TRƯỜNG CAO ĐẲNG CÔNG THƯƠNG TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN

****

**BÁO CÁO KHÓA LUẬN TỐT NGHIỆP**

**ĐỀ TÀI: XÂY DỰNG WEBSITE WEBSITE BÁN ĐIỆN THOẠI, LAPTOP VÀ LINH KIỆN (SỬ DỤNG STRING BOOT API, REACT JS, POSTGRESQL)**

**SVTH*:* ĐỖ VIẾT PHÚ**

**MSSV: 2119110186**

**GVHD*:* NGUYỄN ANH TÚ**

**TP.THỦ ĐỨC, 11 THÁNG 06 NĂM 2022**

# NHẬN XÉT CỦA GIÁO VIÊN

**Tính cấp thiết**: Ứng dụng dựa trên tình hình thực tế của nền tảng thương mại điện tử của Việt Nam. Cùng với sự phát triển của thị trường điện thoại di động là sự phát triển mạnh mẽ, bên cạnh đó các trang thương mại điện tử phát triển nhắm đưa đến người dùng những thông tin chính xác và khả quan của xu hướng công nghệ. Sự ra đời của các trang web thương mại điện tử giúp người sử dụng có thể xem và tìm hiểu các thông tin về sản phẩm, hàng hóa. Xuất phát từ nhận định trên, việc lựa chọn và nghiên cứu vấn đề “Xây dựng website bán lẻ điện thoại di động” là một vấn đề có tính cấp thiết và thực tiễn cao.

**Mục tiêu**: Vấn đề nghiên cứu được vạch định rõ ràng, đạt được kết quả ban đầu đặt ra. Tuy nhiên cần đặt mục tiêu cao hơn, rõ ràng hơn cho website.

**Nội dung**: Làm rõ được cơ sở lý luận liên quan đến vấn đề nghiên cứu của đề tài. Thực trạng của vấn đề được làm rõ. Nội dung bố cục rõ ràng, cụ thể, diễn tả cụ thể với nhiều sơ đồ, hình ảnh với những lập luận chặc chẽ. Cần phát triển cơ sở dữ liệu riêng cho website.

**Kết quả nghiên cứu**: Về cơ bản các nội dung của các chương và mục là phù hợp, làm rõ được vấn đề đặt ra, giao diện đẹp. Phương pháp nghiên cứu phù hợp. Tuy nhiên đề tài nghiên cức chỉ ở mức độ nhóm chức năng phù hợp, đầy đủ với người sử dụng mà chưa đưa ra được một số tính năng như thanh toán trực tuyến.

Ngày 12 tháng 07 năm 2021

***Giáo viên hướng dẫn***



**NGUYỄN ANH TÚ**

# LỜI NÓI ĐẦU

Trong những năm gần đây, sự phát triển mạnh mẽ của thương mại điện tử trên thế giới đã góp phần làm thay đổi cách thức kinh doanh, giao dịch truyền thống và đem lại những lợi ích to lớn cho xã hội. Sự ra đời của thương mại điện tử đã đánh dấu sự bắt đầu của một hệ thống mới nhằm phát triển kinh tế, một trong những điều kiện cơ bản và có ý nghĩa quyết định trong việc phát triển thương mại điện tử là việc hoàn thiện các dịch vụ thanh toán điện tử.

Thực tế đã chứng minh thanh toán điện tử là một trong những điều kiện cốt lõi để phát triển thương mại điện tử với vai trò là một khâu không thể tách rời của quy trình giao dịch và còn là biện pháp xác thực việc ký kết hợp đồng giữa người bán và người mua. Thương mại điện tử giúp giảm thấp chi phí bán hàng và chi phí tiếp thị. Bằng Internet/web, một nhân viên bán hàng có thể giao dịch được với rất nhiều khách hàng, sản phẩm điện tử trên các trang web không những phong phú hơn mà còn thường xuyên được cập nhật so với giới hạn và luôn luôn lỗi thời.

Qua đó thương mại điện tử tạo điều kiện cho việc thiết lập và củng cố mối quan hệ giữa các nhân tố tham gia vào quá trình thương mại. Thông qua mạng, các đối tượng tham gia có thể giao tiếp trực tiếp và liên tục với nhau nhờ đó sự hợp tác lẫn sự quản lý đều được tiến hành nhanh chóng và liên tục

Vì vậy em đã chọn đề tài web thương mại điện tử hiểu thêm và hoàn thiện và phát triển hệ thống thương mại điện tử. Trong quá trình làm luận văn, mặc dù đã tìm kiếm thông tin và đọc nhiều tài liệu tham khảo nhưng do kiến thức còn hạn chế, chưa nắm bắt được nhiều thông tin về thương mại điện tử nên sẽ gặp nhiều thiếu sót.

Em xin chân thành cảm ơn.

# LỜI CẢM ƠN

Trước hết tôi xin tỏ lòng biết ơn sâu sắc đến Thầy Nguyễn Anh Tú, thầy đã tận tình hướng dẫn, giúp đỡ tôi rất nhiều trong suốt quá trình thực hiện luận văn này. Sự hiểu biết sâu sắc về khoa học cũng như kinh nghiệm của thầy chính là tiền đề giúp tôi hoàn thành luận văn.

Tôi cũng xin chân thành cảm ơn thầy đã hướng dẫn và chỉ ra con đường tốt để tôi có thể được làm luận văn và đồng thời học hỏi kinh nghiệm thực tế từ dự án đang hoạt động. Giúp tôi không còn bở ngỡ và lo lắng sau khi ra trường.

Tôi chân thành cảm ơn thầy cô Khoa Công Nghệ Thông Tin Trường Cao Đẳng Công Thương đã tận tình truyền đạt kiến thức trong ba năm học tập. Với vốn kiến thức được tiếp thu trong quá trình học không chỉ là nền tảng cho quá trình nghiên cứu khóa luận mà còn là hành trang để tôi bước vào đời một cách vững chắc và tự tin.

Tôi cũng xin gửi lời cảm ơn đến các bạn lớp CCQ1911F đã chia sẽ và giúp đỡ tôi trong quá trình học tập tại ngôi trường Cao Đẳng Công Thương. Đặc biệt tôi xin cảm ơn các bạn đã giúp tôi nghiên cứu, làm việc trong suốt quá trình thực hiện luận văn. Cuối cùng, tôi xin cảm ơn chân thành với lòng biết ơn sâu sắc đến ba mẹ, chị hai và gia đình đã nuôi dưỡng và dạy bảo tôi nên người và tạo cho tôi có điều kiện tốt để được học tập và phát triển. Với vốn kiến thức hạn hẹp và thời gian thực hiện có hạn nên không tránh khỏi những thiếu sót.

Rất mong nhận được những ý kiến đóng góp của quý thầy cô, đó sẽ là hành trang quý giá giúp tôi hoàn thiện kiến thức của mình sau này.

Em xin chân thành cảm ơn!

Tp Thủ Đức, ngày 11 tháng 06 năm 2022

Sinh viên thực hiện

Đỗ Viết Phú

# MỤC LỤC

[NHẬN XÉT CỦA GIÁO VIÊN i](#_Toc106657465)

[LỜI NÓI ĐẦU ii](#_Toc106657466)

[LỜI CẢM ƠN iii](#_Toc106657467)

[MỤC LỤC iv](#_Toc106657468)

[DANH MỤC CÁC BẢNG vi](#_Toc106657469)

[DANH MỤC CÁC HÌNH vii](#_Toc106657470)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 1](#_Toc106657471)

[1.1. SPRING FRAMEWORK 1](#_Toc106657472)

[1.1.1. Giới thiệu Spring Framework 1](#_Toc106657473)

[1.1.2. Các thành phần cơ bản của Spring Framework 3](#_Toc106657474)

[1.2. RESTFUL API 4](#_Toc106657475)

[1.2.1. Giới thiệu. 4](#_Toc106657476)

[1.2.2. Các thành phần cơ bản Restful, Restful API 5](#_Toc106657477)

[1.3. SPRING BOOT 6](#_Toc106657478)

[1.3.1. Giới thiệu về Spring boot. 6](#_Toc106657479)

[1.3.2. Các thành phần cơ bản của Spring Boot. 7](#_Toc106657480)

[1.4. HIBERNATE FRAMEWORK VÀ JAVA PERSISTENCE API. 8](#_Toc106657481)

[1.4.1. Giới thiệu Hibernate Framework 8](#_Toc106657482)

[1.4.2. Các thành phần cơ bản Hibernate Framework 9](#_Toc106657483)

[1.4.3. Ưu nhược điểm khi dùng Hibernate 11](#_Toc106657484)

[1.4.4. Giới thiệu JPA 11](#_Toc106657485)

[1.4.5. Các thành phần cơ bản JPA 12](#_Toc106657486)

[1.4.6. Các lợi ích khi sử dụng JPA 13](#_Toc106657487)

[1.5. REACT JS. 14](#_Toc106657488)

[1.5.1. Giới thiệu React JS 14](#_Toc106657489)

[1.5.2. Các thành phần cơ bản React JS 15](#_Toc106657490)

[1.6. HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU POSTGRESQL. 16](#_Toc106657491)

[1.6.1. Giới thiệu PostgreSQL 16](#_Toc106657492)

[1.6.2. Các đặc điểm cơ bản PostgreSQL 17](#_Toc106657493)

[1.6.3. Các yếu tố giúp PostgreSQL nổi bật 18](#_Toc106657494)

[CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 19](#_Toc106657495)

[2.1. KHẢO SÁT VÀ PHÂN TÍCH HỆ THỐNG 19](#_Toc106657496)

[2.1.1. Giới thiệu hệ thống 19](#_Toc106657497)

[2.1.2. Mục tiêu xây dựng hệ thống 19](#_Toc106657498)

[2.1.3. Các vai trò người dùng trong hệ thống 20](#_Toc106657499)

[2.1.4. Các yêu cầu chức năng của hệ thống 20](#_Toc106657500)

[2.1.5. Yêu cầu phi chức năng 21](#_Toc106657501)

[2.2. ĐẶC TẢ VÀ THIẾT KẾ HỆ THỐNG 22](#_Toc106657502)

[2.2.1. Biểu đồ use case 22](#_Toc106657503)

[2.2.2. Biểu đồ tổng quan về hệ thống 25](#_Toc106657504)

[2.2.3. Biểu đồ tuần tự 26](#_Toc106657505)

[2.2.4. Biểu đồ hoạt động 29](#_Toc106657506)

[2.3. CƠ SỞ DỮ LIỆU 34](#_Toc106657507)

[2.3.1. Các bảng dữ liệu 34](#_Toc106657508)

[2.3.2. Diagram Database 37](#_Toc106657509)

[CHƯƠNG 3. TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 38](#_Toc106657510)

[3.1. TRIỂN KHAI 38](#_Toc106657511)

[3.1.1. Cài đặt. 38](#_Toc106657512)

[3.1.2. Một số giao diện chính của người dùng 38](#_Toc106657513)

[3.1.3. Một số giao diện chính của quản trị 46](#_Toc106657514)

[3.2. ĐÁNH GIÁ KẾT QUẢ 51](#_Toc106657515)

[3.2.1. Kết quả đạt được và hướng phát triển. 51](#_Toc106657516)

[3.2.2. Hạn chế. 52](#_Toc106657517)

[TÀI LIỆU THAM KHẢO 54](#_Toc106657518)

# DANH MỤC CÁC BẢNG

[Bảng 1. Danh sách các actor 22](#_Toc106657812)

[Bảng 2: Danh sách use case. 23](#_Toc106657813)

[Bảng 3: Bảng dữ liệu Category(Danh mục) 36](#_Toc106657814)

[Bảng 4: Bảng dữ liệu Product(Sản phẩm) 36](#_Toc106657815)

[Bảng 5: Bảng dữ liệu Users 37](#_Toc106657816)

[Bảng 6: Bảng dữ liệu User\_roles 37](#_Toc106657817)

[Bảng 7: Bảng dữ liệu Orders 37](#_Toc106657818)

[Bảng 8: Bảng dữ liệu tin tức 38](#_Toc106657819)

[Bảng 9: Bảng dữ liệu thương hiệu 38](#_Toc106657820)

[Bảng 10: Bảng dữ liệu Contact 38](#_Toc106657821)

[Bảng 11: Bảng dữ liệu Product\_reviews 39](#_Toc106657822)

[Bảng 12: Bảng dữ liệu mô tả sản phẩm 39](#_Toc106657823)

[Bảng 13: Bảng dữ liệu User product 39](#_Toc106657824)

[Bảng 14: Mô tả các phím chức năng (Category) 51](#_Toc106657825)

[Bảng 15: Mô tả các phím chức năng (Product) 52](#_Toc106657826)

[Bảng 16: Mô tả các phím chức năng (User) 54](#_Toc106657827)

[Bảng 17: Mô tả các chức năng (News) 55](#_Toc106657828)

[Bảng 18: Bảng mô tả chức năng trang quản lí liên hệ (Contact) 55](#_Toc106657829)

# DANH MỤC CÁC HÌNH

[Hình 1: Các module của Spring Framework. 3](https://d.docs.live.net/512bd7dcda74e1dc/Computer/DoVietPhu_2119110186.docx#_Toc106657257)

[Hình 2: Biểu đồ hoạt động Restful API 4](#_Toc106657258)

[Hình 3: Restful API hoạt động như thế nào 5](#_Toc106657259)

[Hình 4: Biểu đồ hoạt động Restful API 6](#_Toc106657260)

[Hình 5: Cấu trúc Hibernate 9](#_Toc106657261)

[Hình 6: Các thành phần cơ bản Hibernate Framework 11](#_Toc106657262)

[Hình 7: Tổng quan về JPA 12](#_Toc106657263)

[Hình 8: Thành phần của React (Component) 15](#_Toc106657264)

[Hình 9: Thành phần của React (Props and State) 16](#_Toc106657265)

[Hình 10: Những hệ thống quản lý dữ liệu lớn 18](#_Toc106657266)

[Hình 11: Biểu đồ use case người dùng 23](#_Toc106657267)

[Hình 12: Biểu đồ use case quản trị 24](#_Toc106657268)

[Hình 13 : Biểu đồ tổng quan về hệ thống 25](#_Toc106657269)

[Hình 14: Biểu đồ tuần tự chức năng đăng nhập 26](#_Toc106657270)

[Hình 15: Biểu đồ tuần tự chức năng tìm kiếm 27](#_Toc106657271)

[Hình 16: Biểu đồ tuần tự chức năng quản lý sản phẩm 27](#_Toc106657272)

[Hình 17: Biểu đồ tuần tự chức năng quản lý tài khoản 28](#_Toc106657273)

[Hình 18: Biểu đồ tuần tự chức năng quản lý tin tức 29](#_Toc106657274)

[Hình 19: Biểu đồ hoạt động (Admin) 30](#_Toc106657275)

[Hình 20: Biểu đồ hoạt động(Người dùng) 31](#_Toc106657276)

[Hình 21: Biểu đồ hoạt động chức năng đăng nhập 32](#_Toc106657277)

[Hình 22: Biểu đồ hoạt động chức năng tìm kiếm 33](#_Toc106657278)

[Hình 23: Diagram Database 37](#_Toc106657279)

[Hình 24: Giao diện đăng nhập 38](#_Toc106657280)

[Hình 25: Giao diện đăng ký tài khoản 39](#_Toc106657281)

[Hình 26: Giao diện tìm kiếm sản phẩm 40](#_Toc106657282)

[Hình 27: Giao diện giỏ hàng 41](#_Toc106657283)

[Hình 28: Giao diện thanh toán 42](#_Toc106657284)

[Hình 29: Giao diện thông tin cửa hàng 42](#_Toc106657285)

[Hình 30: Giao diện sản phẩm theo loại 43](#_Toc106657286)

[Hình 31: Giao diện sản phẩm liên quan 44](#_Toc106657287)

[Hình 32: Giao diện đơn hàng đã đặt 45](#_Toc106657288)

[Hình 33:Giao diện trang liên hệ 45](#_Toc106657289)

[Hình 34: Giao diện trang tin tức. 46](#_Toc106657290)

[Hình 35: Giao diện quản trị (Category) 46](#_Toc106657291)

[Hình 36: Form thêm Category 47](#_Toc106657292)

[Hình 37: Giao diện quản trị (Product) 48](#_Toc106657293)

[Hình 38: Form thêm Product 49](#_Toc106657294)

[Hình 39: Giao diện quản trị (User) 50](#_Toc106657295)

[Hình 40: Giao diện quản trị (News) 50](#_Toc106657296)

[Hình 41: Giao diện quản lí liên hệ (Contact) 51](#_Toc106657297)

[Hình 42: Chi phí cho các pha phát triển một hệ thống 53](https://d.docs.live.net/512bd7dcda74e1dc/Computer/DoVietPhu_2119110186.docx#_Toc106657298)

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## SPRING FRAMEWORK

### **Giới thiệu Spring Framework**

Spring framework, hay ngắn hơn là Spring, là một cấu trúc dùng để xây dựng chương trình ứng dụng mã nguồn mở dành cho ngôn ngữ lập trình Java. Phiên bản đầu tiên của nó do Rod Johnson viết và đã được ông xuất bản cùng với quyển sách đầu tay Expert One-on-One JavaEE Design and Development (Nhà xuất bản Wrox Press, Tháng 10 năm 2002).

Kiến trúc của Spring framework được ra mắt công chúng lần đầu tiên hồi tháng 6 năm 2003 dưới Giấy phép Apache - phiên bản 2.0. Phiên bản 1.0 đánh dấu mốc thành đạt đầu tiên được xuất bản vào tháng 3 năm 2004 và tiếp đó vào tháng 9 năm 2004, tháng 3 năm 2005.

Tuy Spring framework không bắt buộc người ta phải tuân theo một mô hình lập trình cụ thể nào, song nó lan truyền rộng rải trong cộng đồng những người viết chương trình dùng Java, như một hình thức chủ yếu thay thế cho mô hình Enterprise Java Bean. Theo thiết kế, bộ framework này giải phóng lập trình viên dùng Java, cho phép họ nhiều quyền tự do hơn và đồng thời cung cấp một giải pháp tiện lợi, đầy đủ dẫn chứng bằng tài liệu, dễ dàng sử dụng, phù hợp với những thực hành thông dụng trong công nghệ phần mềm.

Bên cạnh những đặc trưng nền tảng của Spring framework là những cái có thể dùng được trong bất cứ một chương trình ứng dụng Java nào, rất nhiều các mở rộng và tiến bộ trong việc kiến tạo các trình ứng dụng dành cho nền tảng mạng web (web-based application) dựa trên nền Java Enterprise cũng tồn tại nữa. Spring framework nổi tiếng cũng một phần do chính đặc thù kể trên và được giới thương mại công nhận như một nền tảng kiến trúc có tầm quan trọng trong chiến lược kiến tạo phần mềm.

Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: Dependency Injection và Aspect Oriented Programming. Những tính năng core (cốt lõi) của Spring có thể được sử dụng để phát triển Java Desktop, ứng dụng mobile, Java Web. Mục tiêu chính của Spring là giúp phát triển các ứng dụng J2EE một cách dễ dàng hơn dựa trên mô hình sử dụng POJO (Plain Old Java Object)

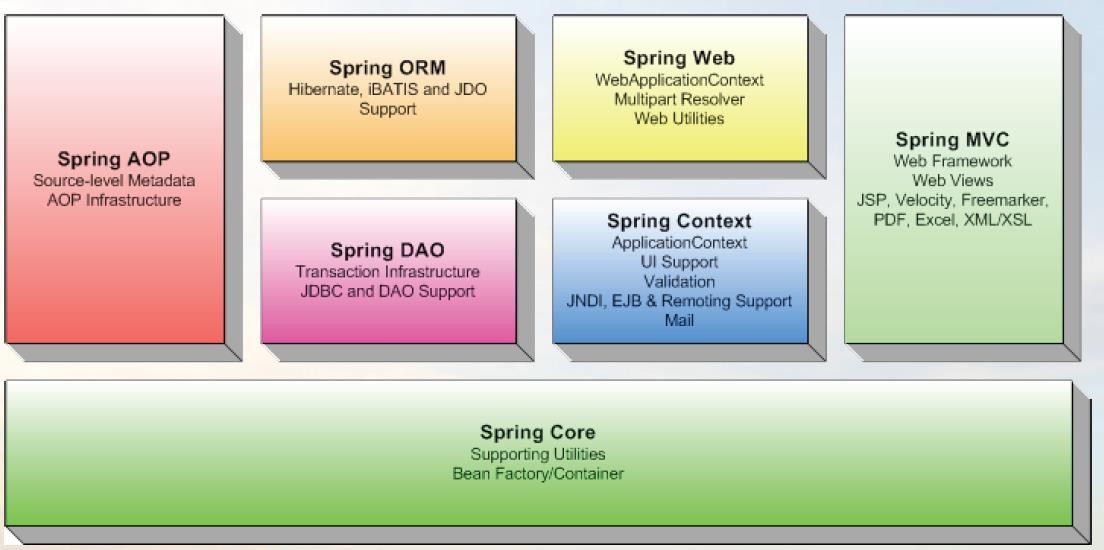
Vào năm 2002, Spring Framework phát hành phiên bản đầu tiên bởi Rod Johnson. Việc xây dựng các ứng dụng doanh nghiệp trở nên đơn giản và dễ dàng hơn. Rod Johnson đã cho xuất bản một cuốn sách được phổ biến rộng rãi có nhan đề "J2EE Development without EJB". Điều này đã cho Spring Framework rất nhiều động lực để thay thế EJB.

Theo đà phát triển đó, Spring đã trở thành framework mã nguồn mở phổ biến nhất để xây dựng các ứng dụng doanh nghiệp. Cách tiếp cận thực tế ban đầu của Rod Johnson tiếp tục được phát triển và hướng tới một bộ công cụ hoàn chỉnh dành cho xây dựng các ứng dụng doanh nghiệp. Theo một số nguồn, trên 50% các ứng dụng web Java hiện nay đang sử dụng Spring.

Để ngăn chặn sự phức tạp trong phát triển các ứng dụng, Spring Framework thường dựa trên các quan điểm như sau:

* Đơn giản hóa công việc phát triển thông qua việc sử dụng các POJO (Plain Old Java Object)
* Nới lỏng ràng buộc giữa các thành phần thông qua việc sử dụng Dependency Injection
* Giảm thiểu các mã boilerplate thông qua việc sử dụng template và aspect.

### **Các thành phần cơ bản của Spring Framework**



Hình 1: Các module của Spring Framework.

**Spring Core**: Core package là phần cơ bản nhất của Spring, cung cấp những đặc tính như IoC (Inversion of Control) và DI (Dependency Injection). Khái niệm cơ bản là BeanFactory, một cài đặt của Factory pattern, cho phép “móc nối” sự phụ thuộc giữa các đối tượng trong file cấu hình.

**Spring Context**: Spring context là một file cấu hình để cung cấp thông tin ngữ cảnh của Spring. Spring context cung cấp các service như JNDI access, EJB integration, e-mail, internalization, validation, và scheduling functionality.

**Spring AOP (Aspect – Oriented Programming):** Spring AOP module tích hợp chức năng lập trình hướng khía cạnh vào Spring framework thông qua cấu hình của nó. Spring AOP module cung cấp các dịch vụ quản lý giao dịch cho các đối tượng trong bất kỳ ứng dụng nào sử dụng Spring. Với Spring AOP chúng ta có thể tích hợp declarative transaction management vào trong ứng dụng mà không cần dựa vào EJB component. Spring AOP module cũng đưa lập trình metadata vào trong Spring. Sử dụng cái này chúng ta có thể thêm annotation vào source code để hướng dẫn Spring nơi và làm thế nào để liên hệ với aspect.

**Spring DAO (Data Access Object):** Tầng JDBC và DAO đưa ra một cây phân cấp exception để quản lý kết nối đến database, điều khiển exception và thông báo lỗi được ném bởi vendor của database. Tầng exception đơn giản điều khiển lỗi và giảm khối lượng code mà chúng ta cần viết như mở và đóng kết nối. Module này cũng cung cấp các dịch vụ quản lý giao dịch cho các đối tượng trong ứng dụng Spring.

**Spring ORM (Object Relational Mapping):** Spring có thể tích hợp với một vài ORM framework để cung cấp Object Relation tool bao gồm: JDO, Hibernate, OJB và iBatis SQL Maps. - Spring Web: Nằm trên application context module, cung cấp context cho các ứng dụng web. Spring cũng hỗ trợ tích hợp với Struts, JSF và Webwork. Web module cũng làm giảm bớt các công việc điều khiển nhiều request và gắn các tham số của request vào các đối tượng domain.

**Spring MVC:** MVC Framework thì cài đặt đầy đủ đặc tính của MVC pattern để xây dựng các ứng dụng Web. MVC framework thì cấu hình thông qua giao diện và chứa được một số kỹ thuật view bao gồm: JSP, Velocity, Tiles và generation of PDF và Excel file.

## RESTFUL API

### **Giới thiệu.**

**RESTful API** là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



Hình 2: Biểu đồ hoạt động Restful API

**RESTful** **API** được xem là một thuật ngữ khá khó hiểu đối với những nhà phát triển mới. Mặc dù họ vẫn luôn tìm hiểu về nó, nhưng với lượng lớn thông tin khác nhau thuộc nhiều website khác nhau khiến cho họ cảm thấy mông lung, khó hiểu thậm chí là chán nản.



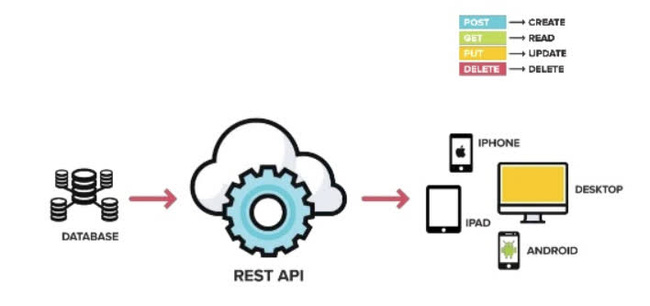
Hình 3: Restful API hoạt động như thế nào

### **Các thành phần cơ bản Restful, Restful API**

**API (Application Programming Interface):** là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như [JSON](https://topdev.vn/blog/json-la-gi/) hay XML.

**REST (REpresentational State Transfer):** là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

**RESTful API:** là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.



Hình 4: Biểu đồ hoạt động Restful API

## SPRING BOOT

### **Giới thiệu về Spring boot.**

**Spring Boot** là một dự án phát triển bởi **JAV**(ngôn ngữ java) trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên chúng ta đơn giản hóa quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển business cho ứng dụng.

Để phát triển một ứng dụng web cơ bản HelloWorld sử dụng Spring framework bạn sẽ cần ít nhất 5 công đoạn sau:

* Tạo một project sử dụng Maven với các dependency cần thiết của Spring MVC và Servlet API.
* Một tập tin web.xml để khai báo DispatcherServlet của Spring MVC.
* Một tập tin cấu hình của Spring MVC.
* Một class Controller trả về một trang “Hello World” khi có request đến.
* Cuối cùng là phải có một web server dùng để triển khai ứng dụng lên chạy.

Trong các công đoạn này, chỉ có công đoạn tạo một class Controller thì có thể khác cho các ứng dụng khác nhau vì mỗi ứng dụng có một yêu cầu khác nhau. Còn các công đoạn khác thì như nhau.

Giờ đây với Spring Boot, chúng ta có thể tạo dự án Spring một cách nhanh chóng và cấu hình cũng đơn giản dùng Sublime Text để phát triển luôn khỏi cần cài đặt eclipse hay netbean.

*Dưới đây là một số tính năng nổi bật của Spring Boot:*

* Tạo các ứng dụng Spring độc lập
* Nhúng trực tiếp Tomcat, Jetty hoặc Undertow (không cần phải deploy ra file WAR)
* Các starter dependency giúp việc cấu hình Maven đơn giản hơn
* Tự động cấu hình Spring khi cần thiết
* Không sinh code cấu hình và không yêu cầu phải cấu hình bằng XML.

*Lợi ích của Sping Boot:*

* Nó rất dễ dàng để phát triển các ứng dụng dựa trên **Spring**với **Java**hoặc **Groovy**.
* Nó giảm thiểu thời gian phát triển và tăng năng xuất
* Nó tránh việc phải viết nhiều mã nguyên mẫu (boilerplate Code), Annotations và các cấu hình XML.
* Nó dễ dàng để bạn tương tác các ứng dụng **Spring Boot** với các hệ sinh thái của **Spring**như **Spring JDBC**, **Spring ORM**, **Spring Data**, **Spring Security** etc.
* Nó đi theo cách tiếp  cận “Nguyên tắc cấu hình mặc định” để giảm thiểu thời gian và nỗ lực cần thiết để phát triển ứng dụng.
* Nó cung cấp các Server nhúng (Embedded HTTP servers) như là **Tomcat**, **Jetty**.... để phát triển và test các ứng dụng web nhanh chóng và dễ dàng.
* Nó cung cấp công cụ **CLI**(Command Line Interface) dể phát triển và test các ứng dụng **Spring Boot(Java**hoặc **Groovy**) từ các dòng lệnh (command prompt) rất dễ dàng và nhanh chóng.
* Nó cung cấp rất nhiều các plugin để phát triển và test các ứng dụng **Spring Boot** nhanh chóng sử dụng các công cụ **Build** như **Maven**và **Gradle**
* Nó cung cấp nhiều plugin để làm việc với các cơ sở dữ liệu nhúng (embedded database)  và các cơ sở dữ liệu lưu trữ trên bộ nhớ (in-memory Databases) một cách dễ dàng.

### **Các thành phần cơ bản của Spring Boot**.

1. *SpringApplication*

SpringApplication là một class cung cấp cách thuận tiện để khởi chạy ứng dụng từ hàm main (). Để start ứng dụng, chỉ cần gọi method run ().

1. *Externalized Configuration*

Spring Boot cho phép chúng ta cấu hình từ bên ngoài (externalize), do đó một ứng dụng có thể chạy trên nhiều môi trường khác nhau. Chúng ta có thể sử dụng file YAML, file properties, các biến môi trường và tham số command-line để thực hiện externalize configuration. Các thuộc tính cấu hình có thể inject trực tiếp vào bean bằng cách sử dụng annotation @Value hoặc thông qua object với @ConfiguarationProperties….

1. *Profiles*

Spring Boot Profiles cung cấp một cách để phân chia các cấu hình cho mỗi môi trường. Các annotation @Component hoặc @Configuation có thể được đánh dấu @Profire đề giới hạn khi nào được tải lên.

1. *Logging*

Spring Boot sử dụng common logging cho tất cả các chức năng log nội bộ. Các depencdeny logging được quản lý mặc định, chúng ta không nên / cần sửa dependency logging nếu khồng có yêu cầu tùy biến (customization) thực sự cần.

## HIBERNATE FRAMEWORK VÀ JAVA PERSISTENCE API.

### **Giới thiệu Hibernate Framework**

**Hibernate** được phát triển bởi một nhóm lập trình Java được đứng đầu bởi Gavin King. Sau đó Jboss mới thuê nhóm này để hỗ trợ họ khi sử dụng hibernate. Hibernate trải qua rất nhiều phiên bản khác nhau đến thời điểm hiên tại. Ở phiên bản này hibernate đã hỗ trợ những tính năng mới hơn như Interceptor/Callback, dữ liệu người dùng có thể định nghĩa, hỗ trợ Anotation của Java5.

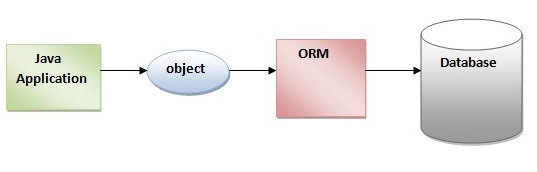
**Hibernate** là 1 [ORM Framework](https://stackjava.com/uncategorized/orm-la-gi-tong-quan-ve-orm-framework.html): thực hiện mapping cơ sở dữ liệu quan hệ sang các object trong ngôn ngữ hướng đối tượng.Thực hiện giao tiếp giữa tầng ứng dụng với tầng dữ liệu (kết nối, truy xuất, lưu trữ…)

**Hibernate** giúp lập trình viên viết ứng dụng Java có thể map các object (POJO) với hệ quản trị cơ sở dữ liệu quan hệ (database), và hỗ trợ thực hiện các khái niệm lập trình hướng đối tượng với cơ dữ liệu quan hệ.

**Hibernate** giúp lưu trữ và truy vấn dữ liệu quan hệ mạnh mẽ và nhanh. Hibernate cho phép bạn truy vẫn dữ liệu thông qua Java Persistence API (JPA) hoặc bằng ngôn ngữ SQL mở rộng của Hibernate (HQL) hoặc bằng SQL thuần (Native SQL)

**Hibernate** vốn là một thư viện sinh ra để làm việc với mọi loại database, nó không phụ thuộc vào bạn chọn loại database nào. Nếu **Java** là “Viết 1 lần chạy mọi nơi” thì **Hibernate** là “Viết 1 lần chạy trên mọi loại database”.

**Tool ORM** giúp đơn giản hoá việc tạo ra dữ liệu, thao tác dữ liệu và truy cập dữ liệu. Đó là một kỹ thuật lập trình để ánh xạ đối tượng vào dữ liệu được lưu trữ trong cơ sở dữ liệu.



Hình 5: Cấu trúc Hibernate

### **Các thành phần cơ bản Hibernate Framework**

Kiến trúc Hibernate bao gồm nhiều đối tượng như đối tượng persistent, session factory, transaction factory, connection factory, session, transaction,...

**Configuration:** Đối tượng Configuration là đối tượng Hibernate đầu tiên bạn tạo trong bất kỳ ứng dụng Hibernate nào và chỉ cần tạo một lần trong quá trình khởi tạo ứng dụng. Nó đại diện cho một tập tin cấu hình hoặc thuộc tính yêu cầu của Hibernate. Đối tượng Configuration cung cấp hai thành phần chính:

* **Database Connection:** Thao tác này được xử lý thông qua một hoặc nhiều tệp cấu hình được Hibernate hỗ trợ. Các tệp này là hibernate.properties và hibernate.cfg.xml.
* **Class Mapping Setup:** Thành phần này tạo ra kết nối giữa các lớp Java và các bảng cơ sở dữ liệu.

**SessionFactory:** Đối tượng Configuration được sử dụng để tạo ra một đối tượng SessionFactory mà cấu hình Hibernate cho ứng dụng bằng cách sử dụng tệp tin cấu hình được cung cấp và cho phép một đối tượng Session được khởi tạo. SessionFactory là một đối tượng luồng an toàn và được sử dụng bởi tất cả các luồng của một ứng dụng.

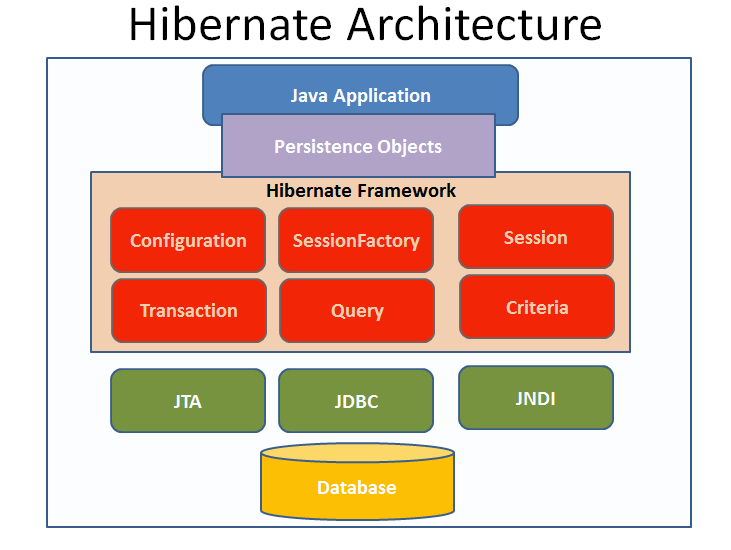
SessionFactory là đối tượng nặng nên thường nó được tạo ra trong quá trình khởi động ứng dụng và lưu giữ để sử dụng sau này. Bạn sẽ cần một đối tượng SessionFactory cho mỗi cơ sở dữ liệu bằng cách sử dụng một tập tin cấu hình riêng biệt. Vì vậy, nếu bạn đang sử dụng nhiều cơ sở dữ liệu thì bạn sẽ phải tạo nhiều đối tượng SessionFactory.

**Session:** Một session được sử dụng để có được một kết nối vật lý với một cơ sở dữ liệu. Đối tượng Session là nhẹ và được thiết kế để được tạo ra thể hiện mỗi khi tương tác với cơ sở dữ liệu. Các đối tượng liên tục được lưu và truy xuất thông qua một đối tượng Session. Các đối tượng Session không nên được mở trong một thời gian dài bởi vì chúng thường không phải là luồng an toàn và chúng cần được tạo ra và được đóng khi cần thiết.

**Transaction**: Một Transaction đại diện cho một đơn vị làm việc với cơ sở dữ liệu và hầu hết các RDBMS hỗ trợ chức năng transaction. Các transaction trong Hibernate được xử lý bởi trình quản lý transaction và transaction (từ JDBC hoặc JTA). Đây là một đối tượng tùy chọn và các ứng dụng Hibernate có thể chọn không sử dụng interface này, thay vào đó quản lý transaction trong code ứng dụng riêng.

**Query**: Các đối tượng Query sử dụng chuỗi truy vấn SQL hoặc Hibernate Query Language (HQL) để lấy dữ liệu từ cơ sở dữ liệu và tạo các đối tượng. Ví dụ truy vấn được sử dụng để ràng buộc các tham số truy vấn, giới hạn số lượng kết quả được trả về bởi truy vấn và cuối cùng thực hiện truy vấn.

**Criteria**: Đối tượng Criteria được sử dụng để tạo và thực hiện truy vấn các tiêu chí định hướng đối tượng để lấy các đối tượng.



Hình 6: Các thành phần cơ bản Hibernate Framework

### **Ưu nhược điểm khi dùng Hibernate**

*a, Ưu điểm*

* Dễ sử dụng: dễ dàng quản lí các kết nối database và dễ fix bug, cung cấp sẵn nhiều API truy vấn
* Tính độc lập: không cần quan tâm tới cơ sở dữ liệu sử dụng khi viết câu lệnh SQL.
* Tính hướng đối tượng: tập trung xử lý theo hướng đối tượng, phù hợp sử dụng trong các case CRUD(Create, Read, Update, Delete)
* Tính tin cậy: Hibernate đã được kiểm thử và khá an toàn trong các truy vấn

*b, Nhược điểm*

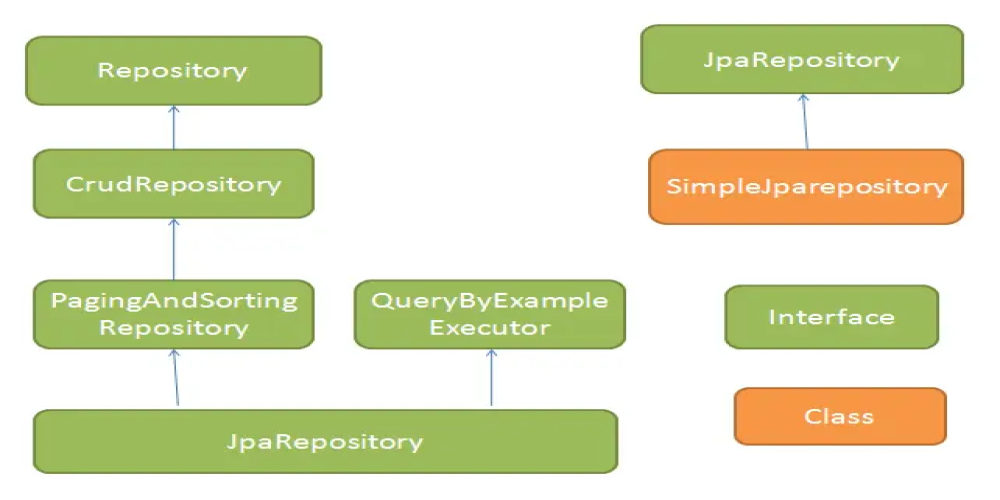
* Không hỗ trợ các câu truy vấn phức tạp
* Một số trường hợp vẫn phải dùng native SQL do Hibernate không thể cover hết tất cả các cú pháp của các hệ quản trị cơ sử dữ liệu.
* Bị hạn chế sự can thiệp vào câu lệnh SQl do nó được tự động sinh ra.

### **Giới thiệu JPA**

**JPA** là viết tắt của **Java Persistence API**, nó là một đặc tả Java cho việc ánh xạ giữa các đối tượng Java với cơ sở dữ liệu quan hệ sử dụng công nghệ phổ biến là **ORM** (Object Relational Mapping).

**JPA** cung cấp đầy đủ các công cụ cho phép chúng ta có thể thao tác với cơ sở dữ liệu một cách đơn giản và nhanh chóng. JPA có thể dùng để **persist** một đối tượng Java (**POJO** – Plain Old Java Object) vào trong cơ sở dữ liệu hoặc lấy dữ liệu từ cơ sở dữ liệu và ánh xạ (**mapping**) ra các đối tượng Java một cách đơn giản.

JPA hoạt động như một **cầu nối** giữa các table/ các mối quan hệ giữa các table trong database và các class/ mối quan hệ giữa các object. Ví dụ: table USER với các column (Id, username, password) sẽ tương ứng với class User.java với các field Id, username, password. Từ đó mỗi khi truy vấn table hay các column ta sẽ gọi trực tiếp các phương thức trên các class, các field của class mà không cần quan tâm tới việc đang dùng loại database nào, kiểu dữ liệu database ra sao.



Hình 7: Tổng quan về JPA

### **Các thành phần cơ bản JPA**

JPA bao gồm ba thành phần chính là: EntityManagerFactory, Entity, EntityManager. Ngoài ra còn có, EntityTransaction, Persistence, Query.

1. *Entity*

Entity là các đối tượng thể hiện tương ứng 1 table trong cơ sở dữ liệu. Entity thường là các class POJO đơn giản, chỉ gồm các phương thức getter, setter.

Dưới đây là một số đặc điểm của một Entity:

* Entity có thể tương tác với cơ sở dữ liệu quan hệ.
* Entity được xác định thông qua một định danh (id), tương đương với khóa chính trong table của cơ sở dữ liệu quan hệ.
* Entity hỗ trợ transaction.
* Entity hỗ trợ kế thừa giống như những class Java khác.

1. *EntityManager*

EntityManager là một interface cung cấp các API cho việc tương tác với các Entity.

Một số chức năng cơ bản của EntityManager như:

* Persist: phương thức này dùng để lưu một thực thể mới tạo vào cơ sở dữ liệu.
* Merge: dùng để cập nhật trạng thái của entity vào cơ sở dữ liệu.
* Remove: xóa một instance của entity.

1. *EntityManagerFactory*

EntityManagerFactory được dùng để tạo ra một instance của EntityManager.

1. *Persistence*

Một Persistence định nghĩa một tập hợp các Entity class được quản lý bởi 1 instacne của EntityManager trong ứng dụng. Persistence (javax.persistence.Persistence) class bao gồm các phương thức static để lấy instance của EntityManagerFactory.

1. *EntityTransaction*

Một Transaction là một tập hợp các thao tác trong đó tất cả các thao tác phải được thực hiện thành công hoặc tất cả thất bại. Một database transaction bao gồm một tập hợp các câu lệnh SQL được committed hoặc rolled back trong một unit.

EntityTransaction có quan hệ 1-1 với EntityManager. Bất kỳ thao tác nào được bắt đầu thông qua đối tượng EntityManager đều được đặt trong một Transaction. Đối tượng EntityManager giúp tạo EntityTransaction.

1. *Query*

Đây là một interface, được mỗi nhà cung cấp JPA implement để có được các đối tượng quan hệ đáp ứng các tiêu chí (criteria) truy vấn.

### **Các lợi ích khi sử dụng JPA**

*Lợi ích*

* Đơn giản hóa công nghệ cho tầng persistence (tầng dữ liệu).
* Không phụ thuộc vào các framework ORM.
* Có nhiều nhà cung cấp hỗ trợ cài đặt JPA.
* Dữ liệu có thể được lưu trữ thông qua việc ORM.
* Viết ít code hơn.
* Performance tốt.
* Độc lập về database.
* Không phải làm việc với SQL.
* Hỗ trợ cấu hình triển khai bằng annotation và xml.
* Có nhiều framework ORM miễn phí hỗ trợ có thể dùng để phát triển nhiều loại ứng dụng khác nhau.
* JPA là một đặc tả đã được chuẩn hóa và là một thành phần trong đặc tả EJB 3.
* Dễ dàng chuyển từ một ORM này sang một ORM khác. Ví dụ từ iBatis sang Hibernate.

## REACT JS.

### **Giới thiệu React JS**

React (Hay ReactJS, React.js) là một thư viện Javascript mã nguồn mở để xây dựng các thành phần giao diện có thể tái sử dụng. Nó được tạo ra bởi Jordan Walke, một kỹ sư phần mềm tại Facebook. Người bị ảnh hưởng bởi XHP (Một nền tảng thành phần HTML cho PHP). React lần đầu tiên được triển khai cho ứng dụng Newsfeed của Facebook năm 2011, sau đó được triển khai cho Instagram.com năm 2012. Nó được mở mã nguồn (open-sourced) tại JSConf US tháng 5 năm 2013.

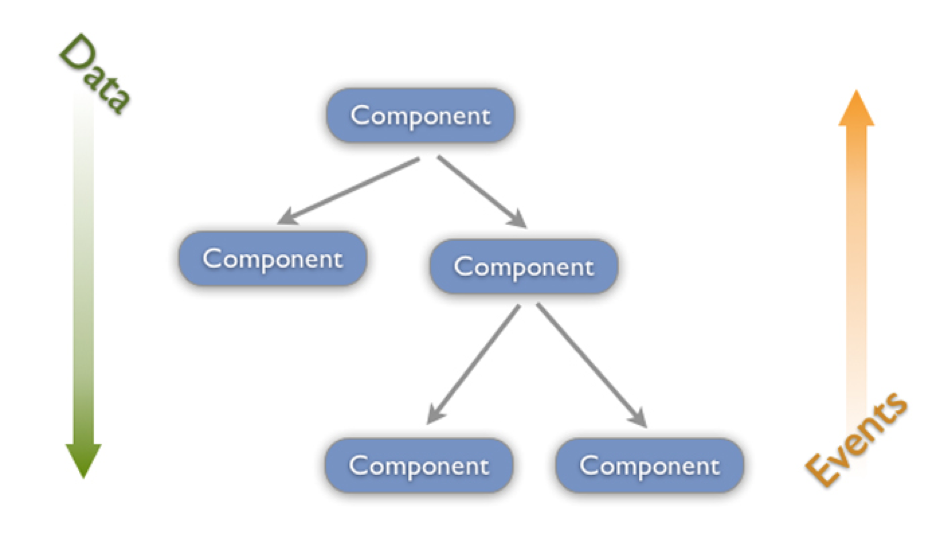
React.js nổi lên trong những năm gần đây với xu hướng Single Page Application. Trong khi những framework khác cố gắng hướng đến một mô hình MVC hoàn thiện thì React nổi bật với sự đơn giản và dễ dàng phối hợp với những thư viện Javascript khác. Nếu như AngularJS là một Framework cho phép nhúng code javasscript trong code html thông qua các attribute như ng-model, ng-repeat...thì với React là một library cho phép nhúng code html trong code javascript nhờ vào JSX, bạn có thể dễ dàng lồng các đoạn HTML vào trong JS.Tích hợp giữa javascript và HTML vào trong JSX làm cho các component dễ hiểu hơn

React là một thư viện UI phát triển tại Facebook để hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được. React được sử dụng tại Facebook trong production, và [www.instagram.com](http://www.instagram.com/) được viết hoàn toàn trên React.

Một trong những điểm hấp dẫn của React là thư viện này không chỉ hoạt động trên phía client, mà còn được render trên server và có thể kết nối với nhau. React so sánh sự thay đổi giữa các giá trị của lần render này với lần render trước và cập nhật ít thay đổi nhất trên DOM.

### **Các thành phần cơ bản React JS**

**Components**: là một thành phần quan trọng nhất trong React, nó giúp phân chia các thành phần giao diện (UI) ra các thành phần nhỏ hơn.Trong một trang bao gồm nhiều các components khác nhau. Một React Component là thành phần nhận vào một props và trả về JSX dùng để hiện thị giao diện người dùng.

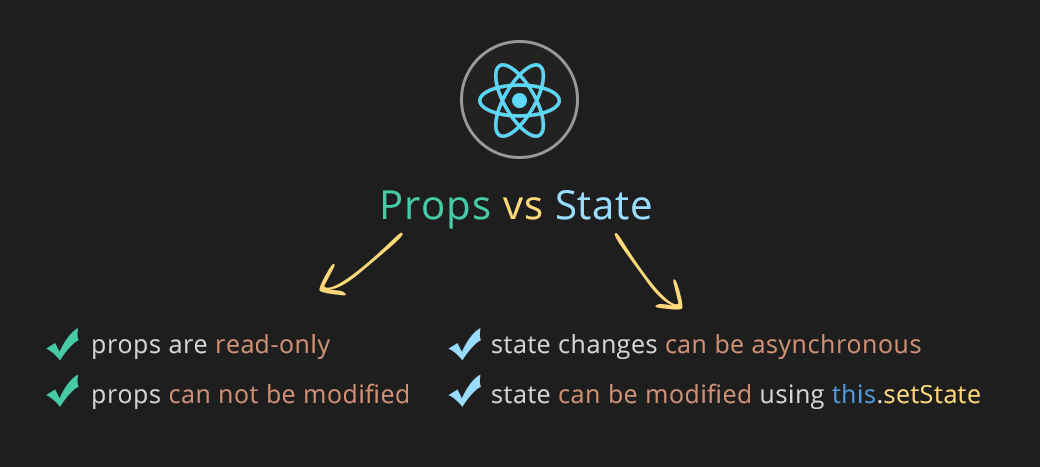


Hình 8: Thành phần của React (Component)

**JSX**: React và React Native đều sử dụng JSX, một cú pháp cho phép viết các phần tử bên trong các đoạn mã Javascript. Bởi vì JSX là Javascript nên có thể sử dụng các biến ngay bên trong nó. Không chỉ các biến mà bất cứ kiểu thức nào trong Javascript cũng có thể được sử dụng trong JSX, giả sử ở đây chúng ta sẽ gọi một hàm bên trong JSX

**Props**: là một object được truyền vào trong một components, mỗi components sẽ nhận vào props và trả về react element. Props cho phép chúng ta giao tiếp giữa các components với nhau bằng cách truyền tham số qua lại giữa các components. Cách truyền một props cũng giống như cách mà bạn thêm một attributes cho một element HTML.

**State**: là một object có thể được sử dụng để chứa dữ liệu hoặc thông tin về components, có thể được thay đổi bất cứ khi nào mong muốn. Khác với props bạn có thể truyền props sang các components khác nhau thì state chỉ tồn tại trong phạm vi của components chứa nó, mỗi khi state thay đổi thì components đó sẽ được render lại. Nó được dùng để xử lý dữ liệu thay đổi theo thời gian hoặc trả lời tương tác từ người dùng.



Hình 9: Thành phần của React (Props and State)

## HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU POSTGRESQL.

### **Giới thiệu PostgreSQL**

[PostgreSQL](https://bizflycloud.vn/tin-tuc/postgresql-la-gi-tim-hieu-ve-co-so-du-lieu-ma-nguon-mo-tien-tien-nhat-the-gioi-20180919175924611.htm) là một hệ thống quản trị cơ sở dữ liệu quan hệ-đối tượng (object-relational database management system) có mục đích chung, hệ thống cơ sở dữ liệu mã nguồn mở tiên tiến nhất hiện nay. PostgreSQL được phát triển dựa trên POSTGRES 4.2 tại phòng khoa học máy tính Berkeley, Đại học California.

PostgreSQL được thiết kế để chạy trên các nền tảng tương tự UNIX. Tuy nhiên, PostgreSQL sau đó cũng được điều chỉnh linh động để có thể chạy được trên nhiều nền tảng khác nhau như Mac OS X, Solaris và Windows.

PostgreSQL là một phần mềm mã nguồn mở miễn phí. Mã nguồn của phần mềm khả dụng theo license của PostgreSQL, một license nguồn mở tự do. Theo đó, bạn sẽ được tự do sử dụng, sửa đổi và phân phối PostgreSQL dưới mọi hình thức.

PostgreSQL không yêu cầu quá nhiều công tác bảo trì bởi có tính ổn định cao. Do đó, nếu bạn phát triển các ứng dụng dựa trên PostgreSQL, chi phí sở hữu sẽ thấp hơn so với các hệ thống quản trị dữ liệu khác.

PostgreSQL sở hữu một hệ tính năng đa dạng giúp hỗ trợ các nhà phát triển xây dựng app, các nhà quản trị bảo vệ toàn vẹn dữ liệu, và tạo ra một môi trường chịu lỗi [fault-tolerant](https://bizflycloud.vn/tin-tuc/fault-tolerance-la-gi-he-thong-chiu-loi-co-tac-dung-gi-20180716163002091.htm) giúp bạn quản lý dữ liệu bất kể tập dữ liệu lớn hay nhỏ. Bên cạnh hệ thống nguồn mở và miễn phí, PostgreSQL cũng có khả năng mở rộng tuyệt vời. Ví dụ, bạn có thể định nghĩa các kiểu dữ liệu riêng của bạn, xây dựng các hàm tùy chỉnh, hay viết mã từ các ngôn ngữ lập trình khác nhau mà không cần biên dịch lại cơ sở dữ liệu.

*Cấu trúc vật lý của PostgreSQL rất đơn giản nó bao gồm các thành phần sau:*

* Shared Memory
* Background processes
* Data directory structure / Data files

### **Các đặc điểm cơ bản PostgreSQL**

PostgreSQL là một chương trình mã nguồn mở xây dựng trên mã nguồn ban đầu của đại học Berkeley. Nó theo [chuẩn SQL99](https://vi.wikipedia.org/w/index.php?title=Chu%E1%BA%A9n_SQL99&action=edit&redlink=1) và có nhiều đặc điểm hiện đại:

* Câu [truy vấn phức hợp](https://vi.wikipedia.org/w/index.php?title=Truy_v%E1%BA%A5n_ph%E1%BB%A9c_h%E1%BB%A3p&action=edit&redlink=1) (*complex query*)
* Khóa ngoại (*foreign key*)
* Thủ tục sự kiện (*trigger*)
* Các [khung nhìn](https://vi.wikipedia.org/w/index.php?title=Khung_nh%C3%ACn&action=edit&redlink=1) (*view*)
* Tính [toàn vẹn](https://vi.wikipedia.org/w/index.php?title=T%C3%ADnh_to%C3%A0n_v%E1%BA%B9n_(giao_d%E1%BB%8Bch)&action=edit&redlink=1) của các [giao dịch](https://vi.wikipedia.org/wiki/Giao_d%E1%BB%8Bch) (*integrity transactions*)
* Việc kiểm tra truy cập đồng thời đa phiên bản (*multiversion concurrency control*)

Hơn nữa, PostgreSQL có thể dùng trong nhiều trường hợp khác, chẳng hạn như tạo ra các khả năng mới như:

* Kiểu dữ liệu
* Hàm
* Toán tử
* Hàm tập hợp
* Phương pháp liệt kê
* Ngôn ngữ theo thủ tục
* Truy vấn xử lý song song (parallel query)
* Sao chép dữ liệu dạng luồng (Streaming replication)

PostgreSQL được phổ biến bằng [giấy phép BSD](https://vi.wikipedia.org/wiki/Gi%E1%BA%A5y_ph%C3%A9p_BSD) cổ điển. Nó không quy định những hạn chế trong việc sử dụng [mã nguồn](https://vi.wikipedia.org/wiki/M%C3%A3_ngu%E1%BB%93n) của [phần mềm](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m). Bởi vậy PostgreSQL có thể được dùng, sửa đổi và phổ biến bởi bất kỳ ai cho bất kỳ mục đích nào.

PostgreSQL cũng là hệ quản trị cơ sở dữ liệu hỗ trợ mạnh trong việc lưu trữ dữ liệu không gian. PostgreSQL kết hợp với module [Postgis](https://vi.wikipedia.org/w/index.php?title=Postgis&action=edit&redlink=1) cho phép người dùng lưu trữ các lớp dữ liệu không gian. Khi sử dụng PostgreSQL, Postgis kết hợp với các phần mềm [GIS](https://vi.wikipedia.org/wiki/H%E1%BB%87_th%E1%BB%91ng_Th%C3%B4ng_tin_%C4%90%E1%BB%8Ba_l%C3%BD) hỗ trợ hiển thị, truy vấn, thống kê hoặc xử lý dữ liệu không gian.

### **Các yếu tố giúp PostgreSQL nổi bật**

PostgreSQL là hệ thống quản lý cơ sở dữ liệu đầu tiên triển khai tính năng kiểm soát đồng thời nhiều phiên bản (MVCC) trước cả Oracle. Tính năng MVCC cũng tương tự với các snapshot riêng biệt trong Oracle.

Là hệ thống quản lý cơ sở dữ liệu quan hệ đối tượng, PostgreSQL cho phép thêm vào các tính năng tùy chỉnh được phát triển bằng các ngôn ngữ chương trình khác nhau như C/C , Java,...



Hình 10: Những hệ thống quản lý dữ liệu lớn

Bên cạnh đó, khi bạn có bất kỳ vấn đề gì cần hỗ trợ, luôn có một cộng đồng nhiệt tình sẵn sàng để trợ giúp. Bạn luôn có thể tìm thấy câu trả lời từ cộng đồng của PostgreSQL về các vấn đề có thể gặp phải khi làm việc với PostgreSQL. Ngoài ra, cũng có rất nhiều công ty cung cấp các dịch vụ hỗ trợ trong trường hợp bạn thực sự cần đến.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## KHẢO SÁT VÀ PHÂN TÍCH HỆ THỐNG

### **Giới thiệu hệ thống**

Thời đại công nghệ số 4.0 với hệ thống mạng phát triển, hoạt động kinh doanh online đang là hướng đi đúng, mang lại doanh thu lớn cho nhiều doanh nghiệp và cá nhân kinh doanh. Việc sở hữu một website thương mại điện tử cũng là một trong những yếu tố giúp doanh nghiệp phát triển, gia tăng doanh số. Ngày càng nhiều doanh nghiệp xây dựng hệ thống website thương mại điện tử với mong muốn mở rộng thị trường kinh doanh, tăng trưởng thị phần và mang lại hiệu quả kinh doanh.

Hệ thống website thương mại điện tử được xây dựng dựa trên các công nghệ như Spring Boot, React, PostgreSQL.

Hệ thống phát triển các chức năng theo nhu cầu của người sử dụng (Xem thông tin sản phẩm, đặt hàng…). Hệ thống phát triển trên nhiều nền tảng khác nhau giúp khách hàng dễ dàng truy cập trang web 1 cách dễ dàng nhất

Trang quản trị: Người quản lý có thể dễ dàng kiểm soát được sản phẩm, thêm, xóa, sửa sản phẩm, cũng như kiểm soát được lượng khách hàng đặt hàng, quản lý user, quản lý bài viết.

### **Mục tiêu xây dựng hệ thống**

Đây là trang web thương mại điện tử được thành lập với mục đích tiện lợi cho việc bán hàng online, tham khảo thông tin sản phẩm, hạn chế ngắn thời gian mua bán hàng hóa và có thể mua bán mọi lúc mọi nơi một cách nhanh chóng.

Đồng thời khách hàng cũng có thể cập nhật thông tin về công nghệ mới nhất trên website. Giúp khách hàng tiếp cận dễ dàng hơn với các thiết bị công nghệ số.

Ngoài ra còn có các lợi ích lớn khi ta xây dựng web thương mại điện tử:

* Tăng lợi nhuận
* Tiết kiệm chi phí
* Gia tăng khả năng tương tác với khách hàng
* Nâng cao tính cạnh tranh với đối thủ
* Quảng bá thương hiệu
* Thao tác thuận tiện và nhanh chóng
* Cung cấp thông tin nhanh và chính xác
* Hình thức thanh toán đơn giản
* Hình ảnh và chi tiết sản phẩm trực quan

### **Các vai trò người dùng trong hệ thống**

Để 1 một website có thể phát triển được, thì người dùng truy cập trang là yếu tố vô cùng quan trọng. Người dùng có thể xem thông tin sản phẩm, đặt hàng, đăng ký, đăng nhập, mua hàng online, bình luận về sản phẩm, bình luận bài viết, tin tức.

Ngoài ra người dùng còn có thể:

* Mua bán tạo tiền để vật chất cho hoạt động phát triển của website
* Quảng bá thương hiệu khi đã từng mua sản phẩm
* Đảm bảo số lượng hàng bán ra: Doanh nghiệp cần mua hàng đủ số lượng và cơ cấu phù hợp quy mô hoạt động. Tránh tình trạng hàng tồn, ứ đọng làm gián đoạn kinh doanh.
* Đảm bảo chất lượng hàng mua vào: Nên lựa chọn hàng hóa có chất lượng tối ưu thay vì tối đa, nhằm đáp ứng mọi nhu cầu của người tiêu dùng
* Đảm bảo mua hàng với chi phí thấp nhất: Đây là yếu tố tạo điều kiện thuận lợi cho việc xác định giá trên thị trường, nâng cao sức cạnh tranh doanh nghiệp so với đối thủ nhưng vẫn đảm bảo lợi nhuận.

### **Các yêu cầu chức năng của hệ thống**

* + - 1. *User*
* Đăng nhập
* Đăng xuất
* Tìm kiếm
* Thanh toán
* Đặt hàng
* Đánh giá
* Liên hệ
* Chi tiết sản phẩm
* Bình luận
* Chi tiết tin tức
* Chính sách bảo hành
* Chính sách đổi trả
  + - 1. *Admin*
* Thêm Sản Phẩm
* Xóa Sản Phẩm
* Sửa Sản Phẩm
* Thêm Danh Mục
* Xóa Danh Mục
* Sửa Danh Mục
* Thêm Tin Tức
* Sửa Tin Tức
* Xóa Tin Tức
* Thêm Sự Kiện
* Sửa Sự Kiện
* Xóa Sự Kiện
* Thêm Tài Khoản
* Sửa Tài Khoản
* Xóa Tài Khoản

### **Yêu cầu phi chức năng**

* An toàn thông tin
* Giao diện
* Màu sắc, thông tin hiển thị phù hợp
* Khả năng mở rộng
* Sức chứa
* Độ khả dụng
* Độ tin cậy
* Khả năng phục hồi
* Khả năng bảo trì
* Dịch vụ có sẵn
* An ninh
* Quy định
* Khả năng quản lí
* Môi trường
* Toàn vẹn dữ liệu
* Khả năng sử dụng
* Khả năng tương tác

## ĐẶC TẢ VÀ THIẾT KẾ HỆ THỐNG

### **Biểu đồ use case**

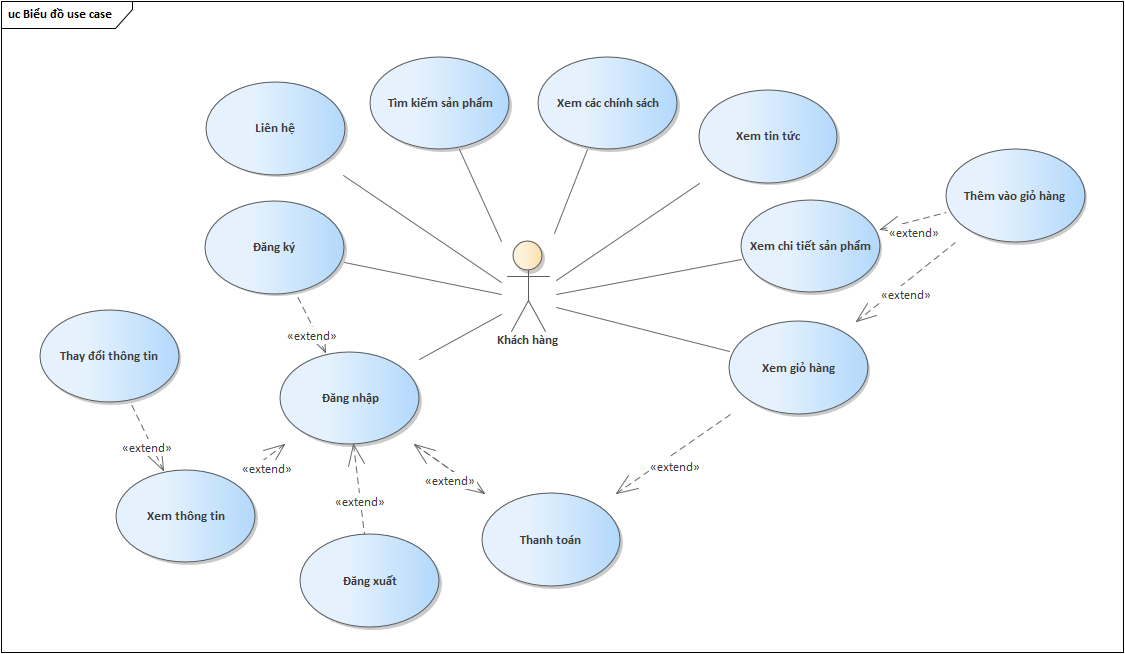
|  |  |
| --- | --- |
| **Actor** | **Vai trò** |
| Khách hàng | User là đối tượng chính tác động trực tiếp và nhiều nhất vào hệ thống.  User có thể thực hiện hầu như các chức năng của ứng dụng. |
| Quản trị | Admin là người quản trị hệ thống website |

Bảng 1. Danh sách các actor

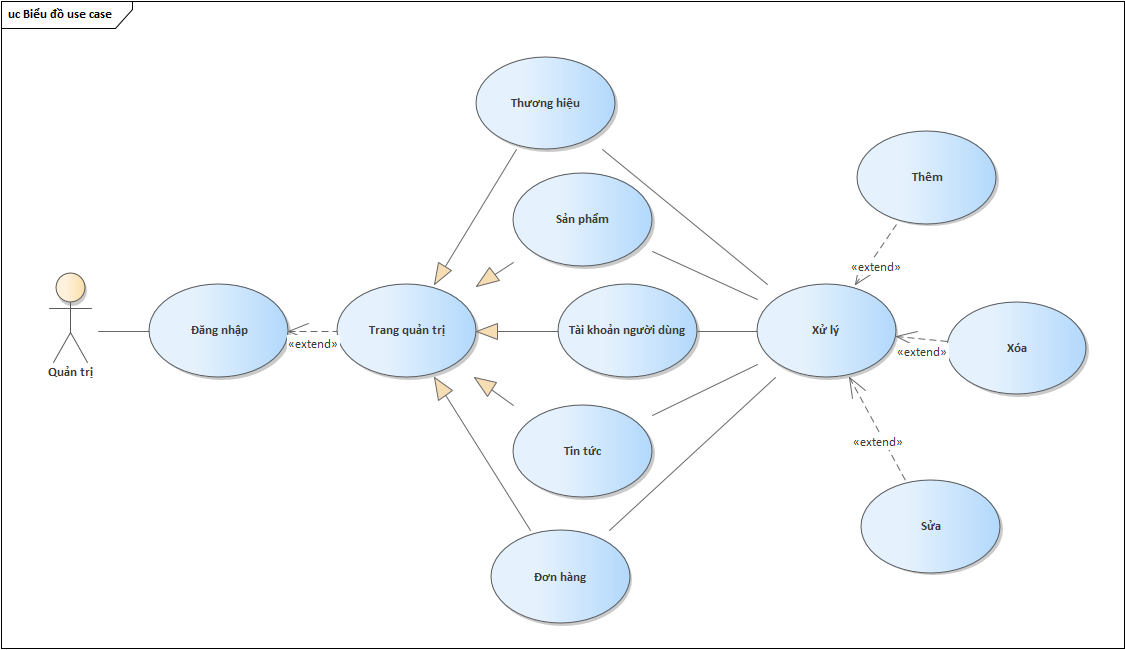
|  |  |
| --- | --- |
| **Chức năng** | **Mô tả** |
| Đăng nhập | Use case này giúp người dùng sử dụng các chức năng của hệ thống cần đến quyền truy cập. |
| Đăng ký | Đăng ký tài khoản |
| Liên hệ | Liên hệ trực tiếp với cửa hang, xem thông tin chi nhánh |
| Tìm kiếm | Use case này mô tả chức năng tìm kiếm sản phẩm của Admin, khách hàng. Chức năng này giúp cho việc tìm kiếm sản phẩm được dễ dàng hơn khi có rất nhiều sản phẩm được lưu trữ trong hệ thống, vì khi đó để tìm kiếm một sản phẩm khi muốn biết thông tin theo từng yêu cầu là không hề đơn giản. |
| Chính sách | Xem các chính sách bảo hành, đổi trả |
| Tin tức | Xem thông tin, chi tiết tin tức |
| Chi tiết sản phẩm | Use case này mô tả chức năng xem thông tin sản phẩm của người sử dụng như xem nguồn gốc sản phẩm, thông số kĩ thuật của sản phẩm… Ngoài ra, khách hàng còn có thể có những đánh giá về sản phẩm. |
| Thêm vào giỏ hàng | Use case này mô tả chức năng đặt hàng của khách hàng, khách hàng có thể thay đổi số lượng sản phẩm theo ý muốn. Có thể hủy bỏ việc đặt hàng nếu thay đổi ý định. |
| Xem giỏ hàng | Chức năng của giỏ hàng là đựng những mặt hàng mà khách hàng chọn. |
| Thanh toán | Thanh toán khi muốn mua 1 sản phẩm |
| Thay đổi thông tin | Thay đổi mật khẩu tài khoản |
| Xem thông tin | Xem thông tin tài khoản, thông tin sản phẩm đã đặt |
| Đăng xuất | Đăng xuất ra khỏi hệ thống |
| Quản lý sản phẩm | Use case này mô tả chức năng tìm kiếm sản phẩm của Admin, khách hàng. Chức năng này giúp cho việc tìm kiếm sản phẩm được dễ dàng hơn khi có rất nhiều sản phẩm được lưu trữ trong hệ thống, vì khi đó để tìm kiếm một sản phẩm khi muốn biết thông tin theo từng yêu cầu là không hề đơn giản. |
| Quản lý đơn hàng | Admin quản lý đơn hàng khi khách hàng đặt mua sản phẩm của cửa hàng. |

Bảng 2: Danh sách use case.

*Dưới đây là sơ đồ mối quan hệ giữa các actor với use case:*

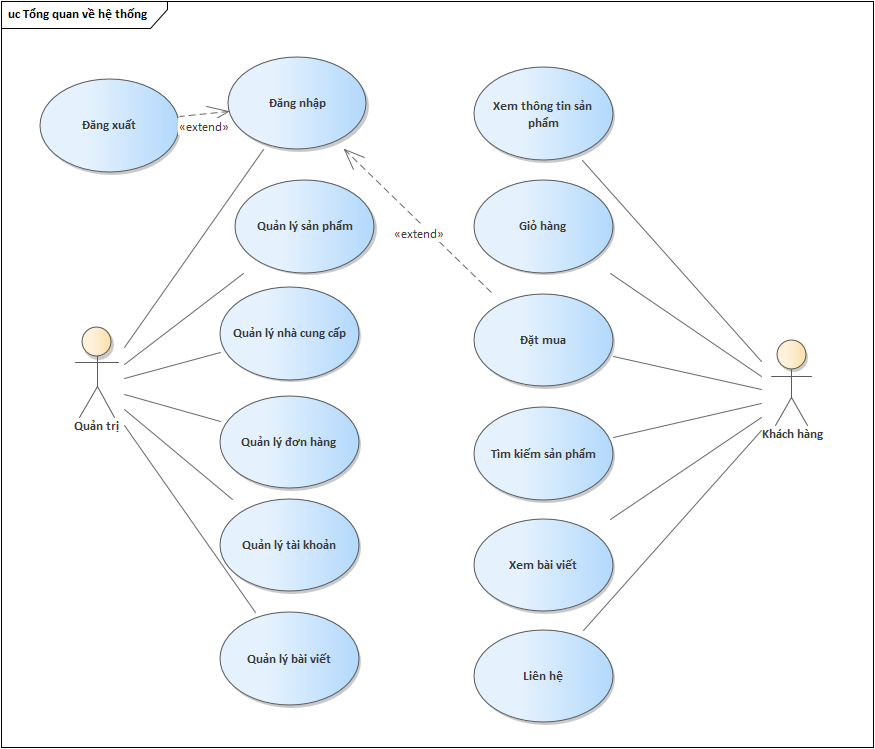


Hình 11: Biểu đồ use case người dùng



Hình 12: Biểu đồ use case quản trị

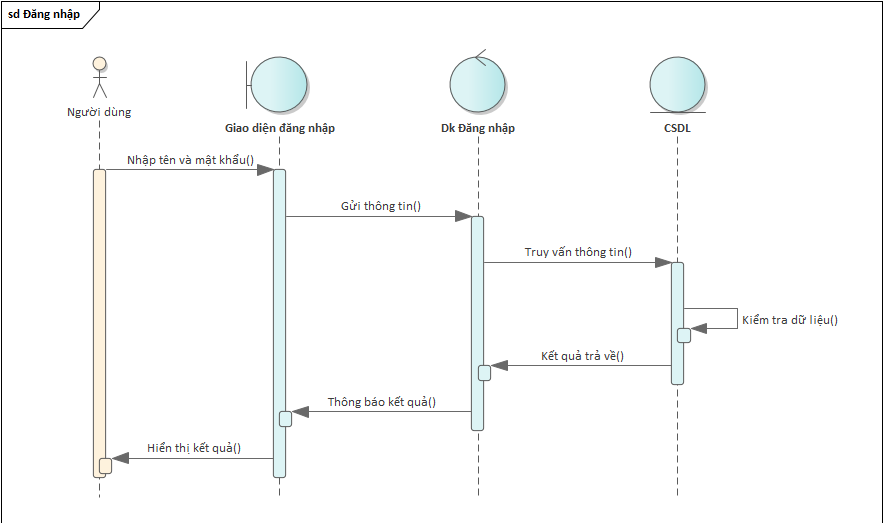
### **Biểu đồ tổng quan về hệ thống**



Hình 13 : Biểu đồ tổng quan về hệ thống

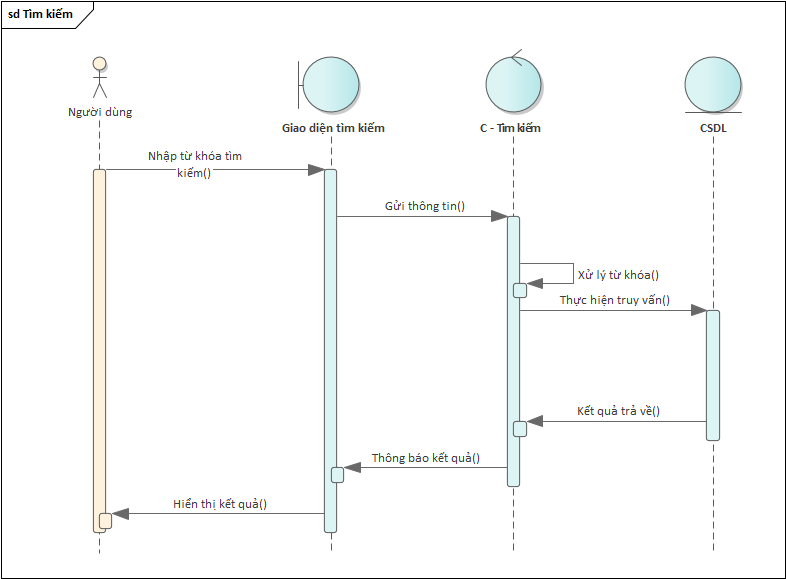
### **Biểu đồ tuần tự**

* 1. *Đăng nhập*



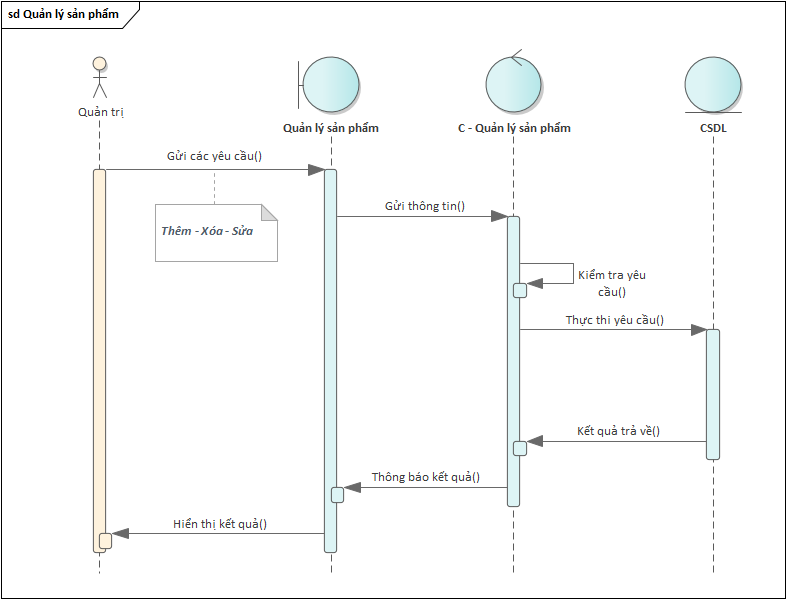
Hình 14: Biểu đồ tuần tự chức năng đăng nhập

* 1. *Tìm kiếm*



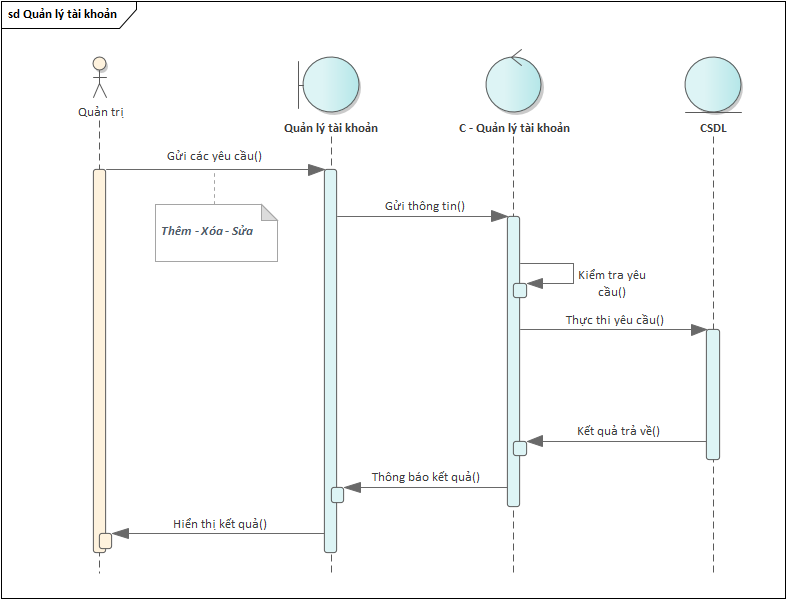
Hình 15: Biểu đồ tuần tự chức năng tìm kiếm

* 1. *Quản lý sản phẩm*



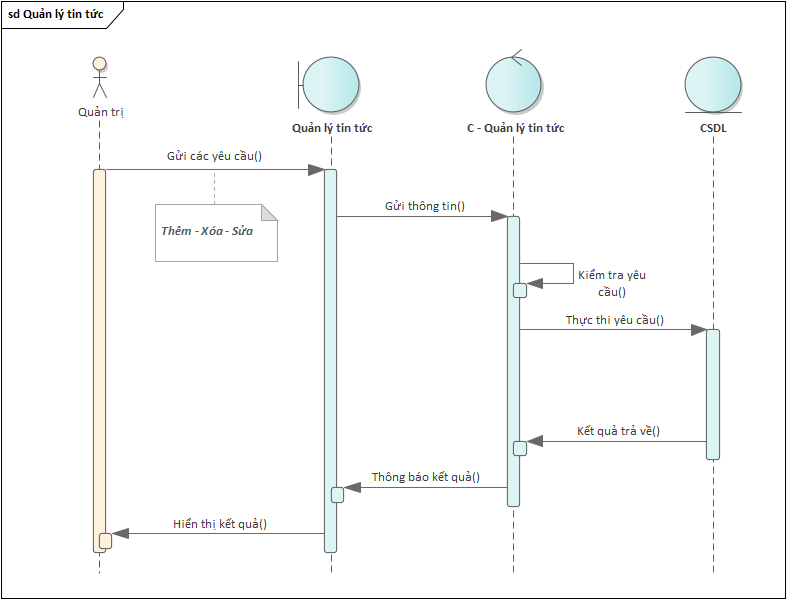
Hình 16: Biểu đồ tuần tự chức năng quản lý sản phẩm

* 1. *Quản lý tài khoản*



Hình 17: Biểu đồ tuần tự chức năng quản lý tài khoản

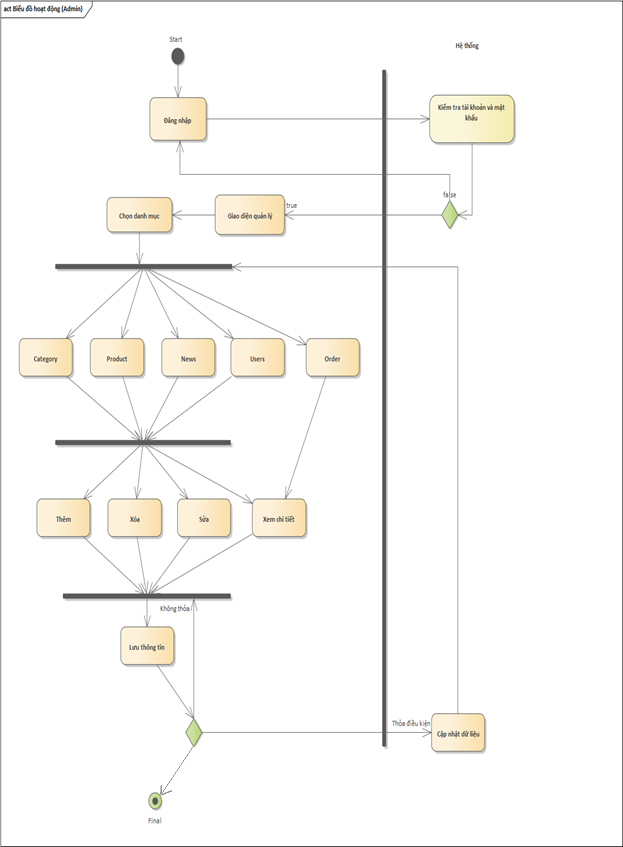
* 1. *Quản lý tin tức*



Hình 18: Biểu đồ tuần tự chức năng quản lý tin tức

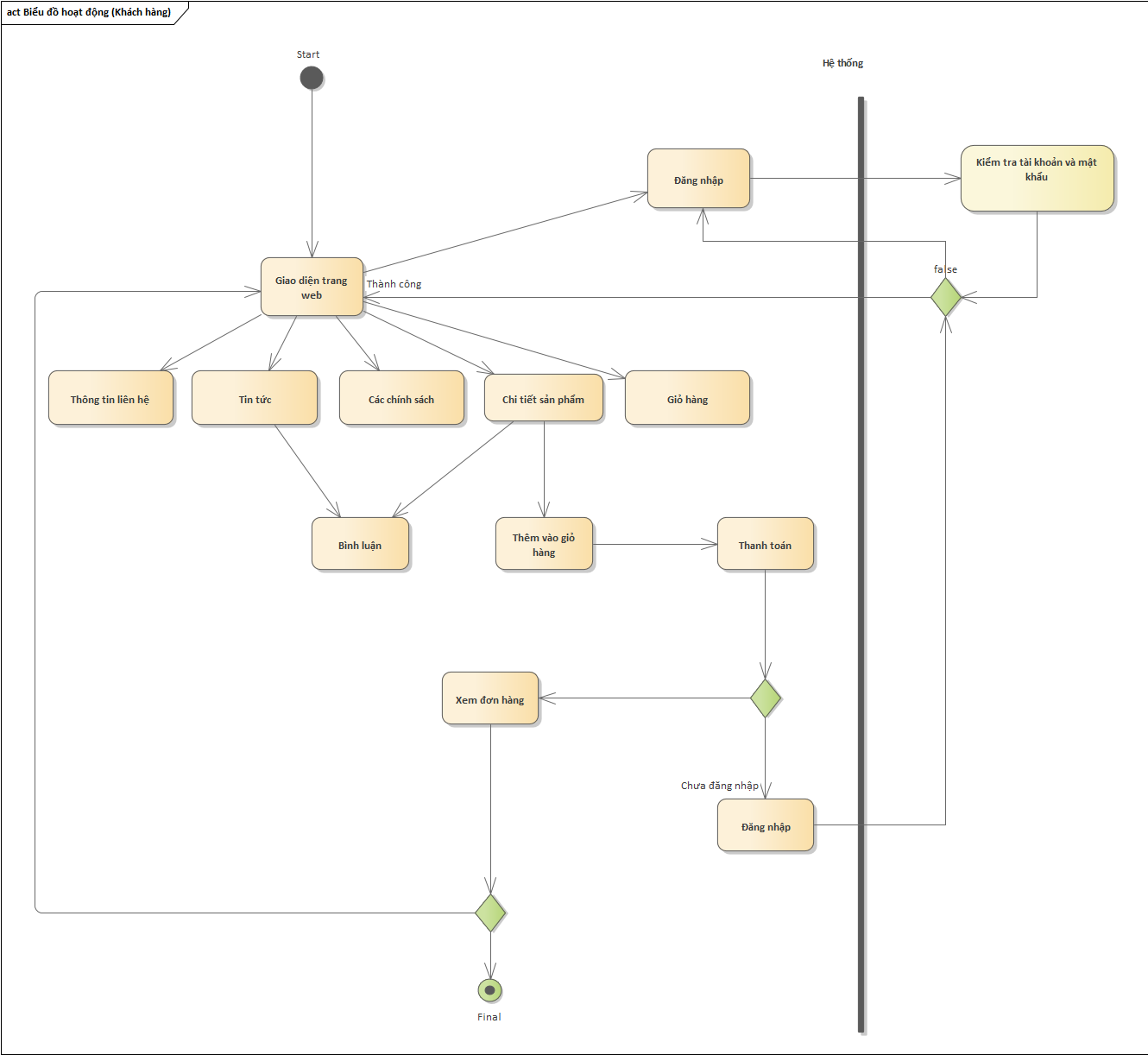
### **Biểu đồ hoạt động**

* 1. *Admin*



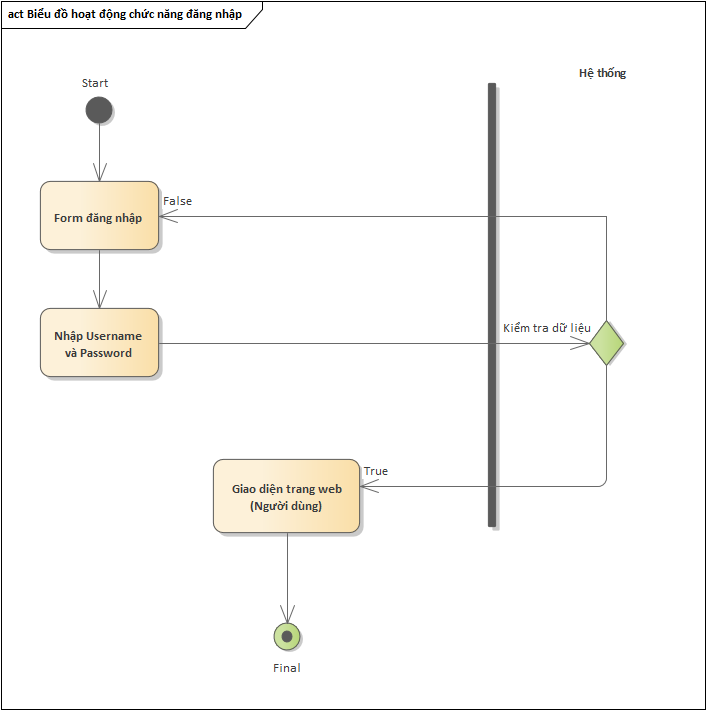
Hình 19: Biểu đồ hoạt động (Admin)

* 1. *Người dùng*



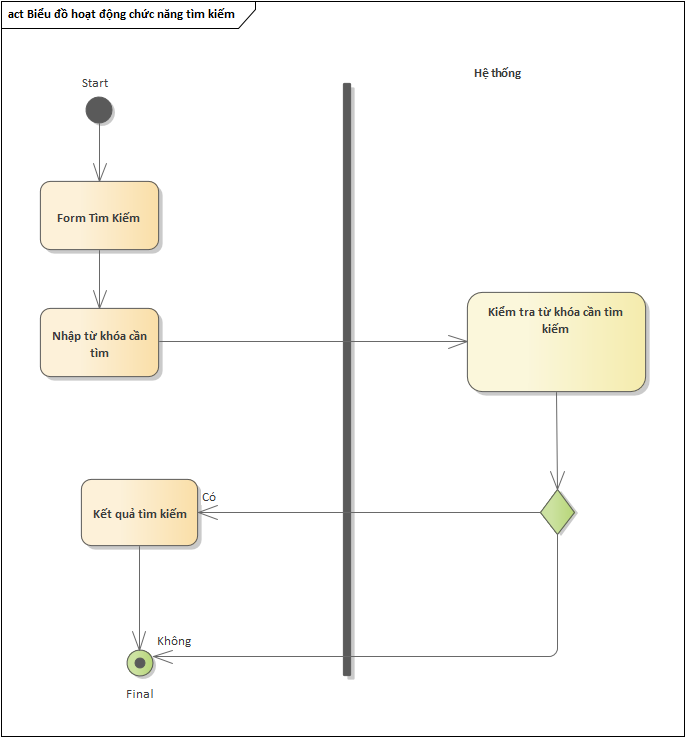
Hình 20: Biểu đồ hoạt động(Người dùng)

* 1. *Biểu đồ hoạt động chức năng đăng nhập*



Hình 21: Biểu đồ hoạt động chức năng đăng nhập

* 1. *Biểu đồ hoạt động chức năng tìm kiếm*



Hình 22: Biểu đồ hoạt động chức năng tìm kiếm

## CƠ SỞ DỮ LIỆU

### **Các bảng dữ liệu**

* 1. *Category*

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Integer | (PK) |
| Name | Varchar (255) | Tên nhãn hàng |
| Title | Varchar (255) | Link nhãn hàng |
| Image | Varchar (255) | Logo nhãn hàng |
| Description | Varchar (255) | Mô tả |

Bảng 3: Bảng dữ liệu Category(Danh mục)

* 1. *Product*

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Integer | (PK) |
| Name | Varchar (255) | Tên sản phẩm |
| Price | Integer | Giá sản phẩm |
| Image | Varchar (255) | Hình ảnh sản phẩm |
| Description | Varchar (255) | Mô tả sản phẩm |
| Title | Varchar (225) | Tiêu đề sản phẩm |
| F-base-notes | Varchar (255) | Tình trạng sản phẩm |
| F-middle-notes | Varchar (255) | Trọng lượng sản phẩm |
| F-top-notes | Varchar (255) | Kích thước sản phẩm |
| Product-Brand | Varchar (255) | Thương hiệu sản phẩm |
| Category | Varchar (255) | Danh mục sản phẩm |
| Type | Varchar (255) | Loại sản phẩm |
| volume | Varchar (255) | Tần số quét |
| Year | Integer | Năm sản xuất |

Bảng 4: Bảng dữ liệu Product(Sản phẩm)

* + - 1. *Users*

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Long | (PK) |
| Username | Varchar (255) | Tên đăng nhập |
| Password | Varchar (255) | Mật khẩu |
| Email | Varchar (255) | Email |

Bảng 5: Bảng dữ liệu Users

* + - 1. *User\_roles*

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| User\_id | Long | Lấy id của User |
| Role\_ | Varchar (255) | Lấy id của Role |

Bảng 6: Bảng dữ liệu User\_roles

* + - 1. *Orders*

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Long | (PK) |
| Address | Varchar (255) | Địa chỉ |
| FirstName | Varchar (255) | Tên đầu của khách hàng |
| LastName | Varchar (255) | Tên cuối của khách hàng |
| Phone-number | Varchar (255) | Số điện thoại |
| Date | date | Ngày đặt |
| City | Varchar (255) | Công ty |
| Post-index | Integer | Id order |
| Total-price | Integer | Tổng giá sản phẩm |
| Email | Varchar (255) | Email khách hàng |
| User-id | Long | Id của user |

Bảng 7: Bảng dữ liệu Orders

* + - 1. News

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Long | (PK) |
| Image | Varchar (255) | Hình ảnh |
| title | Varchar (255) | Tiêu đề |
| description | Varchar (255) | Mô tả |

Bảng 8: Bảng dữ liệu tin tức

* + - 1. Brands

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Long | (PK) |
| Image | Varchar (255) | Hình ảnh |
| title | Varchar (255) | Tiêu đề |
| description | Varchar (255) | Mô tả |
| Name | Varchar (255) | Tên thương hiệu |

Bảng 9: Bảng dữ liệu thương hiệu

* + - 1. Contacts

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Id | Long | (PK) |
| FirstName | Varchar (255) | Tên đầu của khách hàng |
| LastName | Varchar (255) | Tên cuối của khách hàng |
| Phone-number | Varchar (255) | Số điện thoại |
| Date | date | Ngày liên hệ |
| Post-index | Integer | Id contact |
| Title | Varchar (255) | Tiêu đề |
| Email | Varchar (255) | Email khách hàng |
| Description | Varchar (255) | Mô tả |
| User-id | Long | Id của user |

Bảng 10: Bảng dữ liệu Contact

* + - 1. Product\_reviews

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Product\_id | bigint | Id sản phẩm |
| Review\_id | bigint | Id reivew |

Bảng 11: Bảng dữ liệu Product\_reviews

* + - 1. Reviews

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| id | bigint | Id review |
| Author | Varchar (255) | Người nhận xét |
| Date | Date | Ngày nhận xét |
| Message | Varchar (255) | Mô tả |

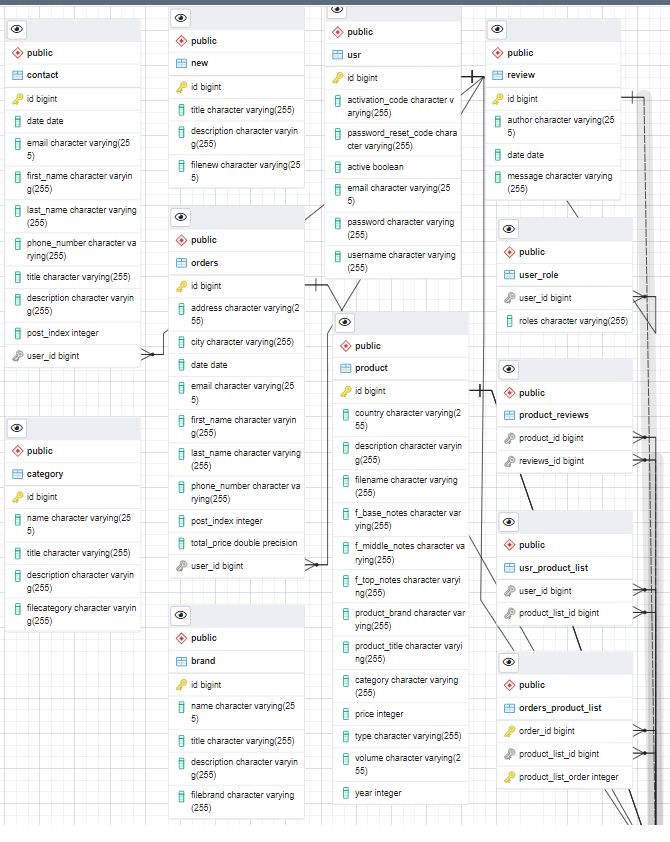
Bảng 12: Bảng dữ liệu mô tả sản phẩm

* + - 1. User\_product\_list

|  |  |  |
| --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Mô tả** |
| Product\_id | bigint | Id sản phẩm |
| user\_id | bigint | Id user |

Bảng 13: Bảng dữ liệu User product

### **Diagram Database**



Hình 23: Diagram Database

# TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

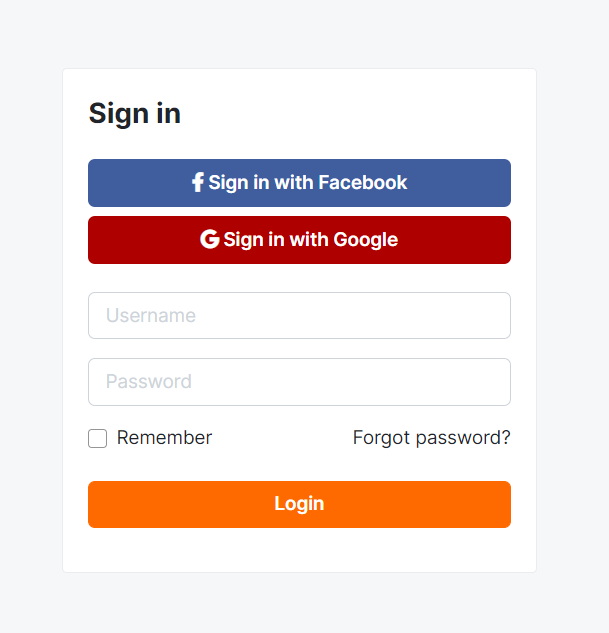
## TRIỂN KHAI

### **Cài đặt.**

Trang web sử dụng phần mềm: Eclipse IDE for Enterprise Java and Web Developers - 2022-03, Visual Studio Code, PostgreSQL.

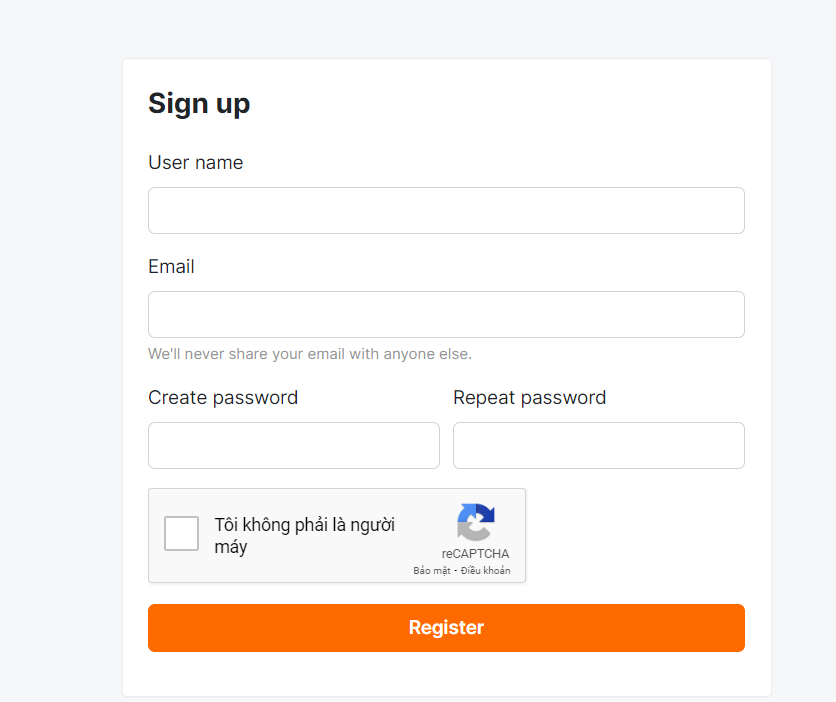
### **Một số giao diện chính của người dùng**

1. *Đăng nhập*



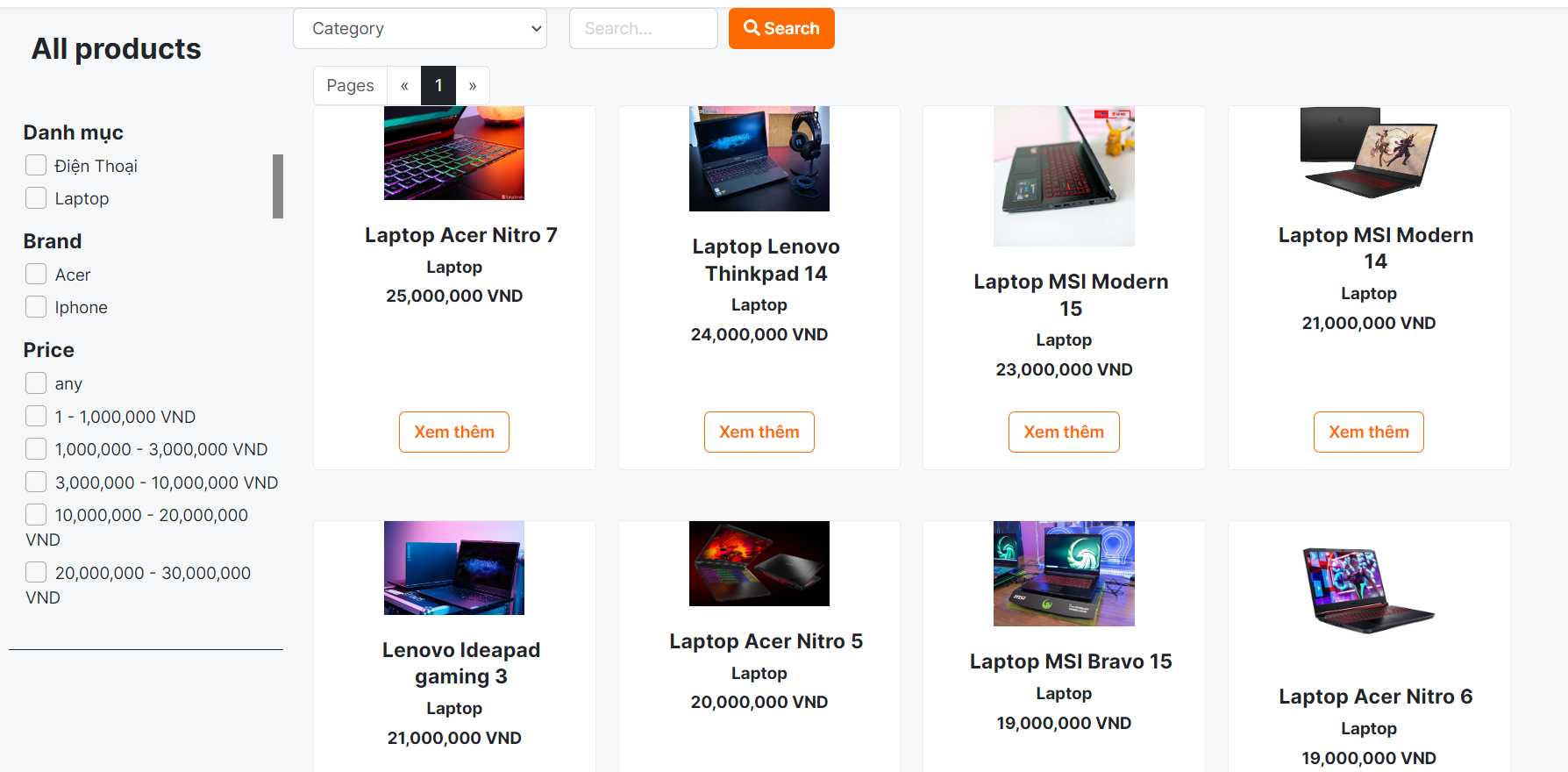
Hình 24: Giao diện đăng nhập

1. *Đăng ký*



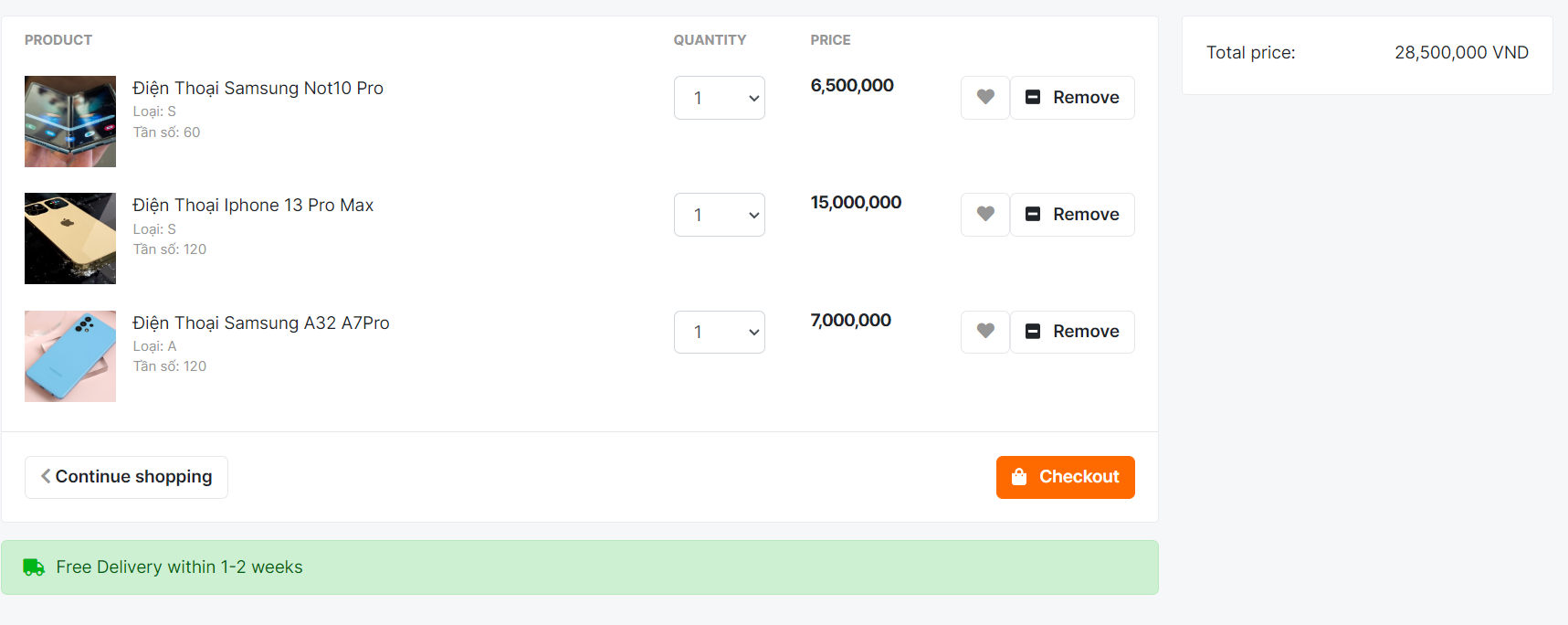
Hình 25: Giao diện đăng ký tài khoản

1. *Tìm kiếm sản phẩm*



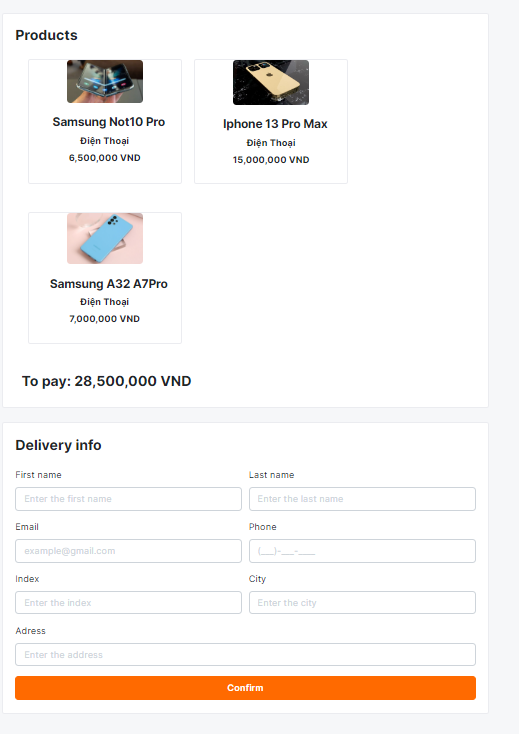
Hình 26: Giao diện tìm kiếm sản phẩm

1. *Giỏ hàng*



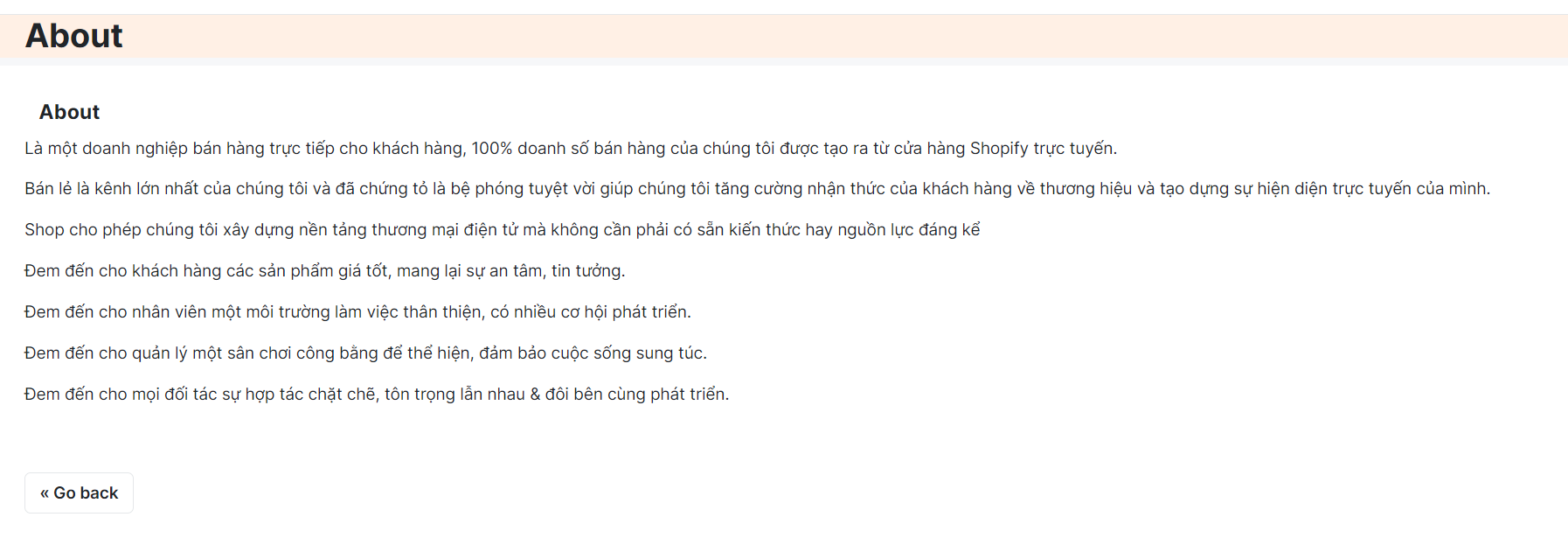
Hình 27: Giao diện giỏ hàng

1. *Thanh toán*



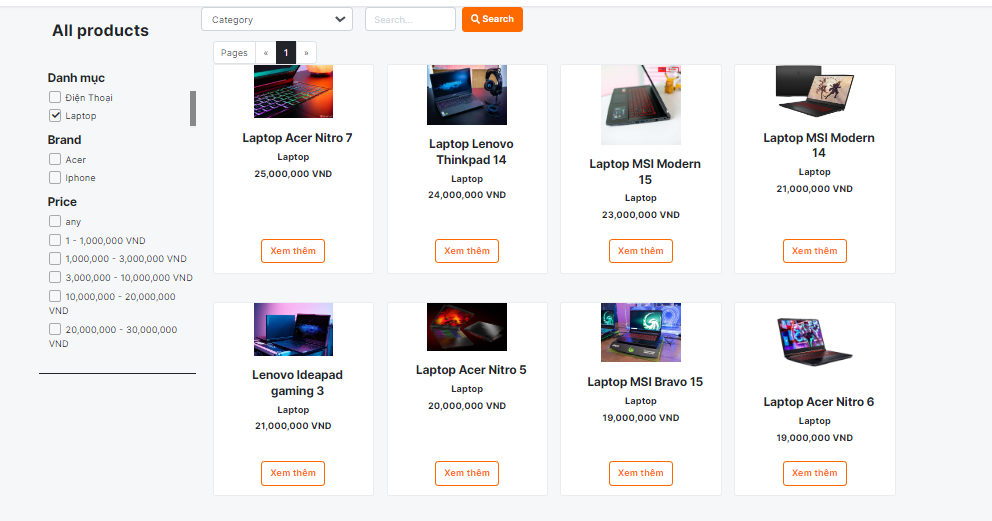
Hình 28: Giao diện thanh toán

1. *Trang địa chỉ*

**

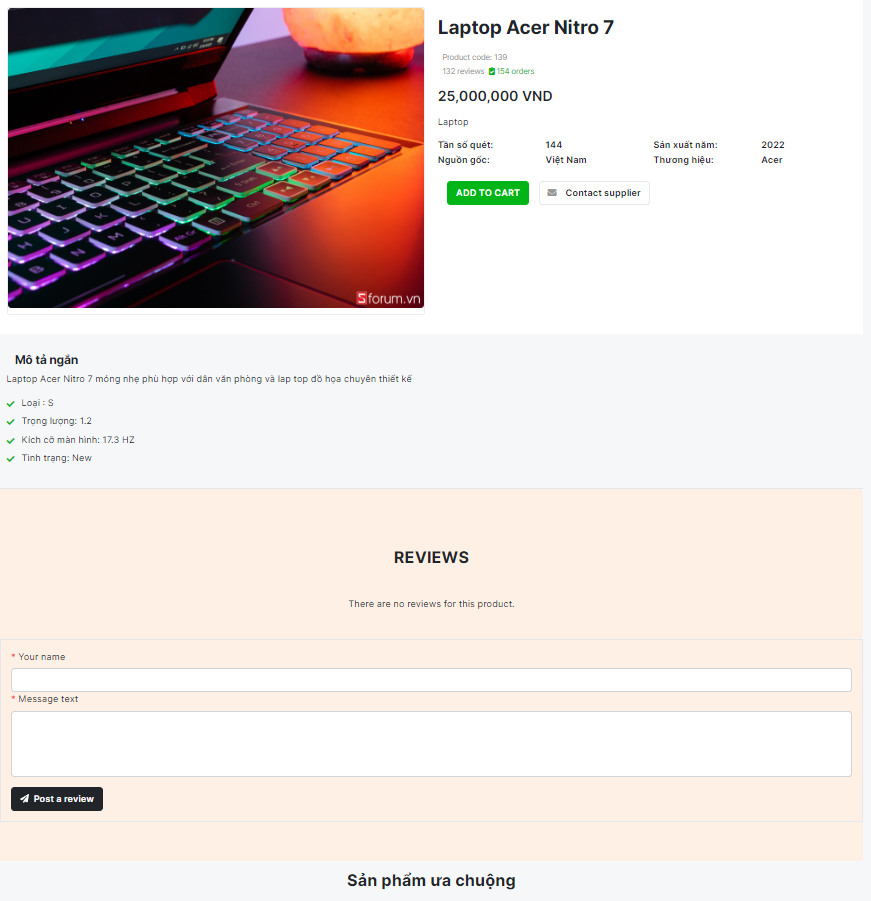
Hình 29: Giao diện thông tin cửa hàng

1. *Sản phẩm theo loại*



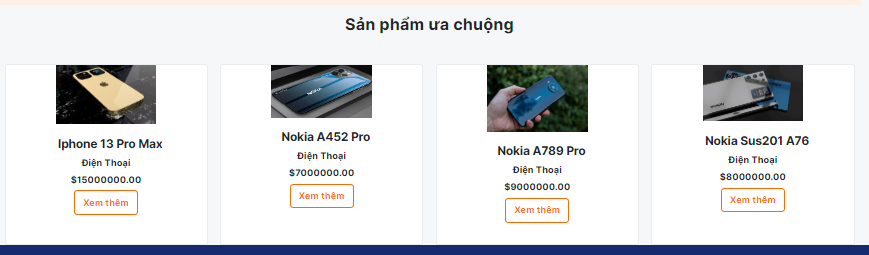
Hình 30: Giao diện sản phẩm theo loại

1. *Chi tiết sản phẩm*



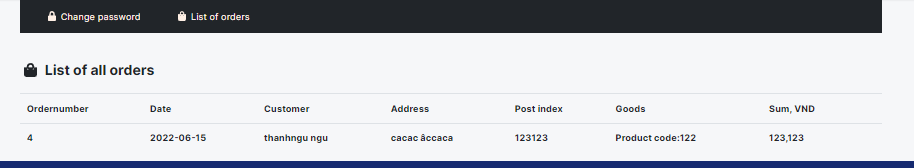
Hình 32: Giao diện chi tiết sản phẩm

1. *Sản phẩm liên quan*



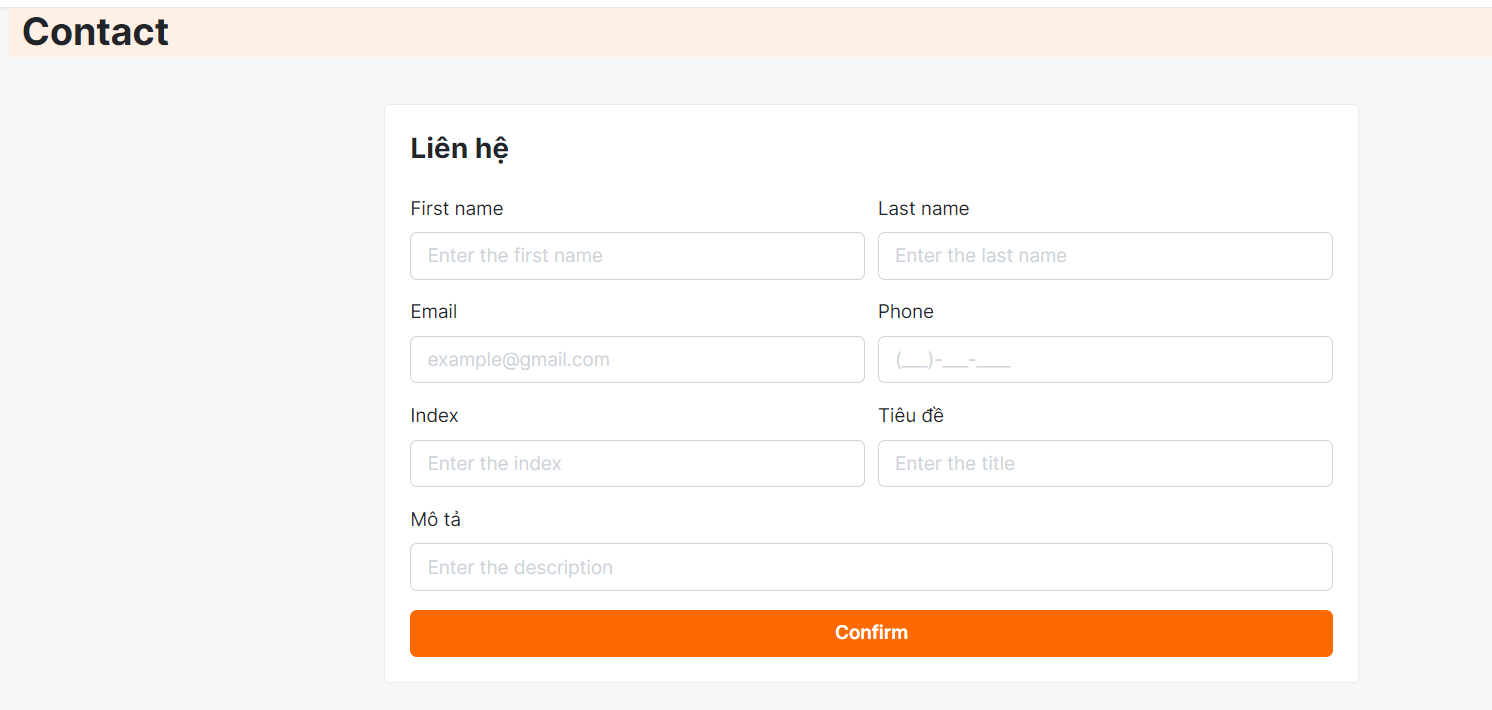
Hình 31: Giao diện sản phẩm liên quan

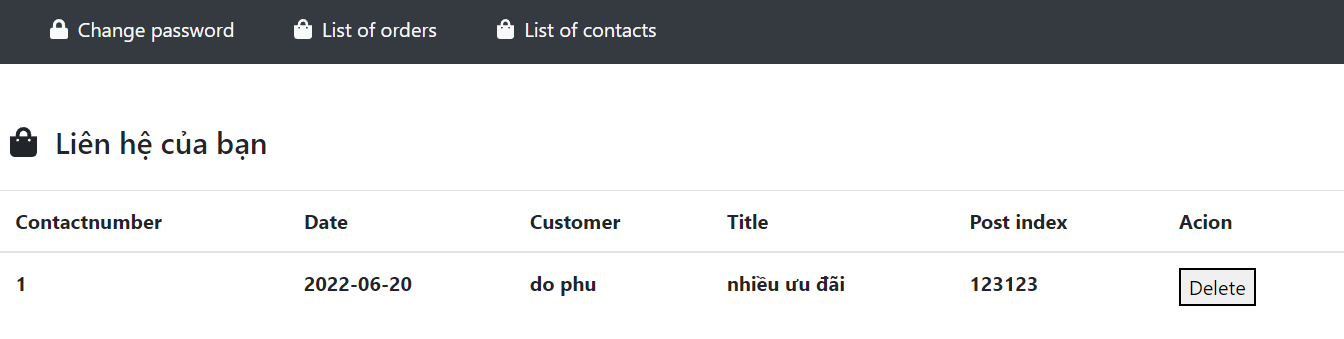
1. *Chi tiết đơn hàng đã đặt*



Hình 32: Giao diện đơn hàng đã đặt

1. *Giao diện trang liên hệ*





Hình 33:Giao diện trang liên hệ

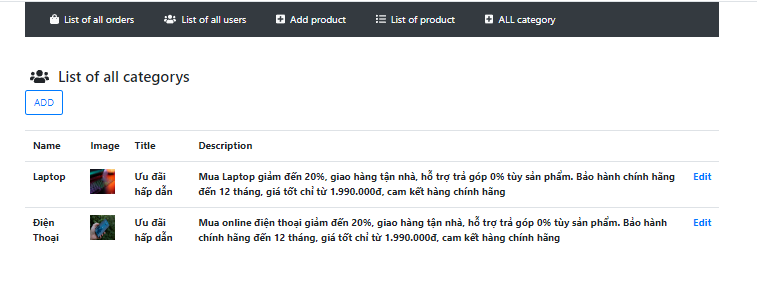
1. *Giao diện trang tin tức*

**

Hình 34: Giao diện trang tin tức.

### **Một số giao diện chính của quản trị**

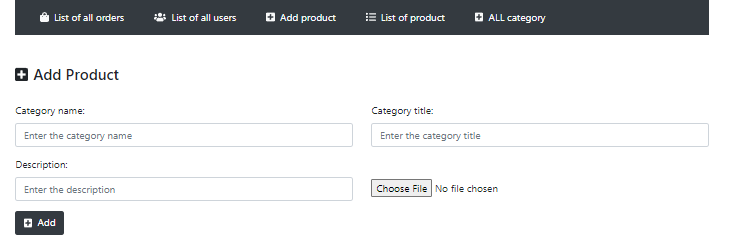
* 1. *Trang quản trị Category*



Hình 35: Giao diện quản trị (Category)

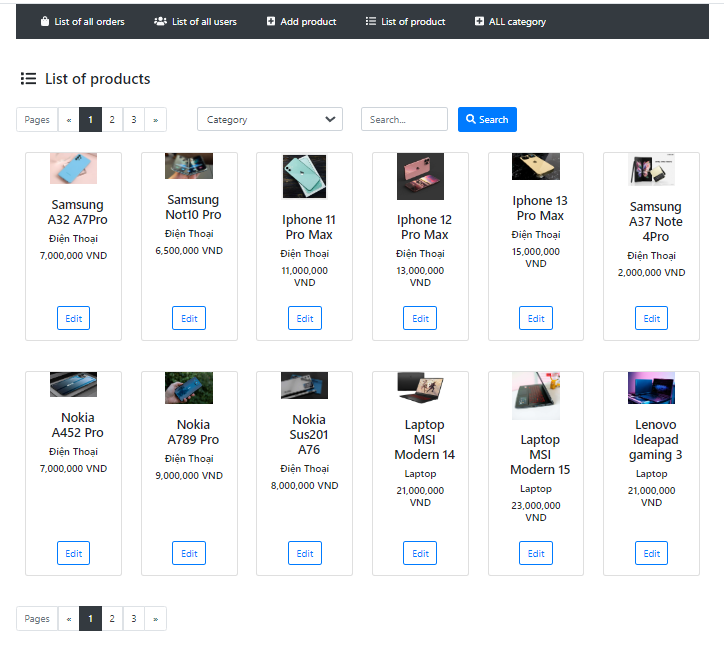
|  |  |  |
| --- | --- | --- |
| **Stt** | **Chức năng** | **Mô tả** |
| 1 | Thêm | Thêm 1 thể loại nhãn hiệu mới |
| 2 | Sửa | Sửa 1 thể loại nhãn hiệu(Link, Image) |
| 3 | Xóa | Xóa 1 thể loại ra khỏi database |

Bảng 14: Mô tả các phím chức năng (Category)



Hình 36: Form thêm Category

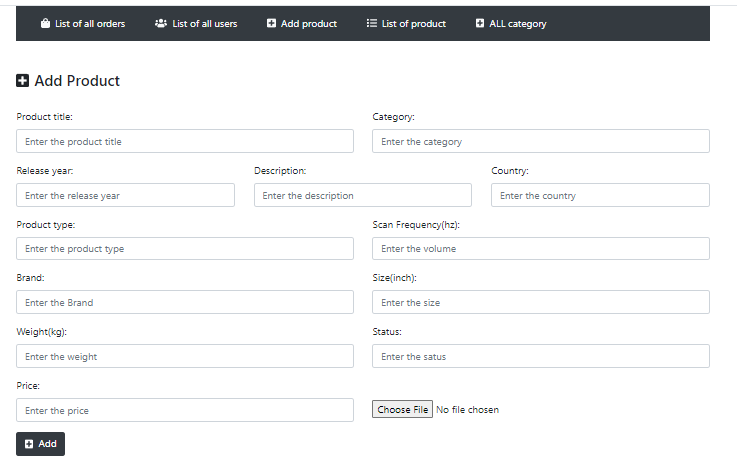
* 1. *Trang quản lý sản phẩm*



Hình 37: Giao diện quản trị (Product)

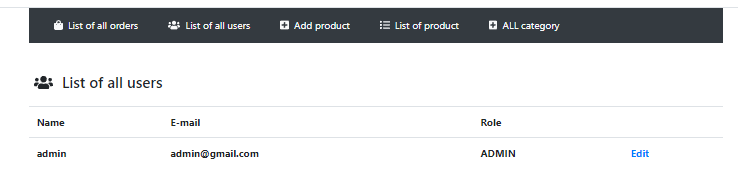
|  |  |  |
| --- | --- | --- |
| **Stt** | **Chức năng** | **Mô tả** |
| 1 | Thêm | Thêm 1 sản phẩm mới |
| 2 | Sửa | Sửa 1 sản phẩm |
| 3 | Xóa | Xóa 1 sản phẩm ra khỏi database |
| 4 | Chi tiết | Xem chi tiết sản phẩm |

Bảng 15: Mô tả các phím chức năng (Product)



Hình 38: Form thêm Product

* 1. *Trang* quản lý tài khoản khách hàng

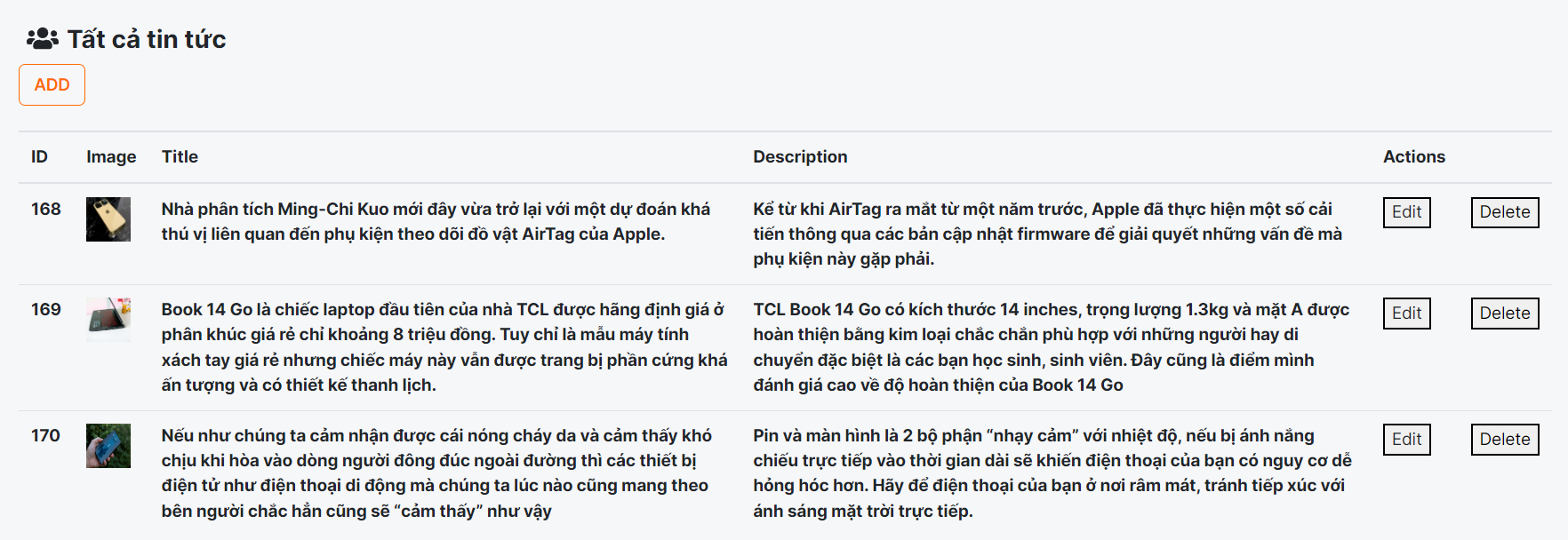


Hình 39: Giao diện quản trị (User)

|  |  |  |
| --- | --- | --- |
| **Stt** | **Chức năng** | **Mô tả** |
| 1 | Sửa | Sửa 1 người dùng |
| 2 | Xóa | Xóa 1 tài khoản người dùng |

Bảng 16: Mô tả các phím chức năng (User)

* 1. Trang quản lí tin tức

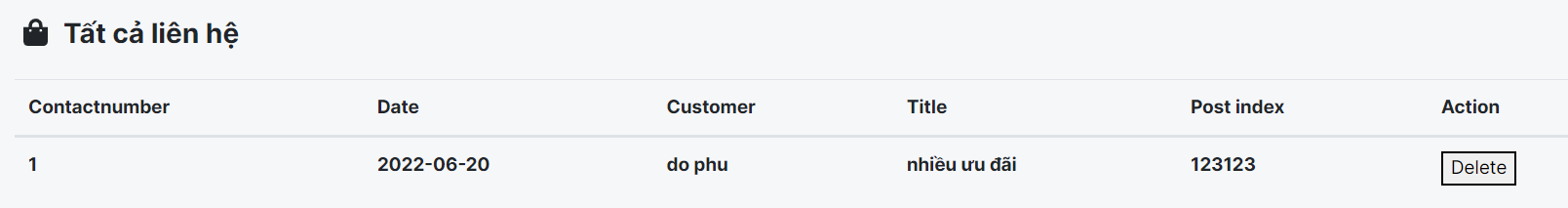


Hình 40: Giao diện quản trị (News)

|  |  |  |
| --- | --- | --- |
| **Stt** | **Chức năng** | **Mô tả** |
| 1 | Thêm | Thêm 1 tin tức |
| 2 | Sửa | Sửa 1 tin tức |
| 3 | Xóa | Xóa 1 tin tức |

Bảng 17: Mô tả các chức năng (News)

* 1. Trang quản lí liên hệ



Hình 41: Giao diện quản lí liên hệ (Contact)

|  |  |  |
| --- | --- | --- |
| **Stt** | **Chức năng** | **Mô tả** |
| 1 | Thêm | Thêm 1 tin tức |
| 2 | Sửa | Sửa 1 tin tức |
| 3 | Xóa | Xóa 1 tin tức |

Bảng 18: Bảng mô tả chức năng trang quản lí liên hệ (Contact)

## ĐÁNH GIÁ KẾT QUẢ

### **Kết quả đạt được và hướng phát triển.**

1. *Kết quả đạt được*

Nghiên cứu và trình bày một cách có hệ thống các nội dung cơ bản về Spring Boot, ReactJS, PostgreSQL và một số công nghệ, kỹ thuật khác trong việc xây dựng một ứng dụng trên nền web.

Về mặt lý thuyết:

* + - Nắm được kiến trúc tổng quan của Spring Boot cũng như các nguyên lý cơ bản và cơ chế hoạt động.
    - Nắm được mô hình web trong Spring Boot và các cơ chế để bảo mật một ứng dụng web được hỗ trợ trong module Spring Security.
    - Hiểu cơ chế giao tiếp giữa client và server trong mô hình web hiện đại theo hướng RESTful service, cách giao tiếp giữa các thành phần của một hệ thống hoặc giữa các hệ thống với nhau.
    - Là kiến thức vô cùng quan trọng để sau này có thể xin vào những công ty công nghệ

Về mặt ứng dụng:

* + - Xây dựng được ứng dụng web mà việc giao tiếp với server thông qua RESTful với sự hỗ trợ của ReactJS.
    - Xây dựng thành công ứng dụng web với đầy đủ các chức năng cơ bản.
    - Xây dựng thành công một ứng dụng thương mại điện tử trên nền web có áp dụng các công nghệ mới trong quá trình phát triển một ứng dụng Web hiện đại.

1. *Hướng phát triển*

Bên cạnh đó tiếp tục tìm hiểu các module còn lại của Spring Framework nhằm tích hợp vào hệ thống để thực hiện các chức năng đặc biệt khác.

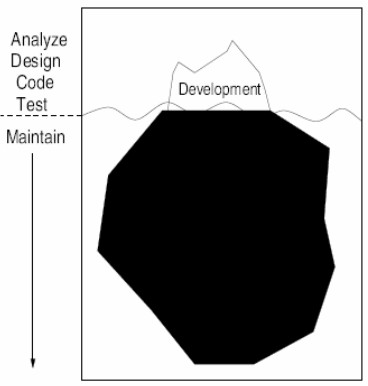
* Xem xét tình hình website hiện nay.
* Lập ra các mục tiêu.
* Khái niệm thương hiệu.
* Khái niệm người dùng mục đích.
* Tối ưu hóa cho các trang nội dung quan trọng và chủ yếu.
* Phân tích đối thủ cạnh tranh.
* Liệt kê các tài nguyên có sẵn của website.

### **Hạn chế.**

Các chức năng cơ bản của ứng dụng đã hoàn thành tốt, tuy nhiên do yêu cầu và phạm vi đề tài chỉ dừng lại ở mức demo công nghệ nên ứng dụng chưa hấp dẫn.

Trang web chưa được hoàn thiện 1 cách toàn diện nhất, còn 1 số chức năng và ràng buộc chưa được chặt chẽ.

Hạn chế về vật chất, không có host trên internet có cài đặt đầy đủ các phần mềm cần thiết tạo khó khăn trong việc thực hiện demo chương trình trên internet. Vấn đề triển khai những công nghệ nêu trên không khó về mặt kỹ thuật nhưng phải xây dựng một cơ sở hạ tầng đầy đủ và tốt để có thể thực thi ứng dụng.



3

3

5

7

15

67

Hình 42: Chi phí cho các pha phát triển một hệ thống

Xđ yêu cầu Đặt tả Thiết kế

Lập trình Kiểm thử Bảo trì

# TÀI LIỆU THAM KHẢO

[1] React Tutorial. [Trực tuyến]. Địa chỉ: <https://www.w3schools.com/react/>. [Truy cập 20/04/2022]

[2] CSS Tutorial. [Trực tuyến]. Địa chỉ: https://www.w3schools.com/css/. [Truy cập 29/05/2022]

[3] HTML Tutorial. [Trực tuyến]. Địa chỉ: https://www.w3schools.com/html/. [Truy cập 29/05/2022]

[4] JavaScript Tutorial. [Trực tuyến]. Địa chỉ: https://www.w3schools.com/js/. [Truy cập 29/05/2022]

[5] Spring Boot. [Trực tuyến]. Địa chỉ: https://spring.io/projects/spring-boot/. [Truy cập 07/06/2022]

[6] Spring Security. [Trực tuyến]. Địa chỉ: https://spring.io/projects/spring-security/. [Truy cập 10/06/2022]

[7] Font Awesome. [Trực tuyến]. Địa chỉ: https://fontawesome.com/. [Truy cập 11/06/2022]

[8] Hibernate Tutorial. [Trực tuyến]. Địa chỉ: https://hibernate.org/. [Truy cập 07/06/2022]