ĐẠI HỌC HUẾ

TRƯỜNG ĐẠI HỌC KHOA HỌC

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙞 🕮 🙜**

**KHOÁ LUẬN**

**TỐT NGHIỆP ĐẠI HỌC**

**Đề tài:**

**TÌM HIỂU JAVA SPRING FRAMEWORK VÀ XÂY DỰNG ỨNG DỤNG WEBSITE QUẢN LÝ NHÀ HÀNG**

Sinh viên thực hiện: HỒ MINH THUẬN

Khóa: K44 – Hệ chính quy

**Huế, 5 – 2024**

ĐẠI HỌC HUẾ

TRƯỜNG ĐẠI HỌC KHOA HỌC

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙞 🕮 🙜**

**KHOÁ LUẬN**

**TỐT NGHIỆP ĐẠI HỌC**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài:**

**TÌM HIỂU JAVA SPRING FRAMEWORK VÀ XÂY DỰNG ỨNG DỤNG WEBSITE QUẢN LÝ NHÀ HÀNG**

Sinh viên thực hiện: HỒ MINH THUẬN

Khóa: K44 - Hệ chính quy

Giáo viên hướng dẫn: PGS. TS. NGUYỄN MẬU HÂN

**Huế, 5 – 2024**

# **LỜI CẢM ƠN**

Qua suốt bốn năm học ở Trường Đại Học Khoa Học Huế khoá 2020-2024, sinh viên chúng em đã được học rất nhiều kiến thức về chuyên ngành của mình và những kiến thức về cuộc sống, ngoài sự nổ lực của bản thân,chúng em còn đón nhận sự giúp đỡ tâm huyết và công sức của nhà trường, thầy cô và bạn bè.

Em xin chân thành cảm ơn tất cả các thầy cô đã từng dạy dỗ sinh viên chúng em, thầy cô Trường Đại Học Khoa Học Huế và nhất là thầy cô và giảng viên trong Khoa Công Nghệ Thông Tin đã cung cấp những kiến thức cần thiết để chúng em có thể hoàn thành được khoá luận tốt nghiệp. Đặt biệt là thầy Nguyễn Mậu Hân giáo viên hướng dẫn em đã đồng hành và giúp em hoàn thành tốt bài khoá luận tốt nghiệp này.

Xin gửi lời cảm ơn đến tất cả các bạn bè đã động viên giúp đỡ và góp ý trong suốt quá trình học tập và quá trình làm khoá luận tốt nghiệp.

Dù em đã cố gắng rất nhiều để hoàn thành tốt khoá luận tốt nghiệp nhưng em cũng không sao tránh khỏi những thiếu xót, sai lầm cũng như có nhiều mặt hạn chế trong quá trình học tập và làm khoá luận. Em xin mong nhận được sự thông cảm, góp ý, giúp đỡ, cùng với sự chỉ dẫn của thầy cô cũng như ý kiến đóng góp chân thành của các bạn để cho em có thể hoàn thành tốt khoá luận tốt nghiệp và giúp em rút ra được kình nghiệm cho bản thân trong quá trình làm việc sau này.

Em xin chân thành cảm ơn!

|  |  |
| --- | --- |
|  | Sinh Viên  Hồ Minh Thuận |

# **BẢNG CHỮ CÁI VIẾT TẮT**

|  |  |
| --- | --- |
| AOP | Aspect Oriented Programming |
| API | Application Programming Interface |
| CSRF | Cross-Sỉte Request Forgery |
| DI | Dependency Injection |
| EJB | Enterprise Java Bean |
| HTML | Hyper Text Markup Language |
| IDE | Integrated Development Environment |
| IoC | Inversion of Control |
| JDBC | Java Database Connectivity |
| JMX | Java Management Extensions |
| JPA | Java Persistence API |
| JVM | Java Virtual Machine |
| MVC | Model-View-Controller |
| OPP | Object-Oriented Programming |
| ORM | Object-Relation Mapping |
| POJO | Plain Old Java Object |
| SPA | Single Page Applications |
| SQL | Structured Query Language |
| UML | Unifed Modeling Language |
| XML | eXtensibe Markup Language |

# **DANH MỤC CÁC BẢNG BIỂU**

[Bảng 3.1. Vai trò của các actor. 28](#_Toc167658636)

[Bảng 3.2: Bảng Users 47](#_Toc167658637)

[Bảng 3.3: Bảng Roles 48](#_Toc167658638)

[Bảng 3.4: Bảng Users\_Roles. 48](#_Toc167658639)

[Bảng 3.5: Bảng Tables 48](#_Toc167658640)

[Bảng 3.6: Bảng Menus 49](#_Toc167658641)

[Bảng 3.7: Bảng Menus\_Group 49](#_Toc167658642)

[Bảng 3.8: Bảng Tables\_Menu 50](#_Toc167658643)

[Bảng 3.9: Bảng Customer 50](#_Toc167658644)

[Bảng 3.10: Bảng Employee 51](#_Toc167658645)

[Bảng 3.11: Bảng Processing\_Newspaper 51](#_Toc167658646)

[Bảng 3.12: Bảng Provisional\_Invoice 52](#_Toc167658647)

[Bảng 3.13: Bảng Invoice 52](#_Toc167658648)

[Bảng 3.14: Bảng Refresh\_Tokens 53](#_Toc167658649)

[Bảng 3.15: Bảng API User 53](#_Toc167658650)

[Bảng 3.16: Bảng API Table 53](#_Toc167658651)

[Bảng 3.17: Bảng API Table\_Menu 54](#_Toc167658652)

[Bảng 3.18: Bảng Menu 54](#_Toc167658653)

[Bảng 3.19: Bảng Menus\_Group 54](#_Toc167658654)

[Bảng 3.20: Bảng API Customer 55](#_Toc167658655)

[Bảng 3.21: Bảng API Employee 55](#_Toc167658656)

[Bảng 3.22: Bảng Processing\_Newspaper 55](#_Toc167658657)

[Bảng 3.23: Bảng API Provisional\_Invoice 56](#_Toc167658658)

[Bảng 3.24: Bảng API Invoice 56](#_Toc167658659)

[Bảng 3.25: Bảng API Login 56](#_Toc167658660)

# **DANH MỤC CÁC HÌNH ẢNH**

[Hình 2.1: Kiến trúc tổng thể của Spring Framework 10](#_Toc167658661)

[Hình 3.2: Use case tổng quát 29](#_Toc167658662)

[Hình 3.3: Use case đăng nhập hệ thống 29](#_Toc167658663)

[Hình 3.4: Use case quản lý bán hàng 30](#_Toc167658664)

[Hình 3.5: Use case quản lý bàn 31](#_Toc167658665)

[Hình 3.6: Use case quản lý khách hàng 31](#_Toc167658666)

[Hình 3.7: Use case thống kê đơn hàng 32](#_Toc167658667)

[Hình 3.8: Use case quản lý báo chế biến 32](#_Toc167658668)

[Hình 3.9: Use case quản lý thực đơn 33](#_Toc167658669)

[Hình 3.10: Use case quản lý nhân viên 33](#_Toc167658670)

[Hình 3.11: Biểu đồ chức năng đăng nhập hệ thống 34](#_Toc167658671)

[Hình 3.12: Biểu đồ hoạt động chức năng đổi mật khẩu 35](#_Toc167658672)

[Hình 3.13: Biểu đồ hoạt động chức năng gọi món thực đơn bàn 35](#_Toc167658673)

[Hình 3.14: Biểu đồ hoạt động chức năng cập nhật thông tin chi tiết thực đơn bàn 36](#_Toc167658674)

[Hình 3.15: Biểu đồ hoạt động chức năng xoá thông tin chi tiết thực đơn bàn 36](#_Toc167658675)

[Hình 3.16: Biểu đồ hoạt động chức năng cập nhật thông tin hoá đơn bàn 37](#_Toc167658676)

[Hình 3.17: Biểu đồ hoạt động chức năng báo chế biến hoá đơn bàn 37](#_Toc167658677)

[Hình 3.18: Biểu đồ hoạt động chức năng tạm tính hoá đơn bàn 38](#_Toc167658678)

[Hình 3.19: Biểu đồ hoạt động chức năng thanh toán hoá đơn bàn 38](#_Toc167658679)

[Hình 3.20: Biểu đồ hoạt động chức năng thêm thông tin bàn mới 39](#_Toc167658680)

[Hình 3.21: Biểu đồ hoạt động chức năng cập nhật thông tin bàn 39](#_Toc167658681)

[Hình 3.22: Biểu đồ hoạt động chức năng xoá thông tin bàn 40](#_Toc167658682)

[Hình 3.23: Biểu đồ hoạt động chức năng thêm thông tin khách hàng mới 40](#_Toc167658683)

[Hình 3.24: Biểu đồ hoạt động chức năng cập nhật thông tin khách hàng 41](#_Toc167658684)

[Hình 3.25: Biểu đồ hoạt động chức năng xác nhận phiếu báo chế biến 41](#_Toc167658685)

[Hình 3.26: Biểu đồ hoạt động chức năng hoàn thành phiếu báo chế biến 42](#_Toc167658686)

[Hình 3.27: Biểu đồ hoạt động chức năng huỷ phiếu báo chế biến 42](#_Toc167658687)

[Hình 3.28: Biểu đồ hoạt động chức năng thêm thông tin thực đơn mới 43](#_Toc167658688)

[Hình 3.29: Biểu đồ hoạt động chức năng cập nhật thông tin thực đơn 43](#_Toc167658689)

[Hình 3.30:Biểu đồ hoạt động chức năng xoá thông tin thực đơn 44](#_Toc167658690)

[Hình 3.31: Biểu đồ hoạt động chức năng thêm thông tin nhân viên mới 44](#_Toc167658691)

[Hình 3.32:Biểu đồ hoạt động chức năng cập nhật thông tin nhân viên 45](#_Toc167658692)

[Hình 3.33: Biểu đồ hoạt động chức năng xoá thông tin nhân viên 45](#_Toc167658693)

[Hình 3.34: Biểu đồ hoạt động chức năng tạo tài khoản đăng nhập mới cho nhân viên. 46](#_Toc167658694)

[Hình 3.35: Biểu đồ hoạt động chức năng thay đổi thông tin đăng nhập của nhân viên 46](#_Toc167658695)

[Hình 3.36: Biểu đồ hoạt động chức năng xoá tài khoản đăng nhập của nhân viên 47](#_Toc167658696)

[Hình 3.37: Giao diện trang đăng nhập hệ thống 57](#_Toc167658697)

[Hình 3.38: Giao diện chính màn hình trang chủ 57](#_Toc167658698)

[Hình 3.39: Giao diện thông tin tài khoản đăng nhập 58](#_Toc167658699)

[Hình 3.40: Giao diện chính màn hình thu ngân 58](#_Toc167658700)

[Hình 3.41: Giao diện hiển thị danh sách các thực đơn để chọn 59](#_Toc167658701)

[Hình 3.42: Giao diện thông tin hoá đơn 59](#_Toc167658702)

[Hình 3.43: Hiển thị in hoá đơn sau khi tạm tính 60](#_Toc167658703)

[Hình 3.44: Giao diện chính màn hình bếp 60](#_Toc167658704)

[Hình 3.45: Giao diện chính trang quản lý bàn 61](#_Toc167658705)

[Hình 3.46: Giao diện chính trang quản lý thực đơn 61](#_Toc167658706)

[Hình 3.47: Giao diện chính trang quản lý khách hàng 62](#_Toc167658707)

[Hình 3.48: Giao diện chính trang quản lý nhân viên 62](#_Toc167658708)

# **MỤC LỤC**

[**LỜI CẢM ƠN** i](#_Toc167658709)

[**BẢNG CHỮ CÁI VIẾT TẮT** ii](#_Toc167658710)

[**DANH MỤC CÁC BẢNG BIỂU** iii](#_Toc167658711)

[**DANH MỤC CÁC HÌNH ẢNH** v](#_Toc167658712)

[**MỤC LỤC** viii](#_Toc167658713)

[CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI 1](#_Toc167658714)

[1.1. Lý do chọn đề tài: 1](#_Toc167658715)

[1.2. Đối tượng và phạm vi đề tài: 1](#_Toc167658716)

[1.3. Phương pháp nghiên cứu: 2](#_Toc167658717)

[1.4. Bố cục khoá luận: 2](#_Toc167658718)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 4](#_Toc167658719)

[2.1. Ngôn ngữ lập trình Java: 4](#_Toc167658722)

[2.1.1. Java và lịch sử phát triển: 4](#_Toc167658723)

[2.1.2. Một số tính năng của Java: 5](#_Toc167658724)

[2.1.3. Ưu và nhược điểm của Java: 6](#_Toc167658725)

[2.2. Spring Framework: 7](#_Toc167658726)

[2.2.1. Tổng quan Spring: 7](#_Toc167658727)

[2.2.2. Lịch sử phát triển: 8](#_Toc167658728)

[2.2.3. Một số khái niệm chính: 9](#_Toc167658729)

[2.2.4. Kiến trúc và các module của Spring Framework : 10](#_Toc167658730)

[2.2.5. Spring Boot: 12](#_Toc167658739)

[2.2.6. Spring Security: 14](#_Toc167658740)

[2.2.7. Spring Data JPA: 18](#_Toc167658741)

[2.3. Angular: 19](#_Toc167658742)

[2.3.1. Tổng quan về Angular: 19](#_Toc167658743)

[2.3.2. Lịch sử phát triển của Angular: 20](#_Toc167658744)

[2.3.3. Ưu điểm của Angular: 21](#_Toc167658745)

[2.3.4. Nhược điểm của Angular: 22](#_Toc167658746)

[2.4. MySQL: 22](#_Toc167658747)

[2.4.1. Tổng quan về SQL: 22](#_Toc167658748)

[2.4.2. MySQL: 23](#_Toc167658749)

[2.4.3. Ưu điểm của MySQL: 24](#_Toc167658750)

[2.4.4. Nhược điểm của MySQL: 24](#_Toc167658751)

[2.5. Bootstrap: 24](#_Toc167658752)

[CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT CHƯƠNG TRÌNH 26](#_Toc167658753)

[3.1. Phân tích bài toán: 26](#_Toc167658755)

[3.1.1. ADMIN 26](#_Toc167658756)

[3.1.2. CASHIER: 27](#_Toc167658757)

[3.1.3. CHEF: 28](#_Toc167658758)

[3.2. Biểu đồ Use Case: 28](#_Toc167658759)

[3.2.1. Use case tổng quát: 29](#_Toc167658760)

[3.2.2. Use case Đăng Nhập Hệ Thống: 29](#_Toc167658761)

[3.2.3. Use case Quản Lý Bán Hàng: 30](#_Toc167658762)

[3.2.4. Use Case Quản Lý Bàn: 31](#_Toc167658763)

[3.2.5. Use case Quản Lý Khách Hàng: 31](#_Toc167658764)

[3.2.6. Use case Thống Kê Đơn Hàng: 32](#_Toc167658765)

[3.2.7. Use case Quản Lý Báo Chế Biến: 32](#_Toc167658766)

[3.2.8. Use case Quản Lý Thực Đơn: 33](#_Toc167658767)

[3.2.9. Use case Quản Lý Nhân Viên: 33](#_Toc167658768)

[3.3. Biểu đồ hoạt động (Activity Diagram): 34](#_Toc167658769)

[3.3.1. Chức năng đăng nhập hệ thống: 34](#_Toc167658770)

[3.3.2. Chức năng đổi mật khẩu: 35](#_Toc167658771)

[3.3.3. Chức năng gọi món thực đơn bàn: 35](#_Toc167658772)

[3.3.4. Chức năng cập nhật thông tin chi tiết thực đơn bàn: 36](#_Toc167658773)

[3.3.5. Chức năng xoá thông tin chi tiết thực đơn bàn: 36](#_Toc167658774)

[3.3.6. Chức năng cập nhật thông tin hoá đơn bàn: 37](#_Toc167658775)

[3.3.7. Chức năng báo chế biến hoá đơn bàn: 37](#_Toc167658776)

[3.3.8. Chức năng tạm tính hoá đơn bàn: 38](#_Toc167658777)

[3.3.9. Chức năng thanh toán hoá đơn bàn: 38](#_Toc167658778)

[3.3.10. Chức năng thêm thông tin bàn mới: 39](#_Toc167658779)

[3.3.11. Chức năng cập nhật thông tin bàn: 39](#_Toc167658780)

[3.3.12. Chức năng xoá thông tin bàn: 40](#_Toc167658781)

[3.3.13. Chức năng thêm thông tin khách hàng mới: 40](#_Toc167658782)

[3.3.14. Chức năng cập nhật thông tin khách hàng: 41](#_Toc167658783)

[3.3.15. Chức năng xác nhận phiếu báo chế biến: 41](#_Toc167658784)

[3.3.16. Chức năng hoàn thành phiếu báo chế biến: 42](#_Toc167658785)

[3.3.17. Chức năng huỷ phiếu báo chế biến: 42](#_Toc167658786)

[3.3.18. Chức năng thêm thông tin thực đơn mới: 43](#_Toc167658787)

[3.3.19. Chức năng cập nhật thông tin thực đơn: 43](#_Toc167658788)

[3.3.20. Chức năng xoá thông tin thực đơn: 44](#_Toc167658789)

[3.3.21. Chức năng thêm thông tin nhân viên mới: 44](#_Toc167658790)

[3.3.22. Chức năng cập nhật thông tin nhân viên: 45](#_Toc167658791)

[3.3.23. Chức năng xoá thông tin nhân viên: 45](#_Toc167658792)

[3.3.24. Chức năng tạo tài khoản đăng nhập mới cho nhân viên: 46](#_Toc167658793)

[3.3.25. Chức năng thay đổi thông tin đăng nhập của nhân viên: 46](#_Toc167658794)

[3.3.26. Chức năng xoá tài khoản đăng nhập của nhân viên: 47](#_Toc167658795)

[3.4. Thiết kế cơ sở dữ liệu: 47](#_Toc167658796)

[3.4.1. Bảng Users: 47](#_Toc167658797)

[3.4.2. Bảng Roles: 48](#_Toc167658798)

[3.4.3. Bảng Users\_Roles: 48](#_Toc167658799)

[3.4.4. Bảng Tables: 48](#_Toc167658800)

[3.4.5. Bảng Menus: 49](#_Toc167658801)

[3.4.6. Bảng Menus\_Group: 49](#_Toc167658802)

[3.4.7. Bảng Tables\_Menu: 50](#_Toc167658803)

[3.4.8. Bảng Customer: 50](#_Toc167658804)

[3.4.9. Bảng Employee: 51](#_Toc167658805)

[3.4.10. Bảng Processing\_Newspaper: 51](#_Toc167658806)

[3.4.11. Bảng Provisional\_Invoice: 52](#_Toc167658807)

[3.4.12. Bảng Invoice: 52](#_Toc167658808)

[3.4.13. Bảng Refresh\_Tokens: 53](#_Toc167658809)

[3.5. RESTful API: 53](#_Toc167658810)

[3.6. Giao diện: 57](#_Toc167658811)

[3.6.1. Trang đăng nhập: 57](#_Toc167658812)

[3.6.2. Trang “Trang Chủ”: 57](#_Toc167658813)

[3.6.3. Trang “Nhà Hàng”: 58](#_Toc167658814)

[3.6.4. Trang “Bàn”: 61](#_Toc167658815)

[3.6.5. Trang “Thực Đơn”: 61](#_Toc167658816)

[3.6.6. Trang “Khách Hàng”: 62](#_Toc167658817)

[3.6.7. Trang “Nhân Viên”: 62](#_Toc167658818)

[**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 63](#_Toc167658819)

[**TÀI LIỆU THAM KHẢO** 65](#_Toc167658820)

# CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

* 1. Lý do chọn đề tài:

Trong thời đại công nghiệp 4.0 hiện nay, Internet ngày càng được phổ biến sử dụng mạnh mẽ, từ đó công nghệ thông tin đang được phát triển nhanh chóng xâm nhập vào nhiều lĩnh vực khoa học, kỹ thuật cũng như trong cuộc sống chúng ta. Công nghệ mới càng ngày được phát triển hơn, nó trở thành môt công cụ đắc lực trong nhiều ngành nghề như giao thông, quân sự, y học,… đặc biệt trong ngành dịch vụ quản lý nó được áp dụng ngày càng rộng rãi.

Trước đây, khi máy tính chưa được ứng dụng rộng rãi, các công việc quản lý nhà hàng đều được làm thủ công từ khâu kinh doanh bán hàng đến nhập khẩu hàng hoá - thực phẩm nên rất mất nhiều thời gian, tiền bạc và tốn kém về nhân lực cũng như tài chính. Ngày nay, với sự phát triển công nghệ thông tin và internet ngày càng rộng rãi mà hầu hết máy tính đều đã được sử dụng nhiều trong các cơ quan, nhà máy, trường học… giúp cho công việc ngày càng tốt hơn. Việc sử dụng máy tính cùng với internet vào công tác quản lý nhà hàng là một phương pháp tốt nhất nhằm xoá bỏ những nhược điểm quản lý lạc hậu, lỗi thời tốn kém về nhân lực và tài chính ở nhiều mặt.

Vì vậy, em đã chọn đề tài: “Tìm hiểu Java Spring Framework và xây dựng ứng dụng website quản lý nhà hàng” với mong muốn áp dụng công nghệ Java Spring vào xây dựng website quản lý nhà hàng nhằm giúp có việc quản lý nhà hàng một cách hiệu quả và hoàn thiện, áp dụng cho các hệ thống nhà hàng kinh doanh.

* 1. Đối tượng và phạm vi đề tài:

Khoá luận sẽ tập trung trình bày kết quả nghiên cứu của em về các nội dung: Spring Framework, Angular Framework , MySQL và Bootstrap. Mỗi phần nội dung sẽ được trình bày giới thiệu sơ lược và giải thích những nội dung cơ bản nhất, những điểm mạnh hay lợi ích mà nó đem lại cho các nhà phát triển phần mềm.

Về Spring Framework em sẽ tập trung tìm hiểu và trình bày các phần: Spring Boot, Spring Security và Spring Data JPA.

Sau khi tìm hiểu em sẽ vận dụng kết quả tìm hiểu được vào việc xây dựng một ứng dụng website quản lý nhà hàng nhằm mục đích minh hoạ phần lý thuyết đã trình bày.

* 1. Phương pháp nghiên cứu:

Tìm hiểu và nghiên cứa các tài liệu về Java Spring Framework và các công nghệ phát triển phần mềm xây dựng website như Anuglar, MySQL, Bootstrap… của các tác giả trong và ngoài nước, các bài viết, các thông tin trang mạng trên cơ sở đó để tìm hiểu chọn lọc và sắp xếp lại theo ý tưởng của mình từ đó phân tích, thiết kế hệ thống ứng dụng website quản lý nhà hàng.

Dựa trên kết quả tìm hiều được để xây dựng một ứng dụng website quản lý nhà hàng có sử dụng các nội dung đã nghiên cứa nhằm mục đích minh hoạ cho phần cơ sở lý thuyết sẽ trình bày trong nội dung khoá luận này.

Các công cụ phần mềm hỗ trợ nghiên cứu:

* Microsoft Word: dùng để viết tài liệu báo cáo khoá luận tốt nghiệp.
* Intellij IDE: dùng để lập trình code Backend Java Spring.
* Visual Studio Code: dùng để lập trình code Fontend Angular.
* MySQL Workbench: dùng để lưu trữ cơ sở dữ liệu.
* Postman: dùng để kiểm thử test các API.
* StarUML: dùng để thiết kế các biểu đồ UML.
  1. Bố cục khoá luận:

Chương 1: Giới thiệu tổng quan về để tài; lý do chọn đề tài; trình bày các vấn đề cụ thể cần giải quyết và hướng giải quyết cho từng vấn đề; đối tượng, phạm vi nghiên cứu và phương pháp nghiên cứu đề tài.

Chương 2: Trình bày về cơ sở lý thuyết được nghiên cứu và sử dụng trong đề tài bao gồm các nội dung liên quan về ngôn ngữ lập trình hướng đối tượng Java, Framework Java Spring, Angular, MySQL, Bootstrap,…

* Trong phần Spring Framework em sẽ giới thiệu cơ bản về công nghệ framework Java Spring này và một số kiến thức liên quan, tiếp theo sẽ trình bày về 3 module dự án của nó là Spring Boot, Spring Security và Spring Data JPA.
* Phần Angular sẽ trình bày tổng quan cơ bản về framework fontend này và những đặc trưng nổi bật của nó.
* Phần tiếp theo sẽ trình bày về hệ quản trị cơ sở dữ liệu MySQL, giới thiệu về MySQL và ưu điểm của hệ quản trị CSDL này.
* Phần tiếp theo sẽ giới thiệu về Bootstrap hỗ trợ trong việc xây dựng giao diện người dùng cho ứng dụng web.

Chương 3: Sau khi tìm hiểu và nghiên cứa về các công nghệ sử dụng ở trước đó thì chương này trình bày phần phân tích, thiêt kế hệ thống và cài đặt ứng dụng website quản lý nhà hàng sử dụng các công nghệ ở trên.

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2. 1. Ngôn ngữ lập trình Java:
      1. Java và lịch sử phát triển:

Java là một ngôn ngữ lập trình dạng lập trình hướng đối tượng . Khác với các ngôn ngữ lập trình thông thường, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch bằng công cụ javac để chuyển thành dạng ByteCode. Sau đó được thực thi trên từng loại máy cụ thể nhờ chương trình thông dịch Java.

Cú pháp Java được xây dựng trên nền tảng của C và C++ nhưng có cú pháp hướng đối tượng đơn giản hơn. Do đó viết một chương trình Java dễ hơn, đơn giản hơn, đỡ tốn thời gian sửa lỗi hơn trong công việc.

Java được khởi đầu bởi James-Gosling và bạn đồng nghiệp ở Sun Microsystems năm 1991. Ban đầu ngôn ngữ này được gọi là Oak (có nghĩa là cây sồi) do phía sau cơ quan của ông James-Gosling có trồng cây đó, sau này ông thấy đã có ngôn ngữ lập trình tên Oak rồi, do vậy nhóm thiết kế quyết định đổi tên và “Java” là cái tên được chọn, Java là tên của một quán café mà nhóm thiết kế java hay đến đó uống.

Java được Sun Microsystems giới thiệu vào tháng 6 năm 1995. Từ đó, nó đã được trở thành một công cụ lập trình của các lập trình viên chuyên nghiệp.

Lịch sử phiên bản Java:

1. JDK Beta : năm 1995
2. JDK 1.0: ngày 23 tháng 1 năm 1996
3. JDK 1.1: ngày 19 tháng 2 năm 1997
4. J2SE 1.2: ngày 8 tháng 12 năm 1998
5. J2SE 1.3: ngày 8 tháng 5 năm 2000
6. J2SE 1.4: ngày 6 tháng 2 năm 2002
7. J2SE 5.0: ngày 30 tháng 9 năm 2004
8. Java SE 6: ngày 11 tháng 12 năm 2006
9. Java SE 7: ngày 28 tháng 7 năm 2011
10. Java SE 8: ngày 18 tháng 3 năm 2014
11. Java SE 9: ngày 21 tháng 9 năm 2017
12. Java SE 10: ngày 20 tháng 3 năm 2018
13. Java SE 11: ngày 25 tháng 9 năm 2018
14. Java SE 12: ngày 19 tháng 3 năm 2019
15. Java SE 13: ngày 17 tháng 9 năm 2019
16. Java SE 14: ngày 17 tháng 3 năm 2020
17. Java SE 15: ngày 15 tháng 9 năm 2020
18. Java SE 16: ngày 16 tháng 3 năm 2021
19. Java SE 17: ngày 14 tháng 9 năm 2021
20. Java SE 18: ngày 22 tháng 3 năm 2022
21. Java SE 19: ngày 20 tháng 9 năm 2022
22. Java SE 20: ngày 21 tháng 3 năm 2023
23. Java SE 21 (LTS): ngày 19 tháng 9 năm 2023
    * 1. Một số tính năng của Java:

* Đa nền tảng: Java được thiết kế để có thể chạy trên nhiều nền tảng khác nhau, vì vậy nó rất phù hợp cho việc phát triển các ứng dụng đa nền tảng. Java sử dụng một máy ảo JVM để chạy mã nguồn, vì vậy mã nguồn được viết một lần và có thể chạy trên nhiều hệ điều hành khác nhau mà không cần thay đổi.
* Quản lý bộ nhớ tự động: Java có tính năng tự động quản lý bộ nhớ, tức là nó tự động thu dọn các vùng nhớ không sử dụng nữa để giảm thiểu các lỗi bộ nhớ. Điều này giúp cho các ứng dụng được viết bằng Java có thể chạy ổn định và tránh các lỗi liên quan đến bộ nhớ.
* Hỗ trợ đa luồng: Java có thể xử lý đa luồng, cho phép chương trình thực hiện nhiều tác vụ cùng một lúc. Điều này giúp cho các ứng dụng có thể chạy nhanh và hiệu quả hơn, đặc biệt là khi phải xử lý nhiều tác vụ cùng một lúc.
* Tính bảo mật cao: Java có các tính năng bảo mật như kiểm tra kiểu tĩnh và kiểm tra lỗi trên đường dẫn. Java được thiết kế để giảm thiểu các lỗ hổng bảo mật và các vấn đề liên quan đến an ninh.
* Hỗ trợ các thư viện và công cụ phát triên mãnh mẽ: Java có các thư viện tiêu chuẩn và các công cụ phát triển như IDE để giúp các nhà phát triển tạo ra các ứng dụng dễ dàng hơn. Các thư viện và công cụ này cung cấp các tính năng như: tạo giao diện người dùng, xử lý dữ liệu, kết nối cơ sở dữ liệu và nhiều hơn nữa...
* Tính kế thừa và đa hình: Java là ngôn ngữ lập trình hướng đối tượng, vì vậy nó có các tính năng như kế thừa, đa hình và đóng gói. Các tính năng này giúp cho mã nguồn được tái sử dụng và giảm thiểu sự trùng lặp trong mã nguồn.
* Tính mở rộng: Java có tính năng mở rộng, cho phép các nhà phát triển thêm các tính năng mới vào ngôn ngữ bằng cách tạo các thư viện và API riêng. Điều này giúp cho Java có thể được sử dụng trong nhiều lĩnh vực khác nhau và được phát triển theo các hướng khác nhau.
  + 1. Ưu và nhược điểm của Java:

Ưu điểm của Java:

* Độ tin cậy cao.
* Tính đa nền tảng.
* Quản lý bộ nhớ tự động.
* Công cụ phát triên phong phú.
* Hỗ trợ đa luồng.

Nhược điểm của Java:

* Tốc độ chậm hơn so với các ngôn ngữ lập trình gần sát với phần cứng, chẳng hạn như C hoặc C++.
* Java có thể chạy trên nhiều nền tảng khác nhau, nhưng ứng dụng này có thể cần đến một trình biên dịch hoặc máy ảo Java riêng biệt để có thể chạy trên các thiết bị di động.
* Sử dụng bộ nhớ lớn hơn so với một số ngôn ngữ lập trình khác.
* Cú pháp phức tạp hơn so với một số ngôn ngữ lập trình khác.
  1. Spring Framework:
     1. Tổng quan Spring:

Spring framework hay còn gọi ngắn hơn là Spring, là một trong những mã nguồn mở được xây dựng dựa trên nền tảng ngôn ngữ lập trình Java và được sử dụng bởi hàng triệu lập trình viên cho đến bây giờ. Phiên bản đầu tiên của Spring được viết bởi Rod Johnson và đã được ông xuất bản cùng với quyển sách đầu tay “Expert One-on-One J2EE Design and Development” vào tháng 10 năm 2002 .

Kiến trúc Spring được phát hành vào tháng 6 năm 2003 bởi Giấy phép Apache 2.0. Phiên bản 1.0 được phát hành tháng 3 năm 2004 đánh dấu cột mốc đầu tiền và lần lượt phát hành kế tiếp tháng 9 năm 2004 và tháng 3 năm 2005.

Spring khá nhẹ với phiên bản chỉ khoảng 2MB với kích thước nhỏ và nó hoạt động trong suốt với một lập trình viên. Spring cung cấp hỗ trợ các cơ sở hạ tầng phần mềm đã được xác định rõ ràng để phát triển ứng dụng trong Java. Nói dễ hiểu hơn, Spring đã xử lý cơ sở hạ tầng để người lập trình viên tập trung tối đa vào phát triển ứng dụng.

Bên cạnh những đặc trưng nền tảng của Spring Framework là những cái có thể dùng trong bất kỳ một chương trình Java nào, rất nhiều các mở rộng và tiến bộ trong việc kiến tạo các trình ứng dụng dành riêng nền tảng website dựa trên nền tản Java Enterprise cũng tồn tại nữa. Spring Framework nổi tiếng cũng một phần do chính đặc thù kể trên và được thị trường công nhận như một nền tảng kiến trúc có tầm quan trọng trong chiến lược kiến tạo phần mềm.

* + 1. Lịch sử phát triển:

Phần đầu tiên của Spring Framework ban đầu nguyên được [Rod Johnson](https://vi.wikipedia.org/w/index.php?title=Rod_Johnson&action=edit&redlink=1) viết vào năm 2000. Trước đây, vào năm 2001, những mô hình lập trình cho các trình ứng dụng cho mạng WEB được đại đa số sử dụng đều do [Java Servlet API](https://vi.wikipedia.org/w/index.php?title=Java_Servlet_API&action=edit&redlink=1) và [Enterprise JavaBean](https://vi.wikipedia.org/w/index.php?title=Enterprise_JavaBean&action=edit&redlink=1) cung cấp. Cả hai đều là đặc tả do [Sun Microsystems](https://vi.wikipedia.org/wiki/Sun_Microsystems) kiến tạo với sự cộng tác của các nhà buôn bán phần mềm và các bên quan tâm, và chúng rất được mến chuộng trong cộng đồng Java.

Vào tháng 2 năm 2003, thành lập dự án tại [Sourceforge](https://vi.wikipedia.org/wiki/SourceForge) để phát triển framwork Spring.  Sau khi phát triển trền nền tảng ban đầu hơn một năm họ đã phát hành phiên bản đầu tiên (1.0) vào tháng 3 năm 2004. Sau khi phát hành nó đã nhận được nhiều sự quan tâm của cộng đồng Java.

Spring Framework đã làm cho những kỹ thuật vốn không được biết đến mấy trở nên những kỹ thuật được mọi người ưa chuộng trong một thời gian ngắn ngủ. Một trong những kỹ thuật nổi tiếng hơn cả là kỹ thuật [đảo ngược quyền điều khiển](https://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BA%A3o_ng%C6%B0%E1%BB%A3c_quy%E1%BB%81n_kh%E1%BB%91ng_ch%E1%BA%BF&action=edit&redlink=1) IoC.

Vào trong năm 2004, Spring Framework được rất nhiều người hưởng ứng và làm theo và cũng do tự chính nó cống hiến khung hình [lập trình hướng khía cạnh](https://vi.wikipedia.org/w/index.php?title=L%E1%BA%ADp_tr%C3%ACnh_%C4%91%E1%BB%8Bnh_d%E1%BA%A1ng&action=edit&redlink=1) AOP. Spring Framework đã tạo cơ hội cho mẫu hình lập trình này trở nên được ưa thích trong cộng đồng Java

Năm 2005 cho thấy mức độ hưởng ứng nâng cao hơn những năm trước, nguyên do cũng vì những xuất bản mới được ra với những cột mốc đáng kể và những tính năng mới được thêm vào. [Diễn đàn Spring](http://forum.springframework.org/) (Spring Forum) ra đời cuối năm 2004 cũng góp phần không nhỏ trong việc nâng cao tính phổ biến của Spring framework và từ đó đến nay đã sinh trưởng trở thành một nguồn thông tin quan trọng, giúp đỡ cho người sử dụng.

 Vào tháng 12 năm 2005, hội thảo đầu tiên về Spring Framework đã được tổ chức tại [Miami, Florida](https://vi.wikipedia.org/wiki/Miami,_Florida) thu hút 300 nhà lập trình viên trong 3 ngày, và tiếp theo đó cuộc hội thảo ở [Antwerp](https://vi.wikipedia.org/wiki/Antwerpen) vào tháng 6 năm 2006, thu hút hơn 400 người.

* + 1. Một số khái niệm chính:

Hai trong những thành phần chủ chốt và là nền tảng tạo nên sức mạnh của Spring là IoC và DI.

* + - 1. Inversion of Control (IoC):

IoC Container trong Spring được xây dựng dựa trên nguyên lý Inversion of Control (đảo ngược điều khiển) đã xuất hiện khá lâu trong các mẫu thiết kế và được phổ biến rộng rãi nhờ Robert C.Martin và Martin Fowler.

IoC là một mẫu thiết kế và là tập hợp các kỹ thuật lập trình, trong đó luồng thực thi của một hệ thống bị đảo ngược so với cách tương tác truyền thống. nó giúp làm thay đổi luồng điều khiển của chương trình một cách linh hoạt.

Khái niệm “đảo ngược” ở đây chính là chuyển nhiệm vụ kiểm soát lưu trình thực thi từ ứng dụng cho một thành phần chuyên trách thường đó là một thư viện khung framework dựng sẵn bên ngoài. Nhờ đó, ứng dụng chính của chúng ta quan tâm phát triển không kiểm soát việc điều khiển luồng thực thi nữa, mà chỉ tập trung vào việc định nghĩa chức năng. Thư viện phần mềm khung chuyên trách kiểm soát điều khiển sẽ dựa trên mô tả trong cấu hình của ứng dụng để thay mặt ứng dụng điều phối luồng thực thi trong chương trình.

Dễ dàng nhận thấy rằng thư viện phần mềm khung này khác với các thư viện thông thường ở chỗ: thư viện thông thường cung cấp các chức năng và chờ được ứng dụng gọi đến, còn thư viện phần mềm khung được tạo dựng luồng thực thi và gọi đến các chức năng của ứng dụng.

IoC trong Spring cũng hoàn toàn mang ý nghĩa như trên. Trong Spring các đối tượng chính để xây dựng ứng dụng thì được quản lý bởi Spring IoC container. Và IoC container gọi các đối tượng đó là các bean. Một bean chỉ đơn giản là một đối tượng được khởi tạo và quản lý bởi Spring IoC container.

* + - 1. [Dependency Injection](https://vi.wikipedia.org/w/index.php?title=Dependency_Injection&action=edit&redlink=1) (DI):

Dependency Injection (tiêm phụ thuộc) là khả năng liên kết giữa các thành phần lại với nhau, đó chính là việc các thuộc tính trong một đối tượng được “tiêm chích” để tham chiếu lần lượt đến các đối tượng khác. Dependency Injection trong Spring hoạt động dựa trên Java Reflection.

* + 1. Kiến trúc và các module của Spring Framework :

Spring Framework được chia làm nhiều module khác nhau, tuỳ theo mục đích phát triển ứng dụng mà ta dùng các module đó.

Dưới đây là kiến trúc tổng thể của Spring Framework:



Hình 2.1: Kiến trúc tổng thể của Spring Framework

2. 2. 4. 1. Core Container:

Core package là phần cơ bản nhất của Spring, cung cấp những đặc tính như IoC và DI. Khái niệm cơ bản là BeanFactory, một cài đặt của mẫu thiết kế Factory cho phép móc nối sự phụ thuộc giữa các đối tượng trong file cấu hình.

Core Container bao gồm các module:**spring-core, spring-beans, spring-context** và**spring-expression (**Spring Expression Language**).**

* Spring-core và Spring-beans cung cấp các phần cơ bản của framework, bao gồm IoC và DI.
* Spring-context hỗ trợ đa ngôn ngữ và tích hợp các tính năng Java EE như EJB và JMX.
* Spring-expression cung cấp hỗ trợ cho việc đặt và lấy giá trị, toán tử logic và truy vấn các tập hợp dữ liệu.
  + - 1. Data Access/Integration:

Data Access/Integration bao gồm JDBC, ORM, OXM, JMS và module Transaction. Những module này cung cấp khả năng giao tiếp với database.

* Spring-jdbc giúp giảm thiểu mã JDBC và hỗ trợ nhiều cơ sở dữ liệu.
* Spring-orm tích hợp với các orm API phổ biến như JPA, JDO và Hibernate.
* Spring-oxm hỗ trợ Object/XML mapping với nhiều công cụ như AXB, Castor, XMLBeans và nhiều công cụ khác.
* Spring-jms cung cấp khả năng tạo và sử dụng các message và tích hợp với spring-messaging.
* Spring-transaction hỗ trợ quản lý giao dịch cho các POJO và các lớp đặc biệt
  + - 1. Web:

Spring cung cấp Spring MVC để xây dựng ứng dụng web. Module bao gồm: spring-web, spring-webmvc, spring-websocket và springwebmvc-portlet.

* Spring-web hỗ trợ tích hợp web, chức năng tải tệp và khởi tạo IoC container sử dụng servlet.
* Spring-webmvc triển khai Model-View-Controller cho ứng dụng web.
* Spring-websocket hỗ trợ WebSocket-based, giao tiếp hai chiều giữa client và server trong các ứng dụng web.
* Springwebmvc-portlet dành cho môi trường portlet và ánh xạ chức năng của module Web-Servlet.
  + - 1. AOP, Aspects, Instrumentation và Messaging:

Những module này hỗ trợ lập trình hướng khía cạnh (Aspect Oriented Programming) và tích hợp với AspectJ.

* Module AOP cung cấp lập trình hướng khía cạnh, cho phép tách rời mã và các chức năng của ứng dụng.
* Module Aspects tích hợp với AspectJ, một khuôn khổ AOP mạnh mẽ.
* Module Instrumentation hỗ trợ đo đạc và triển khai lớp bộ nạp.
* Module Messaging hỗ trợ STOMP và WebSocket, cung cấp mô hình lập trình chú thích cho định tuyến và xử lý tin nhắn STOMP.
  + - 1. Test:

Spring cung cấp khả năng hỗ trợ kiểm thử với Junit và TestNG.

* + 1. Spring Boot:
       1. Khái niệm và tổng quát:

Spring Boot là một dự án phát triển bởi Java trong hệ sinh thái Spring Framework. Nó giúp cho các lập trình viên đơn giản hoá quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển business cho ứng dụng.

Spring Boot hoạt động nhằm tối ưu hóa quy trình phát triển ứng dụng Java. Điều này sẽ giúp nhà phát triển tập trung vào việc xây dựng tính năng chính của ứng dụng mà không cần phải lo lắng về cấu hình phức tạp.

Spring Boot được phát triển bởi công ty Pivotal Software (nay thuộc VMware) với mục đích làm đơn giản hóa quá trình phát triển ứng dụng Spring. Công cụ sẽ giảm khối lượng công việc cần thiết để cấu hình và triển khai ứng dụng. Bằng cách tự động hoá cấu hình và cung cấp các cấu hình mặc định thông minh, Spring Boot giúp nhà phát triển nâng cao sự tập trung vào việc xây dựng các tính năng chính của ứng dụng.

* + - 1. Ưu điểm nổi bật của Spring Boot:

Một số ưu điểm dưới đây đã giúp Spring Boot trở thành sự lựa chọn phổ biến cho việc xây dựng các ứng dụng và dịch vụ web trong hệ sinh thái Java:

* Tối ưu hóa quá trình phát triển: Spring Boot cung cấp cấu hình mặc định thông minh và tự động, giúp giảm thiểu việc cấu hình thủ công và tối ưu quá trình phát triển ứng dụng Java.
* Tích hợp tốt: Spring Boot tích hợp tốt với nhiều công nghệ và thư viện khác trong hệ sinh thái Spring Framework. Nền tảng cho phép hệ thống dễ dàng tích hợp các module và dịch vụ khác nhau mà không cần phải lo lắng về cấu hình phức tạp.
* Embedded server: Spring Boot đi kèm với các máy chủ nhúng như Tomcat, Jetty, hoặc Undertow. Đây là công cụ không thể thiếu trong việc triển khai ứng dụng một cách đơn giản mà không cần cấu hình thêm bất kỳ máy chủ nào khác.
* Tự động cấu hình: Spring Boot sử dụng cơ chế cấu hình tự động thông minh, cho phép ứng dụng tự cấu hình dựa trên các thư viện và module được sử dụng.
* Quản lý phụ thuộc: Spring Boot cung cấp các công cụ quản lý phụ thuộc mạnh mẽ như Maven hoặc Gradle, giúp quản lý các phụ thuộc của ứng dụng một cách hiệu quả.
* Monitoring và quản lý: Spring Boot cung cấp các công cụ hỗ trợ giám sát và quản lý ứng dụng dễ dàng, bao gồm Spring Boot Actuator cho việc giám sát và quản lý ứng dụng.
  + 1. Spring Security:
       1. Tổng quan về Spring Security:

Spring Security được phát triển bởi SpringSource (hiện thuộc [Pivotal](https://en.wikipedia.org/wiki/Pivotal_Software?ref=200lab.io)) và được xem là một trong những framework bảo mật phổ biến nhất cho ứng dụng Java.

Spring Security cung cấp các tính năng xác thực (authentication) và phân quyền (authorization) cho các ứng dụng, cũng như hỗ trợ các tiêu chuẩn và giao thức bảo mật như HTTPS, OAuth2, JWT, LDAP, SAML, OpenID Connect.

Spring Security được thiết kế theo kiến trúc plugin, cho phép tùy biến linh hoạt và dễ dàng theo nhu cầu của ứng dụng và được tích hợp sẵn với các thành phần khác của Spring Framework, như Spring Boot, Spring MVC, Spring Data, Spring Cloud, và Spring WebFlux.

* + - 1. Cơ chế hoạt động của Spring Security:

**Spring Security** hoạt động theo mô hình client-server. Khi một client gửi một request đến server, server sẽ xác thực người dùng và phân quyền để đảm bảo rằng người dùng chỉ có thể truy cập vào những tài nguyên mà họ được phép truy cập.

Cơ chế hoạt động của**Spring Security**dựa trên cơ chế filter và sự kiện để can thiệp vào quá trình xử lý yêu cầu request và phản hồi response của ứng dụng web, tức là khi một yêu cầu được gửi đến ứng dụng web, nó sẽ được chuyển qua một chuỗi các bộ lọc filter chain do Spring Security quản lý. Mỗi bộ lọc có một nhiệm vụ cụ thể, như kiểm tra xác thực, kiểm tra phân quyền, điều hướng đến trang đăng nhập hoặc đăng xuất, xử lý các lỗi bảo mật.

Nếu một yêu cầu không thỏa mãn các điều kiện bảo mật của ứng dụng, **Spring Security** sẽ sinh ra một sự kiện để thông báo cho ứng dụng biết. Ngược lại, nếu một yêu cầu được chấp nhận bởi **Spring Security**, nó sẽ được tiếp tục xử lý bởi ứng dụng web như bình thường. Khi ứng dụng web trả về một phản hồi cho yêu cầu, nó cũng sẽ được chuyển qua lại chuỗi các bộ lọc của **Spring Security** để áp dụng các thiết lập bảo mật cho phản hồi.

Cơ chế hoạt động của Spring Security gồm ba thành phần chính:

* Authentication : Xác thực.
* Authorization : Uỷ quyền.
* Authentication Provider: Nhà cung cấp xác thực.
  + - 1. Authentication:

**Authentication** là quá trình xác thực xem người dùng có quyền truy cập vào ứng dụng hay không. Khi người dùng đăng nhập vào hệ thống, thông tin đăng nhập của họ sẽ được xác thực để đảm bảo rằng họ là người dùng hợp lệ và có quyền truy cập vào các tài nguyên yêu cầu.

**Authentication** thường dựa trên các thông tin nhận dạng (identifier) và thông tin bí mật (credential) của người dùng hoặc ứng dụng, ví dụ như tên đăng nhập và mật khẩu, mã token, vân tay, khuôn mặt tùy theo cách tiếp cận của ứng dụng đó.

SpringSecurity hỗ trợ xác thực thông qua một số cơ chế, bao gồm:

* Form-based authentication: Xác thực thông qua một form đăng nhập.
* HTTP Basic authentication: Xác thực thông qua các header authorization.
* Authentication via a custom login page: Xác thực thông qua một trang đăng nhập tùy chỉnh.
* Pre-authenticated authentication: Xác thực thông qua các giá trị được cung cấp từ phía máy khách.
  + - 1. Authorization:

**Authorization** là quá trình xác định quyền truy cập của người dùng đối với các tài nguyên trong ứng dụng. Khi người dùng truy cập vào một tài nguyên, **Spring Security**sẽ kiểm tra xem người dùng có được phép truy cập vào tài nguyên đó hay không hoặc thực hiện một hành động nào đó trong hệ thống.

**Authorization** thường dựa trên các thông tin về vai trò role, nhóm group, quyền hạn permission, chính sách policy. Ngoài ra, còn giúp đảm bảo rằng chỉ những người dùng hoặc ứng dụng có quyền thích hợp mới có thể truy cập vào tài nguyên hoặc thực hiện hành động được bảo vệ.

**Spring Security** hỗ trợ phân quyền ứng dụng web bằng cách sử dụng các annotation hoặc XML để định nghĩa các quyền truy cập cho người dùng. Các quyền truy cập này được áp dụng cho các tài nguyên trong ứng dụng, chẳng hạn như trang web hoặc API.

* + - 1. Authentication Provider:

**Authentication Provider** là một thành phần quan trọng trong **Spring Security** chịu trách nhiệm xác minh thông tin xác thực của người dùng hoặc ứng dụng.

Ví dụ, khi một người dùng đăng nhập vào hệ thống, Authentication Provider sẽ kiểm tra thông tin đăng nhập của người dùng và trả về kết quả xác thực.

**Authentication Provider** được sử dụng bởi **Authentication Manager** để xử lý yêu cầu xác thực từ người dùng hoặc ứng dụng.

Mỗi Authentication Provider chỉ hỗ trợ một loại Authentication cụ thể, ví dụ như UsernamePasswordAuthenticationToken, JwtAuthenticationToken, PreAuthenticatedAuthenticationToken,..

* + - 1. Các tính năng nâng cao của Spring Secutity:

SpringSecurity không chỉ hỗ trợ xác thực và phân quyền cơ bản, mà còn cung cấp nhiều tính năng nâng cao để bảo vệ ứng dụng web của bạn. Một số tính năng nâng cao của SpringSecurity bao gồm:

* CSRF protection: bảo vệ chống lại tấn công CSRF.
* Session management: quản lý phiên.
* Password encoding: mã hóa mật khẩu.

Chúng ta có thể kích hoạt và tùy chỉnh các tính năng nâng cao này thông qua các annotation, XML, hoặc Java configuration trong Spring Security.

* + - 1. Ưu và nhược điểm của Spring Securty:

Ưu điểm của Spring Security:

* SpringSecurity là một framework bảo mật mạnh mẽ và linh hoạt, hỗ trợ rất nhiều tiêu chuẩn và giao thức bảo mật.
* Được tích hợp sẵn với SpringFramework, giúp việc phát triển ứng dụng web an toàn và hiệu quả hơn.
* SpringSecurity có một cộng đồng lớn và sôi động, với rất nhiều tài liệu hướng dẫn và ví dụ minh họa.

Nhược điểm của Spring Security:

* Cấu hình SpringSecurity có thể khá phức tạp và khó hiểu, đặc biệt là khi làm việc với các tính năng nâng cao.
* Một số tính năng củaSpringSecurity có thể không phù hợp với  loại ứng dụng web, ví dụ như ứng dụng web không sử dụng Spring Framework. Và cả các ứng dụng có quy mô lớn hoặc các ứng dụng yêu cầu tốc độ phản hồi cao.
* Yêu cầu kiến thức chuyên môn về bảo mật để sử dụng hiệu quả.
  + 1. Spring Data JPA:
       1. Tổng quan về Spring Data JPA:

Spring Data JPA là dự án thuộc module Spring Data trong Spring Framework. Nó cung cấp một cách dễ dàng để làm việc với cơ sở dữ liệu trong ứng dụng Spring. Spring Data JPA tự động tạo các truy vấn SQL dựa trên các phương thức trong các repository và cung cấp tích hợp mạnh mẽ với Hibernate.

Spring Data JPA dựa trên tiêu chuẩn JPA cho phép chúng ta truy cập và lưu trữ dữ liệu giữa các đối tượng Java và database. JPA tuân theo ORM, là kỹ thuật được sử dụng trong lập trình để ánh xạ giữa đối tượng trong mã nguồn và các bảng cơ sở dữ liệu quan hệ. Nó giúp chúng ta làm việc với cơ sở dữ liệu một cách trừu tượng hoá, tức là thay vì trước đây sử dụng truy vấn SQL trực tiếp bằng lớp DAO chúng ta sử dụng các đối tượng Java để thực hiện thao tác cơ sở dữ liệu.

Sử dụng JPA đơn giản hơn, gọn gàng hơn và ít tốn sức hơn là JDBC, SQL hay ánh xạ cơ sở dữ liệu viết tay. JPA thích hợp cho các ứng dụng phức tạp không tuân theo hiệu suất. Ưu điểm chính của JPA so với JDBC là: trong JPA, dữ liệu được biểu thị bằng các object và class trong khi trong JDBC dữ liệu được biểu diễn bằng các bảng và bản ghi. Nó sử dụng POJO để biểu diễn dữ liệu liên tục giúp đơn giản hóa việc lập trình database.

* + - 1. Một số lợi ích sử dụng Spring Data JPA:
* Giảm đoạn mã code : Sử dụng JPA, chúng ta có thể sử dụng các đối tượng Java thay vì phải viết các câu truy vấn SQL hoặc sử dụng JDBC trực tiếp. Điều này làm cho code trở nên ngắn gọn hơn và dễ đọc hơn, giảm thiểu tối đa mã code.
* Dễ bảo trì: Với JPA, việc thay đổi cấu trúc cơ sở dữ liệu trở nên dễ dàng hơn. Chúng ta chỉ cần thay đổi định nghĩa của đối tượng Java và JPA sẽ tự động áp dụng các thay đổi này vào cơ sở dữ liệu tương ứng.
* Hiệu suất cao: JPA giúp tối ưu hóa việc truy xuất và lưu trữ dữ liệu trong cơ sở dữ liệu. Nó sử dụng các kỹ thuật như lazy loading và caching để giảm thiểu số lần truy cập vào cơ sở dữ liệu.
* **Tích hợp với Spring Framework**: Spring JPA được tích hợp chặt chẽ với Spring Framework, cho phép bạn sử dụng các tính năng khác của Spring như Dependency Injection, Transaction Management và Security.
  1. Angular:
     1. Tổng quan về Angular:

Công nghệ HTML hỗ trợ tốt cho các trang web tĩnh, kiểu như trước năm 2000 vậy. Khi chúng ta xây dựng một trang web với PHP thì nó cũng chỉ là một trang web tĩnh với nội dung được thay đổi khi chúng ta gửi request về máy chủ, máy chủ sẽ trả lại một trang với nội dung tương ứng. Tuy nhiên mọi thứ đã thay đổi nhiều từ sự ra đời và phát triển của HTML5, nhất là sự hỗ trợ từ các tập đoàn công nghệ lớn như Google. Facebook và sự tập trung đông đảo của cộng đồng mã nguồn mở.

Ngoài ra sự thành công của jQuery đã khiến JavaScript được nhiều người yêu thích vì tính đơn giản và dễ sử dụng lập trình giao diện. Việc phát triển một website sử dụng AJAX thì không khó, chúng ta có thể dùng jQuery để làm việc với nó tuy nhiên làm thế nào để xây dựng một phần mềm có thể mở rộng, dễ kiểm thử, nâng cấp và bảo trì thì không hề đơn giản vì bản thân JavaScript không được thiết kế ngay từ đầu để làm những việc này. Do đó sự ra đời của những framework hỗ trợ lập trình viên xây dựng ứng dụng web một cách có hệ thống đã ra đời như React, Vue.js, Angular…

Angular là một framework mã nguồn mở được viết bằng TypeScript và được sử dụng để thiết kế giao diện web Fontend. Angular là một framework phổ biến được sử dụng để xây dựng ứng dụng web đơn trang SPA và các ứng dụng trang web động. Công cụ được phát triển bởi Google nhằm cung cấp giải pháp tiếp cận cấu trúc mạnh mẽ với tác dụng phát triển các ứng dụng web phức tạp.

Một Single Page Applications là một ứng dụng web hiển thị vừa vặn trên một mặt của trang web với mục đích giúp người dùng có trải nghiệm giống như đang dùng ứng dụng trên desktop và làm tăng hiệu suất của ứng dụng. Hay nói cách khác là ứng dụng chạy bên trong trình duyệt mà không yêu cầu phải tải lại toàn bộ trang web mỗi lần sử dụng.

Angular sử dụng TypeScript (phát triển dựa trên JavaScript) để xây dựng ứng dụng và tích hợp các tính năng như Two-Way Data Binding, Dependency Injection, Routing, Forms. Nền tảng giúp cho các nhà phát triển xây dựng và duy trì mã nguồn một cách dễ dàng và hiệu quả.

* + 1. Lịch sử phát triển của Angular:

Ban đầu, Angular được phát triển bởi đội ngũ của Google và chính thức ra mắt lần đầu tiên dưới tên gọi AngularJS vào năm 2010. AngularJS nhanh chóng trở thành một trong những framework phổ biến nhất cho việc phát triển ứng dụng web đơn trang (SPA). Nền tảng cung cấp phương án xây dựng ứng dụng web phức tạp một cách dễ dàng hơn.

Vào năm 2016, Angular trình làng phiên bản hoàn toàn mới với tên gọi Angular 2 (sau đó đổi thành Angular). Phiên bản này được viết lại hoàn toàn và sử dụng TypeScript thay vì JavaScript như phiên bản AngularJS. Angular 2 (và các phiên bản tiếp theo) đã mang lại nhiều cải tiến và tính năng mới giúp cho việc phát triển ứng dụng web trở nên linh hoạt, mạnh mẽ và hiệu quả hơn.

Từ đó, Angular đã liên tục phát triển và được cập nhật với các phiên bản mới định kỳ mang lại tính năng, hiệu suất cải thiện đáng kể và phiên bản mới nhất hiện nay là Angular 17 phát hành vào ngày 8 tháng 11 năm 2023 . Angular đã trở thành một trong những framework phổ biến nhất trong việc phát triển ứng dụng web. Tiện ích còn được sử dụng rộng rãi trong cộng đồng phát triển phần mềm toàn cầu.

* + 1. Ưu điểm của Angular:

Angular có nhiều ưu điểm mạnh mẽ mà làm nó trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng web. Dưới đây là một số ưu điểm chính của Angular:

* TypeScript: Angular sử dụng TypeScript, một ngôn ngữ phát triển của JavaScript với kiểu dữ liệu tĩnh, cho phép phát triển ứng dụng web theo cách cấu trúc hơn, dễ quản lý hơn và dễ bảo trì hơn.
* Two-way data binding: Angular cung cấp tính năng two-way data binding giữa model và view. Nền tảng tự động đồng bộ dữ liệu giữa các thành phần, tiết kiệm thời gian và công sức cho nhà phát triển.
* Dependency Injection: Angular sử dụng Dependency Injection để quản lý và cung cấp các đối tượng cho các thành phần của ứng dụng một cách linh hoạt. Tiện ích tạo ra mã mãng nguồn dễ bảo trì và tái sử dụng.
* Routing: Angular cung cấp một hệ thống routing mạnh mẽ, cho phép ứng dụng chuyển đổi giữa các trang một cách dễ dàng và linh hoạt. Điều này đã mang đến những trải nghiệm tốt hơn cho người dùng.
* Forms: Angular cung cấp các tính năng mạnh mẽ cho việc xử lý và xác thực dữ liệu nhập từ người dùng, giúp quản lý form một cách hiệu quả và dễ dàng.
* Community and Ecosystem: Angular có một cộng đồng lớn và mạnh mẽ, đi kèm với một hệ sinh thái phong phú của các thư viện và công cụ hỗ trợ. Nền tảng giúp nhà phát triển dễ dàng tìm kiếm giải pháp và thông tin hỗ trợ khi cần.
  + 1. Nhược điểm của Angular:

Mặc dù Angular có nhiều ưu điểm, nhưng cũng cần lưu ý đến một số nhược điểm sau:

* Kích thước: Các ứng dụng Angular có thể có kích thước tải về lớn do cần phải bao gồm và tải về các thư viện và framework lớn như Angular core và các thư viện liên quan. Điều này có thể làm tăng thời gian tải ứng dụng và tốn bộ nhớ.
* Quản lý phiên bản: Do Angular liên tục phát triển và cập nhật chương trình mới có thể mang đến một vài khó khăn trong việc duy trì và nâng cấp phiên bản cho những dự án lớn.
* Hiệu suất: Mặc dù đã có nhiều cải tiến nhưng hiệu suất của ứng dụng Angular vẫn bị ảnh hưởng khi ứng dụng trở nên quá lớn và phức tạp.
  1. MySQL:
     1. Tổng quan về SQL:

SQL là viết tắt của Structured Query Language là ngôn ngữ truy vấn có cấu trúc. Nó là một ngôn ngữ, là tập hợp các lệnh để tương tác với cơ sở dữ liệu. Dùng để lưu trữ, thao tác và truy xuất dữ liệu được lưu trữ trong một cơ sở dữ liệu quan hệ. Trong thức tế, SQL là ngôn ngữ chuẩn được sử dụng hầu hết chp hệ cơ sở dữ liệu quan hệ. Tất cả các hệ thống quản lý cơ sở dữ liệu quan hệ (RDMS) như MySQL, MS Access, Oracle, Postgres và SQL Server,,, đều sử dụng SQL làm ngôn ngữ cơ sở dữ liệu chuẩn.

SQL bắt đầu từ những năm 1970, khi các kỹ sư của IBM là Donal Chamberlin và Raymond Boyce thiết kế phiên bản đầu tiên để tương tác và lấy dữ liệu được lưu trữ trong hệ thống cơ sở dữ liệu của công ty. Họ gọi nó là SEQUEL, mặc dù sau đó họ buộc phải thay đổi nó do các vấn đề bản quyền.

SQL được sử dụng phổ biến vì nó có các ưu điểm sau:

* Cho phép truy cập nhữ liệu trong các hệ thống quản lý cơ sở dữ liệu quan hệ.
* Cho phép mô tả dữ liệu.
* Cho phép xác định dữ liệu trong cơ sở dữ liệu và thao tác dữ liệu đó.
* Cho phép nhúng vào trong các ngôn ngữ sử dụng module SQL, thư viện và trình biên diệc tiếp.
* Cho phép tạo và thả các cơ sở dữ liệu bản.
* Cho phép toạ chế độ view, thủ tục lưu trữ, chức năng trong cơ sở dữ liệu.
* Cho phép thiết lăp quyền trên các bảng, thử tục và view.

Hầu như công ty nào lớn cũng cần xây dựng một hệ thống để lưu trữ cơ sở dữ liệu. Mọi thứ trong cơ sở dữ liệu này sẽ được diễn tả ra thành nhiều bảng, có mối quan hệ với nhau. Để truy vần là lấy dữ liệu từ các bảng này nhằm tổng hợp thành thông tin nào đó, người ta dùng đến SQL thông qua các câu query.

* + 1. MySQL:

MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (Relational Database Management System, viết tắt là RDBMS) hoạt động theo mô hình client-server. RDBMS là một phần mềm hay dịch vụ dùng để tạo và quản lý các cơ sở sữ liệu theo hình thức quản lý các mối liên hệ giữa chúng.

MySQL là một số các phần mềm RDBMS. RDBMS và MySQK thường được cho là một vì độ phổ biến quá lớn của MySQL. Các ứng dụng web lớn nhất như Facebook hay Google đều dùng MySQL cho mục đích lưu trữ dữ liệu. Kể cả khi ban đầu nó chỉ được dùng rất hạn chế nhưng giời nó đã tương thích với nhiều hạ tầng máy tính quan trọng như Linux, Windows, masOS và Ubuntu.

* + 1. Ưu điểm của MySQL:
* Dễ sử dụng: MySQL là cơ sở dữ liệu tốc độ cao, ổn định, dễ sử dụng và hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.
* Độ bảo mật cao: MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên Internet khi sở hữu nhiều tính năng bảo mật kể cả ở cấp cao hơn.
* Đa tính năng: MySQL hỗ trợ nhiều chức năng SQL được mong chờ từ một hệ quản trị cơ sở dữ liệu quan hệ cả trực tiếp và gián tiếp.
* Khả năng mở rộng mạnh mẽ: MySQL có thể xử lý rất nhiều dữ liệu và hơn thế nữa nó có thể mở rộng nếu cần thiết.
* Nhanh chóng: Việc đưa ra một số tiêu chuẩn cho phép MySQL để làm việc rất hiệu quả và tiết kiệm chi phí, do đó nó làm tăng tốc độ thực thi.
  + 1. Nhược điểm của MySQL:
* Giới hạn: theo thiết kế, MySQL không có ý định làm tất cả và nó đi kèm với các hạn chế về chức năng mà ứng dụng có thể cần.
* Độ tin cậy: cách các chức năng cụ thể được xử lý với MySQL làm cho nó kém tin cậy hơn so với một số hệ quản trị cơ sở dữ liệu khác.
* Dung lượng hạn chế: Nếu số bản ghi này của bạn lớn dần lên thì việc truy xuất dữ liệu của bạn là khá khó khăn, khi đó chúng ta sẽ phải áp dụng nhiều biện pháp để tăng tốc độ truy xuất dữ liệu như là chia tải database này ra nhiều server.
  1. Bootstrap:

Bootstrap là một Fontend framework, là bộ sưu tập miễn phí các công cụ để tạo ra các trang web và các ứng dụng web. Bootstrap bao gồm HTML và CSS dựa trên các mẫu thiết kế cho kiểu chữ, hình thức, các button và các thành phần giao diện khác cũng như mở rộng tuỳ chon JavaScript.

Bootstrap được định nghĩa sẵn các dữ liệu CSS giúp người dùng có thể thiết kế giao diện Fontend một cách nhanh chóng. Các thư viện có những đoạn mã mà chúng ta sẵn sàng đưa vào website của mình mà không tốn một thời gian quá nhiều để tự viết.

Với Bootstrap việc phát triển giao diện web để phù hợp với đa thiết bị trở nên dễ dàng hơn. Đây là xu hướng phát triển giao diện website sau này. Bootstrap cung cấp các tính năng responsive nghĩa là làm cho trang web có thể tự co dãn để tương thích với mọi thiết bị khác nhau…

Một ý nghĩa khác là responsive làm cho trang web cung cấp được trải nghiệm tuyệt vời cho người dùng trên nhiều thiết bị, kích thước màn hình khác nhau. Đây là một đặt điểm quan trọng khi chúng ta chưa biết được các thiết bị nào có kích thước màn hình như sao được sử dụng truy cập vào trang web. Một trang web có thể hoạt động tốt kể cả có sự biến đổi sẽ giúp cho trải nghiệm người dùng sử dụng tốt và rõ ràng hơn một số trang web chỉ thiết kế cho một loại thiết bị màn hình cụ thể.

CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT CHƯƠNG TRÌNH

1. 1. Phân tích bài toán:

Hệ thống nhà hàng bao gồm 3 đối tượng sử dụng chính là: quản lý nhà hàng (ADMIN), nhân viên thu ngân (CASHIER) và nhân viên bếp (CHEF).

* + 1. ADMIN

ADMIN là người quản trị hệ thống và thực tế là quản lý nhà hàng có các quyền cấp cao nhất sử dụng tất cả các chức năng của trang web quản lý nhà hàng gồm truy cấp tất cả các trang web, sử dụng các chức năng là:

* Trang “Trang chủ” : xem thống kê doanh thu các đơn hàng đã hoàn thành thanh toán trong thời gian là ngày hiện tại hoặc chọn theo khoảng thời gian thống kê; thống kê tổng các thực đơn bán chạy nhất trong ngày hiện tại.
* Trang “Nhà Hàng”: gồm 2 trang là “Màn hình thu ngân” và “Màn hình bếp”. Trang “Màn hình thu ngân” sử dụng các chức năng nhập, chỉnh sửa và xoá thông tin các đơn hàng order vào hệ thống, báo chế biến thực đơn của bàn, chỉnh sửa thông tin hoá đơn bàn, tạm tính in hoá đơn bán hàng, thanh toán hoá đơn bán hàng. Trang “Màn hình bếp” hiển thị các phiếu báo chế biến của nhân viên order phân biệt 3 danh sách gồm danh sách phiếu báo chế biến chưa được xác nhận báo chế biến, danh sách phiếu báo chế biến đã xác nhận chờ chế biến và danh sách phiếu báo chế biến đã hoàn thành; người dùng sử dụng các chức năng xác nhận phiếu báo chế biến, hoàn thành phiếu báo chế biên và huỷ phiếu báo chế biến khi phiếu báo chế biến chưa hoàn thành.
* Trang “Bàn”: người dùng quản lý danh sách bàn trong hệ thống và được thêm, cập nhật, xoá thông tin bàn và tìm kiếm, lọc danh sách bàn theo các dữ liệu chọn.
* Trang “Thực Đơn”: người dùng quản lý danh sách thực đơn trong hệ thống và sử dụng các chức năng thêm, cập nhật, xoá thông tin thực đơn, tìm kiếm, lọc danh sách theo các dữ liệu và quản lý thêm nhóm thực đơn.
* Trang “Khách Hàng”: người dùng quản lý danh sách khách hàng của nhà hàng và sử dụng các chức năng thêm, cập nhật thông tin khách hàng và tìm kiếm, lọc danh sách khách hàng theo dữ liệu chọn.
* Trang “Nhân viên”: người dùng (ADMIN) quản lý danh sách nhân viên nhà hàng và thêm, cập nhật, xoá thông tin nhân viên, tìm kiếm lọc danh sách nhân viên theo dữ liệu chọn. ADMIN có thể tạo tài khoản đăng nhập cho nhân viên và cấp quyền sử dụng.
  + 1. CASHIER:

Nhân viên thu ngân là người sẽ thực hiện hoạt động bán hàng của nhà hàng và được truy cập vào các trang là:

* Trang “Trang chủ” : xem thống kê doanh thu các đơn hàng đã hoàn thành thanh toán trong thời gian là ngày hiện tại hoặc chọn theo khoảng thời gian thống kê; thống kê tổng các thực đơn bán chạy nhất trong ngày hiện tại.
* Trang “Màn hình thu ngân”: người dùng sử dụng các chức năng nhập, chỉnh sửa và xoá thông tin các đơn hàng order vào hệ thống, báo chế biến thực đơn của bàn, chỉnh sửa thông tin hoá đơn bàn, tạm tính in hoá đơn bán hàng, thanh toán hoá đơn bán hàng.
* Trang “Bàn”: người dùng quản lý danh sách bàn trong hệ thống và được thêm, cập nhật, xoá thông tin bàn và tìm kiếm, lọc danh sách bàn theo các dữ liệu chọn.
* Trang “Khách Hàng”: người dùng quản lý danh sách khách hàng của nhà hàng và sử dụng các chức năng thêm, cập nhật thông tin khách hàng và tìm kiếm, lọc danh sách khách hàng theo dữ liệu chọn.
  + 1. CHEF:

Nhân viên bếp là người sẽ thực hiện hoạt động xử lý các phiếu báo chế biến thực đơn của nhà hàng và được truy cập vào các trang là:

* Trang “Màn hình bếp”: hiển thị các phiếu báo chế biến của nhân viên order phân biệt 3 danh sách gồm danh sách phiếu báo chế biến chưa được xác nhận báo chế biến, danh sách phiếu báo chế biến đã xác nhận chờ chế biến và danh sách phiếu báo chế biến đã hoàn thành; người dùng sử dụng các chức năng xác nhận phiếu báo chế biến, hoàn thành phiếu báo chế biên và huỷ phiếu báo chế biến khi phiếu báo chế biến chưa hoàn thành.
* Trang “Thực Đơn”: người dùng quản lý danh sách thực đơn trong hệ thống và sử dụng các chức năng thêm, cập nhật, xoá thông tin thực đơn, tìm kiếm, lọc danh sách theo các dữ liệu và quản lý thêm nhóm thực đơn.
  1. Biểu đồ Use Case:

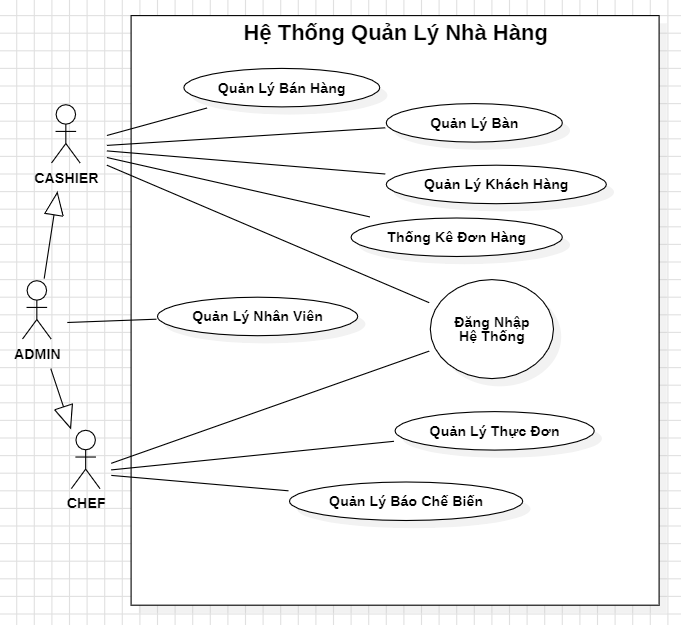
Hệ thống ứng dụng website quản lý nhà hàng gồm 3 Actor: CASHIER, CHEF và ADMIN.

Bảng 3.1. Vai trò của các actor.

|  |  |  |
| --- | --- | --- |
| **STT** | **Actor** | **Vài Trò** |
| 1 | CASHIER | CASHIER (nhân viên thu ngân) là đối tượng sử dụng các chức năng hoạt động bán hàng của hệ thống. Gồm chức năng đăng nhập hệ thống, quản lý bán hàng ,quản lý thông tin bàn, quản lý thông tin khách hàng và thống kê đơn hàng. |
| 2 | CHEF | CHEF (nhân viên bếp) là đối tượng sử dụng các chức năng đăng nhập hệ thống, quản lý phiếu báo chế biến và quản lý thực đơn nhà hàng trong hệ thống. |
| 3 | ADMIN | ADMIN là actor kế thừa actor CASHIER và actor CHEF và thêm các chức năng quản lý nhân viên. |

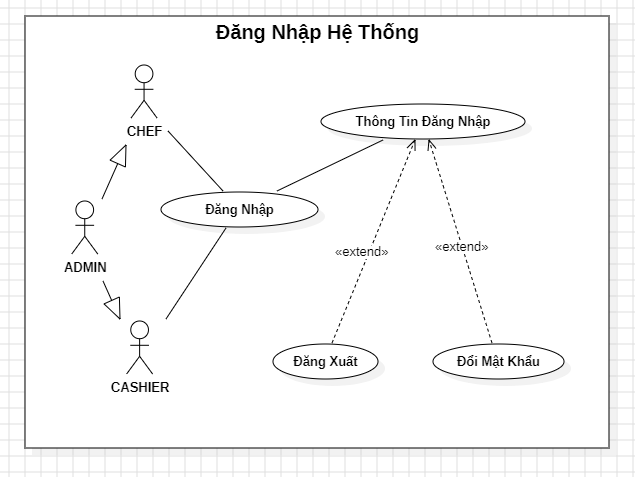
Dưới đây là một số biểu đồ use case hệ thống:

* + 1. Use case tổng quát:



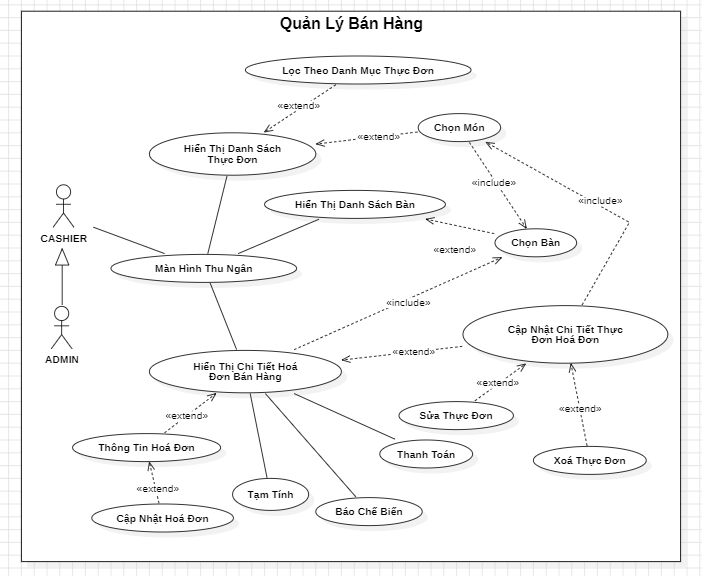
Hình 3.2: Use case tổng quát

* + 1. Use case Đăng Nhập Hệ Thống:



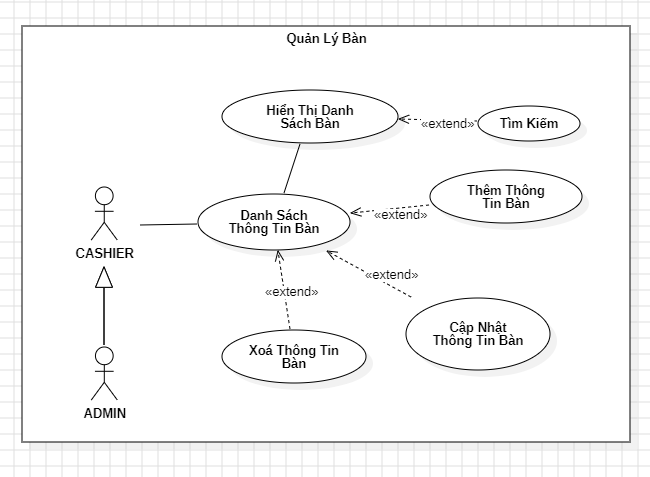
Hình 3.3: Use case đăng nhập hệ thống

* + 1. Use case Quản Lý Bán Hàng:



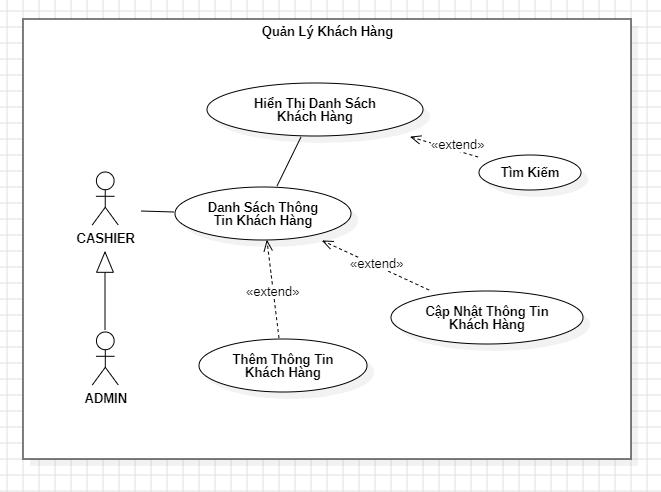
Hình 3.4: Use case quản lý bán hàng

* + 1. Use Case Quản Lý Bàn:



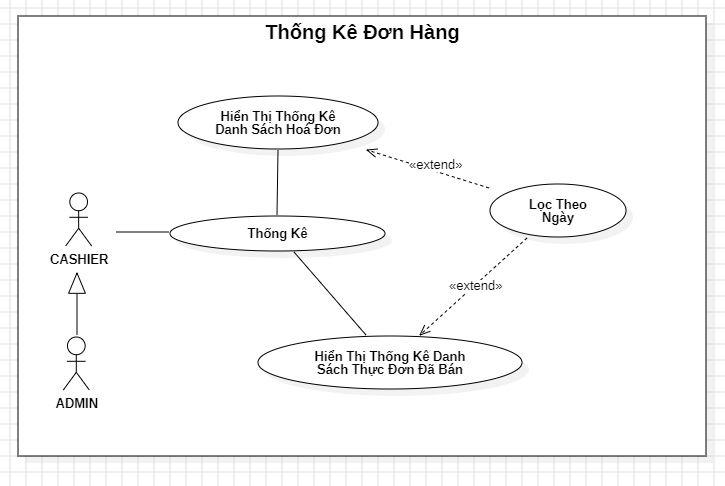
Hình 3.5: Use case quản lý bàn

* + 1. Use case Quản Lý Khách Hàng:



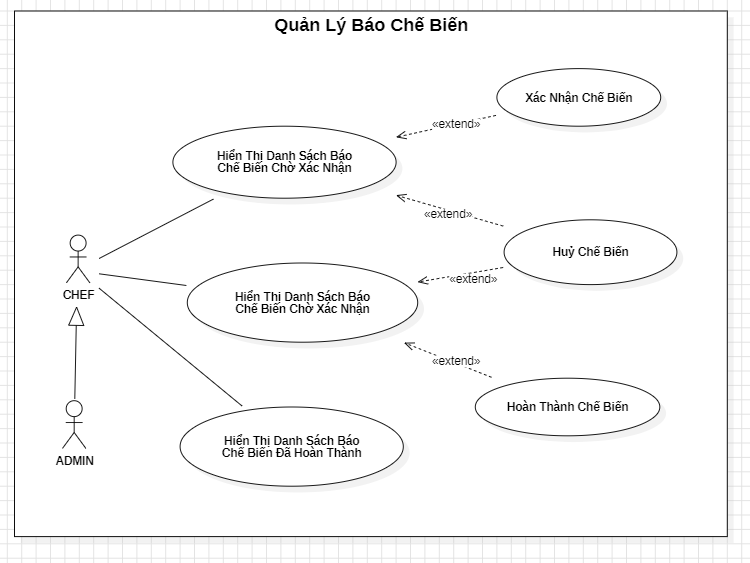
Hình 3.6: Use case quản lý khách hàng

* + 1. Use case Thống Kê Đơn Hàng:



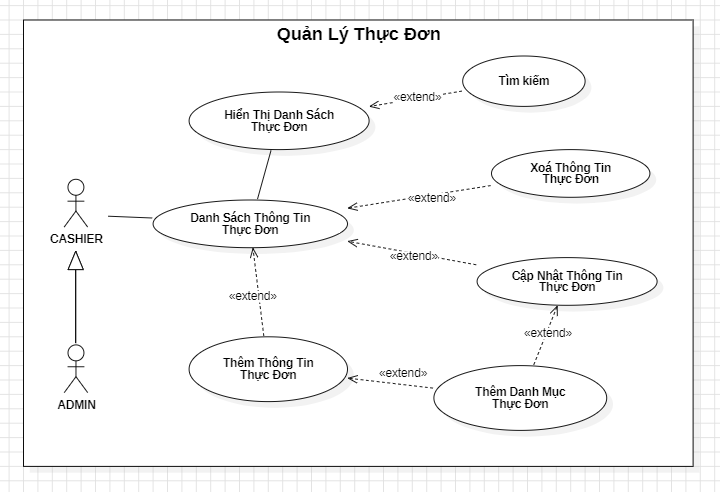
Hình 3.7: Use case thống kê đơn hàng

* + 1. Use case Quản Lý Báo Chế Biến:



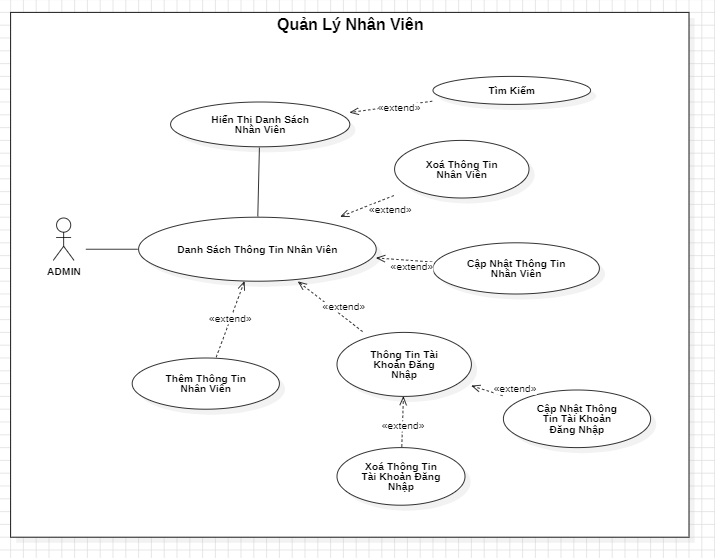
Hình 3.8: Use case quản lý báo chế biến

* + 1. Use case Quản Lý Thực Đơn:



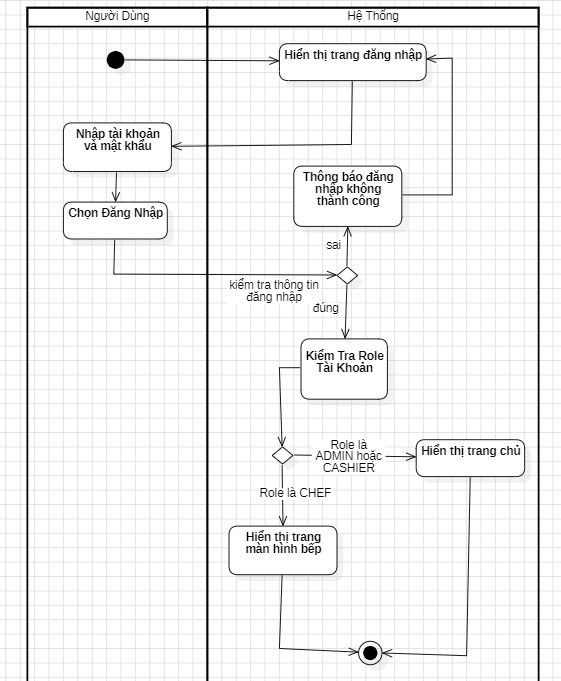
Hình 3.9: Use case quản lý thực đơn

* + 1. Use case Quản Lý Nhân Viên:



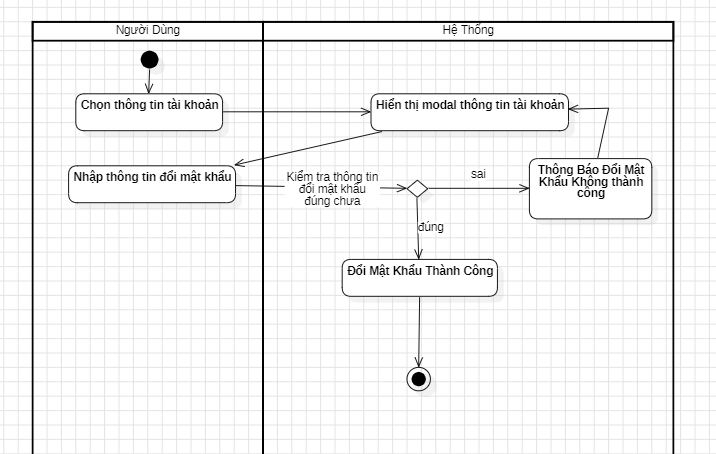
Hình 3.10: Use case quản lý nhân viên

* 1. Biểu đồ hoạt động (Activity Diagram):
     1. Chức năng đăng nhập hệ thống:



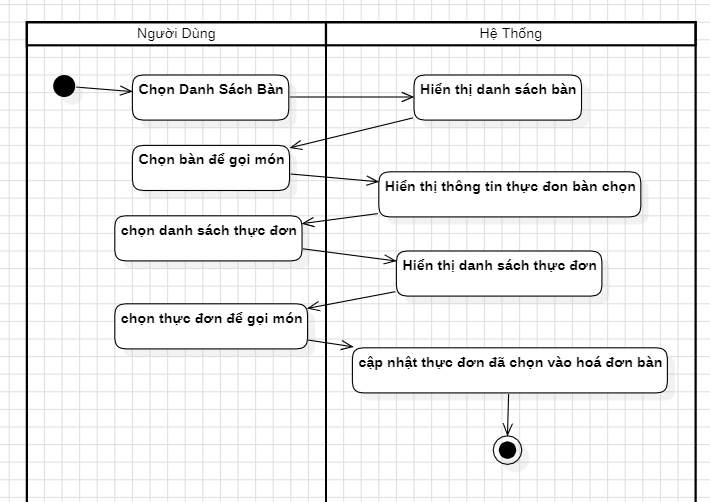
Hình 3.11: Biểu đồ chức năng đăng nhập hệ thống

* + 1. Chức năng đổi mật khẩu:



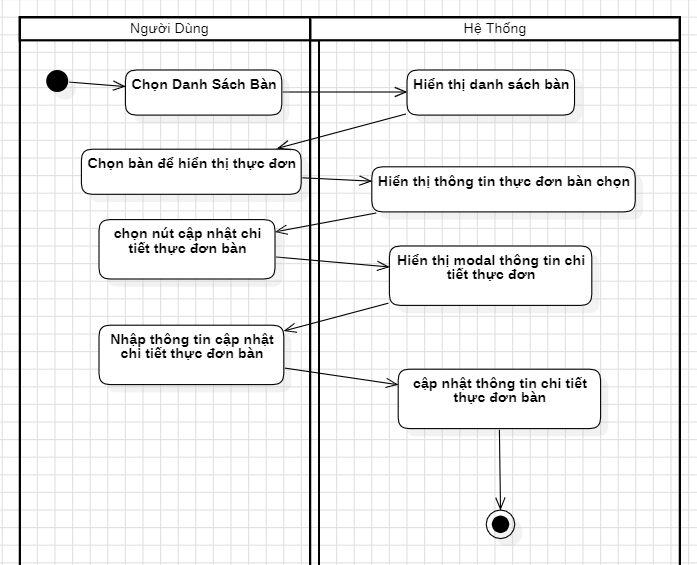
Hình 3.12: Biểu đồ hoạt động chức năng đổi mật khẩu

* + 1. Chức năng gọi món thực đơn bàn:



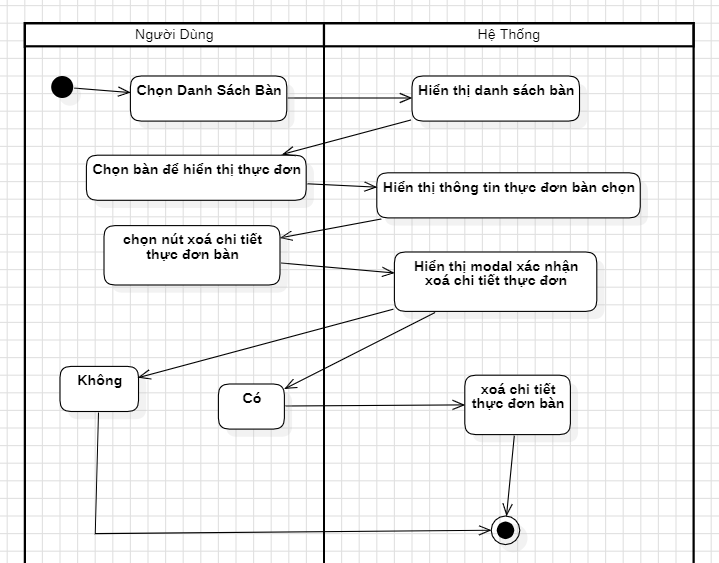
Hình 3.13: Biểu đồ hoạt động chức năng gọi món thực đơn bàn

* + 1. Chức năng cập nhật thông tin chi tiết thực đơn bàn:



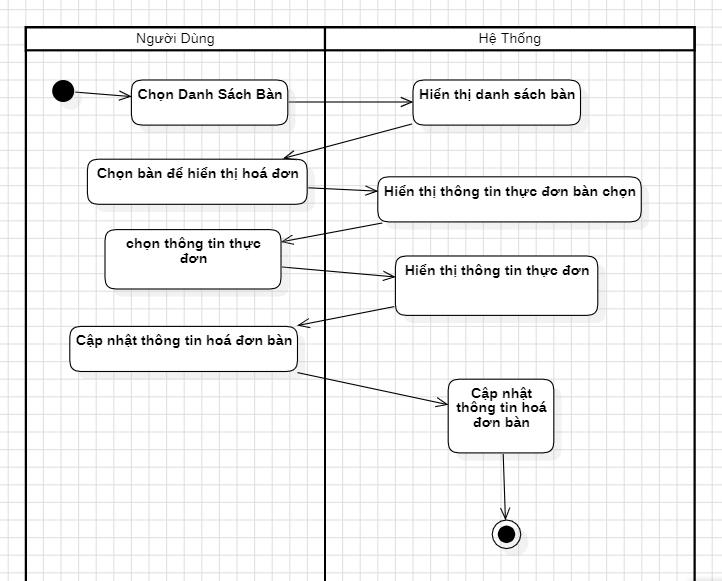
Hình 3.14: Biểu đồ hoạt động chức năng cập nhật thông tin chi tiết thực đơn bàn

* + 1. Chức năng xoá thông tin chi tiết thực đơn bàn:



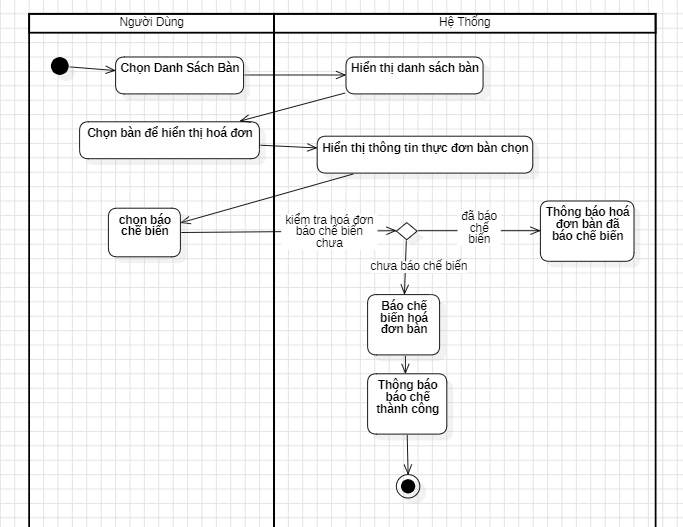
Hình 3.15: Biểu đồ hoạt động chức năng xoá thông tin chi tiết thực đơn bàn

* + 1. Chức năng cập nhật thông tin hoá đơn bàn:



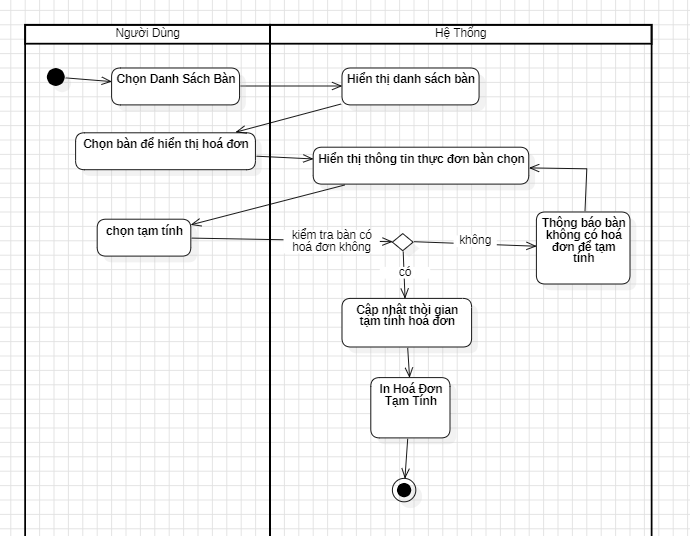
Hình 3.16: Biểu đồ hoạt động chức năng cập nhật thông tin hoá đơn bàn

* + 1. Chức năng báo chế biến hoá đơn bàn:



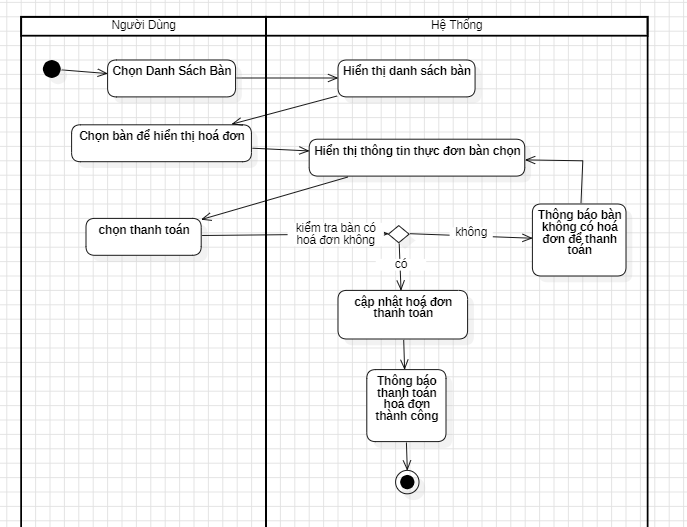
Hình 3.17: Biểu đồ hoạt động chức năng báo chế biến hoá đơn bàn

* + 1. Chức năng tạm tính hoá đơn bàn:



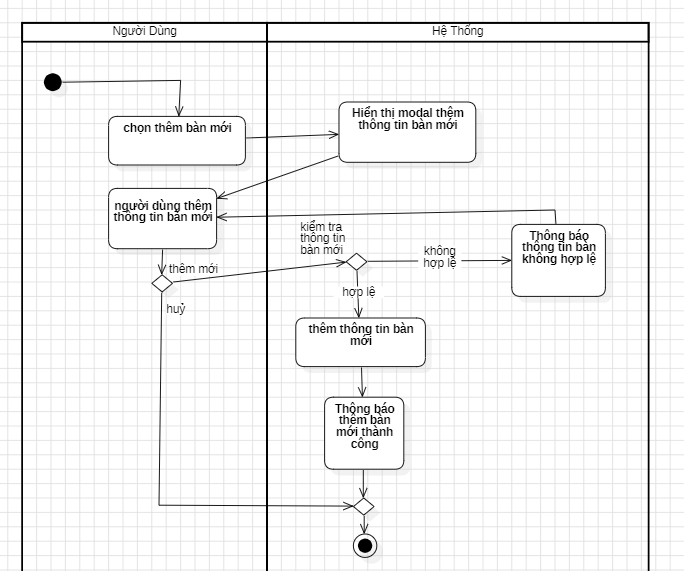
Hình 3.18: Biểu đồ hoạt động chức năng tạm tính hoá đơn bàn

* + 1. Chức năng thanh toán hoá đơn bàn:



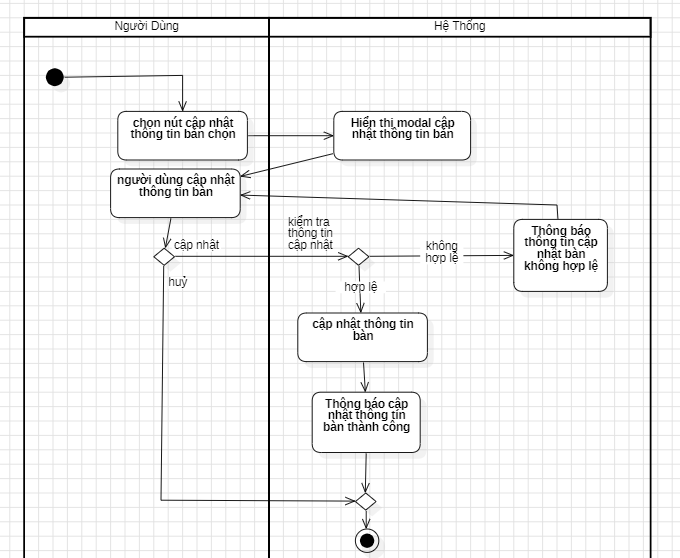
Hình 3.19: Biểu đồ hoạt động chức năng thanh toán hoá đơn bàn

* + 1. Chức năng thêm thông tin bàn mới:



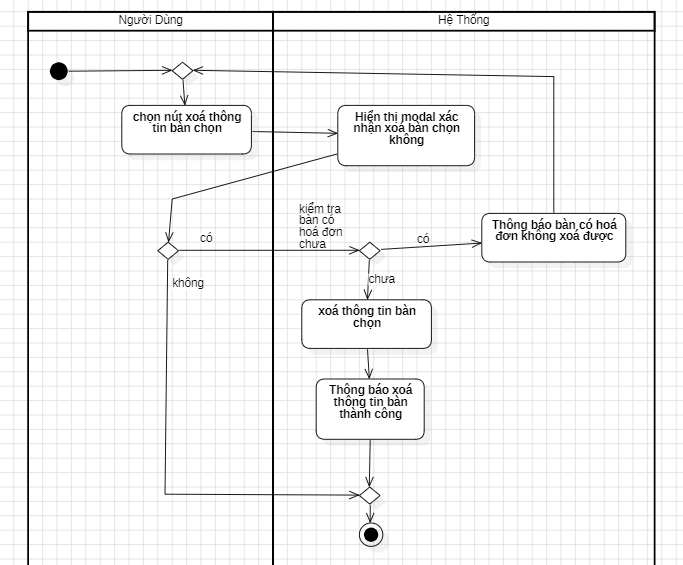
Hình 3.20: Biểu đồ hoạt động chức năng thêm thông tin bàn mới

* + 1. Chức năng cập nhật thông tin bàn:



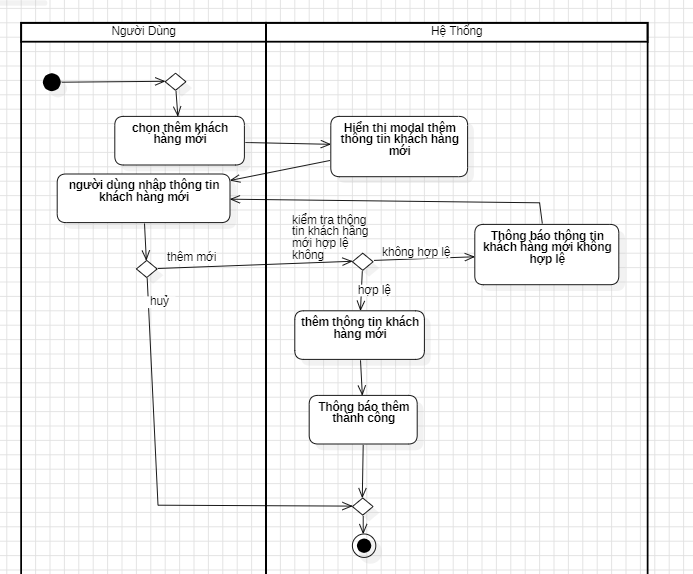
Hình 3.21: Biểu đồ hoạt động chức năng cập nhật thông tin bàn

* + 1. Chức năng xoá thông tin bàn:



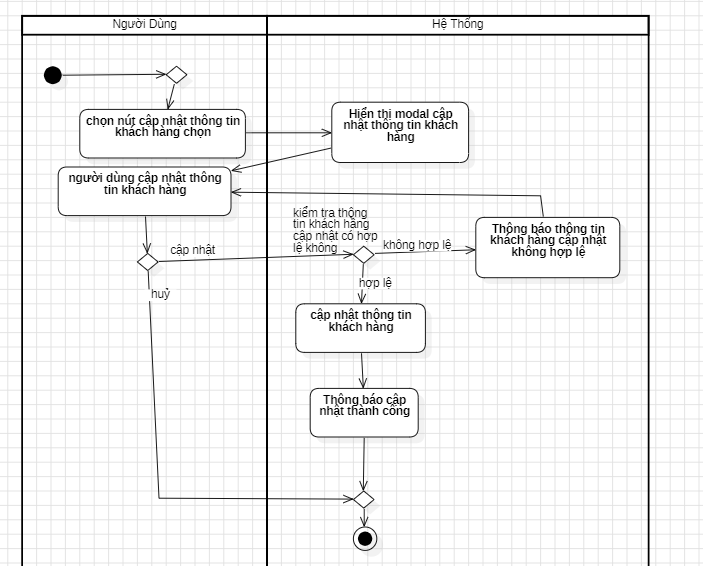
Hình 3.22: Biểu đồ hoạt động chức năng xoá thông tin bàn

* + 1. Chức năng thêm thông tin khách hàng mới:



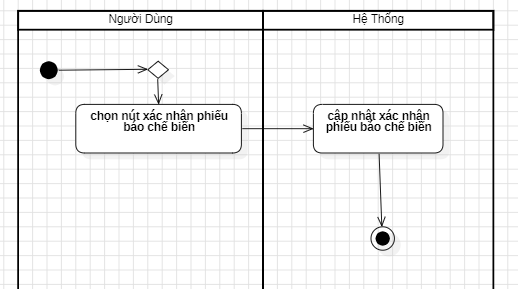
Hình 3.23: Biểu đồ hoạt động chức năng thêm thông tin khách hàng mới

* + 1. Chức năng cập nhật thông tin khách hàng:



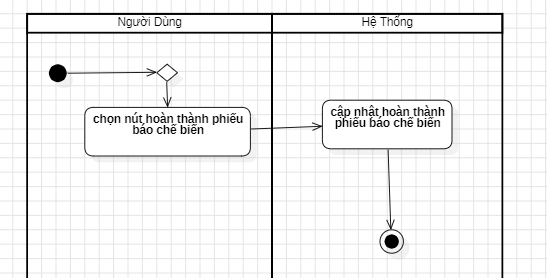
Hình 3.24: Biểu đồ hoạt động chức năng cập nhật thông tin khách hàng

* + 1. Chức năng xác nhận phiếu báo chế biến:



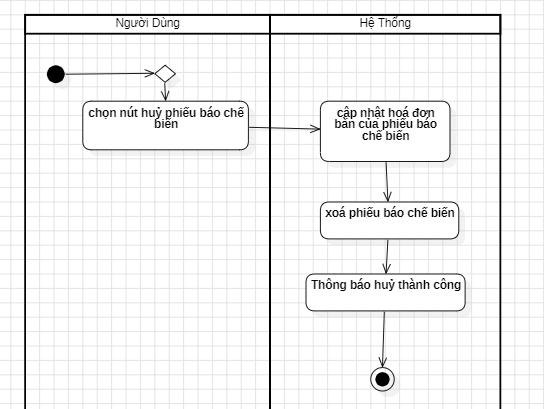
Hình 3.25: Biểu đồ hoạt động chức năng xác nhận phiếu báo chế biến

* + 1. Chức năng hoàn thành phiếu báo chế biến:



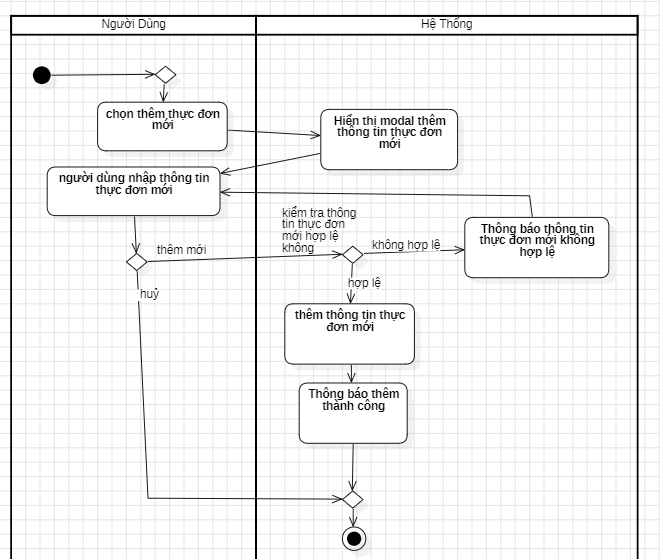
Hình 3.26: Biểu đồ hoạt động chức năng hoàn thành phiếu báo chế biến

* + 1. Chức năng huỷ phiếu báo chế biến:



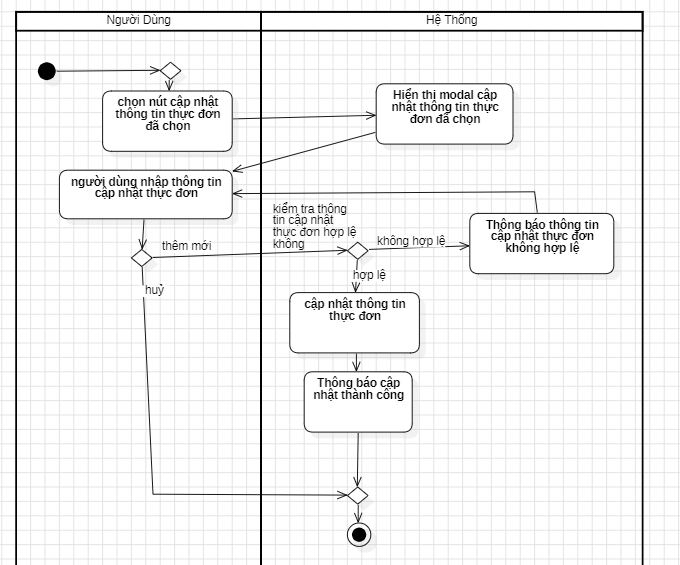
Hình 3.27: Biểu đồ hoạt động chức năng huỷ phiếu báo chế biến

* + 1. Chức năng thêm thông tin thực đơn mới:



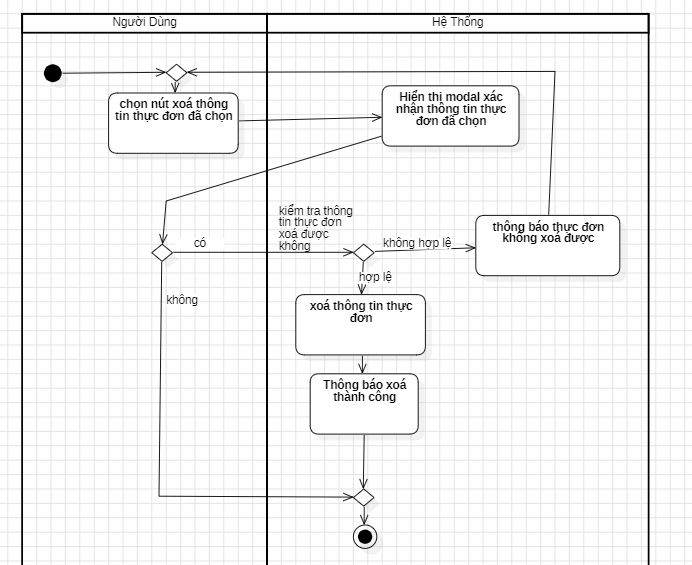
Hình 3.28: Biểu đồ hoạt động chức năng thêm thông tin thực đơn mới

* + 1. Chức năng cập nhật thông tin thực đơn:



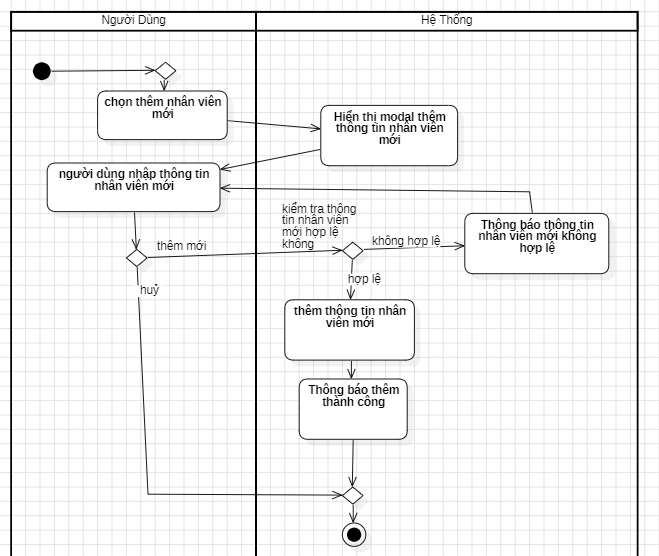
Hình 3.29: Biểu đồ hoạt động chức năng cập nhật thông tin thực đơn

* + 1. Chức năng xoá thông tin thực đơn:



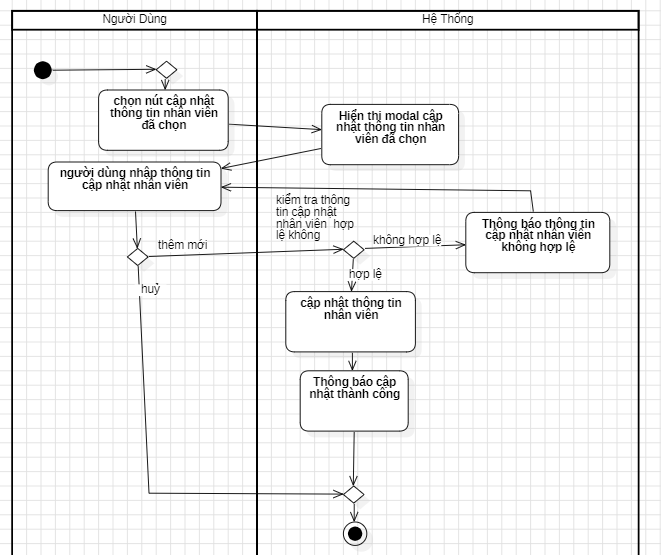
Hình 3.30:Biểu đồ hoạt động chức năng xoá thông tin thực đơn

* + 1. Chức năng thêm thông tin nhân viên mới:



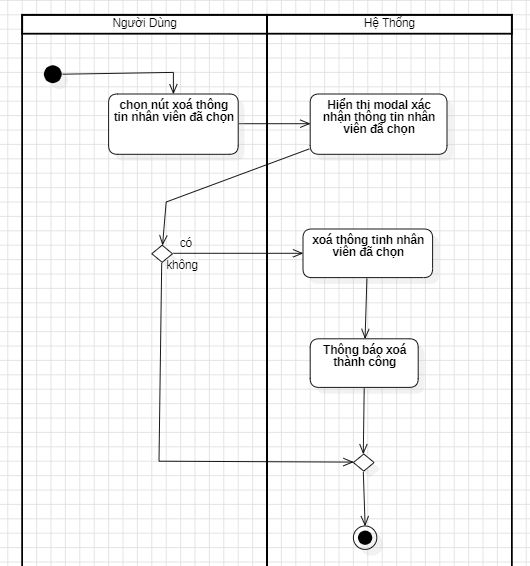
Hình 3.31: Biểu đồ hoạt động chức năng thêm thông tin nhân viên mới

* + 1. Chức năng cập nhật thông tin nhân viên:



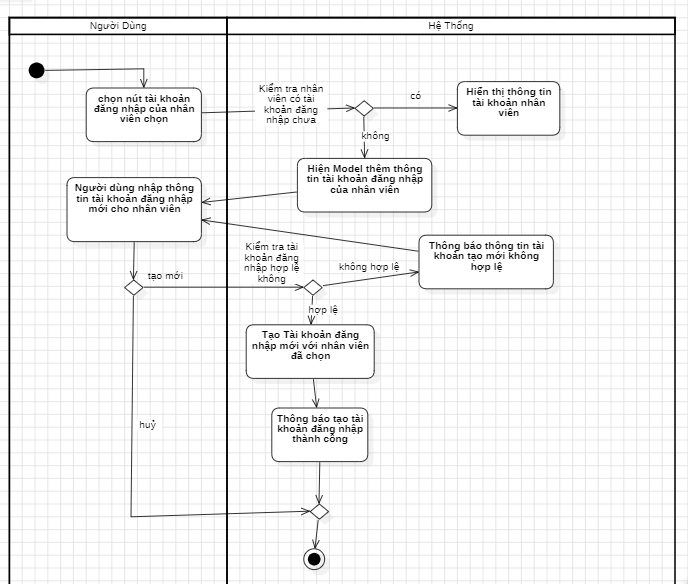
Hình 3.32:Biểu đồ hoạt động chức năng cập nhật thông tin nhân viên

* + 1. Chức năng xoá thông tin nhân viên:



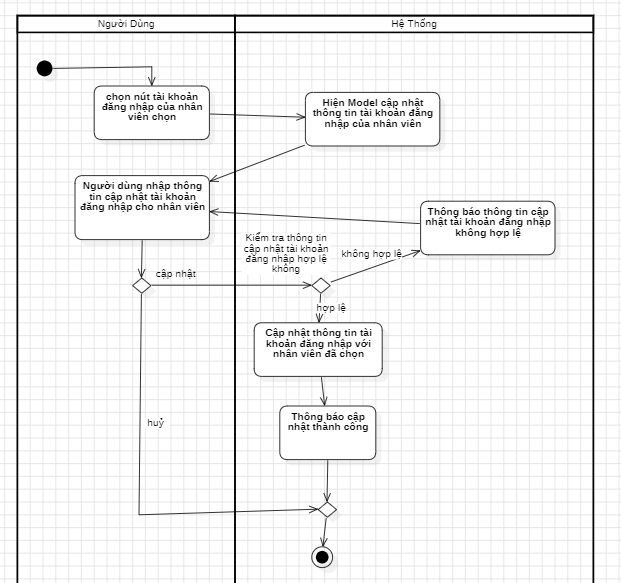
Hình 3.33: Biểu đồ hoạt động chức năng xoá thông tin nhân viên

* + 1. Chức năng tạo tài khoản đăng nhập mới cho nhân viên:



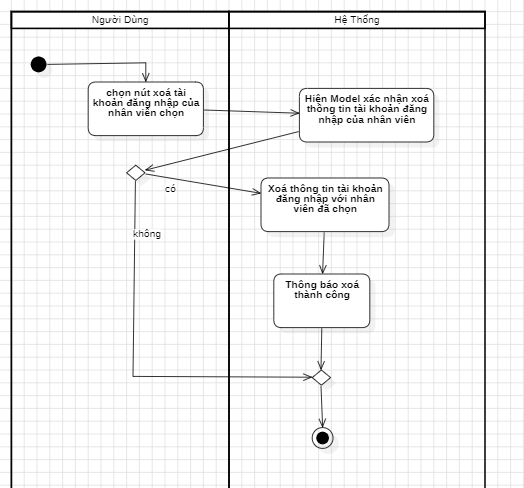
Hình 3.34: Biểu đồ hoạt động chức năng tạo tài khoản đăng nhập mới cho nhân viên.

* + 1. Chức năng thay đổi thông tin đăng nhập của nhân viên:



Hình 3.35: Biểu đồ hoạt động chức năng thay đổi thông tin đăng nhập của nhân viên

* + 1. Chức năng xoá tài khoản đăng nhập của nhân viên:



Hình 3.36: Biểu đồ hoạt động chức năng xoá tài khoản đăng nhập của nhân viên

* 1. Thiết kế cơ sở dữ liệu:
     1. Bảng Users:
* Dùng để lưu tài khoản đăng nhập hệ thống.

Bảng 3.2: Bảng Users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| username | varchar | No |  |  |
| password | varchar | No |  |  |
| employee\_id | int | No |  |  |

* + 1. Bảng Roles:
* Dùng để lưu thông tin các vai trò người dùng.

Bảng 3.3: Bảng Roles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| name | varchar | No |  |  |

* + 1. Bảng Users\_Roles:
* Dùng để nối bảng Users và Roles.

Bảng 3.4: Bảng Users\_Roles.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| user\_info\_id | bigint | No |  | Khoá chính, ràng buộc với khoá ngoại với bảng Users. |
| roles\_id | bigint | No |  | Khoá chính, ràng buộc với khoá ngoại với bảng Roles. |

* + 1. Bảng Tables:
* Dùng để lưu dữ liệu thông tin bàn trong hệ thống.

Bảng 3.5: Bảng Tables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| Id | bigint | No |  | Khoá chính |
| Name | varchar | No |  | Tên bàn |
| is\_empty | bit | Yes | NULL | Bàn có trống không? |
| is\_processing\_newspaper | bit | Yes | NULL | Bàn đã báo chế biến chưa? |
| is\_temporary\_invoice | bit | Yes | NULL | Bàn đã tạm tính hoá đơn chưa? |
| total\_invoice | bigint | No |  | Tổng cộng thành tiền hoá đơn bàn. |

* + 1. Bảng Menus:
* Dùng để lưu dữ liệu thông tin thực đơn trong hệ thống

Bảng 3.6: Bảng Menus

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| name\_menu | varchar | No |  | Tên thực đơn |
| unit | varchar | No |  | Đơn vị tính |
| price\_cost | bigint | No |  | Giá gốc |
| price\_sale | bigint | No |  | Giá bán |
| inventory | bigint | No |  | Số lượng tồn kho |
| image\_url | varchar | No |  | url ảnh thực đơn |
| menus\_group\_id | bigint | No |  | Khoá ngoại ràng buộc tới bảng Menus\_Group |

* + 1. Bảng Menus\_Group:
* Dùng để lưu dữ liệu thông tin danh mục thực đơn trong hệ thống.

Bảng 3.7: Bảng Menus\_Group

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| name\_menu\_group | varchar | No |  | Tên danh mục thực đơn |

* + 1. Bảng Tables\_Menu:
* Dùng để lưu dữ liệu thông tin chi tiết các thức đơn của bàn trong hệ thống.

Bảng 3.8: Bảng Tables\_Menu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| menu\_id | bigint | No |  | Khoá chính, khoá ngoại nối với bảng Menus. |
| table\_id | bigint | No |  | Khoá chính, khoá ngoại nối với bảng Tables. |
| Amount | bigint | No |  | Số lượng thực đơn |
| price\_unit | bigint | No |  | Giá |
| is\_cooking | bit | Yes | NULL | Đã chế biến chưa ? |
| amount\_cooking | bigint | No |  | Số lượng đã chế biến |
| Note | varchar | Yes | NULL | Ghi chú |

* + 1. Bảng Customer:
* Dùng để lưu dữ liệu thông tin khách hàng trong hệ thống.

Bảng 3.9: Bảng Customer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| name\_customer | varchar | No |  | Họ tên khách hàng |
| nickname | varchar | No |  | Nickname khách hàng |
| phone\_number | varchar | No |  | Số điện thoại |
| gender | varchar | Yes | NULL | Giới tính |
| address | varchar | Yes | NULL | Địa chỉ |

* + 1. Bảng Employee:
* Dùng để lưu dữ liệu thông tin nhân viên trong hệ thống.

Bảng 3.10: Bảng Employee

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| name | varchar | No |  | Họ tên nhân viên. |
| gender | varchar | No |  | Giới tính |
| phone\_number | varchar | No |  | Số điện thoại |
| date\_of\_birth | varchar | Yes | NULL | Ngày sinh |
| address | varchar | Yes | NULL | Địa chỉ |
| id\_card | varchar | Yes | NULL | Số CCCD |
| note | varchar | Yes | NULL | Ghi chú |

* + 1. Bảng Processing\_Newspaper:
* Dùng để lưu dữ liệu thông tin các phiếu báo chế biến trong hệ thống.

Bảng 3.11: Bảng Processing\_Newspaper

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| id\_menu | bigint | No |  | Khoá ngoại nối với bảng Menus. |
| id\_table | bigint | No |  | Khoá ngoại nối với bảng Tables |
| amount\_cooking | bigint | No |  | Số lượng chế biến |
| is\_confirm | bit | No |  | Đã xác nhân ? |
| is\_cooking | bit | No |  | Đã hoàn thành ? |
| date\_time\_create | datetime | No |  | Thời gian tạo phiếu |
| date\_time\_completed | datetime | Yes | NULL | Thời gian hoàn thành |
| note | varchar | Yes | NULL | Ghi chú |

* + 1. Bảng Provisional\_Invoice:
* Dùng để lưu dữ liệu thông tin các hoá đơn tạm tính trong hệ thống.

Bảng 3.12: Bảng Provisional\_Invoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| date\_time\_in | datetime | No |  | Thời gian hoá đơn tạo. |
| date\_time\_print\_invoice | datetime | Yes | NULL | Thời gian tạm tính hoá đơn. |
| total\_money | bigint | No |  | Tổng thành tiền hoá đơn |
| discount | int | No |  | Giảm giá |
| surcharge | int | No |  | Phụ thu |
| Table\_id | bigint | No |  | Khoá ngoại nối với bảng Tables |
| Customer\_id | bigint | Yes | NULL | Khoá ngoại nối với bảng khách hàng. |

* + 1. Bảng Invoice:
* Dùng để lưu dữ liệu thông tin các hoá đơn bán hàng trong hệ thống.

Bảng 3.13: Bảng Invoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| date\_time\_create | datetime | No |  | Thời gian hoá đơn hoàn thành. |
| date\_time\_in | datetime | No |  | Thời gian khách hàng tạo hoá đơn, |
| into\_money | bigint | No |  | Tổng thành tiền thực đơn |
| discount | int | No |  | Giảm giá |
| surcharge | int | No |  | Phụ thu |
| total\_money | bigint | No |  | Tổng cộng |
| customer\_id | bigint | Yes | NULL | Khoá ngoại nối với bảng khách hàng. |

* + 1. Bảng Refresh\_Tokens:
* Dùng để lưu dữ liệu thông tin các token đang sử dụng trong hệ thống.

Bảng 3.14: Bảng Refresh\_Tokens

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| id | bigint | No |  | Khoá chính |
| exprity\_date | datetime | No |  | Thời gian token hết hạn. |
| token | varchar | No |  | Dữ liệu token đăng nhập |
| user\_id | bigint | No |  | Khoá ngoại nối với bảng Users. |

* 1. RESTful API:

Các REST API đã được Backend xây dựng để phía Fontend (Angular) sử dụng:

* API User:

Bảng 3.15: Bảng API User

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /user/save |
| 2 | GET | /user/all |
| 3 | GET | /user/profile |
| 4 | DELETE | /user/delete/{userId} |

* API Table:

Bảng 3.16: Bảng API Table

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /table/save |
| 2 | GET | /table /all |
| 3 | POST | /table /detail |
| 4 | GET | /table/processing\_newspaper/{tableId} |
| 5 | DELETE | /table/delete/{tableId} |

* API Table\_Menu:

Bảng 3.17: Bảng API Table\_Menu

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /table\_menu/select |
| 2 | PUT | /table\_menu/save |
| 3 | GET | /table\_menu/all |
| 4 | GET | /table\_menu/detail |
| 5 | PUT | /table\_menu/delete |

* API Menu:

Bảng 3.18: Bảng Menu

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /menus/save |
| 2 | GET | /menus/all |
| 3 | POST | /menus/all\_find\_menus\_group |
| 4 | GET | /menus/detail |
| 5 | DELETE | /menus/delete/{menuId} |

* API Menus\_Group:

Bảng 3.19: Bảng Menus\_Group

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /menus\_group/save |
| 2 | GET | /menus\_group/all |
| 3 | GET | /menus\_group/detail |

* API Customer:

Bảng 3.20: Bảng API Customer

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | GET | /customer/all |
| 2 | GET | /customer/get/{customerId} |
| 3 | POST | /customer/save |
| 4 | DELETE | /customer/delete/{customerId} |

* API Employee:

Bảng 3.21: Bảng API Employee

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | GET | /employee/all |
| 2 | POST | /employee/save |
| 3 | DELETE | /employee/delete/{employeeId} |

* API Processing\_Newspaper:

Bảng 3.22: Bảng Processing\_Newspaper

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /processing\_newspaper/all\_date\_create |
| 2 | POST | /processing\_newspaper/all\_date\_create\_and\_not\_confirm |
| 3 | POST | /processing\_newspaper/all\_date\_create\_and\_confirm\_and\_not\_cooking |
| 4 | POST | /processing\_newspaper/all\_date\_create\_and\_confirm\_and\_cooking |
| 5 | GET | /processing\_newspaper/confirm/{id} |
| 6 | GET | /processing\_newspaper/cooking/{id} |

* API Provisional\_Invoice:

Bảng 3.23: Bảng API Provisional\_Invoice

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /provisional\_invoice/save |
| 2 | GET | /provisional\_invoice/all |
| 3 | POST | /provisional\_invoice/detail |
| 4 | POST | /provisional\_invoice/print\_invoice |

* API Invoice:

Bảng 3.24: Bảng API Invoice

|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /invoice/save |
| 2 | GET | /invoice/all |
| 3 | POST | /invoice/detail |
| 4 | POST | /invoice/all\_date\_create |

* API Login:

Bảng 3.25: Bảng API Login

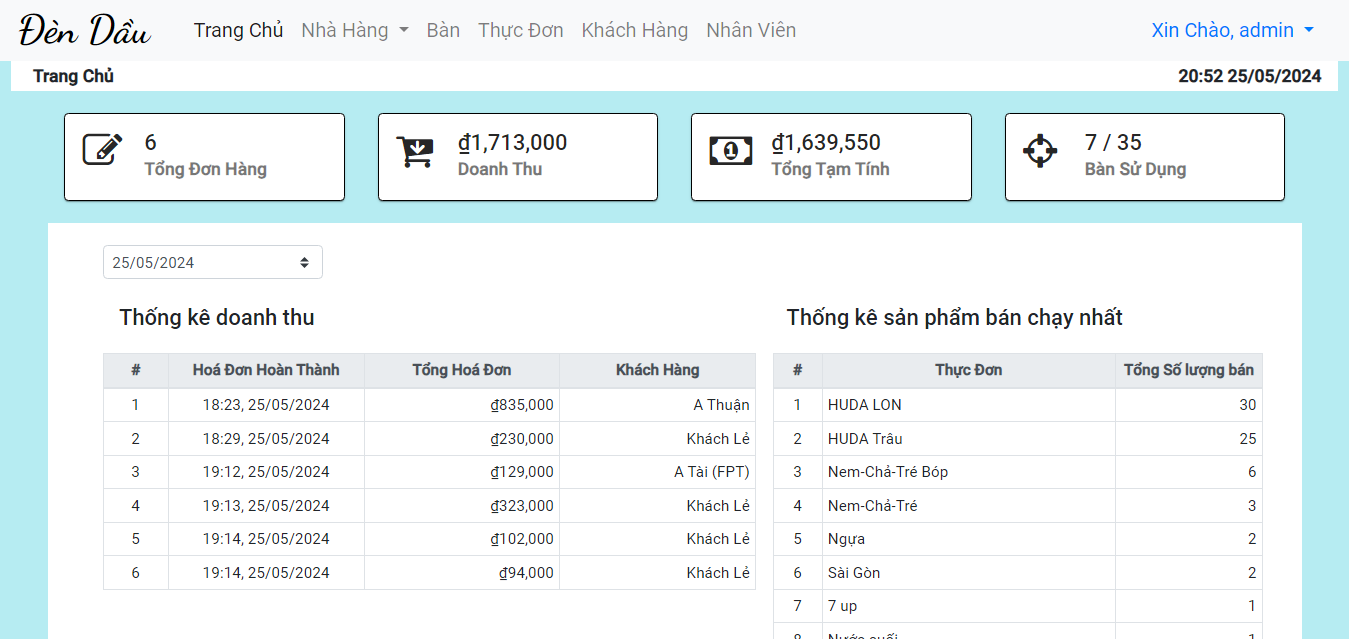
|  |  |  |
| --- | --- | --- |
| **STT** | **HTTP Request** | **API** |
| 1 | POST | /api-login/login |
| 2 | POST | /api-login/refreshToken |
| 3 | POST | /api-login/logout |

* 1. Giao diện:
     1. Trang đăng nhập:

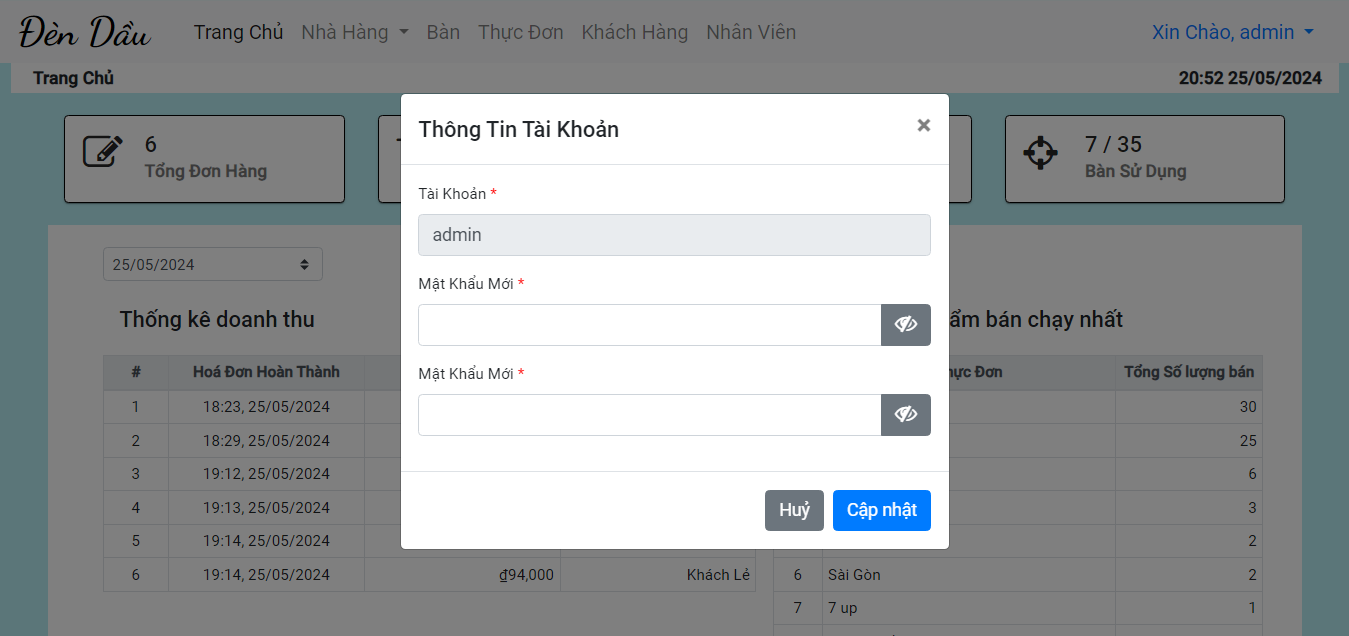


Hình 3.37: Giao diện trang đăng nhập hệ thống

* + 1. Trang “Trang Chủ”:

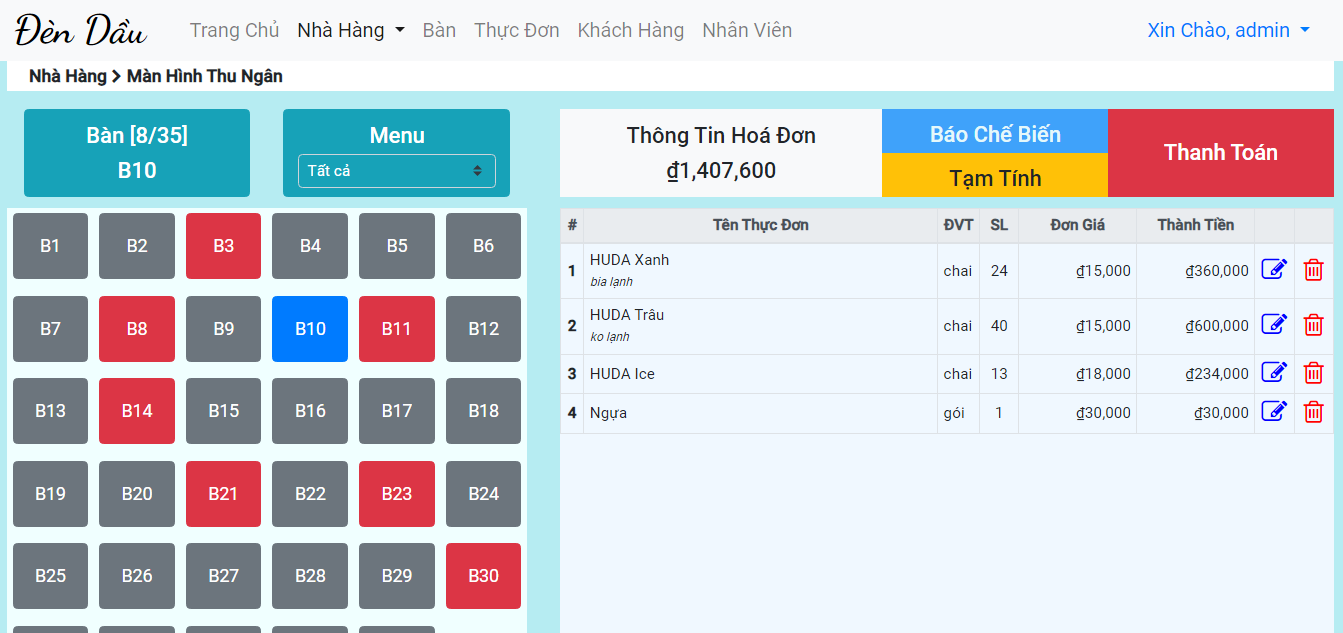


Hình 3.38: Giao diện chính màn hình trang chủ

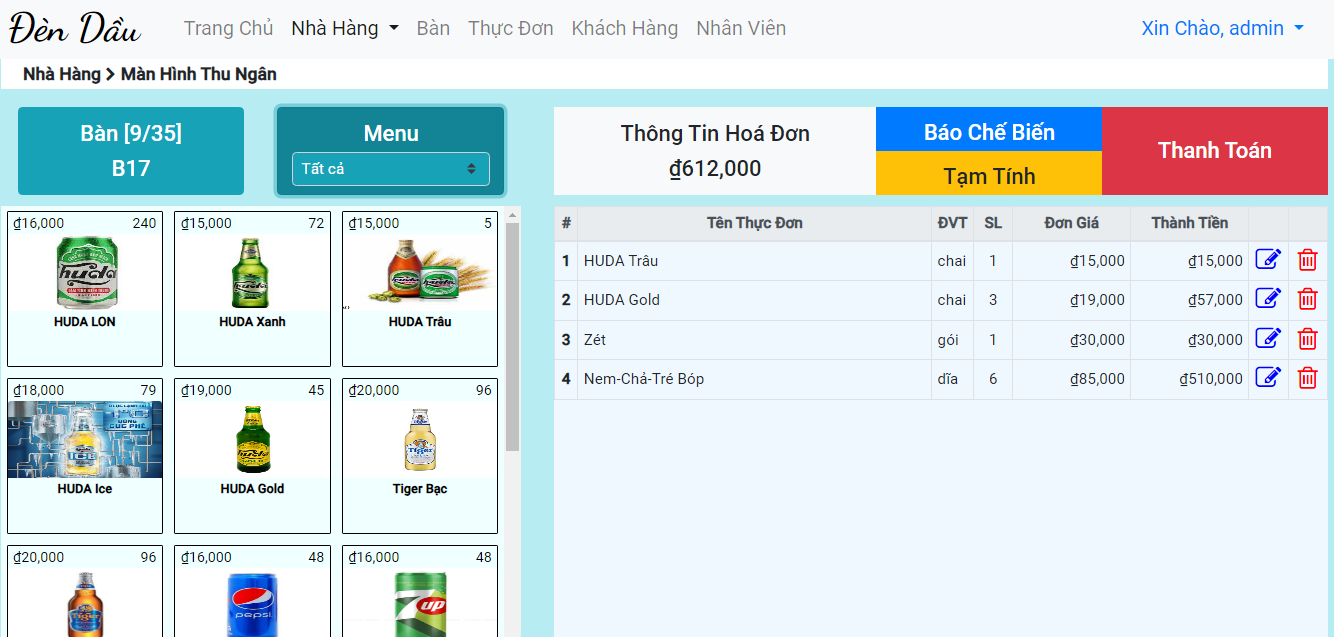


Hình 3.39: Giao diện thông tin tài khoản đăng nhập

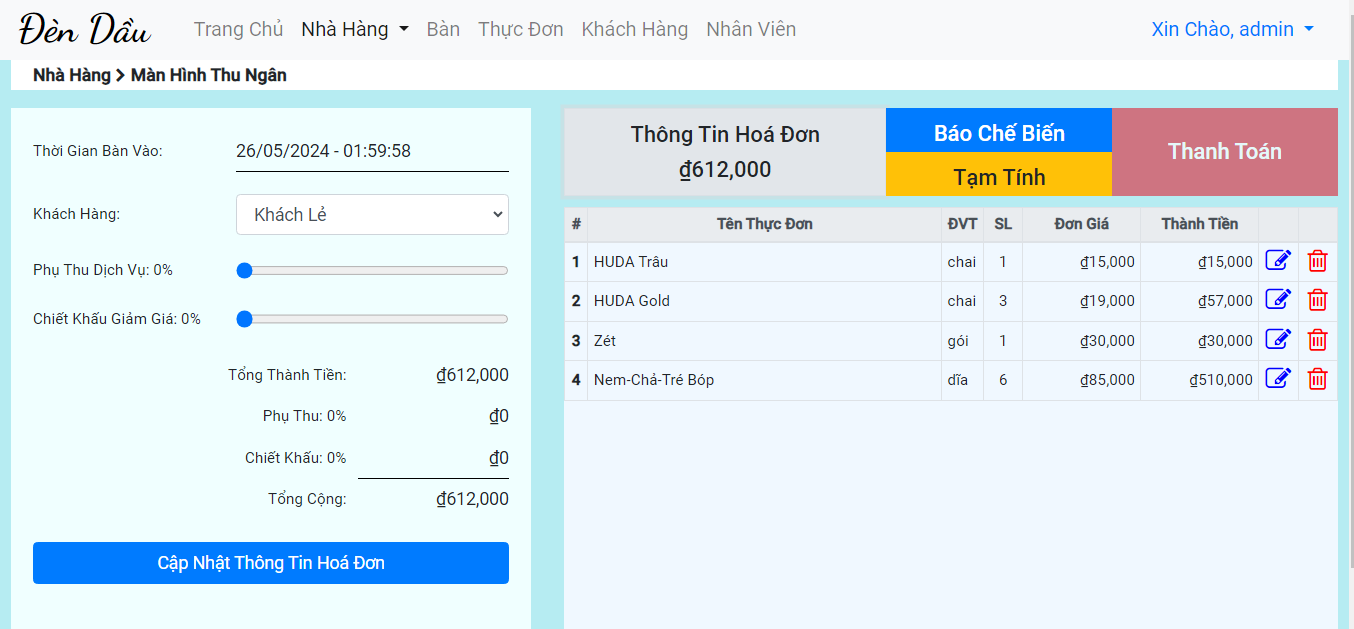
* + 1. Trang “Nhà Hàng”:
       1. Trang “Màn hình thu ngân”:



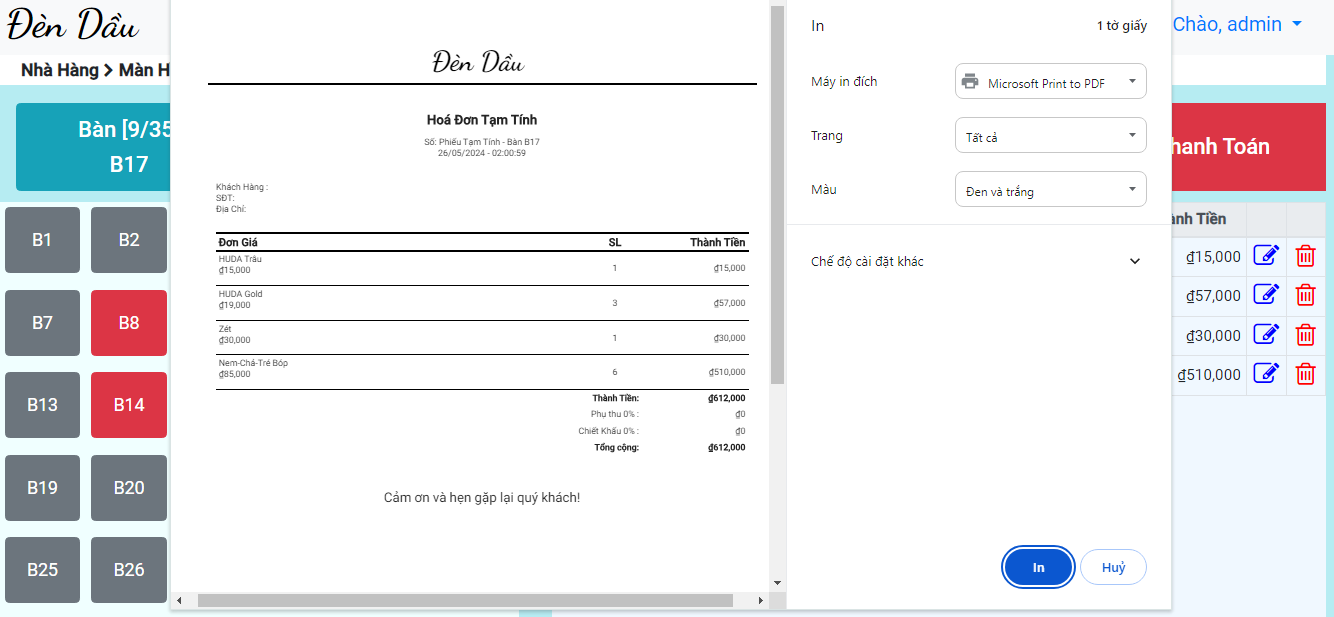
Hình 3.40: Giao diện chính màn hình thu ngân



Hình 3.41: Giao diện hiển thị danh sách các thực đơn để chọn

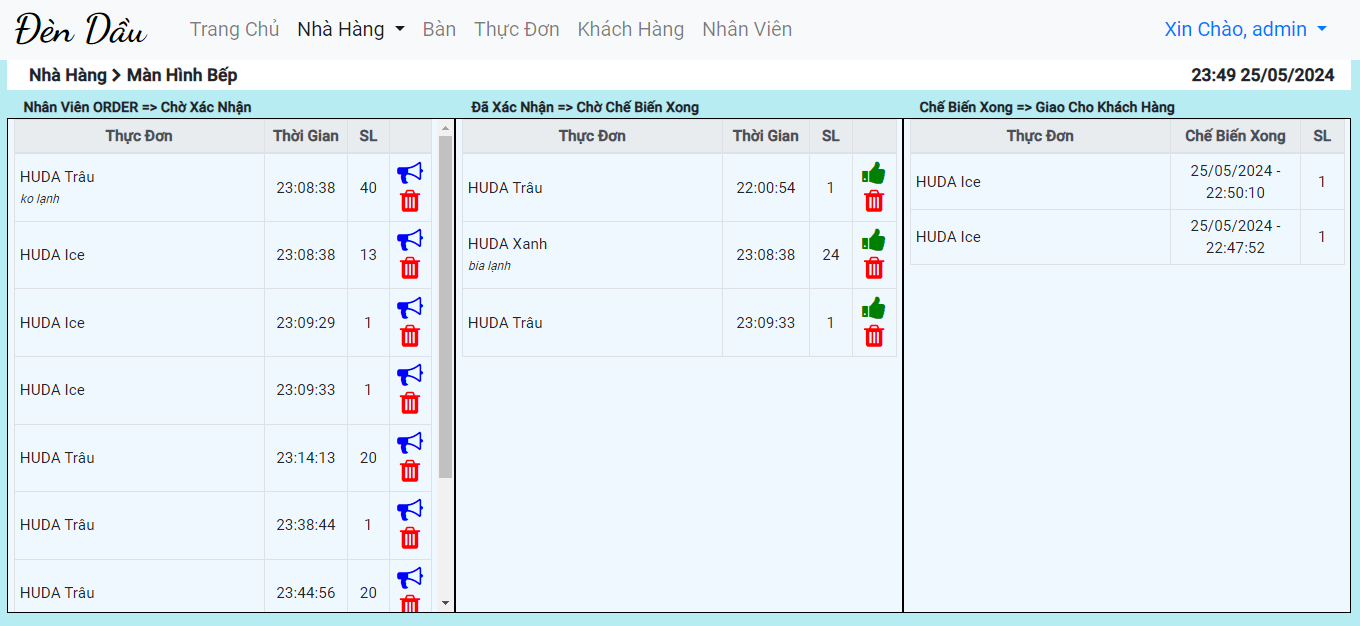


Hình 3.42: Giao diện thông tin hoá đơn



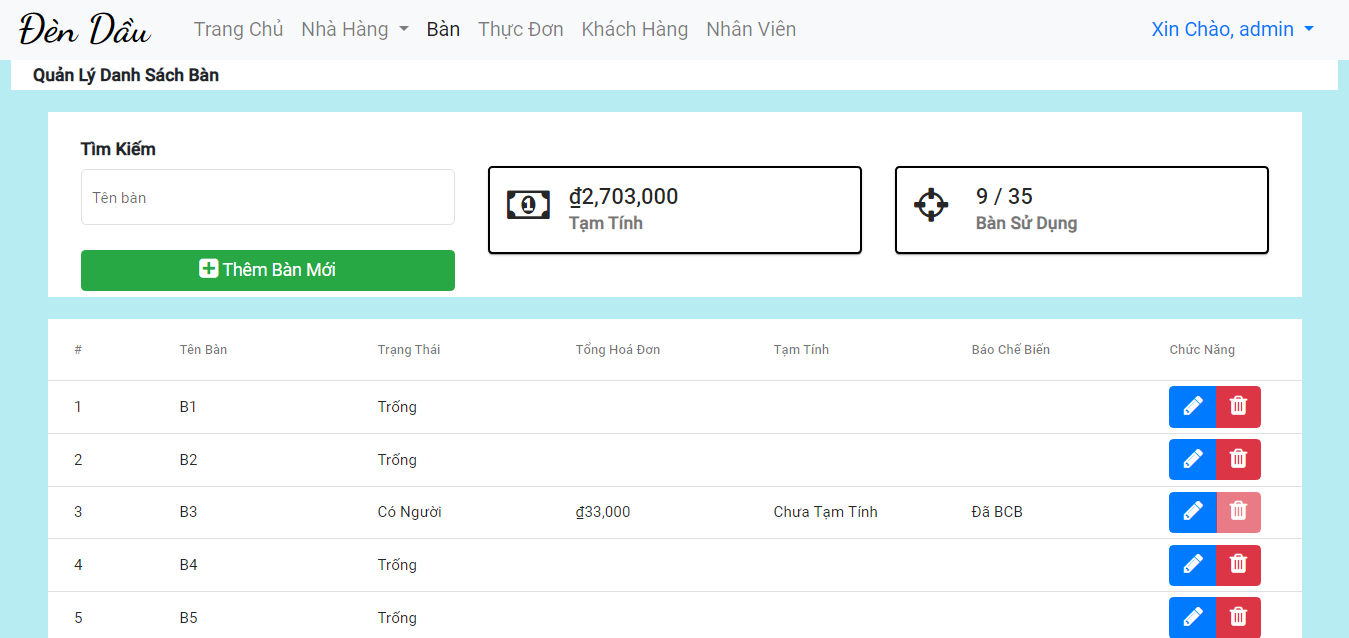
Hình 3.43: Hiển thị in hoá đơn sau khi tạm tính

* + - 1. Trang “Màn hình bếp”:



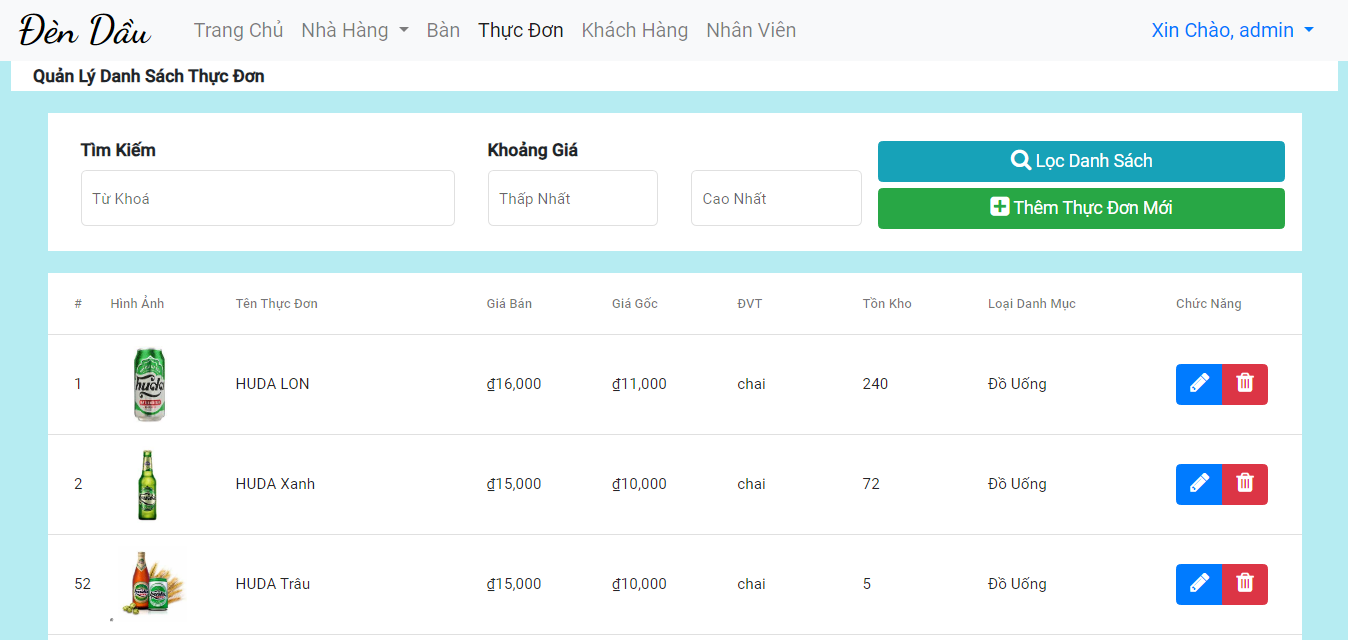
Hình 3.44: Giao diện chính màn hình bếp

* + 1. Trang “Bàn”:



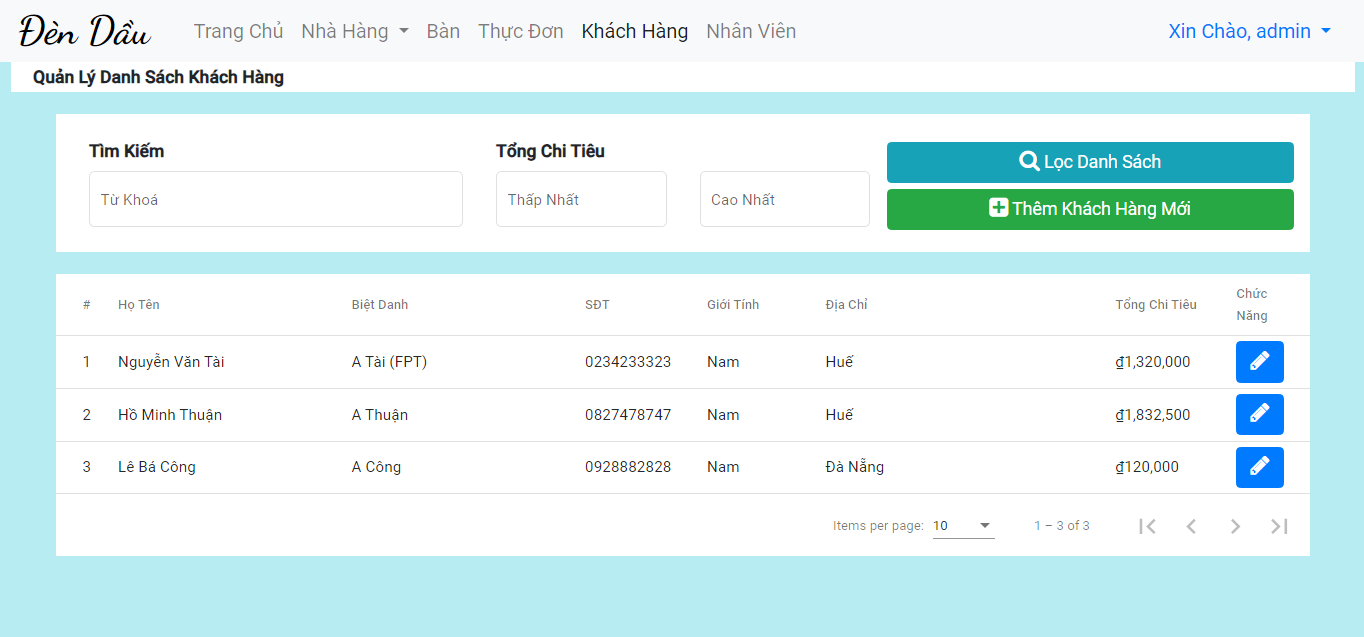
Hình 3.45: Giao diện chính trang quản lý bàn

* + 1. Trang “Thực Đơn”:



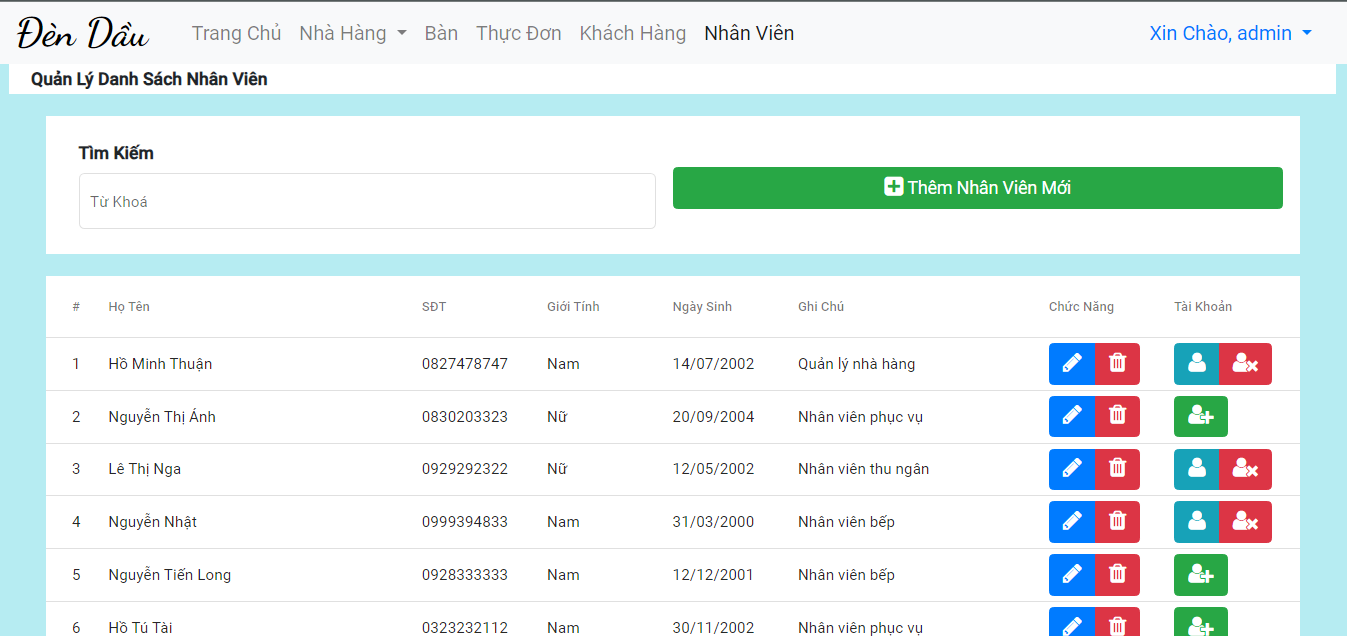
Hình 3.46: Giao diện chính trang quản lý thực đơn

* + 1. Trang “Khách Hàng”:



Hình 3.47: Giao diện chính trang quản lý khách hàng

* + 1. Trang “Nhân Viên”:



Hình 3.48: Giao diện chính trang quản lý nhân viên

# **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. Kết quả đạt được:

Tìm hiểu nghiên cứu và trình bày một cách có hiệu quả hệ thống các nội dung cơ bản về Spring Framework, Angular, MySQL và Bootstrap trong việc xây dựng một ứng dụng website.

Về mặt lý thuyết:

* Nắm được kiến trúc tổng quan của Spring Framework cũng như nguyên lý cơ bản và cơ chế hoạt động của framework này.
* Nắm được cách sử dụng Spring Boot Web và một số dự án khác như Spring Security, Spring Data JPA để xây dựng các chức năng cho ứng dụng web triển khai RESTful API.
* Biết được cấu trúc cơ bản về Angular và sử dụng nó để lập trình thiết kế giao diện Fontend cho ứng dụng. Ngoài ra, dùng Bootstrap framework CSS để thiết kế giao diện một cách nhanh chóng.

Về mặt ứng dụng:

* Xây dựng được ứng dụng web dang SPA – trang web một trang, với việc giao tiếp với Backend Server thông qua RESTful API với sự hỗ trợ của framework FontEnd Anuglar.
* Sử dụng Spring Security để bảo mật trang web bằng cách bảo mật với JWT token.
* Sử dụng Spring Data JPA để dễ dàng tạo và kết nối cơ sở dữ liệu MySQL thông qua cơ chế JPA, tiết kiệm công sức và dành thời gian cho phát triển chức năng ứng dụng.
* Xây dựng thành công một ứng dụng website quản lý nhà hàng với các chức năng cơ bản.

1. Hạn chế:

Các chức năng của ứng dụng web quản lý nhà hàng đã hoàn thành tốt tuy nhiên do yêu cầu đề tài và nguyên nhân khách quan nên ứng dụng chỉ thực hiện một số chức năng so với tên đề tài mà em đã chọn.

Do thiếu sót và hạn chế về vật chất nên em không có host trên internet có cài đặt đầy đủ các phần mềm cần thiết tạo khó khăn cho việc demo ứng dụng lên internet.

1. Hướng phát triển:

Tiếp tục tìm hiểu thêm các dự án của Spring Framwork để thực hiện việc xây dựng các dự án sau này.

Dự án xây dụng web quản lý nhà hàng chỉ mới đang thực hiện các chức năng bán hàng và quản lý cơ bản của một hệ thống nhà hàng. Em sẽ tiếp tục tìm hiểu và xây dựng thêm các chức năng như quản lý nhập hàng, kiểm kê hàng hoá - thực đơn…

# **TÀI LIỆU THAM KHẢO**

[1] Broadcom Inc. (2005-2024). Spring Framework, [Spring Framework Documentation](https://docs.spring.io/spring-framework/reference/index.html).

[2] Broadcom Inc. (2005-2024). Spring Secutity, [Spring Secutity Documentation](https://docs.spring.io/spring-security/reference/servlet/index.html).

[3] Broadcom Inc. (2005-2024). Spring Data JPA, [Spring Data JPA](https://docs.spring.io/spring-data/jpa/reference/index.html).

[4] Google (2010-2020). Angular 9, [Angular Docs version 9](https://v9.angular.io/docs).