positioned below the password field. At the bottom, there is a link "Xác thực lại tài khoản" and a text prompt "Bạn chưa có tài khoản?" followed by a link "Đăng ký ngay".

Xin chào! Cùng bắt đầu trải nghiệm mua hàng thôi nào

Đăng nhập để tiếp tục

Tên đăng nhập

admin

Mật khẩu

.....

Đăng Nhập

[Xác thực lại tài khoản](#)

Bạn chưa có tài khoản? [Đăng ký ngay](#)

Hình 3.1. Giao diện đăng nhập

### 3.2.2. Giao diện đăng ký

Chức năng đăng ký khi đăng kí xong sẽ yêu cầu nhập mã OTP được gửi qua Email để xác thực tài khoản. Chức năng đăng ký hoạt động bình thường



Tạo tài khoản mới

Họ và tên

Email

Số điện thoại

Địa chỉ

Mật khẩu

Xác nhận mật khẩu

Ngày sinh

mm/dd/yyyy

Giới tính

Chọn giới tính

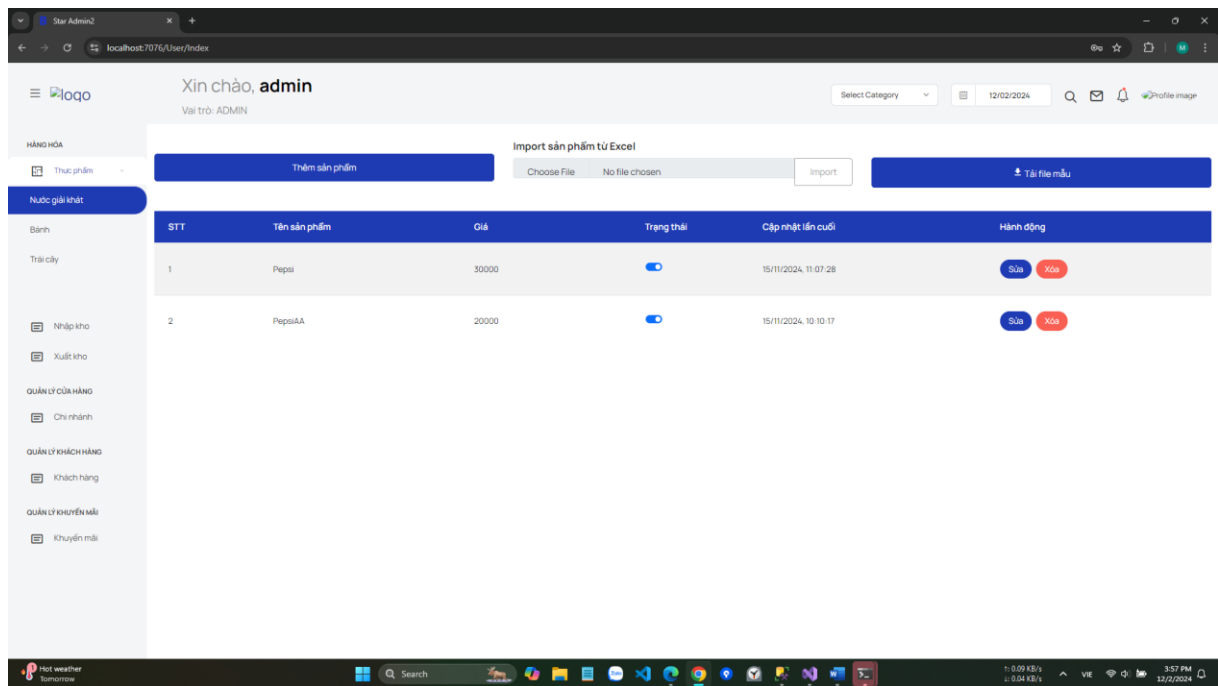
Đóng

Đăng ký

Hình 3.2. Giao diện đăng ký

### 3.2.3. Giao diện quản lý sản phẩm

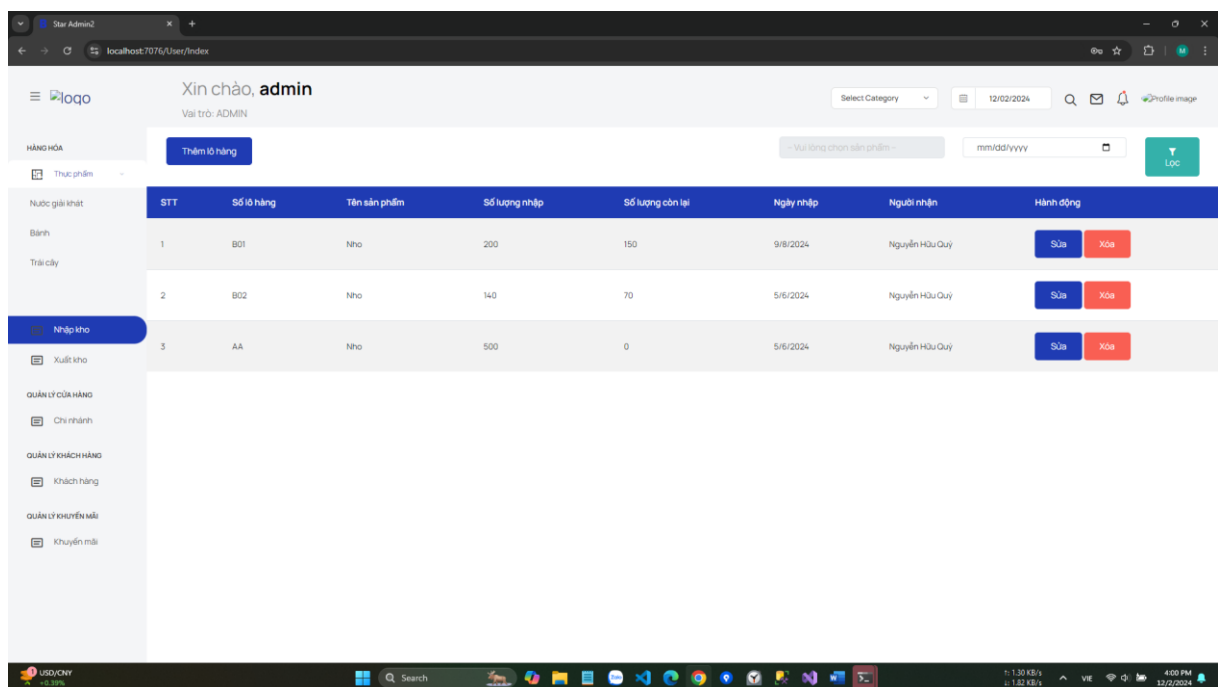
Chức năng quản lý sản phẩm hoạt động ổn định, đã test các chức năng như thêm sản phẩm bằng file Excel, nếu dòng nào sai thông tin hoặc không rõ thì sẽ hiện báo rõ vị trí sai đó.



Hình 3.3. Giao diện quản lý sản phẩm

### 3.2.4. Giao diện quản lý nhập kho

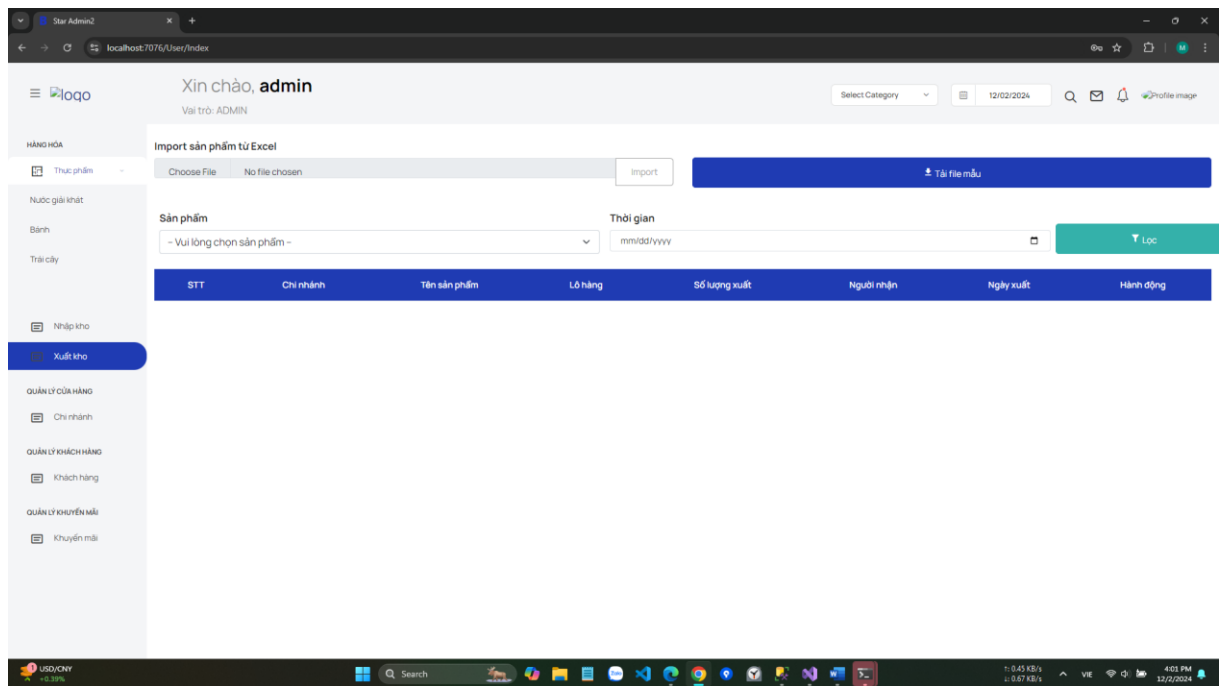
Chức năng nhập kho hoạt động bình thường, đã test các chức năng.



Hình 3.4. Giao diện quản lý nhập kho

### 3.2.5. Giao diện quản lý xuất kho

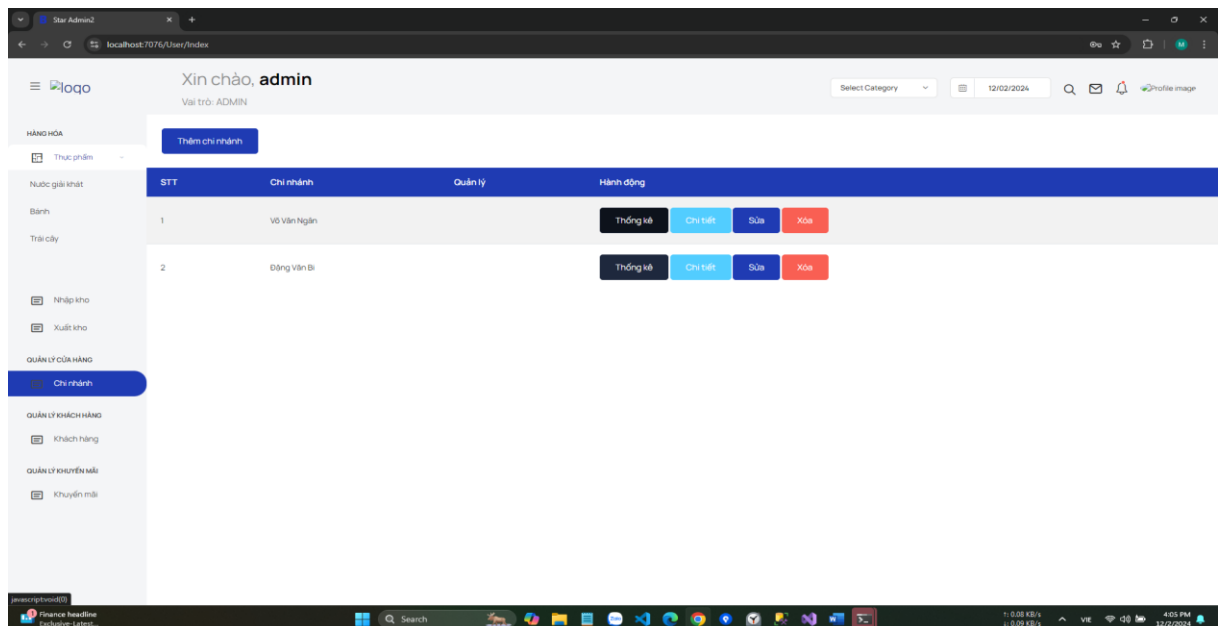
Chức năng quản lý xuất kho hoạt động bình thường, đã test các chức năng.



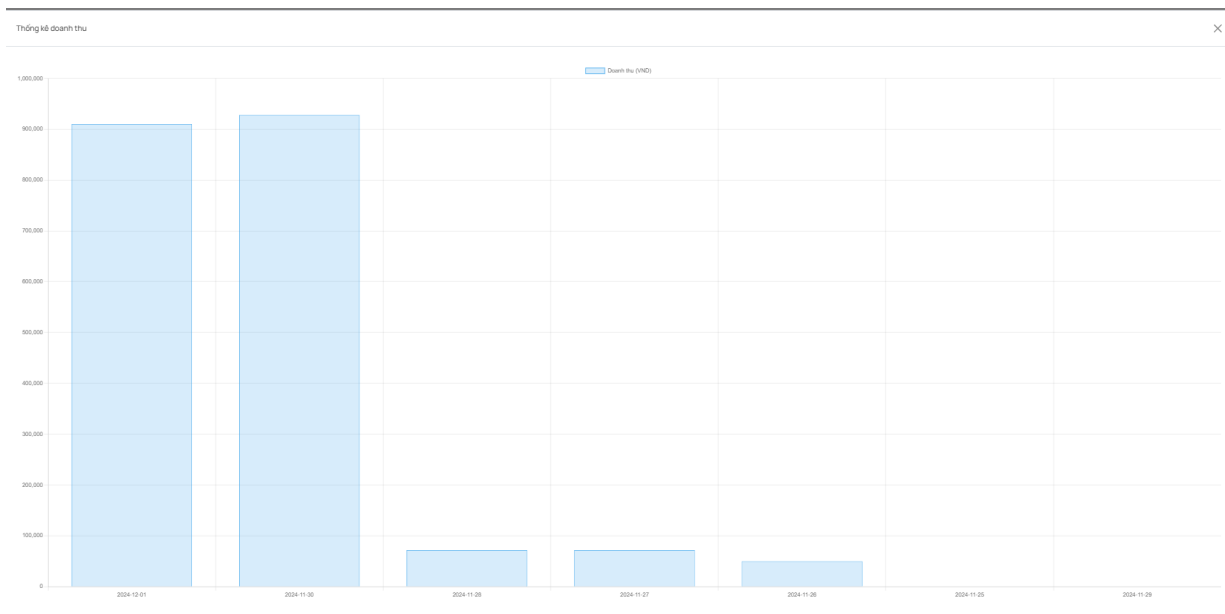
Hình 3.5. Giao diện quản lý xuất kho

### 3.2.6. Giao diện quản lý chi nhánh

Chức năng quản lý chi nhánh hoạt động bình thường. Đã test các chức năng.



Hình 3.6. Giao diện quản lý chi nhánh



Hình 3.7. Giao diện quản lý chi nhánh (Thống kê)

Chi tiết chi nhánh Võ Văn Ngân

Hàng hóa Nhân sự Hóa đơn Nhập hàng

STT	Tên sản phẩm	Số lô hàng	Số lượng hiện tại
1	Nhò	B01	100
2	Bánh	AA	400

Hình 3.8. Giao diện quản lý chi nhánh (Hàng hóa)

Xin chào, admin  
Vai trò: ADMIN

Chọn Category 12/02/2024

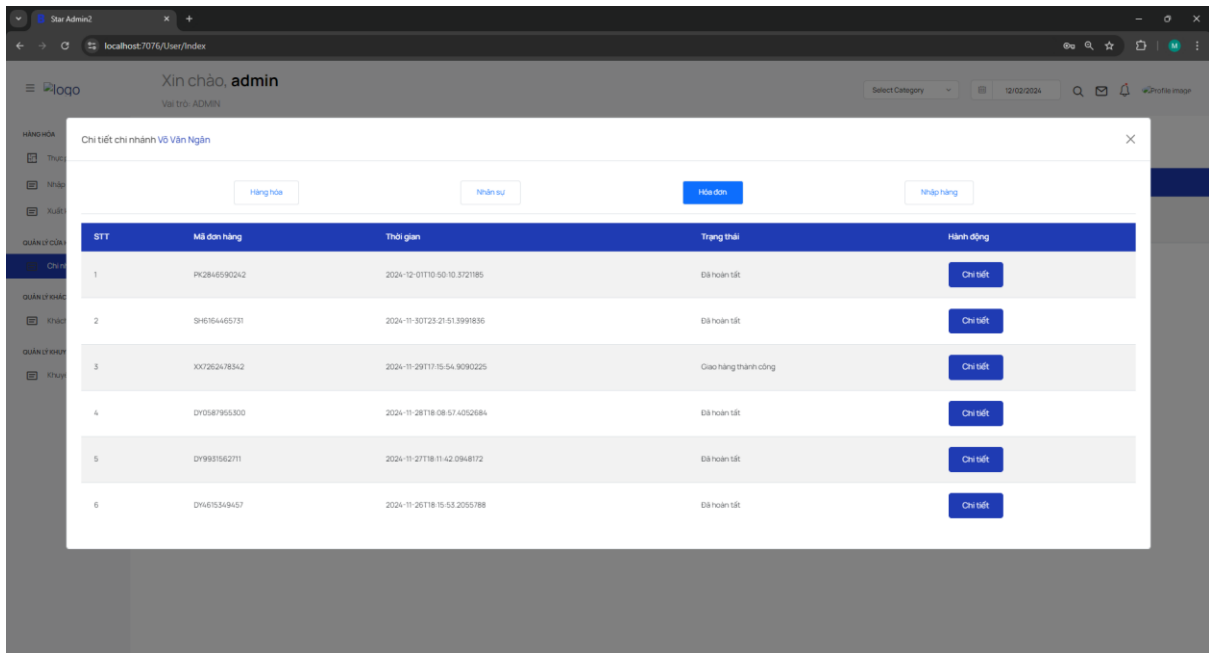
Chi tiết chi nhánh Võ Văn Ngân

Hàng hóa Nhân sự Hóa đơn Nhập hàng

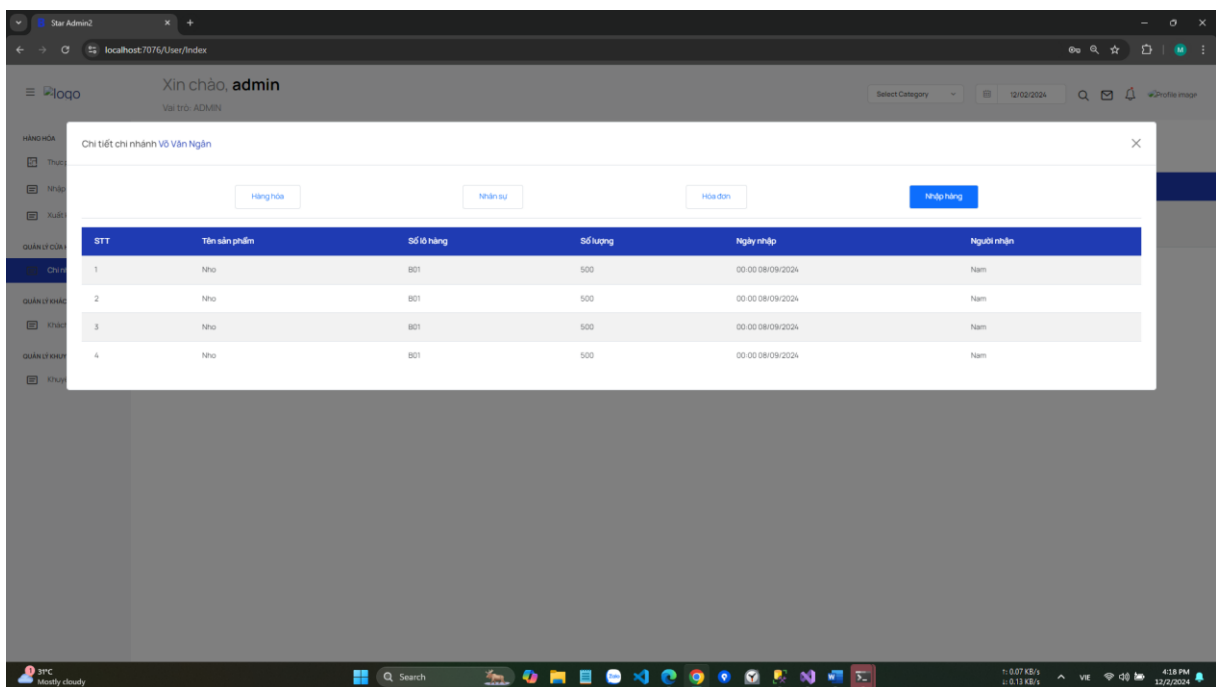
Thêm nhân sự

STT	Họ tên	Email	Ngày sinh	Mức lương	Hành động
1	managerVNN	user1@gmail.com	2024-10-30	12.222	Sửa Xóa
2	emhvn	emhvn@gmail.com	2024-09-08	15.000	Sửa Xóa

Hình 3.9. Giao diện quản lý chi nhánh (Nhân sự)



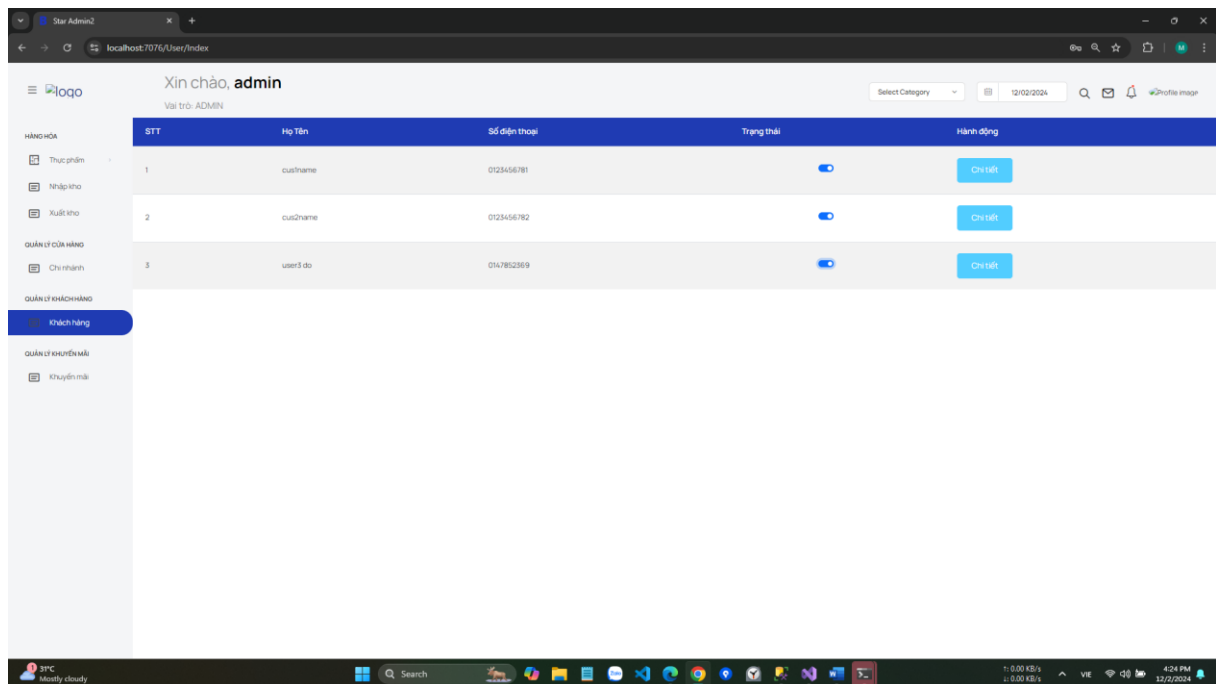
Hình 3.10. Giao diện quản lý chi nhánh (Hóa đơn)



Hình 3.11. Giao diện quản lý chi nhánh (Nhập hàng)

### 3.2.7. Giao diện quản lý khách hàng

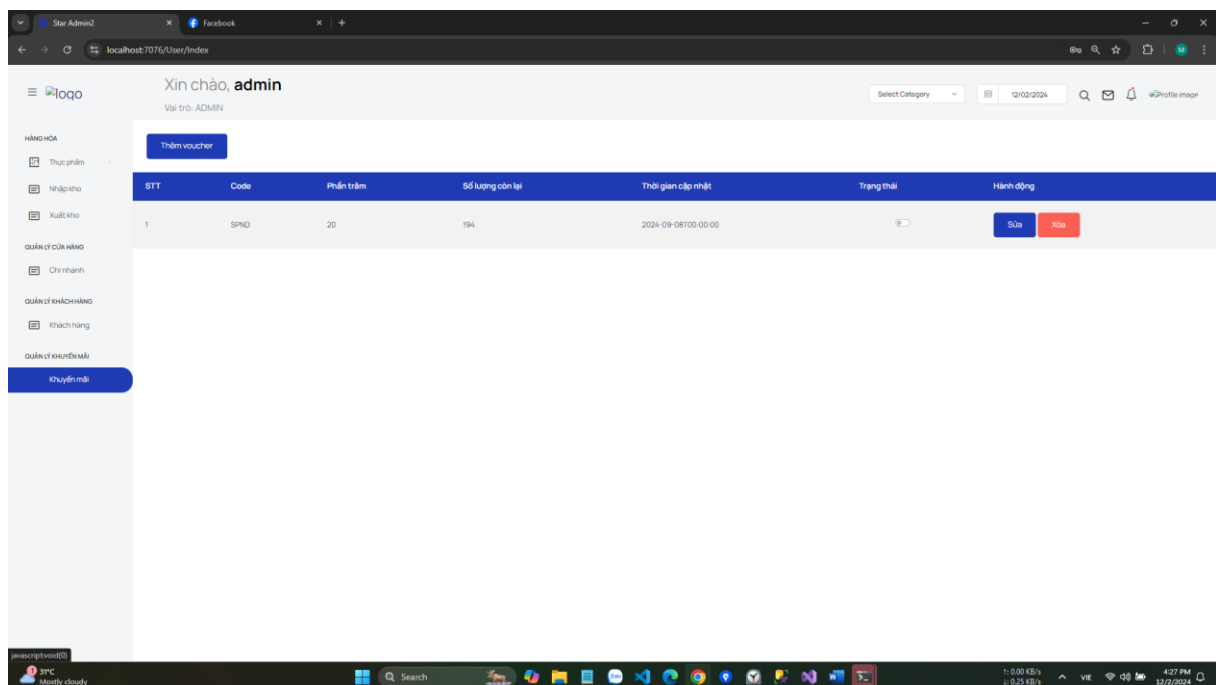
Chức năng quản lý khách hàng chỉ có thể xem thông tin khách hàng, không thể chỉnh sửa. Chức năng hoạt động tốt



Hình 3.12. Giao diện quản lý khách hàng

### 3.2.8. Giao diện quản lý khuyến mãi

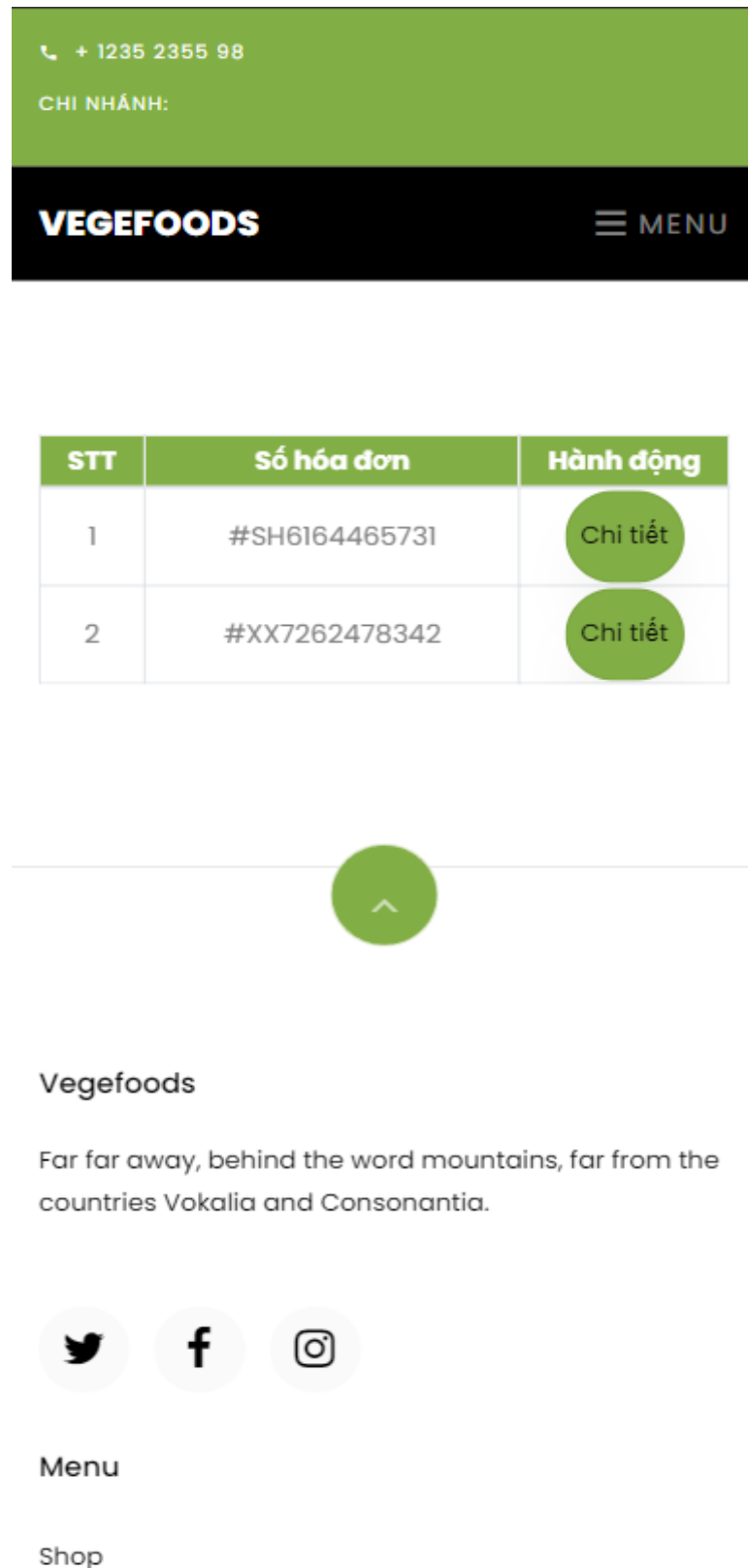
Chức năng quản lý khuyến mãi hoạt động bình thường, có thể thêm khuyến mãi theo phân loại như sản phẩm duy nhất, phân loại duy nhất, tất cả.



Hình 3.13. Giao diện quản lý khuyến mãi

### 3.2.9. Giao diện giao hàng cho khách

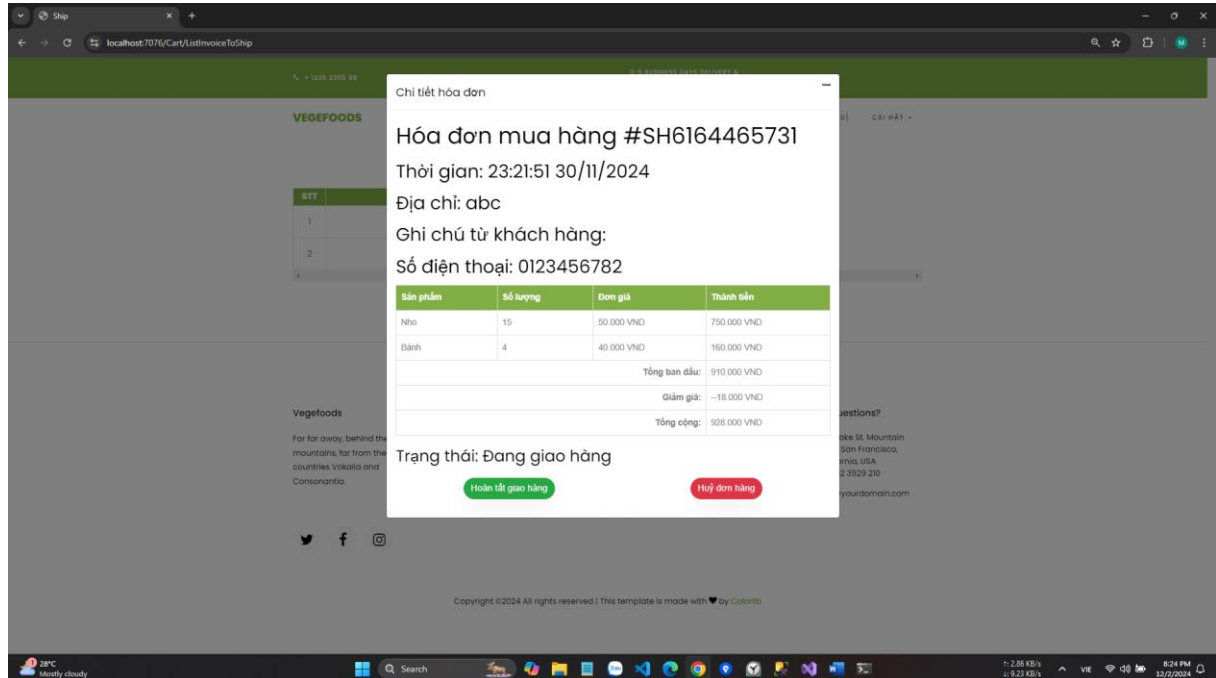
Để giao diện điện thoại là để cho nhân viên giao hàng cho khách bằng điện thoại. Chức năng đã được được test, hoạt động tốt.



Hình 3.14. Giao diện giao hàng cho khách

### 3.2.10. Giao diện chi tiết đơn hàng giao cho khách

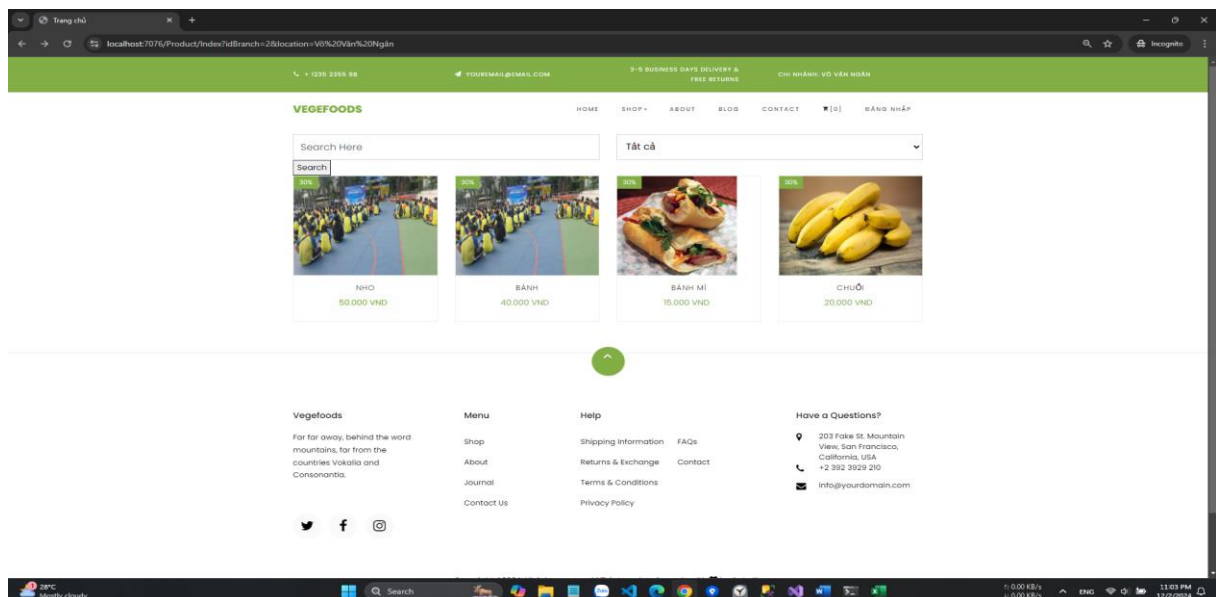
Chức năng xem chi tiết đơn hàng đã được đặt cho cả online lẫn offline, đã test và hoạt động ổn định.



Hình 3.15. Giao diện chi tiết đơn hàng

### 3.2.11. Giao diện chọn hàng hóa

Chức năng lướt hàng hóa để mua hoạt động ổn định.

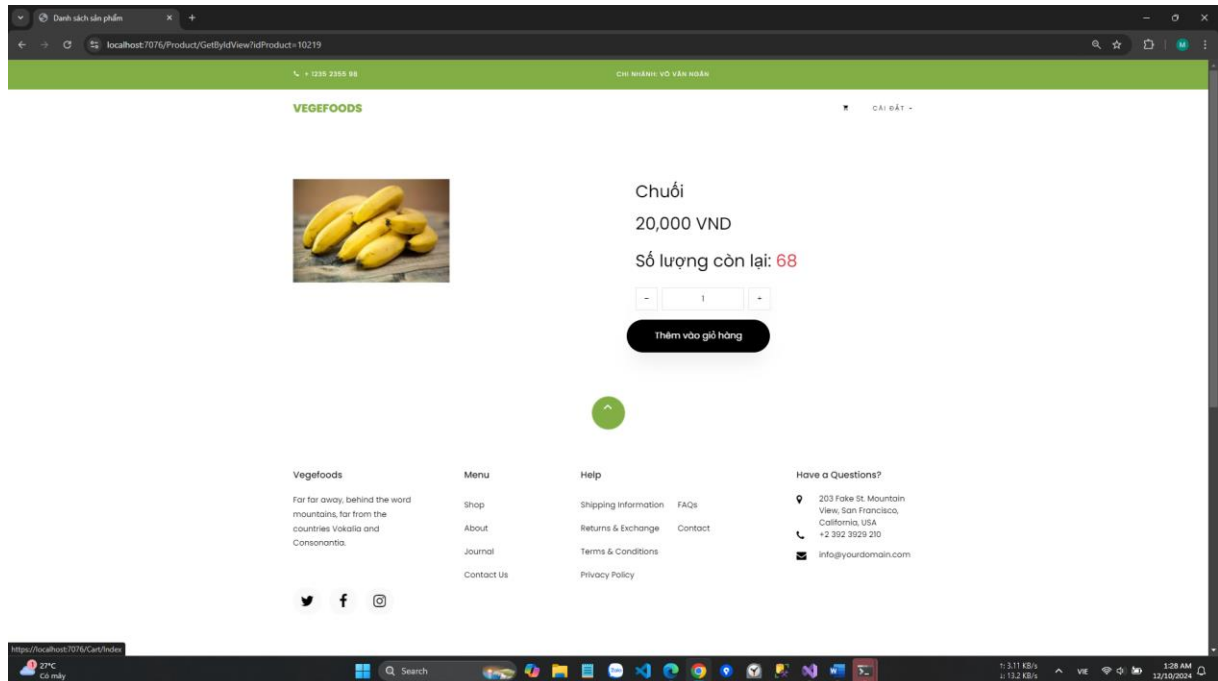


Hình 3.16. Giao diện chọn hàng hóa



### 3.2.12. Giao diện chi tiết hàng hóa

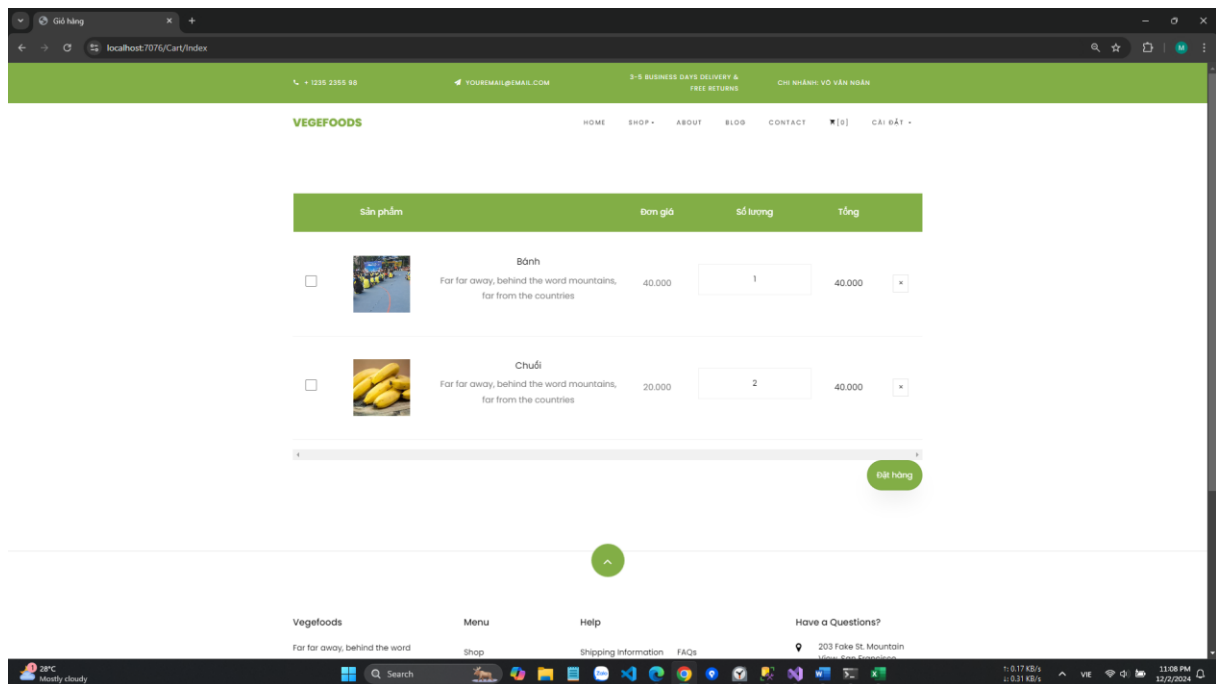
Chức năng xem chi tiết hàng hóa hoạt động ổn định, đã test thêm vào giỏ hàng và tốt, sử dụng FetchAPI để thêm sản phẩm mà không bị reload trang.



Hình 3.17. Giao diện chi tiết hàng hóa

### 3.2.13. Giao diện giỏ hàng

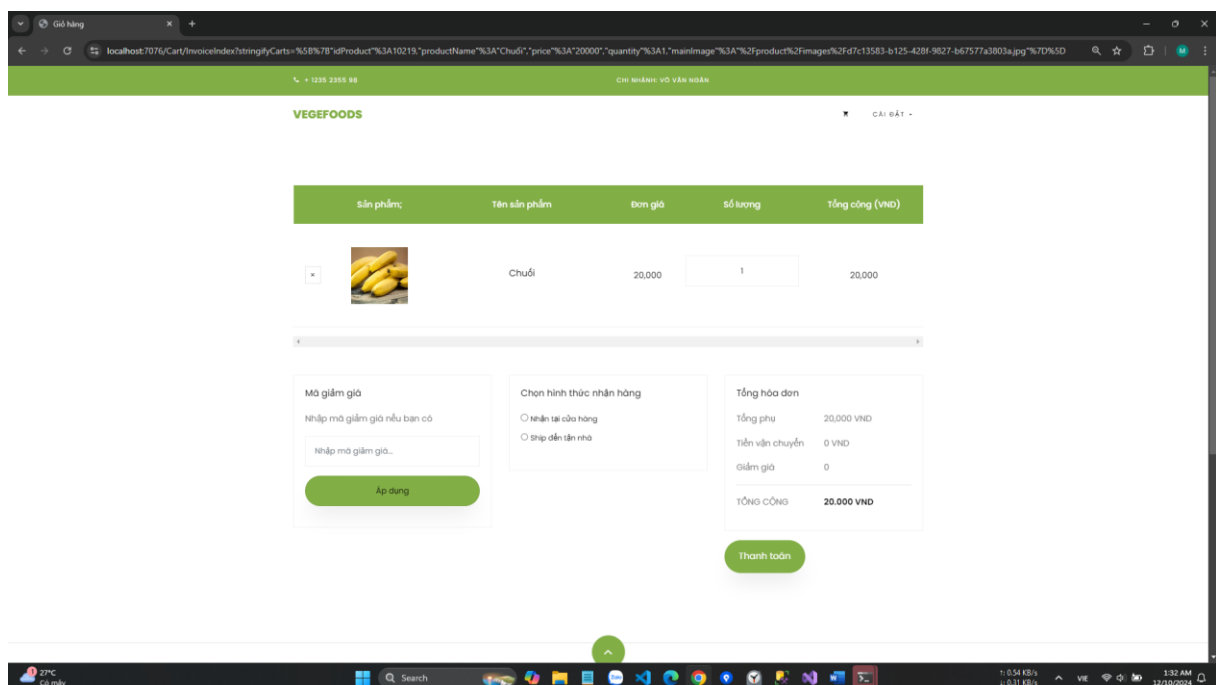
Chức năng xem giỏ hàng hoạt động ổn định, nếu trong giỏ hàng đã tồn tại sản phẩm đó rồi mà khách hàng vẫn muốn thêm sản phẩm thì sản phẩm đó sẽ được cộng thêm. Giỏ hàng này có thể thêm vào được kể cả đã đăng nhập và không đăng nhập (không đăng nhập thì sản phẩm sẽ chỉ được lưu trong session đó thôi). Chức năng hoạt động ổn định



Hình 3.18. Giao diện giỏ hàng

### 3.2.14. Giao diện thanh toán

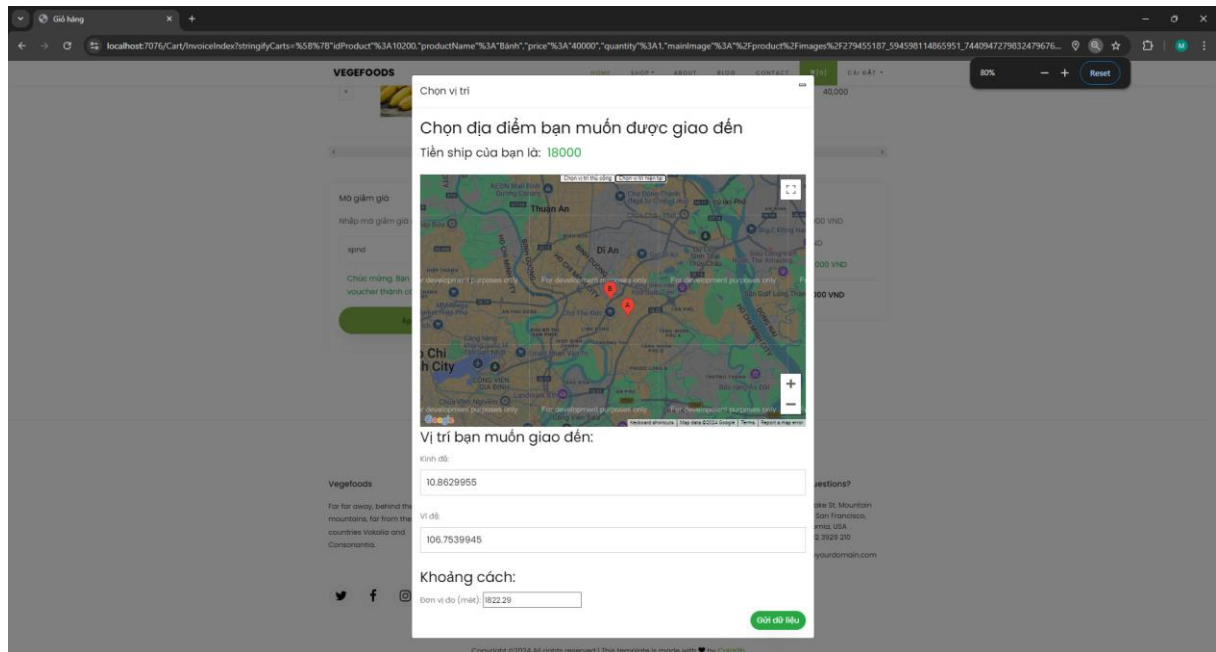
Chức năng thanh toán hoạt động ổn định, khi đã đăng nhập thì có thể sử dụng voucher cũng như chức năng giao hàng tận nơi, còn không đăng nhập thì không thể, khi xác nhận đơn hàng thì sẽ có Email gửi về (đối với đơn hàng đặt online).



Hình 3.19. Giao diện thanh toán

### 3.2.15. Giao diện Google Maps

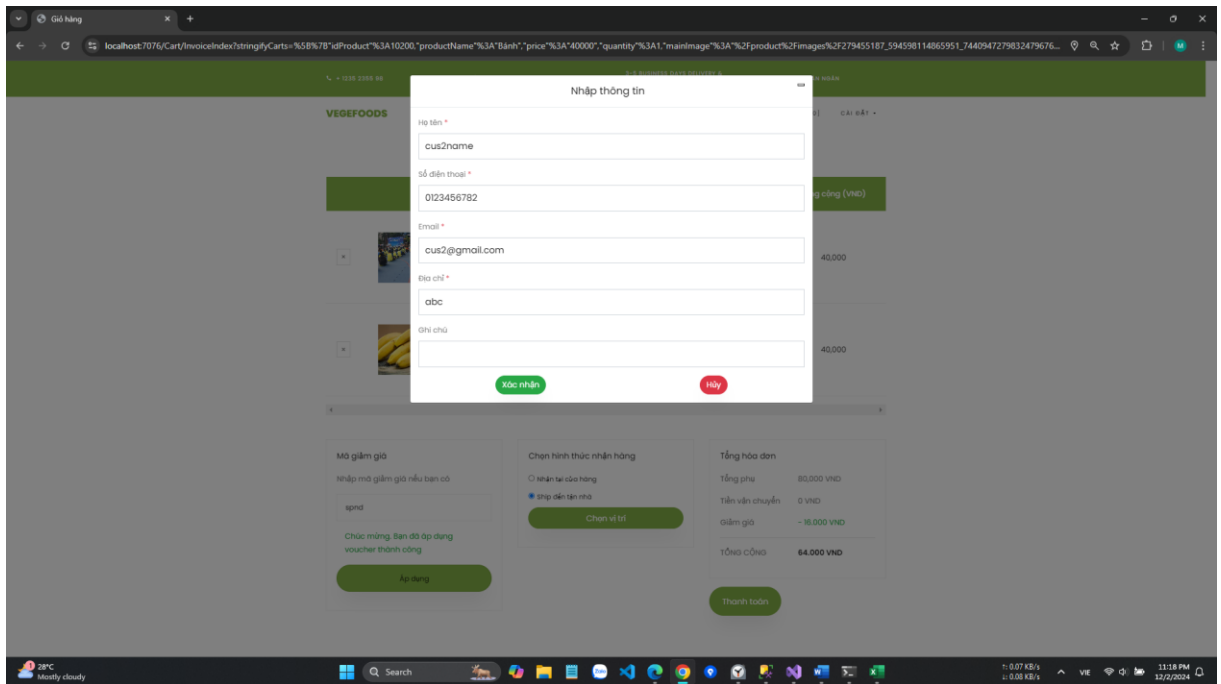
Chức năng giao hàng tính tiền ship dựa vào khoảng cách giữa cửa hàng được chọn và vị trí mong muốn được giao tới, trong phạm vi dưới 3km thì giá sẽ là 18.000, từ 3km – 6 km thì sẽ là 30.000, còn quá 6km thì sẽ không giao được. Cách chức năng hoạt động bình thường.



Hình 3.20. Giao diện Google Maps

### 3.2.16. Giao diện Google Maps

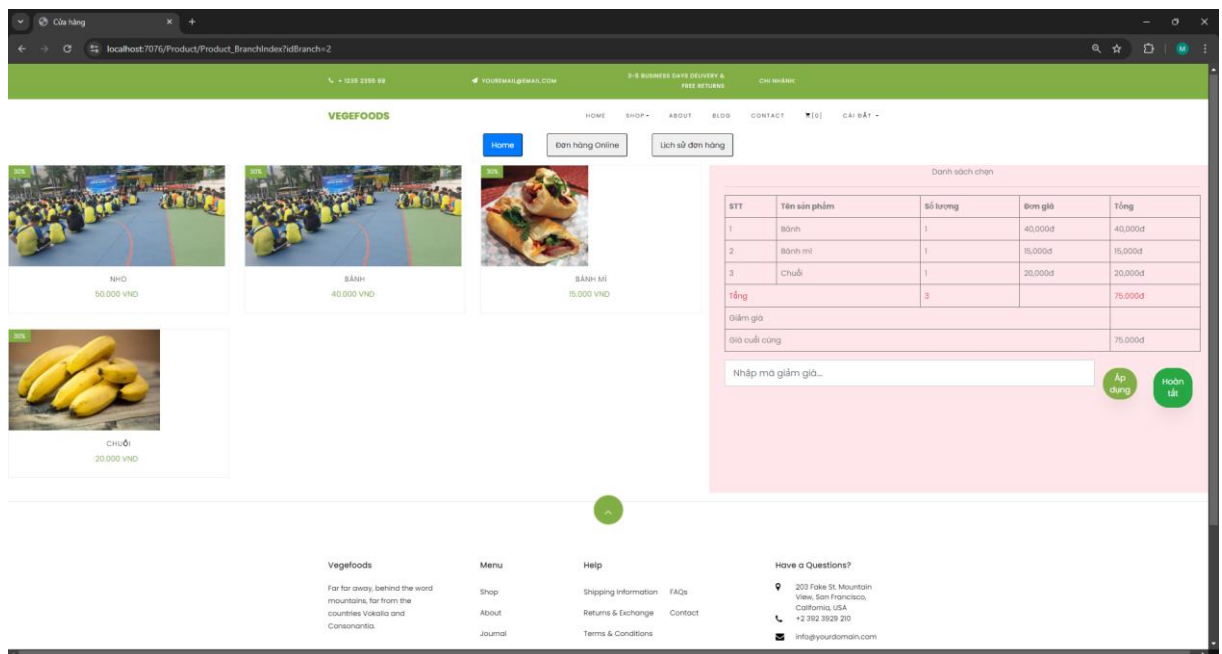
Sau khi bấm xác nhận thông tin thì sẽ hiện form xác nhận thông tin lại 1 lần nữa cũng như ghi chú cho cửa hàng, chức năng đã được test và hoạt động tốt.



Hình 3.21. Giao diện xác nhận thông tin

### 3.2.17. Giao diện cửa hàng (POS)

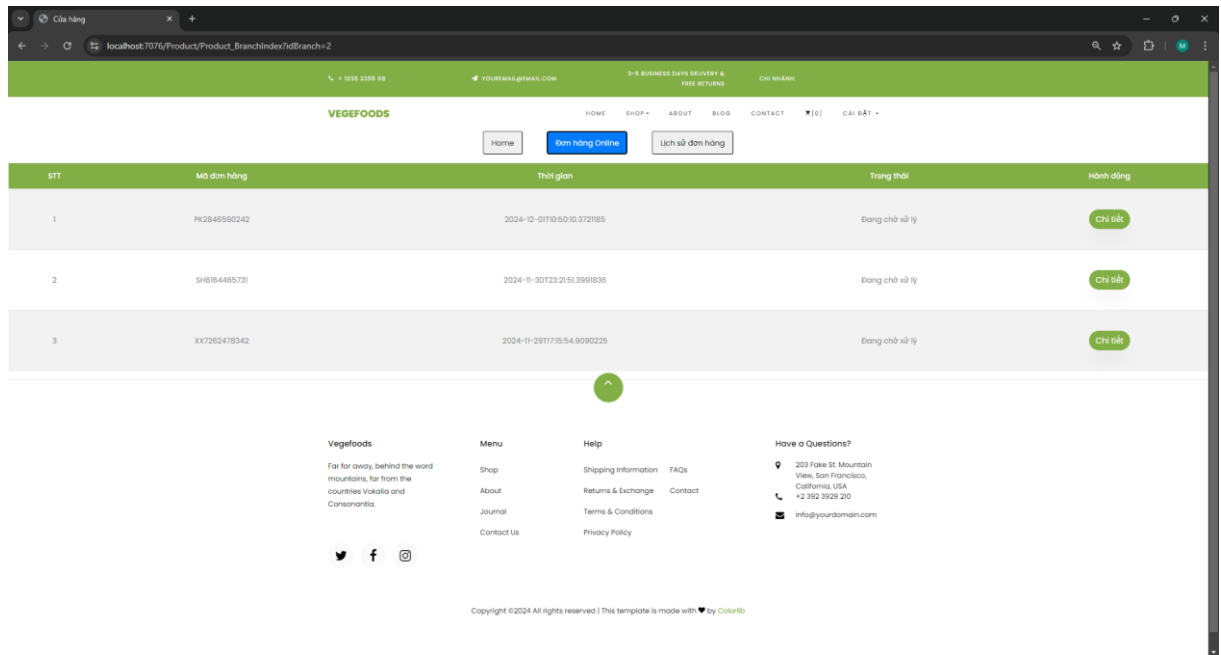
Đây là màn hình chọn sản phẩm cho khách hàng mua offline, chức năng này sẽ dành cho cashier để chọn sản phẩm, chức năng đã được test và hoạt động tốt.



Hình 3.22. Giao diện cửa hàng (POS)

### 3.2.18. Giao diện đơn hàng online (POS)

Đây là chức năng sẽ hiển thị danh sách các sản phẩm đã được đặt tại chi nhánh. Khi khách hàng đặt thì số lượng hàng với lô gần nhất sẽ tạm thời giảm đi, trong trường hợp nếu cửa hàng có bất kì lý do gì mà kho thể giao được thì sẽ hủy đơn hàng và số lượng hàng đó sẽ được rollback lại. Đã test các chức năng và hoạt động bình thường.



Hình 3.23. Giao diện đơn hàng online (POS)

### 3.2.19. Giao diện chi tiết đơn hàng online (POS)

Đây là thông tin đơn hàng đã được đặt và bị hủy bởi cửa hàng. Chi tiết đơn hàng với số lượng hàng. Chức năng này đã được test và hoạt động tốt.

## Chi tiết hóa đơn #PK2846590242

Thời gian: 10:50:10 1/12/2024

Địa chỉ giao hàng: abc

Ghi chú từ khách hàng: cus note

Số điện thoại: 0123456782

Sản phẩm	Số lượng	Đơn giá	Thành tiền
Nho	15	50.000 VND	750.000 VND
Bánh	4	40.000 VND	160.000 VND
Tổng ban đầu:			910.000 VND
Giảm giá:			-0 VND
Tổng cộng:			910.000 VND

Phương thức thanh toán: Nhận tại cửa hàng

Trạng thái: **Hủy bởi cửa hàng**

Hình 3.24. Giao diện chi tiết đơn hàng online (POS)

## 3.2.20. Giao diện tất cả đơn hàng của chi nhánh (POS)

Đây là chức năng hiển thị tất cả các trạng thái đơn hàng cả online lẫn off line. Chức năng hoạt động tốt.

The screenshot displays a web application for VEGEFOODS. At the top, there's a navigation bar with 'Cửa hàng' (Store) and 'VEGEFOODS' logo. Below it, there are buttons for 'Home', 'Đơn hàng Online' (Online Orders), and 'Lịch sử đơn hàng' (Order History). The main content area is a table listing orders. The table has five columns: STT (Serial Number), Mã đơn hàng (Order Code), Thời gian (Time), Trạng thái (Status), and Hành động (Action). There are 7 rows of data. The first three rows show orders in progress ('Đang giao hàng'), and the last four rows show completed orders ('Đã hoàn tất'). Each row includes a 'Chi tiết' (Details) button for further information.

STT	Mã đơn hàng	Thời gian	Trạng thái	Hành động
1	PK2846590242	10:50:10 1/12/2024	Đang giao hàng	<a href="#">Chi tiết</a>
2	SH8164465731	23:21:51 30/11/2024	Đang giao hàng	<a href="#">Chi tiết</a>
3	XX7262478342	17:15:54 29/11/2024	Đang giao hàng	<a href="#">Chi tiết</a>
4	DY0587955300	18:08:57 28/11/2024	Đã hoàn tất	<a href="#">Chi tiết</a>
5	DY9939562711	18:11:42 27/11/2024	Đã hoàn tất	<a href="#">Chi tiết</a>
6	DY4815349457	18:15:53 26/11/2024	Đã hoàn tất	<a href="#">Chi tiết</a>
7	PK7569496543	14:17:20 3/12/2024	Hủy bởi cửa hàng	<a href="#">Chi tiết</a>

Hình 3.25. Giao diện tất cả đơn hàng của chi nhánh (POS)

## **CHƯƠNG 4: KẾT LUẬN**

### **4.1. Các vấn đề đã giải quyết**

Trong đồ án này, em đã xây dựng thành công một website hệ thống quản lý chuỗi của hàng bách hóa với khá đầy đủ các chức năng chính (có thể vẫn còn một vài chức năng chưa có). Website có giao diện người dùng rõ ràng và thân thiện dành cho 5 loại tài khoản Admin, Manager, Employee, Store, Customer (với customer thì vừa có thể đăng nhập cũng như không đăng nhập vẫn có thể mua hàng được). Người dùng có thể sử dụng website một cách dễ dàng. Website đã ứng dụng được API của Google Maps để xác định khoảng cách giúp việc tính tiền ship chính xác hơn, sử dụng JWT để tăng cường bảo mật cho website, và còn nhiều chức công nghệ khác nữa giúp website hoạt động tốt hơn.

### **4.2. Các vấn đề chưa giải quyết**

Bên cạnh những chức năng đã hoàn thành đã nói ở trên, đồ án của em vẫn còn một số vấn đề chưa giải quyết được. Đầu tiên cũng đến từ Google Maps, em chưa tích hợp được chức năng chỉ đường cho nhân viên giao hàng đến cho khách hàng mà chỉ tính được khoảng cách để tính tiền ship (do chi phí hạn chế), chưa tích hợp các chức năng thay đổi mật khẩu, tích hợp các ví điện tử khác để thanh toán, giao diện vẫn còn chưa được thẩm mỹ cho lắm và còn nhiều vấn đề nữa cần được giải quyết. Em sẽ giải quyết những thiếu sót trong tương lai gần để nâng cao hiệu suất và chất lượng hệ thống.

### **4.3. Hướng phát triển trong tương lai**

Trong tương lai, em dự định sẽ hoàn thiện các chức năng còn thiếu và nâng cấp phần mềm để đáp ứng tốt nhu cầu sử dụng. Trước hết, em sẽ phát triển chức năng chỉ đường trong Google Maps, tiếp theo em sẽ tích hợp các ví điện tử để đa dạng hình thức thanh toán, cải thiện giao diện người dùng nhằm nâng cao trải nghiệm người dùng, và còn nhiều chức năng khác nữa em cần phải hoàn thiện thêm. Những cải tiến này sẽ giúp phần mềm trở nên tốt hơn, đáp ứng tốt hơn nhu cầu của doanh nghiệp.



## TÀI LIỆU THAM KHẢO

- [1] <https://aws.amazon.com/microservices/> What are Microservices; truy cập vào ngày 05/12/2024; bởi Amazon; xuất bản 12/2024.
- [2] Parvez, M. O. (2021). Use of machine learning technology for tourist and organizational services: high-tech innovation in the hospitality industry. Tập 7, 2, 240-244
- [3] [API Gateway là gì? Tại sao một hệ thống microservices lại cần API Gateway?](#) API Gateway là gì? Tại sao một hệ thống microservices lại cần API Gateway?; truy cập vào ngày 05/12/2024; bởi Nguyễn Đức Huy Học; xuất bản ngày 15/01/2019.
- [4] [Implementing API Gateways with Ocelot - .NET | Microsoft Learn](#) truy cập vào ngày 05/12/2024; bởi James Montemogno; xuất bản ngày 03/01/2019.
- [5] [ASP.Net MVC vs ASP.Net Core](#) ASP.Net MVC vs ASP.Net Core; truy cập vào ngày 05/12/2024; bởi Anandu G Nath; xuất bản ngày 26/12/2023.
- [6] [kspearrin/Otp.NET: A .NET implementation of TOTP and HOTP for things like two-factor authentication codes.](#); truy cập vào ngày 05/12/2024; bởi Kspearrin; xuất bản vào 04/2024.
- [7] [SMTP là gì? – Giải thích về Máy chủ SMTP – AWS](#) SMTP là gì?; truy cập vào ngày 05/12/2024; bởi Amazon; xuất bản tháng 12/2024.
- [8] [Tìm Hiểu Về Json Web Token \(JWT\) - Viblo](#) Tìm hiểu về json web token (JWT); truy cập vào ngày 05/12/2024; bởi Trần Vương Minh; xuất bản ngày 28/03/2016.
- [9] [RESTful API là gì? Cách thiết kế RESTful API | TopDev](#) truy cập vào ngày 05/12/2024; bởi TopDev, xuất bản ngày 22/04/2019.
- [10] [TÌM HIỂU VỀ GOOGLE MAP API](#) truy cập vào ngày 05/12/2024; bởi Trần Thị Hoa, xuất bản ngày 28/03/2015.
- [11] [AJAX là gì? Cách thức hoạt động và lợi ích của AJAX](#) truy cập vào ngày 05/12/2024; bởi TopDev, xuất bản ngày 02/03/2019.

- [12] [Fetch API - Web APIs | MDN](#) truy cập vào ngày 05/12/2024; bởi Developer.Mozilla, xuất bản ngày 08/10/2024.
- [13] [API Rate Limiting là gì và một số thuật toán hay dùng](#) truy cập vào ngày 05/12/2024; bởi Doan Le, xuất bản ngày 24/09/2020.
- [14] [C# là gì? Tìm hiểu về ngôn ngữ lập trình C#](#) truy cập vào ngày 05/12/2024; bởi CodeGym, xuất bản ngày 24/07/2024.
- [15] [SQL Server là gì? SQL Server giúp bạn làm việc dễ dàng hơn?](#) truy cập vào ngày 05/12/2024; bởi TopDev, xuất bản ngày 01/07/2022.
- [16] [Visual Studio là gì? Những tính năng cần thiết của Visual Studio](#) truy cập vào ngày 05/12/2024; bởi Bizfly Cloud, xuất bản ngày 24/03/2021.
- [17] [Phần mềm StarUML](#) truy cập vào ngày 05/12/2024; bởi TopDev, xuất bản ngày 17/04/2019.
- [18] [About GitHub and Git](#) truy cập vào ngày 05/12/2024; bởi GitHubDocs, xuất bản ngày 25/09/2024.
- [19] [Bootstrap Get Started](#) truy cập vào ngày 05/12/2024; bởi W3School, xuất bản ngày 09/12/2024.