**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

ĐỖ HUY HOÀNG MSSV: N19DCCN063 CHUYÊN NGÀNH: CÔNG NGHỆ PHẦN MỀM 2019-2024 Lớp: D19CQCNPM01-N

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------



**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “XÂY DỰNG ỨNG DỤNG KINH DOANH PHÒNG TRỌ”**

|  |  |
| --- | --- |
| **Người hướng dẫn:** | **TS. NGUYỄN HỒNG SƠN** |
| **Sinh viên thực hiện:** | **ĐỖ HUY HOÀNG** |
| **Mã số sinh viên:** | **N19DCCN063** |
| **Lớp:** | **D19CQCNPM01-N** |
| **Hệ:** | **Đại học chính quy** |

TP. HCM

2023

**TP. HỒ CHÍ MINH, NĂM 2023**

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

-----------------------------

A red circle with a star and a book

Description automatically generated

**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “XÂY DỰNG ỨNG DỤNG KINH DOANH PHÒNG TRỌ”**

|  |  |
| --- | --- |
| **Người hướng dẫn:** | **TS. NGUYỄN HỒNG SƠN** |
| **Sinh viên thực hiện:** | **ĐỖ HUY HOÀNG** |
| **Mã số sinh viên:** | **N19DCCN063** |
| **Lớp:** | **D19CQCNPM01-N** |
| **Hệ:** | **Đại học chính quy** |

**TP. HỒ CHÍ MINH,** **NĂM 2023**

# LỜI CẢM ƠN

Lời đầu tiên em xin phép gửi lời tri ân sâu sắc đến các thầy cô trường Học Viện Công Nghệ Bưu Chính Viễn Thông cơ sở tại TP.HCM đã tận tình dẫn dắt và truyền đạt cho em rất nhiều kiến thức quý báu trong những năm học vừa qua. Thật sự, em đã có cơ hội được trau dồi kiến thức và rèn luyện bản thân dưới sự hướng dẫn tận tâm của các thầy cô, đó là điều vô cùng quý giá đối với em.

Với lòng biết ơn chân thành, em xin gửi lời cảm ơn đặc biệt tới TS. Nguyễn Hồng Sơn, người đã dành thời gian và công sức hướng dẫn em trong quá trình nghiên cứu và thực hành đề tài. Nhờ sự hỗ trợ và động viên của thầy, em đã vượt qua những khó khăn, trở ngại và phát triển kỹ năng cũng như tư duy nghiên cứu. Thầy đã không chỉ truyền đạt kiến thức một cách chuyên sâu mà còn khuyến khích em phát triển tinh thần làm việc nghiêm túc. Điều này sẽ không chỉ có ý nghĩa to lớn mà còn trở thành những phẩm chất quý báu trong cuộc sống và công việc tương lai.

Cuối cùng, em xin chúc quý thầy cô khoa Công nghệ thông tin 2 và thầy Nguyễn Hồng Sơn dồi dào sức khỏe để tiếp tục sứ mệnh cao cả truyền đạt kiến thức cho thế hệ mai sau. Sự đóng góp của quý thầy cô không chỉ giúp đỡ em mà còn hình thành và phát triển tài năng của nhiều thế hệ sinh viên khác. Em hy vọng rằng những giá trị mà em đã nhận được từ quý thầy cô sẽ trở thành nguồn động lực và nền tảng vững chắc cho em trên con đường chinh phục những ước mơ và mục tiêu trong tương lai. Một lần nữa, em xin chân thành cảm ơn và mong rằng quý thầy cô sẽ luôn tràn đầy niềm vui và thành công trong sự nghiệp giáo dục.

Xin cảm ơn!

TP. Hồ Chí Minh, ngày…. tháng 10 năm 2023

**Sinh viên thực hiện**

**Đỗ Huy Hoàng**

# MỤC LỤC

LỜI CẢM ƠN i

MỤC LỤC ii

DANH MỤC CÁC KÝ HIỆU VÀ CHỮ VIẾT TẮT iv

DANH MỤC CÁC BẢNG VẼ v

DANH MỤC CÁC HÌNH vi

MỞ ĐẦU viii

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1

1.1 Bối cảnh đề tài 1

1.2 Mục đích đề tài 1

1.3 Mục tiêu của đề tài 1

1.4 Phương pháp tiến hành 2

1.5 Kết quả phải đạt 2

1.6 Cơ sở lý luận và nền tảng xây dựng ứng dụng 3

1.6.1 Công cụ lập trình 3

1.6.2 Nền tảng công nghệ 3

1.6.3 Giới thiệu ngôn ngữ lập trình Java 3

1.6.4 Giới thiệu hệ quản trị cơ sở dữ liệu MySQL 4

1.6.5 Giới thiệu Eclipse 5

1.6.6 Giới thiệu Visual Studio Code 6

1.6.7 Giới thiệu Framework Angular 6

1.6.8 Giới thiệu Spring boot 13

1.6.9 Giới thiệu Node.js 16

1.6.10 JWT (JSON Web Token) 17

CHƯƠNG 2: KHẢO SÁT HỆ THỐNG 19

2.1 Khảo sát hiện trạng 19

2.2 Giải pháp đề xuất. 19

2.2.1 Yêu cầu phi chức năng: 20

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 21

3.1 Diagram database theo kiến trúc monolithic (MySQL) 21

3.2 Usecase 22

3.2.1 Use case tổng quát 22

3.3 Thiết kế dữ liệu 23

3.3.1 Từ điển dữ liệu 23

3.4 Sơ đồ tuần tự 29

3.4.1 Sơ đồ tuần tự đăng nhập 29

CHƯƠNG 4: THIẾT KẾ PHẦN MỀM 30

4.1 Chức năng chủ trọ 30

4.1.1 Quản lý khu trọ 30

4.1.2 Quản lý phòng trọ 31

4.1.3 Quản lý dịch vụ 32

4.1.4 Quản lý thiết bị 34

4.1.5 Quản lý hợp đồng 36

4.1.6 Quản lý điện nước 37

4.1.7 Xuất hoá đơn 39

4.1.8 Trả phòng 40

4.1.9 Đăng bài 41

4.1.10 Quản lý đặt phòng 43

4.2 Chức năng khách tìm trọ 44

CHƯƠNG 5: KẾT LUẬN VÀ KIẾN NGHỊ 49

TÀI LIỆU THAM KHẢO 50

# DANH MỤC CÁC KÝ HIỆU VÀ CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| **Cụm từ viết tắt hoặc ngoại ngữ** | **Nghĩa** |
| API: Application Programming Interface | Giao diện lập trình ứng dụng |
| User Interface | Giao diện người dùng |
| REST: Representational State Transfer | Một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web |
| MySQL: My Structured Query Language | Ngôn ngữ truy vấn dữ liệu |
| CSDL | Cơ sở dữ liệu |
| HTML: Hypertext Markup Language | Ngôn ngữ siêu văn bản |
| IDE: Integrated Development Environment | Môi trường phát triển tích hợp |
| ERD: Entity Relationship Diagram | Sơ đồ mối quan hệ thực thể |
| ONLINE | Trực tuyến |
| MVC: Model-View-Controller | Mô hình thiết kế phần mềm |
| TELEX: Teleprinter Exchange | Một phương thức truyền thông điện tử |
| FAX: Facsimile | Một phương thức truyền tải |
| FTP: File Transfer Protocol | Giao thức truyền tệp |
| EMAIL: Electronic Mail | Thư điện tử |
| ADMIN: Administrator | Quản trị viên |
| DEMO: Demonstration | Chạy thử |

# DANH MỤC CÁC BẢNG VẼ

[Bảng vẽ 3.1: Bảng từ điển Users 23](#_Toc151586132)

[Bảng vẽ 3.3: Bảng từ điển User Preference 23](#_Toc151586133)

[Bảng vẽ 3.4: Bảng từ điển User Role 23](#_Toc151586134)

[Bảng vẽ 3.5: Bảng từ điển Role 23](#_Toc151586135)

[Bảng vẽ 3.6: Bảng từ điển Refresh Token 24](#_Toc151586136)

[Bảng vẽ 3.7: Bảng từ điển Accomodations 24](#_Toc151586137)

[Bảng vẽ 3.8: Bảng từ điển Address 24](#_Toc151586138)

[Bảng vẽ 3.9: Bảng từ điển Ward 24](#_Toc151586139)

[Bảng vẽ 3.10: Bảng từ điển District 24](#_Toc151586140)

[Bảng vẽ 3.11: Bảng từ điển Province 25](#_Toc151586141)

[Bảng vẽ 3.12: Bảng từ điển Accomodation Services 25](#_Toc151586142)

[Bảng vẽ 3.13: Bảng từ điển Rooms 25](#_Toc151586143)

[Bảng vẽ 3.14: Bảng từ điển Images 25](#_Toc151586144)

[Bảng vẽ 3.15: Bảng từ điển Bills 26](#_Toc151586145)

[Bảng vẽ 3.16: Bảng từ điển Tenants 26](#_Toc151586146)

[Bảng vẽ 3.17: Bảng từ điển Contract 27](#_Toc151586147)

[Bảng vẽ 3.18: Bảng từ điển Contract Service 27](#_Toc151586148)

[Bảng vẽ 3.19: Bảng từ điển Electric Water Num 27](#_Toc151586149)

[Bảng vẽ 3.20: Bảng từ điển Deposit 27](#_Toc151586150)

[Bảng vẽ 3.21: Bảng từ điển Equipment 28](#_Toc151586151)

[Bảng vẽ 3.22: Bảng từ điển Booking 28](#_Toc151586152)

[Bảng vẽ 3.23: Bảng từ điển Bank Account Info 28](#_Toc151586153)

[Bảng vẽ 3.24: Bảng từ điển Post 28](#_Toc151586154)

[Bảng vẽ 3.25: Bảng từ điển Post Utilities 29](#_Toc151586155)

[Bảng vẽ 3.26: Bảng từ điển Invoice Type 29](#_Toc151586156)

[Bảng vẽ 3.27: Bảng từ điển Service Bill 29](#_Toc151586157)

# DANH MỤC CÁC HÌNH

[Hình 1.1: Giới thiệu ngôn ngữ Java 4](#_Toc151586159)

[Hình 1.2: Giới thiệu My SQL 5](#_Toc151586160)

[Hình 1.3: Giới thiệu eclipse 5](#_Toc151586161)

[Hình 1.4: Giới thiệu Visual Studio Code 6](#_Toc151586162)

[Hình 1.5: Giới thiệu Angular 7](#_Toc151586163)

[Hình 1.6: Giới thiệu Spring boot 13](#_Toc151586164)

[Hình 1.7: Giải thích đơn giản về Spring Boot 14](#_Toc151586165)

[Hình 1.8: Kiến trúc của JPA 14](#_Toc151586166)

[Hình 1.9: Spring Integration Objects 15](#_Toc151586167)

[Hình 1.10: Pattern Spring Cloud 15](#_Toc151586168)

[Hình 1.11: Giới thiệu Node JS 17](#_Toc151586169)

[Hình 1.12: Giới thiệu về JWT 18](#_Toc151586170)

[Hình 3.1: Tổng quan Diagram database 21](#_Toc151586171)

[Hình 3.2: Usecase tổng quát 22](#_Toc151586172)

[Hình 3.3: Sơ đồ tuần tự đăng nhập 29](#_Toc151586173)

[Hình 4.1: Giao diện chức năng quản lý khu trọ 30](#_Toc151586174)

[Hình 4.2: Giao diện chức năng thêm khu trọ 30](#_Toc151586175)

[Hình 4.3: Giao diện chức năng chỉnh sửa khu trọ 31](#_Toc151586176)

[Hình 4.4: Giao diện chức năng quản lý phòng trọ 31](#_Toc151586177)

[Hình 4.5: Giao diện chức năng thêm phòng trọ 32](#_Toc151586178)

[Hình 4.6: Giao diện chức năng chỉnh sửa phòng trọ 32](#_Toc151586179)

[Hình 4.7: Giao diện chức năng quản lý dịch vụ 33](#_Toc151586180)

[Hình 4.8: Giao diện chức năng thêm dịch vụ 33](#_Toc151586181)

[Hình 4.9: Giao diện chức năng chỉnh sửa dịch vụ 34](#_Toc151586182)

[Hình 4.10: Giao diện chức năng quản lý thiết bị 34](#_Toc151586183)

[Hình 4.11: Giao diện chức năng thêm thông tin thiết bị 35](#_Toc151586184)

[Hình 4.12: Giao diện chỉnh sửa thông tin thiết bị 35](#_Toc151586185)

[Hình 4.13: Giao diện chức năng quản lý hợp đồng 36](#_Toc151586186)

[Hình 4.14: Giao diện chức năng thêm hợp đồng 36](#_Toc151586187)

[Hình 4.15: Giao diện chức năng chỉnh sửa hợp đồng 37](#_Toc151586188)

[Hình 4.16: Giao diện chức năng quản lý chỉ số điện nước 37](#_Toc151586189)

[Hình 4.17: Giao diện chức năng nhập chỉ số điện nước 38](#_Toc151586190)

[Hình 4.18: Giao diện chức năng chỉnh sửa thông tin chỉ số điện nước 38](#_Toc151586191)

[Hình 4.19: Giao diện xem hoá đơn phòng hàng tháng 39](#_Toc151586192)

[Hình 4.20: Giao diện xem thông tin chi tiết trước khi xuất hoá đơn 39](#_Toc151586193)

[Hình 4.21: Giao diện chỉnh sửa hoá đơn chưa thanh toán 40](#_Toc151586194)

[Hình 4.22: Giao diện quản lý hoá đơn trả phòng 40](#_Toc151586195)

[Hình 4.23: Giao diện chỉnh sửa hoá đơn trả phòng khi chưa thanh toán 41](#_Toc151586196)

[Hình 4.24: Giao diện email hoá đơn 41](#_Toc151586197)

[Hình 4.25: Giao diện quản lý bài đăng 42](#_Toc151586198)

[Hình 4.26: Giao diện tạo bài đăng cho thuê phòng 42](#_Toc151586199)

[Hình 4.27: Giao diện chỉnh sửa bài đăng 43](#_Toc151586200)

[Hình 4.28 Giao diện quản lý thông tin đặt phòng 43](#_Toc151586201)

[Hình 4.29: Giao diện tạo nhanh hợp đồng, đặt cọc, khách thuê từ đặt phòng 44](#_Toc151586202)

[Hình 4.30: Giao diện trang tìm phòng trọ 44](#_Toc151586203)

[Hình 4.31: Giao diện tìm kiếm phòng trọ theo địa chỉ 45](#_Toc151586204)

[Hình 4.32: Giao diện kết quả tìm kiếm theo địa chỉ 45](#_Toc151586205)

[Hình 4.33: Giao diện tìm kiếm theo khoảng giá 46](#_Toc151586206)

[Hình 4.34: Giao diện tìm kiếm phòng trọ theo diện tích 46](#_Toc151586207)

[Hình 4.35: Giao diện xem chi tiết phòng trọ 47](#_Toc151586208)

[Hình 4.36: Giao diện xem chi tiết phòng trọ 47](#_Toc151586209)

[Hình 4.37: Giao diện đặt phòng trọ 48](#_Toc151586210)

[Hình 4.38: Giao diện chọn ngày xem phòng trọ khi đặt phòng 48](#_Toc151586211)

# MỞ ĐẦU

Trong thời đại số hoá hiện nay, việc sử dụng công nghệ số đem lại nhiều lợi ích cho người dùng. Tận dụng các công nghệ kỹ thuật số giúp chúng ta tối ưu hoá quá trình quản lý, giúp việc quản lý trở nên dễ dàng và hiệu quả, hơn thế nữa việc tìm kiếm thông tin cũng trở nên đơn giản và tiện lợi. Do đó, việc quản lý và tìm kiếm phòng trọ ngày càng dễ dàng hơn với sự đóng góp của các công nghệ hiện đại.

Với những nhu cầu thực tế từ việc tìm kiếm phòng cũng như quản lý các phòng trọ cho các chủ trọ, em đã lựa chọn đề tài ‘Xây dựng ứng dụng kinh doanh phòng trọ’, ứng dụng hỗ trợ các chủ trọ quản lý các phòng và khách thuê hiệu quả, trực quan và tiết kiệm thời gian, ngoài ra còn là kênh trung gian giữa các chủ trọ và những người có nhu cầu thuê trọ, việc tìm phòng trọ trở nên đơn giản hơn so với trước đây. Vận dụng các công nghệ mới của Spring framework, cụ thể là Spring Boot để thiết kế và phát triển ứng dụng web. Hệ thống đơn giản với các chức năng cơ bản và đáp ứng đủ các nhu cầu cần thiết hỗ trợ người tìm phòng trọ và giúp các chủ trọ quản lý các phòng trọ thông qua website.

Trong báo cáo này, em sẽ trình bày về quá trình xây dựng web kinh doanh phòng trọ sử dụng Spring Boot, những lợi thế của Spring Boot và áp dụng các ưu điểm đặc trưng để xây dựng website.

# GIỚI THIỆU ĐỀ TÀI

## Bối cảnh đề tài

Với sự phát triển của công nghệ, việc áp dụng công nghệ vào quản lý trở nên dễ dàng hơn, các công cụ hỗ trợ quản lý dần ra đời, góp phần lớn trong việc quản lý một cách chi tiết, minh bạch. Các phần mềm giúp tiết kiệm phần lớn thời gian thu thập dữ liệu, tránh mất mát dữ liệu, hạn chế nhiều giấy tờ cũng như các quy trình phức tạp để quản lý, đặc biệt trong quản lý nhà trọ, cần có quy trình quản lý và tự động giúp các chủ trọ quản lý phòng và đăng tin cho thuê cho các khu trọ dễ dàng, tiện dụng và hiệu quả hơn. Tạo điều kiện cho những người có nhu cầu tìm trọ, việc tìm kiếm trở nên đơn giản hơn, việc xem phòng, đặt cọc và liên hệ với các chủ quản lý nhà trọ dễ dàng hơn với sự tham gia của các phần mềm quản lý.

## Mục đích đề tài

Đề tài này nhằm mục đích nghiên cứu và xây dựng một ứng dụng web về lĩnh vực đặt thuê phòng trọ, giúp cho các chủ trọ có thể quản lý cho thuê phòng,... Cụ thể, đề tài này có các mục đích sau:

* Giúp các chủ trọ có thể quản lý dễ dàng những thông tin như: khu trọ, phòng trọ, người ở… Ngoài ra, ứng dụng web còn hỗ trợ chủ trọ theo dõi được tình trạng các phòng trong khu trọ của của mình như số lượng người ở từng phòng, số phòng trống, số phòng đang được đặt, xem thống kê về số phòng đã cho thuê, tình trạng hư hỏng của từng phòng, gửi hoá đơn hàng tháng cho khách thuê trọ,...
* Giúp người có nhu cầu tìm trọ có kênh liên lạc để dễ dàng tìm kiếm liên hệ đến các chủ trọ, lựa chọn được phòng phù hợp với nhu cầu cũng như tài chính của mình.

## Mục tiêu của đề tài

Đề tài này nhằm mục tiêu xây dựng web application sử dụng Spring Boot, với các tiêu chí sau:

* Ứng dụng web có giao diện thân thiện, dễ sử dụng và phù hợp với nhu cầu của người dùng và người quản trị viên của cửa hàng. Giao diện của ứng dụng web phải tuân thủ các nguyên tắc thiết kế giao diện người dùng (UI design principles), như đơn giản, rõ ràng, nhất quán, khả năng tương tác, khả năng phản hồi.
* Ứng dụng được Dependency injection trong Spring Boot trong khi xây dựng các API cho hệ thống.
* Ứng dụng REST ful Web Service cho giao tiếp giữa front-end và back-end cũng như một số bên thứ ba khác nếu có nhu cầu.
* Ứng dụng auto configuration trong Spring Boot, nhằm tối ưu hoá thời gian cấu hình, dễ dàng quản lý các phụ thuộc, các thư viện

## Phương pháp tiến hành

|  |  |  |  |
| --- | --- | --- | --- |
| **Nội dung** | **Mục đích** | **Cách tiến hành** | **Kết quả** |
| **Phân Tích Và Đặc Tả Yêu Cầu** | Hiểu rõ và chi tiết hóa yêu cầu của hệ thống từ người dùng hoặc người đặt hàng. Từ đó đưa ra được tài liệu đặc tả cho các chức năng | Thu thập yêu cầu thông qua các cuộc hội thoại, khảo sát thực tế, và tham khảo tài liệu. | Một tài liệu yêu cầu rõ ràng và chi tiết, chứa danh sách các chức năng và phi chức năng mà hệ thống phải đáp ứng. |
| **Thiết Kế Kiến Trúc** | Xây dựng kiến trúc tổng quan cho hệ thống. | Phân loại hệ thống thành các module và xác định giao diện giữa chúng. | Một mô hình kiến trúc biểu diễn cấu trúc tổng quan của hệ thống. |
| **Thiết kế các thành phần của PM** | Thiết kế chi tiết các thành phần bên trong phần mềm và cách chúng tương tác. | Bắt đầu từ giao diện cho mỗi usecase (forms); với mỗi request trên giao diện, em hãy chỉ ra cách mà PM sẽ xử lý request này (dùng API nào, gọi stored procedure nào, truy vấn view/table nào,..) để mô tả cách xử lý của PM, nhìn từ góc độ làm ra PM. | Các form, control, api/service |
| **Thiết Kế Hệ Thống Thông Tin** | Thiết kế cơ sở dữ liệu và ràng buộc | Thiết kế ERD  Xây dựng từ điển dữ liệu, chuẩn hóa lược đồ, ràng buộc và phân quyền | Lược đồ ERD, diagram, từ điển dữ liệu |
| **Cài đặt chương trình (lập trình)** | Biến đổi bản thiết kế thành mã nguồn thực sự, có thể thực thi trên máy tính.  Thực hiện các chức năng và tính năng mà phần mềm yêu cầu, dựa trên bản đặc tả đã được xác định trước đó. | Dựa trên yêu cầu của hệ thống, mục tiêu và yêu cầu kỹ thuật, chọn ra các công nghệ sẽ sử dụng để xây dựng hệ thống.  Phân chia công việc lập trình thành các module hoặc thành phần nhỏ, dựa trên bản thiết kế đã có.  Thực hiện việc viết mã, tuân theo các nguyên tắc và chuẩn lập trình đã đề ra, đồng thời đảm bảo mã nguồn dễ đọc, dễ bảo trì. | Mã nguồn hoàn chỉnh cho toàn bộ phần mềm, tuân thủ các yêu cầu kỹ thuật và chức năng đã đặt ra.  Các module hoặc thành phần của phần mềm có thể hoạt động độc lập hoặc kết hợp với nhau.  Phần mềm sẵn sàng cho giai đoạn triển khai và bảo trì tiếp theo. |

## Kết quả phải đạt

Xây dựng hệ thống quản lý kinh doanh phòng trọ hoàn hiện gồm các chức năng sau:

* Web application dành cho chủ trọ gồm các chức năng:
* Quản lý khu trọ
* Quản lý phòng trọ
* Quản lý thiết bị theo khu trọ và phòng trọ
* Quản lý dịch vụ theo khu trọ
* Quản lý đặt cọc giữ phòng trọ
* Quản lý hợp đồng thuê phòng
* Quản lý khách thuê theo hợp đồng
* Quản lý chỉ số điện nước theo các phòng trọ
* Xuất hoá đơn hàng tháng theo từng phòng
* Cấu hình cho phép hệ thống tự động xuất và gửi hoá đơn tiền phòng hàng tháng
* Quản lý trả phòng
* Quản lý đăng tin cho thuê phòng trên trang của hệ thống
* Nhận thông báo cũng như thông tin đặt phòng từ khách đặt trên hệ thống tìm phòng trọ
* Hệ thống gửi mail tự động nhắc nhở nhập chỉ số điện nước dựa theo ngày chốt chỉ số điện nước mà chủ trọ đã cài đặt
* Web application dành cho người muốn thuê trọ gồm các chức năng:
* Tìm kiếm phòng trọ theo các tiêu chí như: địa chỉ, mức giá, diện tích phòng
* Đặt phòng bằng cách để lại thông tin và chọn ngày đặt phòng nếu cần thiết
* Web application dành cho Admin gồm các chứ năng:
* Quản lý tài khoản các chủ trọ
* Cấp tài khoản và khoá hoặc mở khoá tài khoản của chủ trọ
* Hỗ trợ chủ trọ trong quá trình sử dụng

## Cơ sở lý luận và nền tảng xây dựng ứng dụng

### Công cụ lập trình

* Hệ quản trị cơ sở dữ liệu: MySQL.
* Công cụ lập trình: Eclipse 2022, Visual Studio Code 1.78

### Nền tảng công nghệ

* Angular framework version 15
* Node.js version 16.16.0
* Spring framework (Spring Boot) 3.1.1
* Spring security
* JWT (JSON Web Token)
* Azure cloud (cho mục đích lưu trữ ảnh)

### Giới thiệu ngôn ngữ lập trình Java

Java là một nền tảng phát triển các ứng dụng phần mềm có vị trí rất lớn trong những năm cuối thế kỉ 20, đầu thế kỉ 21. Đánh dấu sự trưởng thành của mô hình lập trình hướng đối tượng, nó được coi là một nền tảng mang tính cách mạng trong ngành phần mềm. Mô hình máy ảo Virtual Machine đã cho phép các ứng dụng viết bằng Java có thể chạy trên nhiều hệ điều hành khác nhau.

Một trong những ưu điểm nổi bật của Java là tính hướng đối tượng mạnh mẽ. Java hỗ trợ các khái niệm như lớp, đối tượng, kế thừa, đa hình và đóng gói. Điều này giúp tạo ra mã nguồn có cấu trúc rõ ràng, dễ bảo trì và mở rộng.



Hình 1.1: Giới thiệu ngôn ngữ Java

### Giới thiệu hệ quản trị cơ sở dữ liệu MySQL

MySQL là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng. Vì MySQL là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh. Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên internet. Người dùng có thể tải về MySQL miễn phí từ trang chủ. MySQL có nhiều phiên bản cho các hệ điều hành khác nhau: phiên bản Win32 cho các hệ điều hành dòng Windows, Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS,..

MySQL là một trong những ví dụ rất cơ bản về Hệ Quản trị Cơ sở dữ liệu quan hệ sử dụng Ngôn ngữ truy vấn có cấu trúc (SQL).

MySQL được sử dụng cho việc bổ trợ NodeJs, PHP, Perl, và nhiều ngôn ngữ khác, làm nơi lưu trữ những thông tin trên các trang web viết bằng NodeJs, PHP hay Perl,...



Hình 1.2: Giới thiệu My SQL

### Giới thiệu Eclipse

Eclipse là một công cụ hỗ trợ người dùng lập trình mã nguồn mở. Đây là một môi trường phát triển tích hợp dành cho Java, nhưng vẫn có thể sử dụng để lập trình bằng một số ngôn ngữ khác như C/C++, PHP, Ruby… Người dùng có thể sử dụng Eclipse để mở rộng hơn mã nguồn bằng cách chèn plugins cho project. Đặc biệt, Eclipse là phần mềm hoàn toàn miễn phí, với nhiều plugin hữu ích..

Eclipse có các ưu điểm như đa nền tảng: có thể chạy trên cả hệ điều hành Windows và Linux, tạo thuận lợi cho tích hợp liền mạch các công cụ bên trong và xuyên qua nhiều kiểu nội dung và các nhà cung cấp công cụ khác nhau, tính phổ biến của ngôn ngữ Java (ngôn ngữ sử dụng để viết các công cụ).



Hình 1.3: Giới thiệu eclipse

### Giới thiệu Visual Studio Code

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn mở và miễn phí được phát triển bởi Microsoft. Được thiết kế để hỗ trợ lập trình viên trong việc phát triển phần mềm.

Với VS Code, lập trình viên có thể viết mã nguồn cho nhiều ngôn ngữ lập trình khác như JavaScript, TypeScript, Python, C++, Java và nhiều ngôn ngữ khác. Trình soạn thảo mã của VS Code đi kèm với nhiều tính năng hữu ích như kiểm tra lỗi, gợi ý mã, định dạng mã tự động và tìm kiếm thông minh.

VS Code cũng hỗ trợ tích hợp với các công cụ và dịch vụ phổ biến khác như Git, Docker và Azure. Điều này cho phép lập trình viên làm việc với các hệ thống quản lý phiên bản, triển khai ứng dụng và phát triển trên đám mây một cách dễ dàng và tiện lợi.

A logo for a software company

Description automatically generated

Hình 1.4: Giới thiệu Visual Studio Code

### Giới thiệu Framework Angular

Angular là một framework phát triển ứng dụng web mạnh mẽ và hiệu quả. Được phát triển bởi Google, Angular giúp lập trình viên xây dựng các ứng dụng web đa nền tảng, đáp ứng và dễ bảo trì.

Với Angular, bạn có thể xây dựng các ứng dụng web phức tạp, bao gồm ứng dụng di động. Framework này sử dụng ngôn ngữ TypeScript, một phiên bản mở rộng của JavaScript, để tạo ra mã nguồn dễ đọc, rõ ràng và dễ bảo trì.

Angular cung cấp các tính năng mạnh mẽ như routing, forms handling, HTTP client và dependency injection, giúp bạn xây dựng các ứng dụng web đáng tin cậy và linh hoạt. Nó cũng tích hợp sẵn các công cụ để kiểm thử và tối ưu hóa hiệu suất, giúp đảm bảo chất lượng và tăng cường hiệu suất ứng dụng của mình.

A red and white logo

Description automatically generated

Hình 1.5: Giới thiệu Angular

**Các tính năng nổi bật trong angular**

Two-way data binding: Đây là 1 trong những tính năng tuyệt với nhất trong angular. Dữ liệu được binding một cách tự động và nhanh chóng, những thay đổi trong view sẽ được tự động cập nhật vào trong các component class.

Powerful routing support Angular hỗ trợ mạnh mẽ các routing thông qua cách tải trang không đồng bộ trên cùng 1 trang cho phép chúng ta tạo ra 1 single page application.

Expressive HTML Angular cho phép chúng ta dùng các cấu trúc lập trình như câu lệnh if, vòng lặp for, .. để render và kiểm soát các trang HTML.

Modular by design Angular được thiết kế theo hướng modul hóa để tổ chức và quản lý code 1 cách tốt hơn.

Built in back end support Angular được xây dựng để hỗ trợ việc giao tiếp với back-end servers và thực thi bất kỳ business logic hoặc lấy dữ liệu.

Active community Angular được hỗ trợ bởi google và có 1 cộng đồng đông đảo sẵn sàng hỗ trợ và giải đáp bất cứ câu hỏi nào của bạn.

**Components**

Components là một khối code trong app Angular. Nó là sự kết hợp của bộ template html và nhúng kèm code TypeScript (hoặc Javascript). Các components là độc lập với nhau và độc lập với hệ thống. Nó có thể được cài vào hoặc tháo ra khỏi hệ thống dễ dàng. Một component có thể hiểu như một control trên màn hình hiển thị, gồm giao diện html và code logic xử lý sự kiện đi kèm control đó. Một component cũng có thể lớn như là cả 1 màn hình chứa nhiều control hoặc một nhóm nhiều màn hình. Tức là là một component cũng có thể chứa và gọi được nhiều component khác nối vào. Như vậy Angular rất linh hoạt trong việc chia nhỏ code ra các component.

Trong Angular chúng ta khai báo một Component với cấu trúc như sau:

import { Component } from '@angular/core';

@Component({

selector: 'hello-ng-world',

template: '<h1>Hello Angular world</h1>'

templateUrl: './hello-ng-world.component.html',

styleUrls: '.hello-ng-world.component.css'

})

export class HelloWorld {

}

Chúng ta thấy từ khóa @Component sẽ giúp định nghĩa ra một bộ khung html cho nó. Và bên dưới là một class HelloWorld dùng để viết code logic. Trong định nghĩa bộ khung html, chúng ta có một số thuộc tính cần chú ý sau đây:

* **selector**: Là tên được đặt để gọi một component trong code html. Từ khóa hello-ng-world được đặt tên cho component này. Khi cần gọi component này ra ở màn hình html cha, ta sẽ gọi bằng html tag <hello-ng-world></hello-ng-world>.
* **template**: Là tự định nghĩa khung html cho component dạng string ở trong file này luôn. Ví dụ ở trên chỉ định nghĩa một thẻ html h1 đơn giản. Cách này chỉ dùng cho component đơn giản.
* **templateUrl**: Là đường dẫn url tới file html bên ngoài để load file đó vào làm khung html cho component này. Đây là cách code hay được dùng vì cho phép tách riêng khung html ra khỏi code logic, người làm design sẽ sửa file html riêng, độc lập với người làm code.
* **styles**: Là viết style css luôn vào file component này. Cách này chỉ dùng cho component đơn giản.
* **styleUrls**: Là đường dẫn url đến file style css độc lập cho component này. Cách này khuyên dùng vì file css nên để dành riêng cho người designer đụng vào.

**Binding dữ liệu**

Để bind một chuỗi ra ngoài màn hình html thì rất đơn giản sử dụng 2 dấu ngoặc nhọn {{TenBien}}. Ví dụ chúng ta có một Component đơn giản như sau:

import { Component } from '@angular/core';

@Component({

selector: 'hello-ng-world',

template: `<h1>Hello {{title}} world</h1>`

})

export class HelloWorld {

title = 'Angular 4';

}

Đó là hiển thị chuỗi string. Nhưng nếu muốn hiển thị giá trị đã có ra một ô input text, thì phải viết 2 dấu ngoặc vuông [ ] như sau:

<input type="text" [value]="title">

Chú ý, trong ví dụ này thì value là một từ khóa html chứ không phải tên biến, title mới là tên biến dùng để truyền vào cho [value] nhưng chúng ta không đóng khung title mà đóng khung value. Điều này cho phép chúng ta linh hoạt để gán giá trị cho bất cứ thuộc tính html nào cũng được.

<input type="text" [placeholder]="title" >

**Binding 2 chiều**

Viết theo cú pháp bên dưới đây sẽ giúp bạn bind được 2 chiều, khi gõ vào ô input thì giá trị của h1 sẽ thay đổi tương ứng:

<h1> {{title}} </h1>

<input type="text" [(ngModel)]="title" name="title">

Từ khóa ngModel lúc này không phải là thuộc tính html nữa mà nó là từ khóa của Angular. Khi viết [(ngModel)] chúng ta sẽ gắn chặt giá trị của input html với biến title. Dẫn đến người dùng gõ vào ô này thì biến title thay đổi theo ô h1 hiện giá trị tương ứng của title.

**Services**

Một service là một đoạn code trong ứng dụng Angular mà phục vụ cho tác vụ gì đó, xử lý code logic gì đó. Ví dụ handle (xử lý) data gửi nhận từ/đến một API, hoặc cung cấp hàm authenticate...

Tất nhiên là service thì chỉ có code không hề có giao diện.

Để tạo ra một service thì chúng ta cần import và mô tả một class với từ khóa @injectable lấy từ @angular/core module.

Một ví dụ:

import { Injectable } from '@angular/core';

@Injectable()

export class TimeService {

constructor() { }

getTime(){

return `${new Date().getHours()} : ${new Date().getMinutes()} : ${new Date().getSeconds()}`;

}

}

Sau khi định nghĩa class TimeService là @Injectable() thì chúng ta sẽ gọi được service này ở nhiều chỗ component khác trong ứng dụng.

**Dependency Injection**

Khi một class muốn được gọi (được tiêm vào, inject vào) một component cần gọi hàm bên trong nó, chúng ta cần dùng đến Dependency Injection. Và rất đơn giản chỉ cần gọi ở hàm khởi tạo (constructor) của component là sẽ tiêm được service vào để dùng nó bên trong component đó.

import { Component } from '@angular/core';

import { TimeService } from './time.service';

@Component({

selector: 'app-root',

template: `<div>Date: {{timeValue}}</div>,

})

export class SampleComponent {

timeValue: string;

constructor(private time: TimeService){

this.timeValue = this.time.getTime(); //Gọi biến time từ

}

}

**Directives**

Directives có thể hiểu như là các đoạn mã typescript kèm theo cả html và khi gọi thì gọi như là html luôn, ví dụ:

<div \*ngIf="title"> <!-- Chỗ này là gọi directive ngIf để kiểm tra điều kiện if ngay ở html -->

Time: {{title}}

</div>

Directives được chia làm các loại sau đây:

* **Components**: Không có nghi ngờ gì khi gọi component là directive cũng được, vì rõ ràng là component cho phép định nghĩa selector và gọi ra như một thẻ html tag (<component-name></component-name>).
* **Structural directives**: Là directive cấu trúc, dùng để vẽ html, hiển thị data lên giao diện html. Ví dụ **ngFor**, **ngIf.**
* Attribute directives: Thêm các thuộc tính động cho element html, ví dụ **ngStyle.**

**Ng-if…else**

Đây là một **Structural directives**, có tác dụng kiểm tra điều kiện ngay ở html. Ví dụ:

<div \*ngIf="title; else noTitle">

Time: {{title}}

</div>

<ng-template #noTitle> Click on the button to see time. </ng-template>

Code ở trên, khi biến title có giá trị, thì chuỗi Time: [value] được show ra. Và cục #noTitle template bị ẩn đi, ngược lại thì điều kiện else được chạy và #noTitle được hiện ra.

Như ta thấy dùng cái directive ngIf else này rất tiện lợi khi có thể ẩn hiện html dễ dàng.

**Ng-template**

Đây cũng là một Structural directives. Nó giúp gom cục html cần ẩn hiện.

<div \*ngIf="isTrue; then tmplWhenTrue else tmplWhenFalse"></div>

<ng-template #tmplWhenTrue>show-up when isTrue is true. </ng-template>

<ng-template #tmplWhenFalse> show-up when isTrue is false </ng-template>

**Tạo một directive**

Việc tạo 1 custom directives là cực kỳ đơn giản trong angular.

@Directive({

selector: '[MyDirective]'

})

class MyDirective {

}

**Sử dụng observables ở ngay html**

Chúng ta biết là Observables là luồng dữ liệu được gửi nhận từ api. Observables sẽ chứa data json từ server gửi tới nhưng nó không phải là data json thuần. Nếu đưa luồng data này cho html thì nó sẽ không hiểu được để render ra màn hình. Tuy nhiên với từ khóa pipe async chúng ta có thể in ra trực tiếp như sau:

<div \*ngIf="asyncData | async; else loading; let title">

Title: {{title}}

</div>

<ng-template #loading> Loading... </ng-template>

Lúc này biến Observables asyncData được gọi kèm từ khóa async, data bên trong nó sẽ được in ra html.

Nếu chúng ta có một mảng ở dạng Observables thì cũng hoàn toàn có thể in ra trực tiếp bằng **async** sử dụng ngFor.

<!-- Giả sử mảng colors là Observables và colors = ["red", "blue", "green", "yellow", "violet"]; -->

<ul \*ngFor="let color of colors | async">

<li>{{color}}</li>

</ul>

**Router**

Khi sử dụng Router, chúng ta sẽ có thể dựng nên một trang web SPA (Single Page Application). Khi viết một route template, các components sẽ được map tương ứng với các URL cố định. Và khi click vào link, url sẽ load component mà không cần load lại cả trang. Dẫn đến hiệu ứng là trang web chạy trong 1 tab và các màn hình được load động. Cách hoạt động này khác với MPA (Multiple Page Application) thường thì mỗi màn hình sẽ có url tương ứng và phải bật một tab riêng trên trình duyệt.

Router cũng có cache, tức là component đã load một lần thì sẽ được load lại không cần tải toàn bộ về từ server nữa.

**Router Outlet**

Mỗi một Router sẽ có một URL để load component. Và để biết được là component sẽ render ra chỗ nào thì chúng ta viết đoạn code sau vào khung html cần chèn:

<router-outlet></router-outlet>

**Cài đặt Route cho ứng dụng Angular**

Để cài đặt toàn bộ Router cho một ứng dụng Angular thì chúng ta cần tạo ra một đối tượng JSON chứa các thuộc tính như sau:

* **path**: Đường dẫn URL của component hiện tại.
* **component**: Ứng với đường dẫn bên trên thì load component nào.
* **redirectTo**: Chuyển hướng đến URL này nếu URL ở path không trùng. Ví dụ, khi người dùng gõ URL linh tinh, chúng ta muốn chuyển hướng và load trang Home hoặc trang báo lỗi 404 thì cần ghi rõ URL trang Home hoặc 404 vào redirectTo.
* **pathMatch**: Cài đặt xem chế độ kiểm tra url là như thế nào. khi giá trị là full thì nghĩa là toàn bộ url từ đầu đến cuối sẽ phải chính xác như trong bộ JSON Router. Còn khi giá trị là prefix thì chuỗi đầu tiên của url (dấu sược đầu tiên) sẽ được kiểm tra. Mặc định nếu không nói gì thì prefix sẽ được chọn.

Hãy xem đoạn code ví dụ về Router bên dưới:

const routes: Routes = [

{

path: 'home',

component: Sample1Component,

},

{

path: 'second2',

component: Sample2Component

},

{

path: '',

redirectTo: '/home',

pathMatch: 'full'

}

];

**Params**

Data hoàn toàn có thể được truyền từ màn hình này sang màn hình kia bằng cách cho giá trị vào trong url để gọi (hay còn gọi là cách gọi hàm bằng truyền params URL).

Các bước để làm được việc này như sau:

* **Định nghĩa giá trị cần lấy trong định nghĩa route**: Ví dụ bên dưới sẽ cho ta thấy cách định nghĩa param cần lấy trong url

{

path: details/:id, //từ khóa :id ở đây thể hiện là có một biến id sẽ được đón nhận ở Component Details

component: DetailsComponent

}

* **Đọc data từ URL truyền tới**: Chúng ta cần dùng đến biến **ActivedRoute** của bộ **@angular/router**

// inject activatedRoute

constructor(private activeRoute: ActivatedRoute) { }

// Đọc ra giá trị từ url hiện tại, Route lúc này sẽ trả về một luồng dữ liệu observable nên chúng ta phải subcribe để lấy ra được data bên trong nó.

this.activeRoute

.params

.subscribe((data) => console.log(data[id]));

### Giới thiệu Spring boot

Spring Boot là một extension của Spring Framework giúp các lập trình viên loại bỏ các bước cấu hình phức tạp mà Spring bắt buộc. Spring Boot là dự án phát triển bởi ngôn ngữ Java (JAV) trong hệ sinh thái Spring framework.

Spring Boot được dùng để tạo các ứng dụng độc lập dựa trên Spring, Spring Boot không yêu cầu cấu hình XML.

Spring Boot có các ưu điểm như có các tính năng của Spring Framework, tạo ứng dụng độc lập, có thể chạy bằng java -jar (cho cả java web), cấu hình ít, tự động cậu hình bất kì khi nào có thể (Giảm thời gian viết code, tăng năng suất), không yêu cầu XML config….



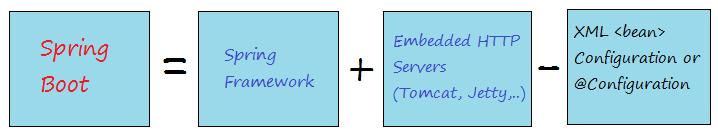
Hình 1.6: Giới thiệu Spring boot

Spring Boot là một Project nằm trên tầng IO Execution (Tầng thực thi) của Spring IO Framework.

Spring Boot là một bước tiếp theo của Spring, để làm cho Spring dễ dàng hơn trong việc thiết lập và phát triển ứng dụng. Với Spring Boot các cấu hình Spring được giảm thiểu tối đa. Spring Boot hỗ trợ các bộ chứa nhúng (embedded containers) điều này cho phép các ứng dụng web có thể chạy độc lập mà không cần phải trên khai lên các Web Server.

**Ưu điểm:**

* Dễ dàng cho việc phát triển ứng dụng dựa trên Spring với Java hoặc Groovy.
* Giảm thiểu thời gian phát triển, tăng năng suất phát triển.
* Tránh việc phải viết nhiều bản mẫu code, cấu hình Annotaion hoặc XML…
* Dễ dàng trong việc tích hợp với hệ sinh thái của Spring như: Spring JDBC, Spring ORM, Spring Data, Spring Security...
* Cung cấp các Embedded HTTP servers như Tomcat, Jetty... để phát triển và test một cách dễ dàng.
* Cung cấp công cụ CLI (command Line Interface) cho việc phát triển và test ứng dụng nhanh chóng và dễ dàng từ command line.
* Cung cấp rất nhiều các plugins để phát triển và test các ứng dụng Spring Boot nhanh chóng sử dụng các công cụ Build như Maven và Gradle.
* Cung cấp nhiều plugins để làm việc với các embedded and in-memory databases.



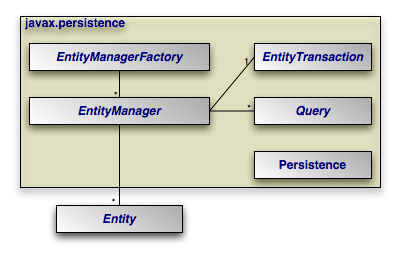
Hình 1.7: Giải thích đơn giản về Spring Boot

**-**  **Spring Data JPA**

Java Persistence API (JPA) là 1 giao diện lập trình ứng dụng Java, nó là một tập hợp các lớp và các phương thức để lưu trữ liên tục một lượng lớn dữ liệu vào cơ sở dữ liệu.

JPA cung cấp một mô hình POJO persistence cho phép ánh xạ các table/các mối quan hệ giữa các table trong database sang các class/mối quan hệ giữa các object.

Ví dụ: table Users với các column (Id, name, age…) sẽ tương ứng với class Users.java với các field Id, name, age… từ đó mỗi khi truy vấn table hay các column ta sẽ truy vấn trực tiếp trên các class, các field của class mà không cần quan tâm tới việc đang dùng loại database nào, dữ liệu database ra sao.



Hình 1.8: Kiến trúc của JPA

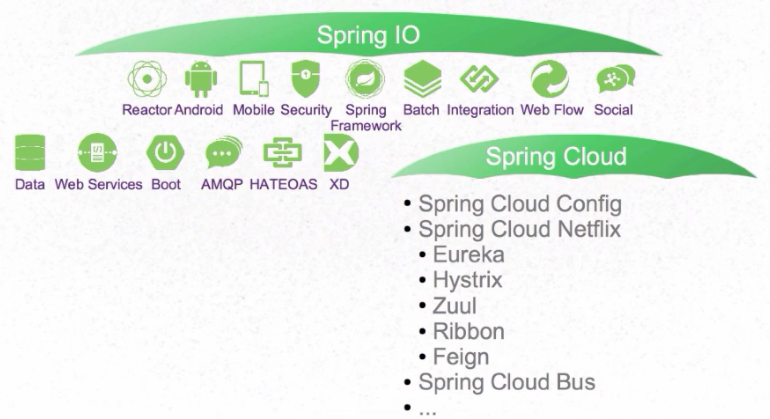
**Một số khái niệm trong JPA:**

* Entity: Entity là các đối tượng thể hiện tương ứng 1 table trong cơ sở dữ liệu. Khi lập trình, entity thường là các class POJO đơn giản, chỉ gồm các method getter, setter.
* EntityManager: EntityManager là một giao diện (interface) cung cấp các API cho việc tương tác với các Entity như Persist (lưu một đối tượng mới), merge (cập nhật một đối tượng), remove (xóa 1 đối tượng).
* EntityManagerFactory: EntityManagerFactory được dùng để tạo ra một thể hiện của EntityManager.
* Persistence: lưu trữ bản sao của đối tượng cơ sở dữ liệu vào bộ nhớ tạm thời.

**Spring Data JPA** là một phần của Spring Data Family, nó không phải là một JPA Provider mà là một đặc tả, nó là một thư viện/framework bổ sung thêm một lớp trừu tượng trên đỉnh của JPA Provider. Spring Data JPA chỉ làm việc với các cơ sở dữ liệu quan hệ. Spring Data JPA hỗ trợ Java Persistent Query Language**.**

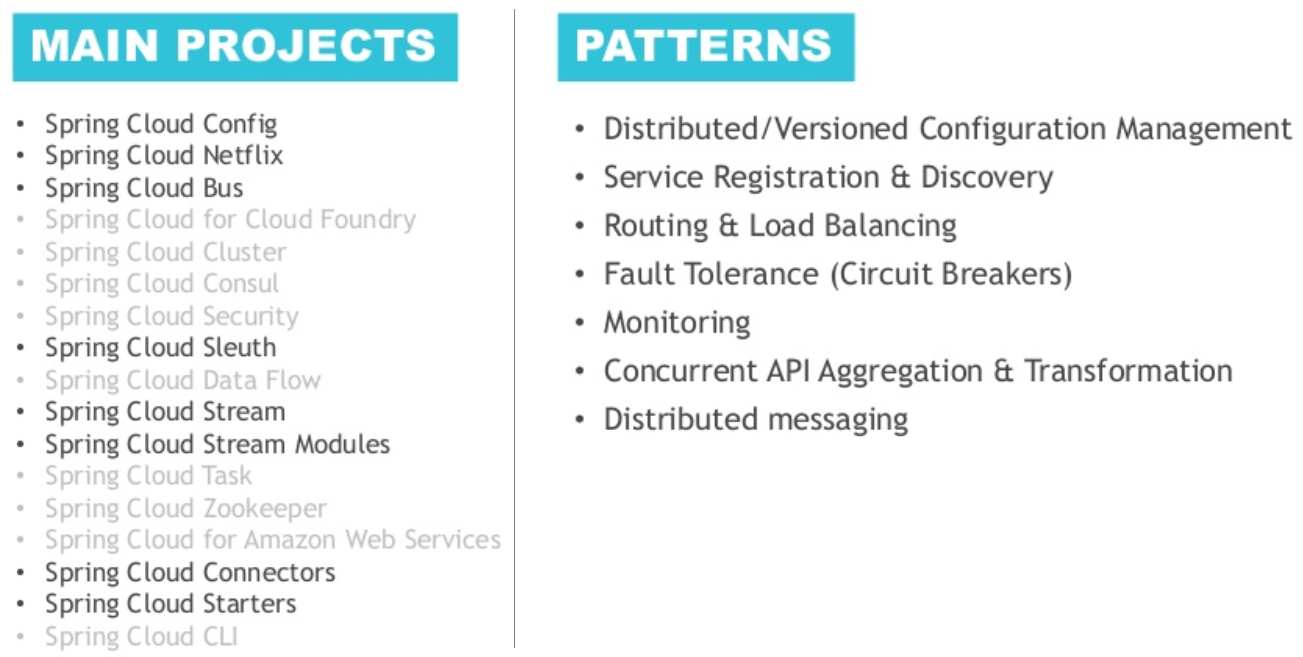
**Spring Cloud**

Spring Cloud chính là một dự án con (sub-project) nằm trong cái ô Spring IO (Spring IO Umbrella), và tự nó cũng là một cái ô, một cái ô con (Sub-Umbrella).



Hình 1.9: Spring Integration Objects

Dưới đây là danh sách các sub-project và các khuôn mẫu (pattern) trong Spring Cloud:



Hình 1.10: Pattern Spring Cloud

Spring Cloud là một công nghệ phần mềm sử dụng để phát triển các ứng dụng phân tán. Một ứng dụng được gọi là phân tán (Distributed application) khi các phần của nó có thể được phát triển trên các ngôn ngữ khác nhau, và được triển khai trên các máy chủ khác nhau. Vì vậy mục tiêu của Spring Cloud là làm sao để các thành phần của ứng dụng có thể giao tiếp với nhau.

**-** **Spring Security**

* Giới thiệu

Spring Security là một dự án nổi bật trong hệ sinh thái Spring. Spring Security cung cấp các dịch vụ bảo mật toàn diện cho các ứng dụng doanh nghiệp có nền tảng Java EE.

Spring Security cung cấp 2 cơ chế cơ bản:

* Authentication (xác thực): là tiến trình thiết lập một principal. Principal có thể hiểu là một người, hoặc một thiết bị, hoặc một hệ thống nào đó có thể thực hiện một hành động trong ứng dụng của bạn.
* Authorization (phân quyền) hay Access-control: là tiến trình quyết định xem một principal có được phép thực hiện một hành động trong ứng dụng của bạn hay không. Trước khi diễn tiến tới Authorization, principal cần phải được thiết lập bởi Authentication.

Có thể thấy đây là 2 cơ chế khá phổ biến trong các dịch vụ bảo mật, không chỉ riêng Spring Security.

**-** **Lợi ích của Spring Security**

Lợi ích lớn nhất của Spring Security là tích hợp tính năng xác thực và phân quyền một cách dễ dàng vào ứng dụng.

Ngoài ra Spring Security còn giúp:

* Chống lại CSRF attack.
* Bảo vệ Session Fixation.
* Mã hóa mật khẩu.
* Cache control.
* XSS-Protection.

### Giới thiệu Node.js

Node.js là một framework phát triển ứng dụng web mạnh mẽ dựa trên ngôn ngữ JavaScript. Được xây dựng trên nền tảng Chrome V8 JavaScript runtime, Node.js cho phép lập trình viên xây dựng các ứng dụng web hiệu quả, đáp ứng và có khả năng xử lý đồng thời nhiều yêu cầu.

Một trong những đặc điểm nổi bật của Node.js là kiến trúc không đồng bộ (asynchronous architecture). Framework này sử dụng các hàm không đồng bộ và sự kiện để xử lý nhiều yêu cầu mà không cần chờ đợi, giúp tăng cường hiệu suất ứng dụng và khả năng mở rộng.

Node.js cung cấp một số thư viện mạnh mẽ như Express.js để xây dựng các ứng dụng web, Socket.IO để tạo kết nối thời gian thực.

A logo of a software company

Description automatically generated

Hình 1.11: Giới thiệu Node JS

### JWT (JSON Web Token)

JSON Web Mã (JWT) là một chuẩn mở (RFC 7519) định nghĩa một cách nhỏ gọn và khép kín để truyền một cách an toàn thông tin giữa các bên dưới dạng đối tượng JSON. Thông tin này có thể được xác minh và đáng tin cậy vì nó có chứa chữ ký số. JWTs có thể được ký bằng một thuật toán bí mật (với thuật toán HMAC) hoặc một public / private key sử dụng mã hoá RSA.

Authentication: khi user logged in, mỗi request tiếp đó đều kèm theo chuỗi token JWT, cho phép người dùng có thể truy cập đường dẫn, dịch vụ và tài nguyên được phép ứng với token đó.

Trong việc xác thực, khi user đăng nhập thành công (browser sẽ post username và mật khẩu về server), server sẽ trả về một chuỗi JWT về browser, và token JWT này được lưu lại trong browser của người dùng (thường trong local-storage đối với access-token hay cookies đối với refresh-token), thay vì cách truyền thống là tạo một session trên server và trả về cookie.

Bất cứ khi nào mà sser muốn truy cập vào route được bảo vệ (mà chỉ có sser đã đăng nhập mới được phép), browser sẽ gửi token JWT này trong Header Authorization, Bearer schema của request gửi đi.

Authorization: Bearer <token>

Đây là cách mà stateless (phi trạng thái) authentication làm việc, trạng thái của user không được lưu trong bộ nhớ của server mà được đóng gói hẳn vào trong JWT. Server sẽ kiểm tra token JWT này có hợp lệ hay không (Bởi vì JWT có tính chất self-contained, mọi thông tin cần thiết để kiểm tra JWT đều đã được chứa trong token JWT).

A screenshot of a computer

Description automatically generated

Hình 1.12: Giới thiệu về JWT

# KHẢO SÁT HỆ THỐNG

## Khảo sát hiện trạng

Sau khi thu thập kết quả từ khảo sát từ thực thế, ứng dụng đặt thuê phòng trọ sẽ được tổ chức như sau:

***Thông tin về Chủ trọ:***

* Mỗi chủ trọ có thể có nhiều dãy, khu trọ.
* Chủ trọ cần đăng ký với quản trị web để được cấp tài khoản và mật khẩu đăng nhập.
* Mỗi khu trọ có thể có nhiều phòng trọ.
* Quản lý thiết bị theo khu trọ và theo phòng trọ.
* Hoá đơn phòng hàng tháng sẽ được xuất tự động hoặc theo thời gian chủ trọ đã cài đặt.
* Quản lý hoá đơn và có thể in hoá đơn nếu cần thiết.
* Mỗi phòng trọ có thể có giá, số lượng người ở khác nhau.
* Chủ trọ sẽ nhận thông báo cũng như thông tin về người đặt phòng trên hệ thống và qua email
* Chỉ số điện nước được lưu trữ trên hệ thống làm cơ sở để tính hoá đơn hàng tháng
* Hợp đồng mỗi phòng trọ sẽ được tạo và quản lý trên hệ thống, có thể in hợp đồng nếu cần thiết
* Chủ trọ có thể đăng tin cho thuê trọ lên bảng tin trang web.

***Thông tin về người muốn tìm trọ:***

* Khách muốn thuê trọ tìm kiếm, lọc phòng trọ theo mức giá, diện tích, vị trí.
* Sau khi chọn được phòng trọ phù hợp có thể đặt phòng, để lại thông tin và chờ chủ trọ liên hệ.

***Thông tin quản trị viên web:***

* Quản trị viên web sẽ cấp tài khoản mật khẩu cho các chủ trọ có nhu cầu muốn đăng ký.
* Quản trị viên web quản lý thông tin về các chủ trọ.

## Giải pháp đề xuất.

Chủ trọ:

1. Đăng nhập vào ứng dụng
2. Từ trang dashboard, chủ trọ có thể theo dõi tổng quan số lượng phòng, số lượng phòng trống, thông tin đặt phòng mới theo ngày
3. Từ trang quản lý khu trọ, chủ trọ có thể xem, chỉnh sửa thông tin về khu trọ
4. Từ trang quản lý phòng trọ, chủ trọ có thể xem, chỉnh sửa thông tin về phòng trọ như diện tích, giá, số người ở tối đa theo từng khu trọ.
5. Từ trang quản lý dịch vụ, chủ trọ có thể xem, thêm, chỉnh sửa các dịch vụ theo từng khu trọ.
6. Từ trang quản lý thiết bị, chủ trọ có thể xem, thêm, chỉnh sửa thiết bị, thêm thiết bị vào phòng theo từng khu trọ
7. Từ trang quản lý hợp đồng, chủ trọ có thể xem, chỉnh sửa, tạo hợp đồng theo từng khu trọ, in hợp đồng nếu cần thiết
8. Từ trang quản lý đặt cọc phòng, chủ trọ có thể xem, chỉnh sửa, thêm đặt cọc phòng nhằm mục địch quản lý tiền đặt cọc giữ phòng theo khu
9. Từ trang quản lý điện nước, chủ trọ có thể xem, chỉnh sửa, nhập chỉ số điện nước cho từng phòng trọ theo tháng và theo khu trọ
10. Từ trang quản lý hoá đơn, chủ trọ có thể chủ động xuất hoá đơn hàng tháng cho các phòng, có thể in hoá đơn nếu cần thiết
11. Từ trang trả phòng, chủ trọ có thể tạo hoá đơn trả phòng, quản lý trả phòng theo từng khu trọ
12. Từ trang quản lý bài đăng, chủ trọ có thể quản lý các bài đăng cho thuê, cho phép chỉnh sửa, đăng bài, gỡ bài
13. Từ trang thông báo đặt phòng, chủ trọ có thể theo dõi các đặt phòng từ hệ thống tìm trọ, xem thông tin khách hàng cũng như thời gian xem phòng nếu có

### Yêu cầu phi chức năng:

Hệ thống đáng tin cậy, chính xác, giao diện thân thiện, dễ sử dụng, truy cập dữ liệu nhanh chóng, đảm bảo tính bảo mật, hệ thống hoạt động ổn định với tính linh hoạt cao.

Hệ thống bảo mật thông tin của người sử dụng, cam kết những thông tin được lưu trữ an toàn, không rao bán thông tin nhằm trục lợi.

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Diagram database theo kiến trúc monolithic (MySQL)

A computer screen shot of a computer flow chart

Description automatically generated

Hình 3.1: Tổng quan Diagram database

## Usecase

### Use case tổng quát

A diagram of a diagram

Description automatically generated

Hình 3.2: Usecase tổng quát

## Thiết kế dữ liệu

### Từ điển dữ liệu

* Bảng: USERS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| active | TINYINT(1) | Not null |
| created\_at | DATETIME(6) | Not null |
| address | VARCHAR(255) | Not null |
| email | VARCHAR(255) | Not null |
| firstname | VARCHAR(255) | Not null |
| lastname | VARCHAR(255) | Not null |
| image | VARCHAR(255) | Not null |
| password | VARCHAR(255) | Not null |
| phone | VARCHAR(255) | Not null |
| username | VARCHAR(255) | Not null |
| role\_id | INT | Khoá ngoại |
| user\_preference\_id | INT | Khoá ngoại |

Bảng vẽ 3.1: Bảng từ điển Users

* Bảng USER\_PREFERENCE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| email\_date | DATETIME(6) | Not null |
| issue\_invoice\_date | DATETIME(6) | Not null |

Bảng vẽ 3.3: Bảng từ điển User Preference

* Bảng: USER\_ROLE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| role\_id | INT | Khoá chính |
| users\_id | INT | Khoá chính |

Bảng vẽ 3.4: Bảng từ điển User Role

* Bảng: ROLE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| name | ENUM | Not null |

Bảng vẽ 3.5: Bảng từ điển Role

* Bảng: REFRESH\_TOKEN

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| user\_id | INT | Khoá ngoại |
| token | VARCHAR(255) | Not null |

Bảng vẽ 3.6: Bảng từ điển Refresh Token

* Bảng: ACCOMODATIONS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| active | TINYINT(1) | Not null |
| address\_id | INT | Khoá ngoại |
| user\_id | INT | Khoá ngoại |
| created\_at | DATETIME(6) | Not null |
| name | VARCHAR(255) | Not null |

Bảng vẽ 3.7: Bảng từ điển Accomodations

* Bảng: ADDRESS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| adress\_line | VARCHAR(255) | Not null |
| ward\_id | INT | Khoá ngoại |

Bảng vẽ 3.8: Bảng từ điển Address

* Bảng: WARD

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| ward | VARCHAR(255) | Not null |
| ward\_code | INT | Not null |
| district\_id | INT | Khoá ngoại |

Bảng vẽ 3.9:Bảng từ điển Ward

* Bảng: DISTRICT

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| district | VARCHAR(255) | Not null |
| district\_code | INT | Not null |
| province\_id | INT | Khoá ngoại |

Bảng vẽ 3.10: Bảng từ điển District

* Bảng: PROVINCE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| province | VARCHAR(255) | Not null |
| province\_code | INT | Not null |

Bảng vẽ 3.11: Bảng từ điển Province

* Bảng: ACCOMODATION\_SERVICES

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| accomodation\_id | INT | Khoá ngoại |
| id | INT | Khoá chính |
| is\_default | INT | Not null |
| price | DOUBLE | Not null |
| description | VARCHAR(255) |  |
| name | VARCHAR(255) | Not null |
| unit | VARCHAR(255) | Not null |

Bảng vẽ 3.12: Bảng từ điển Accomodation Services

* Bảng: ROOMS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| is\_rent | TINYINT(1) | Not null |
| max\_capacity | INT | Not null |
| price | DOUBLE | Not null |
| acreage | DOUBLE | Not null |
| accomodation\_id | INT | Khoá ngoại |
| name | VARCHAR(255) | Not null |

Bảng vẽ 3.13: Bảng từ điển Rooms

* Bảng: IMAGES

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| created\_at | DATETIME(6) | Not null |
| image\_name | VARCHAR(255) | Not null |
| image\_url | VARCHAR(255) | Not null |
| post\_id | INT | Khoá ngoại |

Bảng vẽ 3.14: Bảng từ điển Images

* Bảng: BILLS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| paid\_money | DOUBLE | Not null |
| debt | DOUBLE | Not null |
| quantity\_sent | INT | Not null |
| total\_payment | DOUBLE | Not null |
| total\_service | DOUBLE | Not null |
| total\_price | DOUBLE | Not null |
| is\_pay | TINYINT(1) | Not null |
| payment\_date | DATETIME(6) |  |
| room\_id | INT | Khoá ngoại |
| discount | DOUBLE | Not null |
| date | DATETIME(6) | Not null |
| created\_at | DATETIME(6) | Not null |
| is\_sent | TINYINT(1) | Not null |
| note | VARCHAR(255) |  |
| id | INT | Khoá chính |

Bảng vẽ 3.15: Bảng từ điển Bills

* Bảng: TENANTS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| identify\_num | INT | Not null |
| is\_stayed | TINYINT(1) | Not null |
| contract\_id | INT | Khoá ngoại |
| accomodation\_id | INT | Khoá ngoại |
| start\_date | DATETIME(6) | Not null |
| end\_date | DATETIME(6) | Not null |
| gender | ENUM(‘MALE’, ‘FEMAIL’, ‘UNKNOWN’) | Not null |
| first\_name | VARCHAR(255) | Not null |
| last\_name | VARCHAR(255) | Not null |
| email | VARCHAR(255) | Not null |
| phone | VARCHAR(255) | Not null |

Bảng vẽ 3.16: Bảng từ điển Tenants

* Bảng: CONTRACT

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| deposits | DOUBLE | Not null |
| duration | INT | Not null |
| first\_electric\_num | INT | Not null |
| first\_water\_num | INT | Not null |
| is\_active | TINYINT(1) | Not null |
| day\_stayed\_before | DATETIME(6) | Not null |
| first\_come\_payment | DOUBLE | Not null |
| keep\_room\_deposit | DOUBLE | Not null |
| room\_id | INT | Khoá ngoại |
| end\_date | DATETIME(6) | Not null |
| start\_date | DATETIME(6) | Not null |
| representative | INT | Not null |

Bảng vẽ 3.17: Bảng từ điển Contract

* Bảng: CONTRACT\_SERVICE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| contract\_id | INT | Khoá ngoại |
| quantity | INT | Not null |
| service\_id | INT | Khoá ngoại |

Bảng vẽ 3.18: Bảng từ điển Contract Service

* Bảng: ELECTRIC\_WATER\_NUM

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| first\_electric\_num | INT | Not null |
| last\_electric\_num | INT | Not null |
| electric\_num | INT | Not null |
| first\_water\_num | INT | Not null |
| last\_water\_num | INT | Not null |
| water\_num | INT | Not null |
| month | DATETIME(6) | Not null |
| room\_id | INT | Khoá ngoại |

Bảng vẽ 3.19: Bảng từ điển Electric Water Num

* Bảng: DEPOSIT

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| deposit | DOUBLE | Not null |
| is\_active | TINYINT(1) | Not null |
| is\_repaid | TINYINT(1) | Not null |
| room\_id | INT | Khoá ngoại |
| tenant\_id | INT | Khoá ngoại |
| created\_at | DATETIME(6) | Not null |
| due\_date | DATETIME(6) | Not null |
| start\_date | DATETIME(6) | Not null |
| note | VARCHAR(255) |  |

Bảng vẽ 3.20: Bảng từ điển Deposit

* Bảng: EQUIPMENTS

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| price | DOUBLE | Not null |
| room\_id | INT | Khoá ngoại |
| accomodation\_id | INT | Khoá ngoại |
| description | VARCHAR(255) |  |
| name | VARCHAR(255) | Not null |
| status | VARCHAR(255) | Not null |
| unit | VARCHAR(255) | Not null |

Bảng vẽ 3.21: Bảng từ điển Equipment

* Bảng: BOOKING

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| customer\_email | VARCHAR(255) | Not null |
| customer\_name | VARCHAR(255) | Not null |
| customer\_phone | VARCHAR(255) | Not null |
| room\_id | INT | Khoá ngoại |

Bảng vẽ 3.22: Bảng từ điển Booking

* Bảng: BANK\_ACCOUNT\_INFO

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| account\_owner | VARCHAR(255) | Not null |
| bank\_name | VARCHAR(255) | Not null |
| bank\_number | VARCHAR(255) | Not null |
| user\_id | INT | Khoá ngoại |

Bảng vẽ 3.23: Bảng từ điển Bank Account Info

* Bảng: POST

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| content | VARCHAR(255) | Not null |
| created\_at | DATETIME(6) | Not null |
| is\_active | TINYINT(1) | Not null |
| title | VARCHAR(255) | Not null |
| user\_id | INT | Khoá ngoại |
| room\_id | INT | Khoá ngoại |
| last\_change | DATETIME(6) | Not null |

Bảng vẽ 3.24: Bảng từ điển Post

* Bảng: POST\_UTILITIES

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| post\_id | INT | Khoá chính |
| service\_id | INT | Khoá ngoại |

Bảng vẽ 3.25: Bảng từ điển Post Utilities

* Bảng: INVOICE\_TYPE

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| type | VARCHAR(255) | Not null |

Bảng vẽ 3.26: Bảng từ điển Invoice Type

* Bảng: SERVICE\_BILL

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | INT | Khoá chính |
| price | DOUBLE | Not null |
| quantity | INT | Not null |
| name | VARCHAR(255) | Not null |
| bill\_id | INT | Khoá ngoại |

Bảng vẽ 3.27: Bảng từ điển Service Bill

## Sơ đồ tuần tự

### Sơ đồ tuần tự đăng nhập

A diagram of a diagram

Description automatically generated

Hình 3.3: Sơ đồ tuần tự đăng nhập

# THIẾT KẾ PHẦN MỀM

## Chức năng chủ trọ

### Quản lý khu trọ

A screenshot of a computer

Description automatically generated

Hình 4.1: Giao diện chức năng quản lý khu trọ

A screenshot of a computer

Description automatically generated

Hình 4.2: Giao diện chức năng thêm khu trọ

A screenshot of a computer

Description automatically generated

Hình 4.3: Giao diện chức năng chỉnh sửa khu trọ

### Quản lý phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.4: Giao diện chức năng quản lý phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.5: Giao diện chức năng thêm phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.6: Giao diện chức năng chỉnh sửa phòng trọ

### Quản lý dịch vụ

A screenshot of a computer

Description automatically generated

Hình 4.7: Giao diện chức năng quản lý dịch vụ

A screenshot of a computer

Description automatically generated

Hình 4.8: Giao diện chức năng thêm dịch vụ

A screenshot of a computer

Description automatically generated

Hình 4.9: Giao diện chức năng chỉnh sửa dịch vụ

### Quản lý thiết bị

A screenshot of a computer

Description automatically generated

Hình 4.10: Giao diện chức năng quản lý thiết bị

A screenshot of a computer

Description automatically generated

Hình 4.11: Giao diện chức năng thêm thông tin thiết bị

A screenshot of a computer

Description automatically generated

Hình 4.12: Giao diện chỉnh sửa thông tin thiết bị

### Quản lý hợp đồng

A screenshot of a computer

Description automatically generated

Hình 4.13: Giao diện chức năng quản lý hợp đồng

A screenshot of a computer

Description automatically generated

Hình 4.14: Giao diện chức năng thêm hợp đồng

A screenshot of a computer

Description automatically generated

Hình 4.15: Giao diện chức năng chỉnh sửa hợp đồng

### Quản lý điện nước

A screenshot of a computer

Description automatically generated

Hình 4.16: Giao diện chức năng quản lý chỉ số điện nước

A screenshot of a computer

Description automatically generated

Hình 4.17: Giao diện chức năng nhập chỉ số điện nước

A screenshot of a computer

Description automatically generated

Hình 4.18: Giao diện chức năng chỉnh sửa thông tin chỉ số điện nước

### Xuất hoá đơn

A screenshot of a computer

Description automatically generated

Hình 4.19: Giao diện xem hoá đơn phòng hàng tháng

A screenshot of a computer

Description automatically generated

Hình 4.20: Giao diện xem thông tin chi tiết trước khi xuất hoá đơn

A screenshot of a computer

Description automatically generated

Hình 4.21: Giao diện chỉnh sửa hoá đơn chưa thanh toán

### Trả phòng

A screenshot of a computer

Description automatically generated

Hình 4.22: Giao diện quản lý hoá đơn trả phòng

A screenshot of a computer

Description automatically generated

Hình 4.23: Giao diện chỉnh sửa hoá đơn trả phòng khi chưa thanh toán

A screenshot of a computer

Description automatically generated

Hình 4.24: Giao diện email hoá đơn

### Đăng bài

A screenshot of a computer

Description automatically generated

Hình 4.25: Giao diện quản lý bài đăng

A screenshot of a computer

Description automatically generated

Hình 4.26: Giao diện tạo bài đăng cho thuê phòng

A screenshot of a computer

Description automatically generated

Hình 4.27: Giao diện chỉnh sửa bài đăng

### Quản lý đặt phòng

A screenshot of a computer

Description automatically generated

Hình 4.28 Giao diện quản lý thông tin đặt phòng

A screenshot of a computer

Description automatically generated

Hình 4.29: Giao diện tạo nhanh hợp đồng, đặt cọc, khách thuê từ đặt phòng

## Chức năng khách tìm trọ

A screenshot of a computer

Description automatically generated

Hình 4.30: Giao diện trang tìm phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.31: Giao diện tìm kiếm phòng trọ theo địa chỉ

A screenshot of a computer

Description automatically generated

Hình 4.32: Giao diện kết quả tìm kiếm theo địa chỉ

A screenshot of a computer

Description automatically generated

Hình 4.33: Giao diện tìm kiếm theo khoảng giá

A screenshot of a computer

Description automatically generated

Hình 4.34: Giao diện tìm kiếm phòng trọ theo diện tích

A screenshot of a computer

Description automatically generated

Hình 4.35: Giao diện xem chi tiết phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.36: Giao diện xem chi tiết phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.37: Giao diện đặt phòng trọ

A screenshot of a computer

Description automatically generated

Hình 4.38: Giao diện chọn ngày xem phòng trọ khi đặt phòng

# KẾT LUẬN VÀ KIẾN NGHỊ

# TÀI LIỆU THAM KHẢO

1. MySQL: <https://www.mysql.com/>
2. Stack Overflow: <https://stackoverflow.com/>
3. W3School: <https://www.w3schools.com/>
4. Angular: <https://angular.io/>
5. Spring boot: <https://spring.io/projects/spring-boot>