|  |
| --- |
| BỘ GIÁO DỤC VÀ ĐÀO TẠO |
| **TRƯỜNG ĐẠI HỌC NÔNG LÂM TP.HCM** |
| **KHOA CÔNG NGHỆ THÔNG TIN** |



|  |
| --- |
| **LUẬN VĂN TỐT NGHIỆP** |
| **Ứng dụng các công nghệ React, NestJS và Kubernetes trong kiến trúc Microservices để xây dựng sàn thương mại điện tử.** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ngành** | **:** | **Công nghệ thông tin** | |
|  | **Niên khóa** | **:** | **2019 - 2023** | |
|  | **Lớp** | **:** | **DH19DTA** | |
|  | **Sinh viên thực hiện** | **:** | **Cao Huy Tấn Lộc** | **19130120** |
| **Châu Văn Lộc** | **19130121** |

|  |
| --- |
| TP. HỒ CHÍ MINH, tháng 08 năm 2024 |

|  |
| --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** |
| **TRƯỜNG ĐẠI HỌC NÔNG LÂM TP.HCM** |
| **KHOA CÔNG NGHỆ THÔNG TIN** |



|  |
| --- |
| **LUẬN VĂN TỐT NGHIỆP** |
| **Ứng dụng các công nghệ React, NestJS và Kubernetes trong kiến trúc Microservices để xây dựng sàn thương mại điện tử.** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Giảng viên hướng dẫn:** |  | **Sinh viên thực hiện:** | |
|  | **TS. Nguyễn Thị Phương Trâm** |  | **Cao Huy Tấn Lộc** | **19130120** |
|  | **Châu Văn Lộc** | **19130121** |

|  |
| --- |
| TP. HỒ CHÍ MINH, tháng 08 năm 2024 |

**CÔNG TRÌNH HOÀN TẤT TẠI**

**TRƯỜNG ĐẠI HỌC NÔNG LÂM TP.HCM**

Cán bộ hướng dẫn: TS. Nguyễn Thị Phương Trâm

Cán bộ phản biện:

Luận văn cử nhân được bảo vệ tại HỘI ĐỒNG CHẤM LUẬN VĂN CỬ NHÂN TRƯỜNG ĐẠI HỌC NÔNG LÂM TP.HCM ngày 8 tháng 09 năm 2023

**Nhận xét của giảng viên hướng dẫn:**

.....................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

**Nhận xét của giảng viên phản biện:**

....................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **TRƯỜNG ĐH NÔNG LÂM TP.HCM** | **Độc lập - Tự do - Hạnh phúc** |

**NHIỆM VỤ LUẬN VĂN CỬ NHÂN**

|  |  |  |
| --- | --- | --- |
| Họ tên sinh viên: **CAO HUY TẤN LỘC** |  | Giới tính: **Nam** |
| Ngày tháng năm sinh: 06/06/2001 |  | Nơi sinh: Đồng Nai |
| Chuyên ngành: Công nghệ thông tin |  | Ngành: Công nghệ thông tin |
| Email: 19130120@st.hcmuaf.edu.vn |  |  |

|  |  |  |
| --- | --- | --- |
| Họ tên sinh viên: **CHÂU VĂN LỘC** |  | Giới tính: **Nam** |
| Ngày tháng năm sinh: 01/09/2001 |  | Nơi sinh: Tiền Giang |
| Chuyên ngành: Công nghệ thông tin |  | Ngành: Công nghệ thông tin |
| Email: 19130121@st.hcmuaf.edu.vn |  |  |

1. TÊN ĐỀ TÀI: **Ứng dụng các công nghệ React Native, React và Spring Boot trong kiến trúc Microservices để xây dựng phần mềm quản lý bán hàng cho nhà hàng và dịch vụ đồ uống.**
2. NHIỆM VỤ VÀ NỘI DUNG

* Nhiệm vụ:

1. Nội dung: TNGÀY GIAO NHIỆM VỤ: **30/01/2023**
2. NGÀY HOÀN THÀNH NHIỆM VỤ: **08/09/2023**
3. HỌ VÀ TÊN CÁN BỘ HƯỚNG DẪN:

|  |  |  |
| --- | --- | --- |
| Ngày …/…/… | Ngày …/…/… | Ngày …/…/… |
| CÁN BỘ HƯỚNG DẪN | CÁN BỘ PHẢN BIỆT | KHOA CNTT |
| (Ký và ghi rõ họ tên) | (Ký và ghi rõ họ tên) | (Ký và ghi rõ họ tên) |

# 

# LỜI CẢM ƠN

Lời nói đầu, chúng em xin gửi lời cảm ơn chân thành đến quý Thầy, Cô Khoa Công nghệ thông tin, Trường Đại học Nông Lâm Thành phố Hồ Chí Minh đã truyền dạy kiến thức trong suốt thời gian học tập tại trường. Đây là hành trang không thể thiếu giúp chúng em hoàn thành khóa luận và tự tin lập nghiệp sau này.

Xin tỏ lòng biết ơn và gửi lời tri ân sâu sắc đến ThS. Phan Đình Long đã luôn kiên nhẫn, chỉ bảo, hướng dẫn chúng em trong suốt quá trình thực hiện khóa luận tốt nghiệp.

Xin chân thành cảm ơn!

Nhóm thực hiện luận văn tốt nghiệp

# DANH SÁCH CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **STT** | **Ký hiệu chữ viết tắt** | **Chữ viết đầy đủ** |
| 1 | API | **A**pplication **P**rogramming **I**nterface |
| 2 | ID | Mã định danh |
| 3 | IDE | **I**ntegrated **D**evelopment **E**nvironment |
| 4 | JDBC | **J**ava **D**atabase **C**onnectivity |
| 6 | MVC | **M**odel **V**iew **C**ontroller |
| 7 | REST | **RE**presentational **S**tate **T**ransfer |
| 8 | SOAP | **S**imple **O**bject **A**ccess **P**rotocol |
| 9 | SPA | **S**ingle **P**age **A**pplication |
| 10 | STT | **S**ố **T**hứ **T**ự |
| 11 | HTTP | **H**yper **T**ext **T**ransfer **P**rotocol |
| 12 | gRPC | **g**oogle **R**emote **P**rocedure **C**all |
| 13 | DDD | **D**omain-**D**riven **D**esign |
| 14 | ORM | **O**bject **R**elational **M**apping |
| 15 | OXM | Spring **O**bject **X**ML **M**appers |
| 16 | XML | e**X**tensible **M**arkup **L**anguage |
| 17 | AOP | **A**spect **O**riented **P**rogramming |
| 18 | JMS | **J**ava **M**essage **S**ervice |
| 19 | JAXB | **J**ava **A**rchitecture for **X**ML **B**inding |

# DANH MỤC CÁC BẢNG

[Bảng 3.1: Danh sách những yêu cầu chức năng chính của hệ thống 44](#_Toc143846306)

[Bảng 3.2: Đặc tả Use Case Đăng nhập 54](#_Toc143846307)

[Bảng 3.3: Đặc tả Use Case Đăng ký 55](#_Toc143846308)

[Bảng 3.4: Đặc tả Use Case Đăng xuất 56](#_Toc143846309)

[Bảng 3.5: Đặc tả Use Case Yêu cầu cấp lại mật khẩu 57](#_Toc143846310)

[Bảng 3.6: Đặc tả Use Case Quản lý giỏ hàng 58](#_Toc143846311)

[Bảng 3.7: Mô tả lớp Product 67](#_Toc143846312)

[Bảng 3.8: Mô tả lớp Category 67](#_Toc143846313)

[Bảng 3.9: Mô tả lớp Image 67](#_Toc143846314)

[Bảng 3.10: Mô tả lớp TimeSearch 68](#_Toc143846315)

[Bảng 3.11: Mô tả lớp Material 69](#_Toc143846316)

[Bảng 3.12: Mô tả lớp MaterialProduct 70](#_Toc143846317)

[Bảng 3.13: Mô tả lớp MaterialReturn 70](#_Toc143846318)

[Bảng 3.14: Mô tả lớp Supplier 70](#_Toc143846319)

[Bảng 3.15: Mô tả lớp MaterialSupplier 71](#_Toc143846320)

[Bảng 3.16: Mô tả lớp Image 71](#_Toc143846321)

[Bảng 3.17: Mô tả lớp Role 73](#_Toc143846322)

[Bảng 3.18: Mô tả lớp Group 73](#_Toc143846323)

[Bảng 3.19: Mô tả lớp User 73](#_Toc143846324)

[Bảng 3.20: Mô tả lớp Order 75](#_Toc143846325)

[Bảng 3.21: Mô tả lớp Table 75](#_Toc143846326)

[Bảng 3.22:Mô tả lớp GroupTable 75](#_Toc143846327)

[Bảng 3.23: Mô tả lớp PaySlipOrder 77](#_Toc143846328)

[Bảng 3.24: Mô tả lớp PaySlipInventory 78](#_Toc143846329)

[Bảng 3.25: Mô tả lớp ReceiptOrder 78](#_Toc143846330)

[Bảng 3.26: Mô tả lớp ReceiptInventory 79](#_Toc143846331)

[Bảng 3.27: Thông số cấu hình các dịch trên hệ thống 95](#_Toc143846332)

[Bảng 3.28: Mô tả những dịch vụ của hệ thống 96](#_Toc143846333)

[Bảng 3.29: Những giao thức giao tiếp trong hệ thống 98](#_Toc143846334)

[Bảng 4.1: Thông số cấu hình các dịch vụ trên máy chủ Microservices 111](#_Toc144139202)

[Bảng 4.2: Thông số cấu hình các dịch vụ trên máy chủ Keycloak 113](#_Toc144139203)

[Bảng 4.3: Thông số cấu hình các dịch vụ trên máy chủ Keycloak 113](#_Toc144139204)

# DANH MỤC HÌNH ẢNH

[Hình 2.1: Minh họa về kiến trúc Microservices 5](#_Toc144138871)

[Hình 2.2: So sánh giữa Monolithic và Microservices 6](#_Toc144138872)

[Hình 2.3: Logo của Spring Framework 10](#_Toc144138873)

[Hình 2. 4: Tổng quan về kiến trúc của Spring Framework 10](#_Toc144138874)

[Hình 2.5: Minh họa về API Gateway 13](#_Toc144138875)

[Hình 2.6: Minh họa về cách hoạt động của Eureka server 14](#_Toc144138876)

[Hình 2.7: Logo của React Framework 20](#_Toc144138877)

[Hình 2.8: Logo của thư viện Redux-Saga 25](#_Toc144138878)

[Hình 2.9: Hình ảnh minh họa về GraphQL 28](#_Toc144138879)

[Hình 2.10: Logo của Docker 31](#_Toc144138880)

[Hình 2.11: Tổng quan về kiến trúc Docker 32](#_Toc144138881)

[Hình 2.12: Logo của Gradle 36](#_Toc144138882)

[Hình 2.13: Logo của JDBI 36](#_Toc144138883)

[Hình 2.14: Logo của gRPC 37](#_Toc144138884)

[Hình 2.15: Tổng quan về kiến trúc của Apache Kafka. 38](#_Toc144138885)

[Hình 2.16: Logo của Keycloak 38](#_Toc144138886)

[Hình 2.17: Logo của Prometheus 39](#_Toc144138887)

[Hình 2.18: Logo của Grafana 40](#_Toc144138888)

[Hình 2.19: Logo của Zipkin 40](#_Toc144138889)

[Hình 2.20: Logo của MySQL 41](#_Toc144138890)

[Hình 3.1: Sơ đồ Use Case tổng quan cho hệ thống 61](#_Toc144139150)

[Hình 3.2: Sơ đồ chi tiết Use Case Đặt hàng 62](#_Toc144139151)

[Hình 3.3: Sơ đồ chi tiết Use Case Xem sản phẩm 62](#_Toc144139152)

[Hình 3.4: Sơ đồ chi tiết Use Case Quản lý giỏ hàng 62](#_Toc144139153)

[Hình 3.5: Sơ đồ chi tiết Use Case Đặt bàn 63](#_Toc144139154)

[Hình 3.6: Sơ đồ chi tiết Use Case Quản lý thông tin cá nhân 63](#_Toc144139155)

[Hình 3.7: Sơ đồ chi tiết Use Case Quản lý đơn hàng 63](#_Toc144139156)

[Hình 3.8: Sơ đồ chi tiết Use Case Quản lý người dùng 64](#_Toc144139157)

[Hình 3. 9: Sơ đồ chi tiết Use Case Quản lý sản phẩm 65](#_Toc144139158)

[Hình 3.10: Sơ đồ chi tiết Use Case Quản lý nhập nguyên liệu 65](#_Toc144139159)

[Hình 3.11: Sơ đồ tuần tự chức năng Đăng ký 71](#_Toc144139160)

[Hình 3.12: Sơ đồ tuần tự chức năng Đăng nhập 72](#_Toc144139161)

[Hình 3.13: Sơ đồ tuần tự chức năng Quên mật khẩu 72](#_Toc144139162)

[Hình 3.14: Sơ đồ tuần tự chức năng Thanh toán PayPal 73](#_Toc144139163)

[Hình 3.15: Sơ đồ tuần tự chức năng Đặt hàng 73](#_Toc144139164)

[Hình 3.16: Sơ đồ tuần tự chức năng Hủy đặt hàng 74](#_Toc144139165)

[Hình 3.17: Sơ đồ tuần tự chức năng Nhập kho 74](#_Toc144139166)

[Hình 3.18: Sơ đồ tuần tự chức năng Sửa người dùng 75](#_Toc144139167)

[Hình 3.19: Sơ đồ tuần tự chức năng Đặt bàn 75](#_Toc144139168)

[Hình 3.20: Kiến trúc Three-Layer hiện thực trong Spring Boot 77](#_Toc144139169)

[Hình 3.21: Mô hình mẫu về kiến trúc Hexagonal. Nguồn herbertograca.com 79](#_Toc144139170)

[Hình 3.22: Kiến trúc Hexagonal xây dựng cho dịch vụ đặt hàng 81](#_Toc144139171)

[Hình 3.23: Những interface đại diện cho Use Case của chức năng đặt hàng 81](#_Toc144139172)

[Hình 3.24: Những Use Case được nhóm thành bốn nhóm Service khác nhau 82](#_Toc144139173)

[Hình 3.25: Port và Adapter trong chức năng đặt hàng 83](#_Toc144139174)

[Hình 3.26: Sơ đồ lớp tổng quan của product-service 84](#_Toc144139175)

[Hình 3.27: Sơ đồ chi tiết Domain Layer của product-service 84](#_Toc144139176)

[Hình 3.28: Sơ đồ lớp tổng quan của inventory-service 86](#_Toc144139177)

[Hình 3.29: Sơ đồ chi tiết Domain Layer của inventory-service 87](#_Toc144139178)

[Hình 3.30: Sơ đồ lớp tổng quan của user-service 90](#_Toc144139179)

[Hình 3.31: Sơ đồ chi tiết Domain Layer của user-service 90](#_Toc144139180)

[Hình 3.32: Sơ đồ lớp tổng quan của order-service 92](#_Toc144139181)

[Hình 3.33: Sơ đồ chi tiết Domain Layer của order-service 92](#_Toc144139182)

[Hình 3.34:Sơ đồ lớp tổng quan của payment-service 94](#_Toc144139183)

[Hình 3.35: Sơ đồ chi tiết Domain Layer của payment-service 95](#_Toc144139184)

[Hình 3.36: Sơ đồ cơ sở dữ liệu của product-service 98](#_Toc144139185)

[Hình 3.37: Sơ đồ cơ sở dữ liệu của user-service 98](#_Toc144139186)

[Hình 3.38: Sơ đồ cơ sở dữ liệu của inventory-service 99](#_Toc144139187)

[Hình 3.39: Sơ đồ cơ sở dữ liệu của order-service 99](#_Toc144139188)

[Hình 3.40: Sơ đồ cơ sở dữ liệu của payment-service 100](#_Toc144139189)

[Hình 4.1: Kiến trúc phần mềm tổng quan 101](#_Toc144139193)

[Hình 4.2: Mỗi dịch vụ kết nối đến một database riêng 104](#_Toc144139194)

[Hình 4.3: Minh họa các dịch vụ giao tiếp với Eureka Server trong hệ thống 105](#_Toc144139195)

[Hình 4.4: Hoạt động của Config Server trong hệ thống 106](#_Toc144139196)

[Hình 4.5: Mô tả hoạt động của Keycloak trong hệ thống 108](#_Toc144139197)

[Hình 4.6: Tổng quan kiến trúc hệ thống 109](#_Toc144139198)

[Hình 4.7: Chi tiết các dịch vụ trên máy chủ Microservices 111](#_Toc144139199)

[Hình 4.8: Chi tiết máy chủ Keycloak và các tương tác liên quan 112](#_Toc144139200)

[Hình 4. 9: Chi tiết máy chủ Config-Server 113](#_Toc144139201)

# TÓM TẮT

Việt Nam là một trong các quốc gia có tốc độ chuyển đối số và phát triển thương mại điện tử thuộc top đầu của Đông Nam Á. Sự phát triển của smartphone, website hiện tại chính là nền tảng để khách hàng có thể dễ dàng tiếp cận với các sản phẩm trực

Khi thị trường thương mại điện tử đang bùng nổ như hiện nay, việc sở hữu một nền tảng mua sắm trực tuyến không chỉ là một lợi thế cạnh tranh mà còn là một yếu tố sống còn. Với sự gia tăng không ngừng của người tiêu dùng trực tuyến, các doanh nghiệp cần một giải pháp công nghệ linh hoạt, mạnh mẽ để không chỉ giữ chân khách hàng mà còn thu hút khách hàng mới. tuyến, tạo cho khách hàng một thói quen tiêu dùng mới – chạm để mua hàng

Chính vì lẽ đó, đồ án của chúng em hướng đến sàn thương mại điện tử được xây dựng trên nền tảng kiến trúc microservice, sử dụng các công nghệ hàng đầu như ReactJS cho giao diện người dùng, Kubernetes để quản lý container hóa, NestJS làm nền tảng backend, và KongAPI làm cổng API gateway. Sự kết hợp này không chỉ đảm bảo tính ổn định, mà còn tối ưu hóa trải nghiệm người dùng và tăng cường khả năng mở rộng.

# MỤC LỤC

[LỜI CẢM ƠN i](#_Toc144143094)

[DANH SÁCH CHỮ VIẾT TẮT ii](#_Toc144143095)

[DANH MỤC CÁC BẢNG iii](#_Toc144143096)

[DANH MỤC HÌNH ẢNH v](#_Toc144143097)

[TÓM TẮT viii](#_Toc144143098)

[MỤC LỤC ix](#_Toc144143099)

[CHƯƠNG 1 GIỚI THIỆU 1](#_Toc144143100)

[1.1. Đặt vấn đề 1](#_Toc144143101)

[1.2. Mục tiêu 2](#_Toc144143102)

[1.3. Đối tượng nghiên cứu 2](#_Toc144143103)

[1.4. Phạm vi nghiên cứu 3](#_Toc144143104)

[CHƯƠNG 2 TỔNG QUAN CÔNG NGHỆ 4](#_Toc144143105)

[2.1. Tổng quan về Microservices 4](#_Toc144143106)

[2.1.1. Giới thiệu về Microservices 4](#_Toc144143107)

[2.1.2. Những đặc trưng của Microservices 5](#_Toc144143108)

[2.1.3. So sánh kiến trúc Microservices và kiến trúc nguyên khối (Monolithic) 6](#_Toc144143109)

[2.1.4. So sánh kiến trúc Microservices và kiến trúc hướng dịch vụ (SOA - Service-Oriented Architecture) 8](#_Toc144143110)

[2.1.5. Những khó khăn khi triển khai Microservices 9](#_Toc144143111)

[2.1.6. Kết luận về Microservices 9](#_Toc144143112)

[2.2. Spring Framework 9](#_Toc144143113)

[2.2.1. Giới thiệu về Spring Framework 9](#_Toc144143114)

[2.2.2. Kiến trúc tổng quan của Spring Framework 10](#_Toc144143115)

[2.2.3. Spring Cloud 12](#_Toc144143116)

[2.2.4. Spring Boot 15](#_Toc144143117)

[2.2.5. Spring Security 16](#_Toc144143118)

[2.2.6. Một số thành phần khác của Spring Framework 17](#_Toc144143119)

[2.2.7. Những tính năng nổi bật khi sử dụng Spring Framework 18](#_Toc144143120)

[2.2.8. Những khó khăn khi tiếp cận Spring Framework 18](#_Toc144143121)

[2.2.9. Kết luận về Spring Framework 19](#_Toc144143122)

[2.3. React Framework 20](#_Toc144143123)

[2.3.1. Giới thiệu về React Framework 20](#_Toc144143124)

[2.3.2. React JS 21](#_Toc144143125)

[2.3.3. React Native 23](#_Toc144143126)

[2.3.4. Redux Saga – thư viện quản lý tác vụ bất đồng bộ trong React JS và React Native 25](#_Toc144143127)

[2.3.5. Kết luận về React Framework 27](#_Toc144143128)

[2.4. Graphql 27](#_Toc144143129)

[2.4.1. Tổng quan về Graphql 27](#_Toc144143130)

[2.4.2. Các đặc điểm quan trọng của Graphql 28](#_Toc144143131)

[2.4.3. So sánh giữa Graphql và REST 29](#_Toc144143132)

[2.4.4. Khó khăn ban đầu khi làm việc với Graphql 30](#_Toc144143133)

[2.4.5. Kết luận về GraphQL 31](#_Toc144143134)

[2.5. Docker 31](#_Toc144143135)

[2.6.1. Tổng quan về Docker 31](#_Toc144143136)

[2.6.2. Kiến trúc của Docker 32](#_Toc144143137)

[2.6.3. So sánh Docker và Virtual Machine 33](#_Toc144143138)

[2.6.4. Đặc điểm khiến Docker tương thích với Microservices 35](#_Toc144143139)

[2.6.5. Kết luận về Docker 35](#_Toc144143140)

[2.6. Các công nghệ liên quan khác 35](#_Toc144143141)

[2.6.1. Gradle - Dependency management 35](#_Toc144143142)

[2.6.2. JDBI 36](#_Toc144143143)

[2.6.3. gRPC 36](#_Toc144143144)

[2.6.4. Kafka - Distributed event streaming 37](#_Toc144143145)

[2.6.5. Keycloak 38](#_Toc144143146)

[2.6.6. Prometheus 39](#_Toc144143147)

[2.6.7. Grafana 40](#_Toc144143148)

[2.6.8. Zipkin 40](#_Toc144143149)

[2.6.9. MySQL 41](#_Toc144143150)

[CHƯƠNG 3 PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 42](#_Toc144143151)

[3.1. Bài toán và yêu cầu 42](#_Toc144143152)

[3.1.1. Giới thiệu bài toán 42](#_Toc144143153)

[3.1.2. Yêu cầu bài toán 43](#_Toc144143154)

[3.2. Yêu cầu chức năng 44](#_Toc144143155)

[3.2.1. Ứng dụng đặt hàng, đặt bàn cho người dùng 44](#_Toc144143156)

[3.2.2. Ứng dụng quản lý bán hàng cho quản lý hệ thống và nhân viên 51](#_Toc144143157)

[3.3. Yêu cầu phi chức năng 59](#_Toc144143158)

[3.4. UseCase Diagram 60](#_Toc144143159)

[3.4.1. Sơ đồ tổng quát 60](#_Toc144143160)

[3.4.2. Sơ đồ chi tiết Use Case 61](#_Toc144143161)

[3.4.3. Đặc tả Use Case 65](#_Toc144143162)

[3.5. Sequence Diagram 71](#_Toc144143163)

[3.6. Class Diagram 76](#_Toc144143164)

[3.6.1. Tổng quan về Hexagonal 76](#_Toc144143165)

[3.6.2. Áp dụng Hexagonal trong các dịch vụ của hệ thống 80](#_Toc144143166)

[3.6.3. Class diagram của từng dịch vụ 84](#_Toc144143167)

[3.7. Database Diagram 98](#_Toc144143168)

[3.7.1. Dịch vụ sản phẩm 98](#_Toc144143169)

[3.7.2. Dịch vụ người dùng 98](#_Toc144143170)

[3.7.3. Dịch vụ kho 99](#_Toc144143171)

[3.7.4. Dịch vụ đặt hàng, đặt bàn 99](#_Toc144143172)

[3.7.5. Dịch vụ thanh toán 100](#_Toc144143173)

[CHƯƠNG 4 KIẾN TRÚC TỔNG THỂ CỦA HỆ THỐNG 101](#_Toc144143174)

[4.1. Kiến trúc phần mềm 101](#_Toc144143175)

[4.1.1. Tổng quan kiến trúc phần mềm 101](#_Toc144143176)

[4.1.2. Mỗi dịch vụ lưu trữ dữ liệu ở một database riêng biệt 104](#_Toc144143177)

[4.1.3. Sử dụng Eureka Server định danh cho các dịch vụ 105](#_Toc144143178)

[4.1.4. Sử dụng Config Server quản lý cấu hình các dịch vụ 106](#_Toc144143179)

[4.1.5. Áp dụng OAuth2.0 bảo mật và ủy quyền tới các dịch vụ trong hệ thống 107](#_Toc144143180)

[4.2. Kiến trúc hệ thống 109](#_Toc144143181)

[4.2.1. Tổng quan kiến trúc hệ thống 109](#_Toc144143182)

[4.2.2. Máy chủ Microservices 111](#_Toc144143183)

[4.2.3. Máy chủ Keycloak 112](#_Toc144143184)

[4.2.4. Máy chủ Config-Server 113](#_Toc144143185)

[CHƯƠNG 5 KẾT QUẢ ĐẠT ĐƯỢC 114](#_Toc144143186)

[5.1. Kết quả đạt được 114](#_Toc144143187)

[5.1.1. Về kiến thức 114](#_Toc144143188)

[5.1.2. Về thái độ 115](#_Toc144143189)

[5.1.3. Về sản phẩm 115](#_Toc144143190)

[5.2. Những điểm mạnh của đề tài 117](#_Toc144143191)

[5.3. Những hạn chế của đề tài 118](#_Toc144143192)

[CHƯƠNG 6 KẾT LUẬN 119](#_Toc144143193)

[6.1. Kết luận 119](#_Toc144143194)

[6.2. Thuận lợi 119](#_Toc144143195)

[6.3. Khó khăn và thách thức 119](#_Toc144143196)

[6.4. Hướng phát triển 120](#_Toc144143197)

[TÀI LIỆU THAM KHẢO 121](#_Toc144143198)

# CHƯƠNG 1 GIỚI THIỆU

## Đặt vấn đề

Trong thời đại mà khoa học công nghệ phát triển mạnh mẽ như hiện nay, Việt Nam đã có những bước tiến hội nhập vào dòng chảy của khoa học và công nghệ tiên tiến trên thế giới. Một trong những lĩnh vực quan trọng và phát triển mạnh mẽ là công nghệ thông tin, đóng vai trò quan trọng trong các ngành khoa học công nghệ và ảnh hưởng đến tất cả các khía cạnh trong đời sống xã hội cũng như các công tác tổ chức quản lý.

Trong lĩnh vực quản lý, công nghệ thông tin đã được ứng dụng rộng rãi và góp phần tạo nên sự khoa học, chính xác và hiệu quả trong công việc của các nhà quản lý. Các công ty, tổ chức và cơ quan đều muốn áp dụng công nghệ thông tin vào hoạt động của mình để nâng cao hiệu quả và tăng cường quản lý.

Với sự phát triển của các công ty trong các lĩnh vực sản xuất, kinh doanh và dịch vụ, quy mô của chúng ngày càng lớn. Quá trình sản xuất và bán hàng diễn ra qua nhiều khâu và công đoạn khác nhau. Số lượng hàng hóa sản xuất và khách hàng sử dụng ngày càng tăng lên. Điều này khiến cho công tác quản lý phức tạp hơn. Do đó, việc áp dụng công nghệ thông tin trong công việc quản lý trở thành một nhu cầu thiết yếu và quan trọng đối với các công ty.

Một trong những thách thức trong quản lý là việc lưu trữ và tra cứu thông tin hóa đơn. Khi có nhiều cơ sở và hàng ngày phải viết nhiều hóa đơn, việc lưu trữ và tìm kiếm các hóa đơn này có thể tốn nhiều thời gian và đôi khi gặp khó khăn. Thống kê và báo cáo hàng tháng hoặc hàng năm cũng trở nên khó khăn khi số lượng hóa đơn bán hàng ngày càng tăng. Việc lặp đi lặp lại quá nhiều lần gây khó khăn cho người quản lý trong việc lưu trữ và quản lý thông tin.

Do đó, công tác quản lý bán hàng đóng vai trò quan trọng trong mọi cửa hàng. Để nâng cao hiệu quả, cần phát triển và hoàn thiện quản lý bán hàng để đáp ứng với thực tế của từng cửa hàng. Hiện nay, có nhiều hệ thống quản lý bán hàng tiêu biểu như KiotViet, PopApp, … Tuy nhiên, những hệ thống này thường đòi hỏi sự kết hợp với các thiết bị đặc biệt và chi phí lắp đặt, bảo trì cao.

Với mục tiêu giảm thiểu chi phí, tăng hiệu suất hoạt động, dễ dàng bảo trì và dễ dàng mở rộng chúng em muốn xây dựng một phần mềm quản lý bán hàng với chức năng vận hành và quản lý toàn bộ quy trình hoạt động của cửa hàng trên chính thiết bị điện thoại di động và máy tính cá nhân. Phần mềm giúp cửa hàng thu thập dữ liệu bán hàng một cách tự động, liên tục từ việc quản lý hàng hóa, quản lý khách hàng đến việc quản lý doanh thu và lợi nhuận. Từ đó cửa hàng tiết kiệm được thời gian và công sức, tăng cường độ chính xác, giảm thiểu sai sót. Người chủ cửa hàng sẽ có cái nhìn rõ ràng về hiệu suất kinh doanh, xu hướng tiêu dùng, và các chỉ số quan trọng khác, giúp đưa ra những quyết định kinh doanh thông minh, tối ưu hóa chiến lược và định hướng phát triển tiếp theo.

Chúng em tin rằng việc áp dụng công nghệ thông tin vào quản lý bán hàng là một bước cần thiết để cải thiện hiệu suất và thành công của cửa hàng trong thị trường cạnh tranh ngày này, đó cũng là lý do nhóm em quyết định chọn đề tài “Ứng dụng các công nghệ React Native, React và Spring Boot trong kiến trúc Microservices để xây dựng phần mềm quản lý bán hàng cho nhà hàng và dịch vụ đồ uống" cho khóa luận của nhóm.

## Mục tiêu

Nghiên cứu nắm vững quy trình bán hàng, nhập hàng, quản lý thông tin.

Nghiên cứu các công cụ, nền tảng lập trình được sử dụng để xây dựng hệ thống như là React Native, React, Spring Boot và kiến trúc Microservices.

Xây dựng phần mềm quản lý bán hàng trên các nền tảng mobile, tablet và website, quản lý được hầu hết các thông tin và xử lý các quy trình nghiệp vụ trong quy trình quản lý bán hàng như là quản lý thông tin sản phẩm, quản lý nhân viên, khách hàng, xử lý đặt hàng, xuất ra các thống kê, báo cáo dưới dạng số liệu tổng hợp, đồ thị trực quan nhằm giúp chủ cửa hàng có thể nắm bắt được tình hình kinh doanh và đứa ra các quyết định đúng đắn nhất.

## Đối tượng nghiên cứu

Cơ chế, cách thức hoạt động của các nghiệp vụ bán hàng, nhập hàng, quản lý thông tin của các cửa hàng đồ ăn và dịch vụ đồ uống.

Tìm hiểu về các công nghệ phía Back-End, bao gồm ngôn ngữ lập trình Java, Spring Boot, Spring Security, Spring Cloud, MySQL, kiến trúc Microservices.

Tìm hiểu về các công nghệ phía Front-End, bao gồm HTML, CSS, JavaScript, TypeScript, React, React Native, Redux Saga.

Tìm hiểu một số công nghệ phụ trợ khác để việc lập trình, quản lý code, quản lý hệ thống trở nên dễ dàng và thuận tiện hơn, bao gồm Docker, Docker compose, Gradle, Keycloak, Zipkin, Prometheus, Grafana.

## Phạm vi nghiên cứu

Tập chung nghiên cứu và xây dựng hệ thống quản lý bán hàng cho doanh nghiệp kinh doanh dịch vụ nhà hàng và đồ uống, không thực hiện cho các ngành khác.

Thời gian nghiên cứu: 30/01/2023 - 08/09/2023

# 

# CHƯƠNG 2 TỔNG QUAN CÔNG NGHỆ

## Tổng quan về Microservices

### Giới thiệu về Microservices

Microservice là một kỹ thuật phát triển phần mềm, là một biến thể của kiến trúc hướng dịch vụ (SOA - xuất hiện trong những năm 1990). SOA tập trung vào việc chia ứng dụng thành các dịch vụ nhỏ và độc lập, nhằm tạo sự linh hoạt và tái sử dụng trong phát triển phần mềm. Tuy nhiên, SOA thường gặp khó khăn trong việc triển khai thực tế do sự phức tạp của các dịch vụ lớn và sự phụ thuộc mạnh mẽ giữa các dịch vụ. Bắt đầu vào khoảng giữa những năm 2000, một số công ty công nghệ lớn như Amazon, Netflix và Twitter đã bắt đầu áp dụng kiến trúc microservice để tăng cường tính mở rộng, khả năng module và triển khai liên tục trong hệ thống của họ. Việc triển khai thành công của những công ty này đã thu hút sự quan tâm và lan rộng kiến trúc microservice sang các lĩnh vực phát triển phần mềm khác. Đến những năm 2010, kiến trúc microservice đã trở nên phổ biến và trở thành một trong những phong cách kiến trúc phổ biến trong công nghiệp phát triển phần mềm. Các công ty và tổ chức phát triển phần mềm khắp thế giới đã bắt đầu chuyển từ kiến trúc monolithic truyền thống sang kiến trúc microservice để tận dụng các lợi ích vượt trội của kiến trúc này trong việc xây dựng các ứng dụng phức tạp và linh hoạt. Cho đến hiện nay, kiến trúc microservice tiếp tục phát triển và tiến xa hơn trong việc tạo ra các hệ thống phần mềm phức tạp, đáp ứng nhanh chóng và hiệu quả yêu cầu của thị trường và người dùng.

Trong kiến trúc microservices, các logic ứng dụng được phân chia thành các phần riêng biệt, được gọi là các microservices. Thay vì phải hiện thực tất cả các logic ứng dụng trong một ứng dụng nguyên khối (monolithic application), mỗi microservice độc lập triển khai một phần cụ thể của ứng dụng hoặc một khả năng kinh doanh. Các microservices có thể được viết bằng nhiều ngôn ngữ lập trình và được triển khai và mở rộng độc lập với nhau. Mỗi microservice hoạt động như một ứng dụng con độc lập, có khả năng giao tiếp với các microservices khác thông qua các giao thức giao tiếp nhẹ nhàng như HTTP, RPC hoặc các hàng đợi tin nhắn. Việc phân chia ứng dụng thành các microservices độc lập giúp cải thiện tính module và tách biệt các chức năng, giúp ứng dụng dễ hiểu, phát triển, kiểm thử và duy trì.

Hình 2.: Minh họa về kiến trúc Microservices

*(Nguồn:* [*https://microservices.io/i/Microservice\_Architecture.png*](https://microservices.io/i/Microservice_Architecture.png)*)*

### Những đặc trưng của Microservices

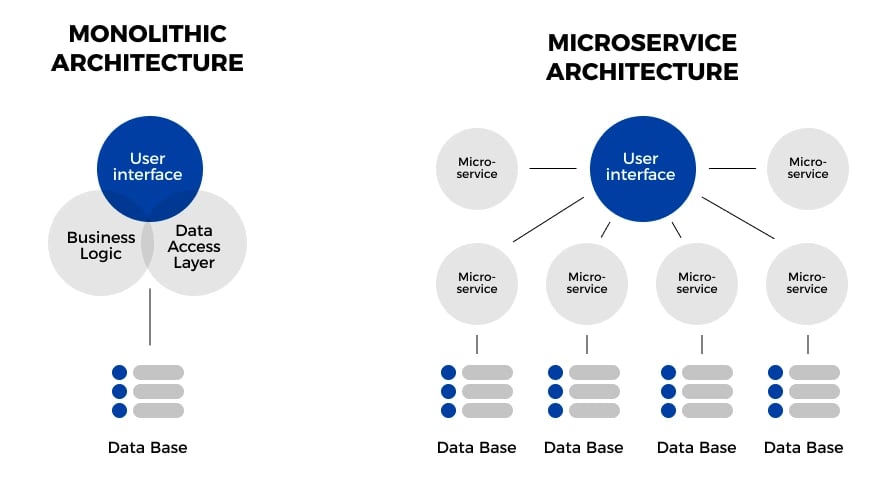
Kiến trúc Microservices cung cấp một cách tiếp cận linh hoạt và hiệu quả để quản lý và phát triển các ứng dụng phức tạp. Trong môi trường thương mại điện tử, nơi mà số lượng người dùng và các giao dịch có thể tăng vọt trong thời gian ngắn, khả năng chịu tải và mở rộng kịp thời là cực kỳ quan trọng. Microservices cho phép từng dịch vụ trong hệ thống hoạt động độc lập, qua đó:

1. **Mở rộng linh hoạt**: Các dịch vụ có thể được mở rộng một cách độc lập, giúp tối ưu hóa tài nguyên và quản lý tải hiệu quả hơn so với kiến trúc monolithic truyền thống.
2. **Độc lập về cơ sở dữ liệu**: Mỗi microservice có thể sử dụng loại cơ sở dữ liệu phù hợp nhất với nhu cầu và tải trọng của nó, tăng cường hiệu suất và tính sẵn sàng của dữ liệu.
3. **Đa ngôn ngữ lập trình**: Hỗ trợ phát triển bằng nhiều ngôn ngữ lập trình khác nhau, giúp tận dụng tối đa sức mạnh và chuyên môn của từng ngôn ngữ để giải quyết các vấn đề cụ thể.
4. **Cách ly và chịu lỗi**: Sự cô lập giữa các services giúp đảm bảo rằng sự cố ở một dịch vụ không làm ảnh hưởng đến toàn bộ hệ thống, qua đó giảm thiểu downtime và mất mát dữ liệu.
5. **Bảo mật và giám sát**: Việc phân tách các dịch vụ cũng giúp dễ dàng triển khai các giải pháp bảo mật tốt hơn, từ việc xác thực người dùng cho đến việc mã hóa dữ liệu.

**Ứng dụng kiến trúc Microservices trong thương mại điện tử đòi hỏi sự đầu tư kỹ lưỡng vào cơ sở hạ tầng công nghệ, đặc biệt là các công nghệ hỗ trợ tính năng mở rộng tự động và quản lý tài nguyên hiệu quả. Việc này bao gồm:**

1. Load Balancing và API Gateway: Các giải pháp Load Balancer và API Gateway đóng vai trò quan trọng trong việc phân phối tải truy cập một cách đồng đều giữa các dịch vụ, đồng thời cung cấp một điểm truy nhập thống nhất cho các client. API Gateway giúp điều phối các yêu cầu đến dịch vụ phù hợp, trong khi đó Load Balancer đảm bảo các dịch vụ không bị quá tải, góp phần vào việc cải thiện độ ổn định và hiệu suất của hệ thống.
2. Message Queuing: Hệ thống hàng đợi tin nhắn (message queuing) cho phép các dịch vụ giao tiếp một cách không đồng bộ, giúp giảm bớt sự phụ thuộc lẫn nhau giữa các services và cải thiện khả năng chịu lỗi của hệ thống. Khi một dịch vụ xử lý chậm, các yêu cầu có thể được lưu vào hàng đợi và xử lý sau, mà không làm ảnh hưởng đến trải nghiệm người dùng cuối.
3. Container Orchestration: Công nghệ điều phối container, như Kubernetes, cung cấp khả năng tự động hóa triển khai, mở rộng và quản lý vòng đời của các microservices. Sử dụng containerization không chỉ giúp đơn giản hóa việc phát triển và triển khai dịch vụ mà còn đảm bảo tính nhất quán và cô lập môi trường giữa phát triển và sản xuất, giúp giảm thiểu các vấn đề liên quan đến "nó hoạt động trên máy tôi".

### So sánh kiến trúc Microservices và kiến trúc nguyên khối (Monolithic)



Hình 2.: So sánh giữa Monolithic và Microservices

*(Nguồn:* [*https://www.openlegacy.com/hs-fs/hubfs/Picture1.webp?width=889&height=478&name=Picture1.webp*](https://www.openlegacy.com/hs-fs/hubfs/Picture1.webp?width=889&height=478&name=Picture1.webp)*)*

Microservices và Monolithic là hai phong cách kiến trúc phát triển phần mềm có nhiều sự khác biệt quan trọng:

* **Tính tách biệt về bản chất (Concern by nature):**

*Microservices*: Microservices phân chia ứng dụng thành các dịch vụ độc lập với nhau, mỗi dịch vụ đảm nhận một phần của logic ứng dụng. Nhờ vậy mà tạo ra tính tách biệt về bản chất giữa các dịch vụ, giúp dễ dàng quản lý, phát triển và triển khai từng dịch vụ một cách độc lập.

*Monolithic*: Monolithic là kiểu kiến trúc truyền thống, toàn bộ ứng dụng được hiện thực trong một khối duy nhất. Vậy nên làm cho các phần của ứng dụng chịu sự phụ thuộc chặt chẽ vào nhau, khiến việc phát triển và triển khai trở nên phức tạp hơn.

* **Tính linh hoạt và khả năng phát triển độc lập:**

*Microservices*: Mỗi dịch vụ trong kiến trúc microservices có thể được phát triển bằng các công nghệ và công cụ khác nhau. Ví dụ, một dịch vụ có thể sử dụng ngôn ngữ lập trình Java, trong khi dịch vụ khác sử dụng Node.js. Từ đó cho phép các nhóm phát triển tận dụng những công nghệ mà họ có kiến thức và kỹ năng chuyên sâu.

*Monolithic*: Trong kiến trúc monolithic, toàn bộ ứng dụng phải sử dụng cùng một ngôn ngữ lập trình, công nghệ và quy ước do vậy có thể gây nên hạn chế sự linh hoạt và sự đa dạng trong việc phát triển và triển khai ứng dụng.

* **Khả năng mở rộng và triển khai:**

*Microservices*: Microservices hỗ trợ phân phối và triển khai liên tục, cho phép ứng dụng mở rộng dễ dàng bằng cách triển khai thêm các dịch vụ mới khi cần thiết, giúp ứng dụng linh hoạt và đáp ứng tốt hơn với điều kiện trong môi trường tải cao.

*Monolithic*: Trong monolithic, việc mở rộng quy mô toàn bộ ứng dụng gây nhiều khó khăn do sự kết hợp mạnh mẽ giữa các thành phần. Việc thêm chức năng mới thường yêu cầu phải triển khai lại toàn bộ ứng dụng, hơn nữa có thể dẫn đến việc lãng phí tài nguyên và không hiệu quả.

* **Tính tích hợp và khả năng tái sử dụng:**

*Microservices*: Microservices giúp ngăn chặn sự phụ thuộc vào một nhà cung cấp dịch vụ, ngôn ngữ lập trình hay framework cụ thể. Nhờ vậy giúp tránh tình trạng bị khóa vào một nhà cung cấp cụ thể và tạo điều kiện thuận lợi cho tái sử dụng dịch vụ giữa các ứng dụng khác nhau.

*Monolithic*: Trong monolithic, sự phụ thuộc chặt chẽ vào một ngôn ngữ, framework hay nhà cung cấp có thể dẫn đến sự khó khăn trong việc tái sử dụng các thành phần giữa các ứng dụng khác nhau.

### So sánh kiến trúc Microservices và kiến trúc hướng dịch vụ (SOA - Service-Oriented Architecture)

Những điểm khác biệt quan trọng giữa hai kiến trúc bao gồm:

* **Quy mô và phạm vi:**

*Microservices* tập trung vào chia nhỏ ứng dụng thành các dịch vụ nhỏ lẻ và độc lập. Mỗi dịch vụ tập trung vào một chức năng cụ thể và có thể được phát triển, triển khai, và quản lý độc lập nhau. Chúng có khả năng mở rộng dễ dàng và có thể sử dụng các công nghệ khác nhau cho từng dịch vụ.

*SOA* bên cạnh việc chia nhỏ các dịch vụ, còn nhấn mạnh việc phân tách các chức năng kinh doanh thành các dịch vụ độc lập, nhưng các dịch vụ này có thể phụ thuộc vào nhau và chia sẻ dữ liệu thông qua giao thức tầng ứng dụng như SOAP hoặc REST. SOA thường được triển khai trong môi trường doanh nghiệp lớn hơn, có quy mô phức tạp hơn và chịu sự quản lý tập trung hơn.

* **Tương tác giữa các dịch vụ:**

*Microservices* thường sử dụng cơ chế giao tiếp qua mạng như gRPC hoặc RESTful API để giao tiếp với nhau. Sự tương tác giữa các dịch vụ trong kiến trúc microservices thường là bất đồng bộ, giúp giảm thiểu sự phụ thuộc và tăng tính linh hoạt của hệ thống.

*SOA* thường sử dụng các giao thức giao tiếp đồng bộ truyền thống như SOAP hay XML-RPC, điều này đôi khi làm giảm tính linh hoạt của hệ thống và tạo ra sự phụ thuộc giữa các dịch vụ.

* **Độ phức tạp trong triển khai:**

*Microservices* có thể phức tạp hơn so với SOA do yêu cầu phải quản lý nhiều dịch vụ độc lập nhau. Việc phân tách dịch vụ nhỏ và quản lý chúng có thể đòi hỏi sự phối hợp và quản lý chặt chẽ hơn.

*SOA* có thể đơn giản hơn trong việc triển khai vì các dịch vụ có thể phụ thuộc vào nhau và được quản lý theo một phạm vi tổng thể hơn.

* **Mục tiêu và ứng dụng:**

*Microservices* thường được sử dụng trong các ứng dụng có tính mở rộng cao, cần tích hợp các công nghệ mới và chịu sự biến đổi nhanh chóng.

*SOA* thường được sử dụng trong các doanh nghiệp lớn, có nhu cầu tích hợp các ứng dụng và dịch vụ có sẵn trong hệ thống doanh nghiệp.

### Những khó khăn khi triển khai Microservices

**Sự phức tạp trong quản lý và điều hành:** Khi chia nhỏ ứng dụng thành nhiều service, có thể dẫn đến sự phức tạp về quản lý và điều hành. Số lượng service tăng lên và có thể trở nên khó kiểm soát, đặc biệt khi phải xử lý các sự phụ thuộc giữa các service.

**Tăng chi phí giao tiếp**: Với kiến trúc Microservices, giao tiếp giữa các service phải thông qua mạng, điều này có thể tăng chi phí giao tiếp và ảnh hưởng đến hiệu suất của hệ thống.

**Triển khai phức tạp hơn**: Triển khai và quản lý các service đòi hỏi sự phối hợp và quản lý khá phức tạp và phải đảm bảo rằng các service hoạt động một cách đồng nhất và tương tác tốt với nhau.

### Kết luận về Microservices

Microservices mang lại tính linh hoạt, khả năng phát triển và triển khai độc lập, cũng như tích hợp và khả năng tái sử dụng tốt hơn so với kiến trúc monolithic. Tuy nhiên, việc triển khai microservices có thể đòi hỏi quản lý và phát triển phức tạp hơn so với monolithic, do sự tách biệt và đa dạng của các dịch vụ. Trong đề tài này, hệ thống áp dụng kiến trúc Microservice để xây dựng hệ thống quản lý bán hàng, có khả năng mở rộng cao, đồng thời tích hợp nhiều công cụ quản lý hệ thống và công cụ DevOps giúp quản lý và giám sát các dịch vụ trong hệ thống một cách trực quan và hiệu quả, đồng thời cũng đảm bảo tính ổn định của hệ thống.

# CHƯƠNG 3 PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Bài toán và yêu cầu

### Giới thiệu bài toán

Trong bối cảnh của sự phát triển không ngừng của các công ty trong lĩnh vực sản xuất, kinh doanh và dịch vụ, việc quản lý quy mô ngày càng lớn đã trở thành một thách thức. Các công đoạn sản xuất và bán hàng diễn ra phức tạp và tương tác qua nhiều khâu khác nhau,