TRƯỜNG ĐẠI HỌC CÔNG NGHỆ SÀI GÒN

**KHOA CÔNG NGHỆ THÔNG TIN**

---oOo---

**LUẬN VĂN TỐT NGHIỆP**

*Tên đề tài:*

TÌM HIỂU ASP.NET FRAMWORK

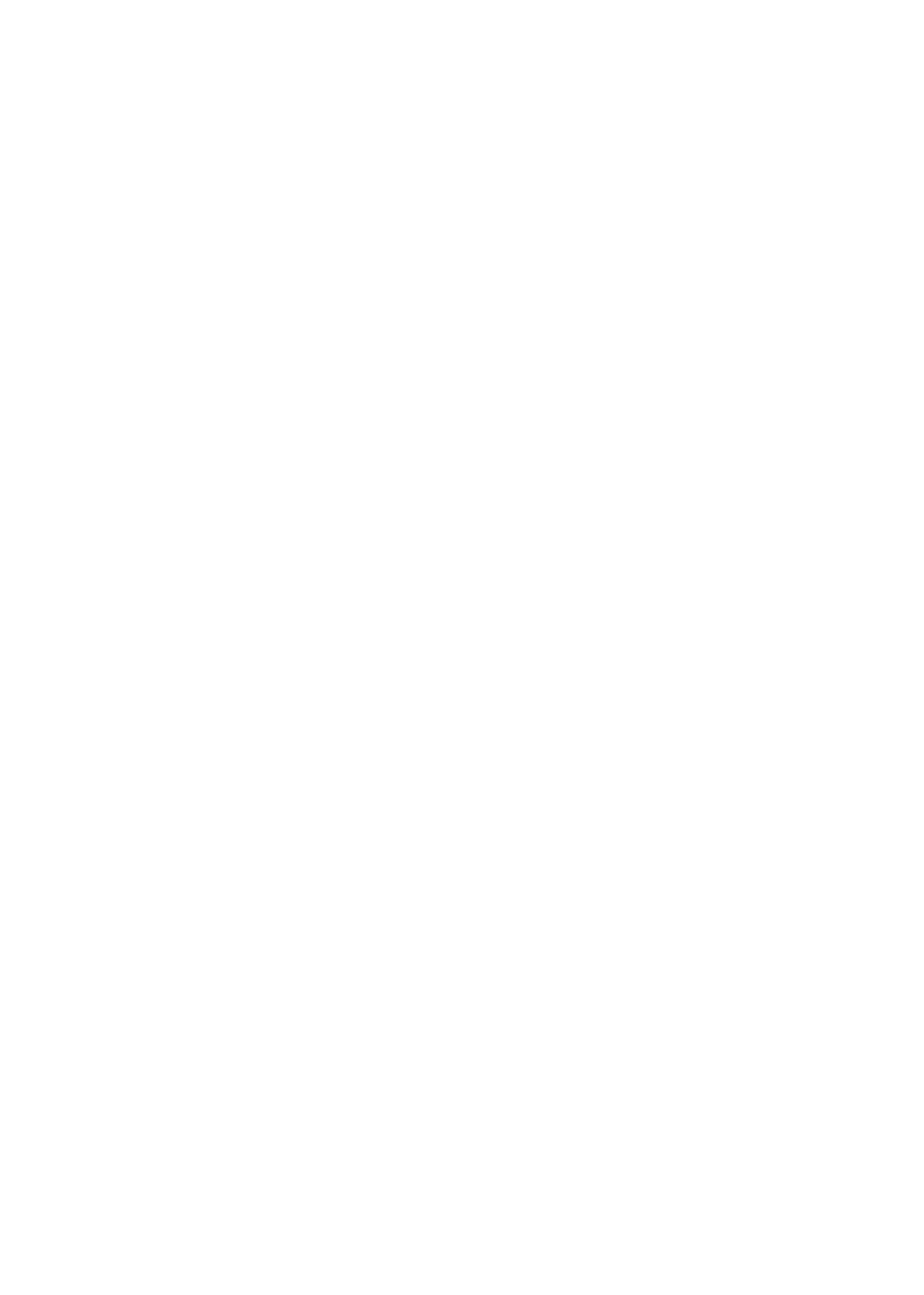
VÀ ỨNG DỤNG VÀO WEBSITE BÁN ĐIỆN THOẠI

Người hướng dẫn: **ThS. NGUYỄN TRỌNG NGHĨA**

Sinh viên thực hiện: **TẠ VĂN HOÀNG HÊN**

TP HỒ CHÍ MINH – NĂM 2019

**LỜI CẢM ƠN**



Luận văn này là tất cả sự say mê và tự tin cho em tìm hiểu về ngôn ngữ mới. Từ nền tảng mà Thầy Cô Khoa Công Nghệ Thông Tin Trường Đại Học Công Nghệ Sài Gòn đã giảng dạy tận tình và đã trang bị những kiến thức cần thiết giúp em tự tin đến với ngôn ngữ mới và từ đó nhận ra có nhiều vấn đề mà em chưa biết. Vì vậy, đây cũng là thử thách rất lớn với em. Chúng em còn phải cố gắng học hỏi thật nhiều từ các thầy cô, bạn bè, thông tin sách báo và các bậc đi trước…

Em xin chân thành cảm ơn tất cả Thầy Cô trường Đại Học Công Nghệ Sài Gòn trong thời gian qua đã truyền đạt cho em những kiến thức cũng như kinh nghiệm quý báu trong suốt quá trình học tập. Đặc biệt, em xin chân thành cảm ơn sự quan tâm, chỉ dạy tận tình của thầy Nguyễn Trọng Nghĩa trong suốt thời gian phụ trách hướng dẫn,

Cuối cùng tôi xin cảm ơn tất cả những người bạn đã luôn ở bên cạnh chúng tôi đã động viên, ủng hộ và đóng góp ý kiến để tôi hoàn thành luận văn tốt nghiệp.

Xin chân thành cảm ơn!

Tp.HCM Ngày 15 tháng 12 năm 2019

MỤC LỤC

[Chương 1. GIỚI THIỆU 1](#_Toc14946387)

[1.1 ĐẶT VẤN ĐỀ 1](#_Toc14946388)

[1.2 MỤC TIÊU ĐỀ TÀI 1](#_Toc14946389)

[1.3 PHƯƠNG PHÁP NGHIÊN CỨU 2](#_Toc14946390)

[Chương 2. MÔ TẢ NGHIỆP VỤ 3](#_Toc14946391)

[2.1 TỔNG QUAN 3](#_Toc14946392)

[2.2 QUY TRÌNH 4](#_Toc14946393)

[Chương 3. GIẢI PHÁP ĐỀ XUẤT 8](#_Toc14946394)

[3.1 KIẾN TRÚC TỔNG THỂ 8](#_Toc14946395)

[3.2 GIẢI PHÁP CÔNG NGHỆ 9](#_Toc14946396)

[3.3 SPRING FRAMEWORK 11](#_Toc14946397)

[Chương 4. THÀNH PHẦN TÁC NHÂN 24](#_Toc14946398)

[4.1 ĐẶC TẢ ACTOR 24](#_Toc14946399)

[4.2 SƠ ĐỒ USE-CASE TỔNG QUÁT 27](#_Toc14946400)

[4.3 ĐẶC TẢ USE CASE CHI TIẾT 29](#_Toc14946401)

[Chương 5. THÀNH PHẦN XỬ LÝ 44](#_Toc14946402)

[5.1 SƠ ĐỒ TUẦN TỰ 44](#_Toc14946403)

[Chương 6. THÀNH PHẦN DỮ LIỆU 53](#_Toc14946404)

[6.1 PHÂN TÍCH Ở MỨC QUAN NIỆM 53](#_Toc14946405)

[6.2 MÔ TẢ CÁC RÀNG BUỘC DỮ LIỆU 56](#_Toc14946406)

[6.3 SƠ ĐỒ HOẠT ĐỘNG 56](#_Toc14946407)

[Chương 7. THÀNH PHẦN GIAO DIỆN 62](#_Toc14946408)

[7.1 CÁC MÀN HÌNH NHẬP LIỆU 62](#_Toc14946409)

[7.2 TẠO BÁO CÁO 68](#_Toc14946410)

[7.3 TẠO MENU SẢN PHẨM 69](#_Toc14946411)

[Tài liệu tham khảo 70](#_Toc14946412)

MỤC LỤC CÁC HÌNH VẼ

[Hình 2‑1: Mô hình quy trình xem và đặt hàng của Khách hàng 4](#_Toc14946475)

[Hình 2‑2: Mô hình mua hàng 5](#_Toc14946476)

[Hình 2‑3: Quy trình đăng nhập 6](#_Toc14946477)

Hình 2-4: Quy trình tìm kiếm đơn hàng 7

[Hình 3‑1: Kiến trúc tổng thể 8](#_Toc14946478)

[Hình 3‑2: Mô hình cấu trúc trong Spring framework 14](#_Toc14946479)

[Hình 3‑3: Mô hình MVC 16](#_Toc14946480)

[Hình 3‑4: Sơ đồ xử lý request 17](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946481)

[Hình 3‑5: RequestMapping ánh xạ đường dẫn 18](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946482)

[Hình 3‑6: Phương thức GET/POST 19](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946483)

[Hình 3‑7: Phương thức trong HandlerInterceptors 20](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946484)

[Hình 4‑1: Use case Khách hàng 27](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946485)

[Hình 4‑2: Use case Khách vãng lai 28](#_Toc14946486)

[Hình 4‑3: Use case quản lí hệ thống 28](#_Toc14946487)

[Hình 4‑4: Sơ đồ usecase quản lý hãng 29](#_Toc14946488)

[Hình 4‑5: Sơ đồ usecase quản lý hóa đơn 31](#_Toc14946489)

[Hình 4‑6: Sơ đồ usecase quản lý sản phẩm 33](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946490)

[Hình 4‑7: Use case tìm kiếm sản phẩm 37](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946491)

[Hình 4‑8: Use case đặt hàng 39](#_Toc14946492)

[Hình 4‑9: Use case đăng nhập 41](#_Toc14946493)

[Hình 5‑1: Sơ đồ thêm sản phẩm vào giỏ hàng 44](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946494)

[Hình 5‑2: Xem giỏ hàng 45](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946495)

[Hình 5‑3: Xóa sản phẩm khỏi giỏ hàng 45](#_Toc14946496)

[Hình 5‑4: Chỉnh sửa thông tin giỏ hàng 46](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946497)

[Hình 5‑5: Tìm kiếm sản phẩm 46](#_Toc14946498)

[Hình 5‑6: Sơ đồ đăng ký thành viên của Khách hàng 47](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946499)

[Hình 5‑7: Chỉnh sửa thông tin cá nhân của Khách hàng 47](#_Toc14946500)

[Hình 5‑8: Đăng nhập thành viên của Khách hàng 48](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946501)

[Hình 5‑9: Xem sản phẩm của Khách hàng 48](#_Toc14946502)

[Hình 5‑10: Xem chi tiết sản phẩm 49](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946503)

[Hình 5‑11: Thêm danh mục của người quản lý 49](#_Toc14946504)

[Hình 5‑12: Xóa danh mục của người quản lý 50](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946505)

[Hình 5‑13: Sửa danh mục của người quản lý 50](#_Toc14946506)

[Hình 5‑14: Thêm sản phẩm mới 51](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946507)

[Hình 5‑15: Xóa sản phẩm trong trang quản lý 51](#_Toc14946508)

[Hình 5‑16: Sửa sản phẩm của người quản lý 52](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946509)

[Hình 6‑1: Sơ đồ Cơ sở dữ liệu 53](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946510)

[Hình 6‑2: Mô tả hoạt động đăng nhập 56](#_Toc14946511)

[Hình 6‑3: Mô tả hoạt động xóa sản phẩm 57](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946512)

[Hình 6‑4: Mô tả hoạt động thêm sản phẩm 57](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946513)

[Hình 6‑5: Mô tả hoạt động sửa sản phẩm 58](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946514)

[Hình 6‑6: Mô tả hoạt động tìm sản phẩm 58](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946515)

[Hình 6‑7: Mô tả hoạt động tra cứu thành viên 59](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946516)

[Hình 6‑8: Mô tả hoạt động cập nhật thành viên 59](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946517)

[Hình 6‑9: Mô tả hoạt động tìm đơn đặt hàng 60](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946518)

[Hình 6‑10: Mô tả hoạt động xóa thành viên 60](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946519)

[Hình 6‑11: Mô tả hoạt động xử lý đơn đặt hàng 61](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946520)

[Hình 6‑12: Mô tả hoạt động xem chi tiết đơn đặt hàng 61](#_Toc14946521)

[Hình 7‑1: Màn hình đăng ký 62](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946522)

[Hình 7‑2: Màn hình tìm kiếm sản phẩm 63](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946523)

[Hình 7‑3: Màn hình đăng nhập 63](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946524)

[Hình 7‑4: Màn hình hiển thị những sản phẩm trong giỏ hàng 64](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946525)

[Hình 7‑5: Màn hình hiển thị đặt hàng sau khi mua hàng 64](#_Toc14946526)

[Hình 7‑6: Màn hình đăng nhập trang quản trị 65](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946527)

[Hình 7‑7: Màn hình hiển thị quản lý danh mục 65](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946528)

[Hình 7‑8: Màn hình hiển thị trang sửa loại sản phẩm 66](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946529)

[Hình 7‑9: Màn hình hiển thị trang quản lý sản phẩm 66](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946530)

[Hình 7‑10: Màn hình nhập liệu Khách hàng 67](#_Toc14946531)

[Hình 7‑11: Màn hình chỉnh sửa sản phẩm 67](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946532)

[Hình 7‑12: Màn hình chỉnh sửa Khách hàng 68](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946533)

[Hình 7‑13: Báo cáo doanh thu theo loại 68](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946534)

[Hình 7‑14:Báo cáo doanh thu theo tháng 69](file:///C:\Users\truon\Downloads\doanpt%20(1).doc#_Toc14946535)

[Hình 7‑15: Menu sản phẩm trang Khách hàng 69](#_Toc14946536)

# GIỚI THIỆU

## ĐẶT VẤN ĐỀ

Ngày nay công nghệ thông tin đã có những bước phát triển mạnh mẽ theo cả chiều rộng và chiều sâu. Máy tính điện tử không còn là một thứ phương tiện quý hiếm mà đang ngày càng trở thành một công cụ làm việc và giải trí thông dụng của con người không chỉ ở công sở mà ngay cả trong gia đình. Đứng trước vai trò của thông tin hoạt động cạnh tranh gay gắt, các tổ chức và các doanh nghiệp đều tìm mọi biện pháp để xây dựng và hoàn thiện hệ thống thông tin của mình nhằm tin học hóa các hoạt động tác vụ của đơn vị.

Hiện nay các công ty tin học hàng đầu thế giới không ngừng đầu tư và cải thiện các giải pháp cũng như các sản phẩm nhằm cho phép tiến hành thương mại hóa trên Internet. Thông qua các sản phẩm và công nghệ này, chúng ta dễ nhận ra tầm quan trọng và tính tất yếu của thương mại điện tử. Với những thao tác đơn giản trên máy có nối mạng internet bạn sẽ có trong tầm tay những gì mà mình cần mà không phải mất nhiều thời gian. Bạn chỉ cần vào các trang dịch vụ thương mại điện tử, làm theo hướng dẫn và click vào những gì bạn cần. Các nhà dịch vụ sẽ mang đến tận nhà cho bạn.

Ở Việt Nam cũng có rất nhiều doanh nghiệp đang tiến hành thương mại hóa trên Internet nhưng do những khó khăn về cơ sở hạ tầng như viễn thông chưa phát triển mạnh, các dịch vụ thanh toán điện tử qua ngân hàng chưa phổ biến nên chỉ dừng lại ở mức độ giới thiệu sản phẩm và tiếp cận đơn hàng thông qua web. Để tiếp cận và đóng góp đẩy mạnh sự phổ biến của thương mại điện tử ở việt nam, em đã tìm hiểu và cài đặt ‘Website bán điện thoại trực tuyến’.

## MỤC TIÊU ĐỀ TÀI

Mục tiêu chính là xây dựng website quản lý việc mua bán hàng online của cửa hàng bán điện thoại gồm:

* Tìm hiểu tổng quát về ngôn ngữ C#.
* Tìm hiểu về SQL Server.
* Tìm hiểu về LINQ.
* Tìm hiểu về JQUERY.
* Tìm hiểu về ADO.NET.
* Tìm hiểu về Entity Framework Core.
* Tìm hiểu về ASP.NET CORE 2.1.
* Tìm hiểu về giao thức API trong ASP.NET.
* Phân tích điểm mạnh điểm yếu trong ASP.NET CORE.
* Xây dựng web bán điện thoại trên mạng cho cửa hàng.
* Thiết kế thân thiện với người dùng và dễ dàng xem, tìm kiếm sản phẩm.
* Xây dựng chức năng quản lý sản phẩm, quản lý cửa hàng cho người quản trị.

## PHƯƠNG PHÁP NGHIÊN CỨU

Về lý thuyết:

* Nắm vững các kiến thức về hệ thống thông tin.
* Nắm vững kỹ thuật lập trình theo mô hình MVC.
* Nắm vững các phương pháp tổ chức, phân tích và thiết kế cơ sở dữ liệu.
* Tham khảo một số web bán hàng trực tuyến, các web bán điện thoại đã và đang hoạt động trên internet như: thế giới di động, di động thông minh, cellphones…

Về kỹ thuật:

* Sử dụng ASP.NET Core 2.1 và Entity Framework Core để cài đặt và thực hiện các chương trình ứng dụng.
* Sử dụng giao thức API để xử lý ứng dụng.
* Sử dụng hệ quản trị cơ sở dữ liệu SQL Server 2017 để tạo và quản lý cơ sở dữ liệu.
* Dùng phần mền Visual studio 2017 để hổ trợ lập trình.
* Dùng postman để test dữ liệu API.
* Dùng trình duyệt google Chrome để chạy localhost.
* Dùng github để quản lý source code.

# MÔ TẢ NGHIỆP VỤ

## TỔNG QUAN

Admin (Nhà quản trị): là người quản trị trang web, admin đăng nhập tài khoản có toàn quyền sử dụng cấu hình và thêm xóa sửa cơ sở dữ liệu, xem các chi tiết đơn hàng, quản lý các thông tin danh mục, sản phẩm, nhà cung cấp, đơn đặt hàng…

Customer (Khách hàng): là khách hàng hay các công ty khác cần mua sản phẩm trên mạng. Họ dạo quanh cửa hàng mua từng sản phẩm của các hãng điện thoại khác nhau, chọn sản phẩm bỏ vào giỏ hàng, tạo đơn đặt hàng cho nhà quản trị. Customer là khách hàng thành viên đã đăng ký tài khoản và mật khẩu để tiện cho lần giao dịch tiếp theo. Họ có thể thừa hưởng những chương trình khuyến mãi, giảm giá của cửa hàng (nếu có).

Free User (Khách hàng vãng lai): là những người chỉ xem sản phẩm trong danh mục nhưng chưa mua, có thể xem quảng cáo giá cả chất lượng sản phẩm. Khách vãng lai muốn mua phải đăng ký thành viên.

Đăng nhập và tạo tài khoản người dùng: Khách hàng có thể tạo tài khoản trên web để có thể dễ dàng mua sản phẩm mỗi khi truy cập vào mà không cần phải nhập nhiều thông tin khi thanh toán sản phẩm.

Chức năng giỏ hàng: Hiển thị những sản phẩm mà khách hàng đã chọn từ trước và có thể dễ dàng kiểm tra xem mình đã mua những gì, chọn lọc những sản phẩm không cần thiết mà mình đã chọn trước đó.

Chức năng tìm kiếm đơn hàng: người quản trị có thể tìm kiếm những đơn hàng mà mình muốn tìm kiếm, không cần mất quá nhiều thời gian để tìm kiếm trên đơn hàng trong danh mục đơn hàng, chỉ cần có thông tin đơn hàng và ngày tháng đặt hàng là có thể tìm được đơn hàng mà mình muốn tìm kiếm.

Chức năng thanh toán: Chức năng thanh toán sản phẩm sau khi khách hàng đã chọn được sản phẩm mà mình muốn và thanh toán sản phẩm, chức năng thanh toán là thanh toán trực tiếp khi khách hàng nhận được đúng sản phẩm của mình.

Chức năng cập nhật thông tin khuyến mãi: Sản phẩm khuyến mãi sẽ được cập nhật thường xuyên trên trang web mỗi khi cửa hàng có chương trình khuyến mãi, để khách hàng có thể cập nhật được thông tin và mua sản phẩm đúng.

Chức năng đánh giá: khách hàng có thể đánh giá sản phẩm của cửa hàng mọi lúc, khi khách hàng đã sử dụng sản phẩm khách hàng có thể đánh giá được sản phẩm bao nhiêu điểm, khách hàng có thể bình luận hay phản hồi về những sản phẩm của cửa hàng thông qua website.

Chức năng xuất hóa đơn: sau khi thanh toán khách hàng sẽ được xuất hóa đơn cho sản phẩm.

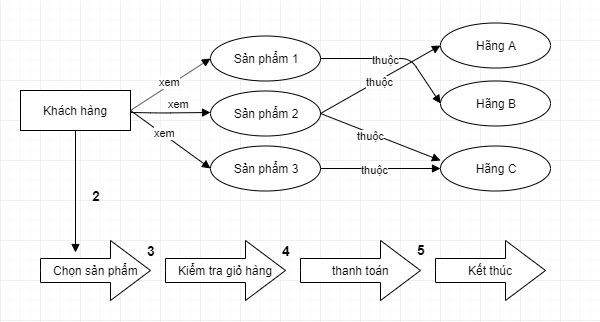
Quản lý sản phẩm: Admin có thể quản lý được những sản phẩm trên trang web, thêm, xóa, sửa những sản phẩm hiện thị trên web cho khách hàng.

Quản lý danh mục sản phẩm: giúp Admin quản lý được những loại sản phẩm mà web mình đang hiển thị, có thể điều chỉnh thay đổi.

## QUY TRÌNH

### **Quy trình đặt hàng**

Mô hình quy trình xem và đặt hàng của khách hàng



***Hình 2‑1:*** Mô hình quy trình xem và đặt hàng của Khách hàng

Mô hình được mô tả như sau:

Quy trình 1: Khi trang chủ được hiển thị, khách hàng có thể xem sản phẩm của các hãng điện thoại có trong danh mục, các sản phẩm này có thể trùng nhau về tên, về loại sản phẩm nhưng chỉ có người quản trị phân biệt được sản phẩm nào thuộc hãng điện thoại nào.

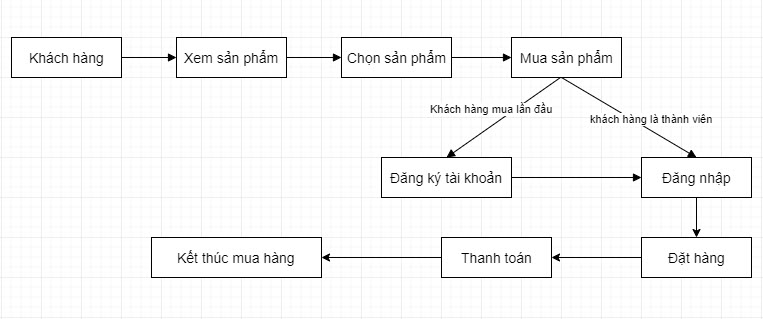
Quy trình 2: Sau khi đã chọn được sản phẩm, khách hàng sẽ cho vào giỏ hàng.

Quy trình 3: Trong giỏ hàng của mình, khách hàng có thể cập nhật số lượng, thêm hoặc bớt đi sản phẩm.

Quy trình 4: Khách hàng sẽ lập đơn đặt hàng sau khi đã quyết định chọn những sản phẩm có trong giỏ hàng.

Quy trình 5: Phiên giao dịch sẽ kết thúc khi khách hàng đặt xong đơn đặt hàng.

Mô hình mua hàng:



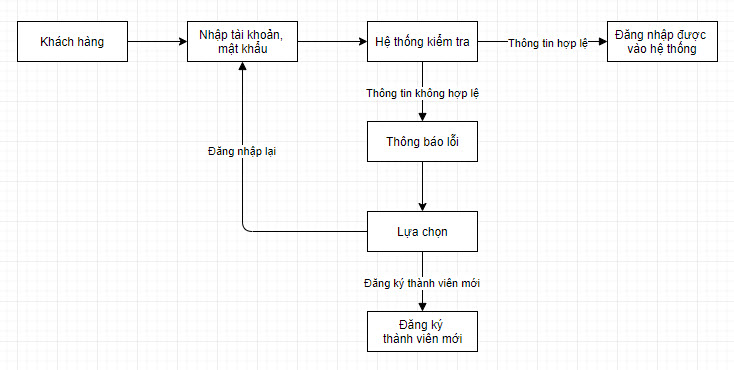
***Hình 2‑2***: Mô hình mua hàng

Quy trình mua hàng của khách hàng trong cửa hàng như sau: có hai loại khách hàng:

Nếu khách hàng đã là thành viên của cửa hàng khi đó khách hàng đăng nhập vào tài khoản của mình (tên đăng nhập, mật khẩu) thành công sẽ chào mừng khách hàng vào cửa hàng. Sau đó, khách hàng tham quan và dạo quanh cửa hàng trực tuyến, xem các thông tin chi tiết của từng sản phẩm của tất cả các hãng điện thoại. Nếu khách hàng đồng ý sản phẩm nào thì cho hàng vào giỏ hàng ảo tương ứng với nút “Thêm vào”. Khách hàng có thể chọn nhiều sản phẩm vào giỏ hàng và cập nhật lại số lượng sản phẩm (trở lại mua hàng tiếp, xóa một sản phẩm trong giỏ hàng…). Sau khi hoàn thành xong việc chọn hàng nếu khách hàng đồng ý đặt hàng sẽ click vào nút “Tạo đơn đặt hàng” để lập đơn đặt hàng. Cuối cùng kết thúc phiên giao dịch và giỏ hàng ảo sẽ được xóa trong cơ sở dữ liệu, đơn đặt hàng sẽ được lưu vào dữ liệu đơn đặt hàng của cửa hàng.

Nếu khách hàng mua hàng lần đầu tiên của cửa hàng thì khách hàng phải đăng ký thành viên của cửa hàng để tiện lợi cho những lần giao dịch tiếp theo thì cửa hàng yêu cầu nhập đầy đủ thông tin khách hàng. Khi việc đăng ký thành công khách hàng sẽ có một tài khoản trên cửa hàng là tên đăng nhập và mật khẩu cho những lần giao dịch tiếp theo. Sau đó quy trình mua hàng diễn ra giống như mô tả khách hàng đã là thành viên như trên.

### **Quy trình đăng nhập**

****

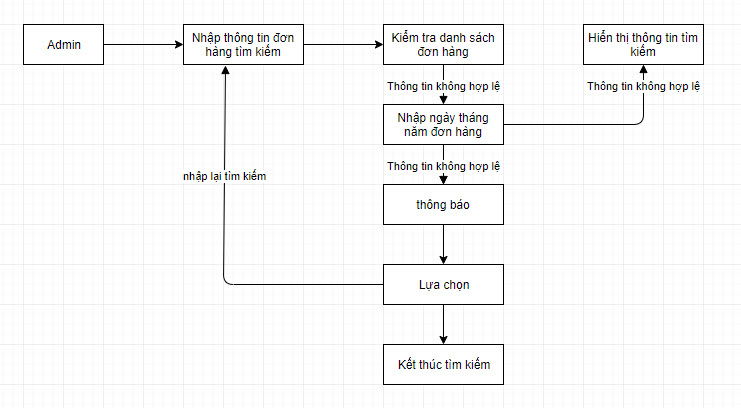
***Hình 2‑3***: Quy trình đăng nhập

Mô tả: cho khách hàng đăng nhập vào trang web, khách hàng nhập thông tin đăng nhập hệ thống tiếp nhận và kiểm tra tên đăng nhập và mật khẩu, nếu hợp lệ thì được đăng nhập vào hệ thống, còn không hợp lệ sẽ bị báo lỗi và được chọn là đăng nhập lại hoặc đăng ký mới.

### **Quy trình tìm kiếm đơn hàng**

Để không mất quá nhiều thời gian để có thể tìm kiếm đơn hàng mà mình muốn tìm kiếm, người quản trị có thể biết được thông tin về đơn hàng mà mình cần bằng cách nhập thông tin mã đơn hàng vào vào ô tìm kiếm trên trang web hoặc nhập thông tin ngày tháng của đơn hàng.

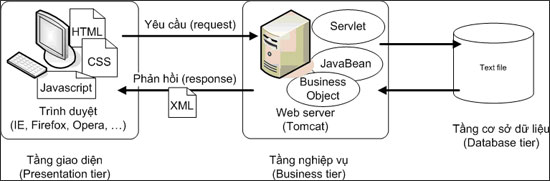
Khi khách hàng nhập tên đơn hàng mà mình muốn tìm kiếm, hệ thống sẽ so sánh những ký tự mà khách hàng nhập với dữ liệu của database. Nếu nhận được kết quả so sánh thì gửi thông báo kết quả tìm kiếm lên cho người dùng nếu không thì hiển thị thông báo “Không tìm thấy sản phẩm”.



***Hình 2‑4***: Quy trình tìm kiếm đơn hàng

# GIẢI PHÁP ĐỀ XUẤT

## KIẾN TRÚC TỔNG THỂ



***Hình 3‑1***: Kiến trúc tổng thể

* Các đối tượng tham gia khai thác trong hệ thống gồm:

Admin: là thành viên quản trị của hệ thống, có các quyền và chức năng như: tạo các tài khoản, quản lý sản phẩm, quản trị người dùng, quản lý hoá đơn, quản lý gmail, quản lý khuyến mãi…

Thành viên: là hệ thống thành viên, có chức năng: Đăng kí, đăng nhập, sửa thông tin cá nhân, xem giỏ hàng, đặt hàng, xem thông tin về các hóa đơn đã lập.

Khách hàng mới: là khách vãng lai có chức năng: Đăng kí, xem thông tin sản phẩm, xem giỏ hàng.

### **Module giới thiệu sản phẩm và dịch vụ**

Cung cấp cho khách hàng những thông tin cần thiết về các sản phẩm dịch vụ được chia theo danh mục mỗi danh mục có nhiều sản phẩm bên trong trình bày dưới dạng list sản phẩm, dịch vụ. Các thành phần này có thể bao gồm hình ảnh, mô tả và giá thành và không giới hạn danh mục sản phẩm.

### **Module giỏ hàng**

Bao gồm các chức năng như giỏ hàng, đơn hàng, quản lý khách hàng, khách hàng có thể lựa chọn sản phẩm mà mình muốn mua và đăng ký đặt hàng một cách dễ dàng.

### **Module quản lý sản phẩm**

Có thể thêm, xóa, sửa sản phẩm các loại sản phẩm không giới hạn sản phẩm.

### **Module tìm kiếm đơn hàng**

Cho phép tìm các thông tin trên website bằng các từ khóa có liên quan hoặc lọc theo ngày tháng năm.

### **Module sản phẩm mới**

Hiển thị các sản phẩm/dịch vụ mới nhất của công ty bằng các hình ảnh tiêu biểu, khách hàng có thể click vào hình sản phẩm để xem thông tin và hình ảnh mô tả chi tiết về sản phẩm đó bạn có thể ghi thêm điểm nổi bậc cấu hình của sản phẩm mới.

## GIẢI PHÁP CÔNG NGHỆ

### **HTML, CSS, Javascript, bootstrap:**

HTML: Là chữ viết tắt của cụm từ HyperText Markup (dịch là Ngôn ngữ đánh dấu siêu văn bản) được sử dụng để tạo ra các trang web, trên một website có thể chứa nhiều trang và mỗi trang được quy ra thành một tập tin HTML. Cha đẻ của HTML là Tim Berners-Lee, cũng là người khai sinh ra World Wide Web và chủ tịch của World Wide Web Consortium (W3C – tổ chức thiết lập ra các chuẩn trên môi trường Internet). Một tập tin HTML sẽ bao gồm các phần tử HTML và được lưu lại dưới đuôi mở rộng là .html hoặc .htm.

  HTML giúp định dạng văn bản trong trang Web nhờ các thẻ. Hơn nữa, các thẻ html có thể liên kết với các trang web khác nhau trên Internet. Đa số các thẻ HTML có dạng thẻ đóng mở. Thẻ đóng dùng chung từ lệnh giống như thẻ mở, nhưng thêm dấu xiên phải (/). Ngôn ngữ HTML qui định cú pháp không phân biệt chữ hoa chữ thường. Ví dụ, có thể khai báo <html> hoặc <HTML>.  Không có khoảng trắng trong định nghĩa thẻ.

CSS (Cascading Style Sheet). Website được cấu tạo từ các thẻ html nhưng với những thẻ html thì mới chỉ thể hiện được bộ khung của website. Để căn chỉnh, trình bày cho đẹp mắt thì ta cần sử dụng ngôn ngữ CSS. Đây là ngôn ngữ được dụng rất nhiều trong lập trình web, thường đi cùng với ngôn ngữ html.

JavaScript là ngôn ngữ lập trình kịch bản, hỗ trợ người lập trình trong việc tạo ra các hiệu ứng của website, kiểm tra các thông tin đầu vào thường được viết cho phần giao diện của website

Bootstrap là một framework cho phép thiết kế website reponsive nhanh hơn và dễ dàng hơn. Bootstrap bao gồm các HTML Teamplates, CSS Teamplates và Javasript tạo ra những cái cơ bản và có sẳn như: typography, forms, bootons và nhiều thứ khác.

### **Tổng Quát SQL Server.**

SQL Server là một hệ thống quản lý cơ sở dữ liệu quan hệ, hỗ trợ một số lượng lớn các quy trình xử lý giao dịch, ứng dụng doanh nghiệp và ứng dụng phân tích trong các công ty IT. SQL Server là một trong 3 công nghệ dữ liệu dẫn đầu hiện này cùng với Oracle Database và IBM's DB2. SQL Server được phát triển bởi [Microsoft](https://en.wikipedia.org/wiki/Microsoft), là một [máy chủ cơ sở dữ liệu](https://en.wikipedia.org/wiki/Database_server) , nó là một [sản phẩm phần mềm](https://en.wikipedia.org/wiki/Software_product) có chức năng chính là lưu trữ và truy xuất dữ liệu theo yêu cầu của các [ứng dụng phần mềm](https://en.wikipedia.org/wiki/Software_application) khác. Có thể chạy trên cùng một máy tính hoặc trên một máy tính khác trên mạng (bao gồm cả Internet).

SQL cung cấp các tập lệnh phục vụ cho các công việc hỏi đáp dữ liệu như:

- Chèn, xóa và cập nhật các hàng trong 1 quan hệ.

- Tạo, thêm, xóa và sửa đổi các đối tượng trong của cơ sở dữ liệu.

- Điều khiển việc truy xuất tới cơ sở dữ liệu và các đối tượng của cơ sở dữ liệu để đảm bảo tính bảo mật, tính nhất quán và sự ràng buộc của cơ sở dữ liệu.

Đối tượng của SQL server là các bảng dữ liệu với các cột và các hàng. Cột được gọi là trường dữ liệu và hàng là bản ghi của bảng. Cột dữ liệu và kiểu dữ liệu xác định tạo nên cấu trúc của bảng. Khi bảng được tổ chức thành một hệ thống cho một mục đích sử dụng cụ thể vào công việc nào đó sẽ trở thành một cơ sở dữ liệu.

### **Tổng Quát Về Ngôn Ngữ C#**

C# (đọc là Cee Sharp) là ngôn ngữ lập trình với nền tảng .Net platform phiên bản đầu tiên năm 2002, C# trải qua các phiên bản 1.0, 2.0, 3.0 … và hiện nay là 8.0. C# được phát triển của đội ngũ kỹ sư Microsoft, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamuth.

Ngôn ngữ C# khá đơn giản, nhưng nó có ý nghĩa cao khi thực thi những khái niệm lập trình hiện đại. C# bao gồm tất cả những hỗ trợ cho cấu trúc, thành phần component, lập trình hướng đối tượng. Những tính chất đó hiện diện trong một ngôn ngữ lập trình hiện đại. Và C# hội đủ những điều kiện như vậy, hơn nữa nó được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

C# cũng hỗ trợ giao diện interface. Một lớp chỉ có thể kế thừa duy nhất từ một lớp cha (tức là không cho đa kế thừa như trong ngôn ngữ C++), tuy nhiên một lớp có thể thực thi nhiều giao diện. Khi một lớp thực thi một giao diện thì nó sẽ cung cấp chức năng thực thi giao diện. C# cũng hỗ trợ cấu trúc, nhưng khái niệm về ngữ nghĩa của nó thay đổi khác với C++. Trong C#, một cấu trúc được giới hạn, là kiểu dữ liệu nhỏ gọn, và khi tạo thể hiện thì nó yêu cầu ít hơn về hệ điều hành và bộ nhớ so với một lớp. Một cấu trúc không thể kế thừa từ một lớp (hoặc kế thừa một cấu trúc khác), nhưng một cấu trúc có thể thực thi một giao diện.

Ngôn ngữ ra đời cùng với .NET:

* Kết hợp C++ và Java.
* Hướng đối tượng, Hướng thành phần.
* Chỉ cho phép đơn kế thừa kể cả dữ liệu cơ bản.
* Lớp Oject là cha của tất cả các lớp mọi lớp đều dẫn xuất từ Oject.

C# mạnh mẽ và bền vững:

* Garbage Collector: tự động thu hồi vùng nhớ không dùng.
* Kiểm soát và xử lý ngoại lệ exception: đoạn mã lỗi sẽ không được thực thi.
* Type safe: không cho gán các kiểu dữ liệu khác nhau.
* Versioning: dảm bảo sự tương thích giữa các lớp cha.

Ưu điểm khi sử dụng C#:

* Đễ đọc, dễ học hơn C++.
* Kiểm tra nhanh hơn và có khả năng ít lỗi hơn so với C++ và java.
* Tất cả mọi thứ điều được truyền bằng tham chiếu ngoại trừ các loại giá trị, theo mặc định.
* Chương trình được biên dịch thành các nhị phân riêng được tối ưu hóa cho các nên tảng khi nó chạy được (và nó chạy chậm hơn một chút so với mã c++).
* Dễ dàng tạo nhiều chủ đề.
* Có nhiều phương tiện tạm dừng các luồng để chờ tính hiệu và như vậy.
* Nếu chương trình gặp sự cố, nó sẽ bật lên một hợp thoại cho người dùng biết mã đó ở đâu và tại sao (trên windows) hoặc ghi dấu vết ngân xếp vào bàn điều khiển (với mono nếu chạy --debug).

Nhược điểm của khi sử dụng C#:

* Sử dụng nhiều bộ nhớ hơn C++.
* Trình thu gôm rác sử dụng chu kỳ vào bộ nhớ (nhưng hầu như không có, ít nhất là thời CPU).
* Một số thứ không hoạt động trong các chức năng gọi đơn trong một số của sổ sẽ không tồn tại trên linux, việc triển khai các biểu mẫu đơn của windows cần có cách giải quyết trong mã (nhưng có lẽ chúng ta không sử dụng chúng).
* Phải sử dụng .net 2.0 thuộc generic (trong số những thứ khác), 1.1. không có chúng không sử dụng bất cứ thứ gì mới hơn 2.0 vì hổ trợ thả 3.0 trở lên cho windows 98, ME và có thể 2000.
* Không thể khai khách các chức năng đồng bộ hóa.
* Không có con trỏ, nhưng chúng được thay thế bởi các tham số ref và out, vì vạy chúng không cần nhiều.
* Yêu cầu (phiên bản chính xác của) .NET framework phải được cài đặt để chạy chương trình.

### **Tổng Quát về LinQ.**

Language-Integrated Query (LINQ) là một thành phần của Microsoft .NET Framework cho phép bổ sung khả năng truy vấn dữ liệu từ cơ sở dữ liệu vào các ngôn ngữ lập trình .NET được giới thiệu lần đầu tiên trong .NET Framework 3.5 vào ngày 19 tháng 11 năm 2007.

LINQ truy xuất dữ liệu bằng các biểu thức truy vấn, giống như các câu lệnh SQL và có thể được sử dụng để trích xuất và xử lý dữ liệu một cách thuận tiện từ các mảng, các lớp được liệt kê, tài liệu XML, cơ sở dữ liệu quan hệ và nguồn dữ liệu của bên thứ ba. Các ứng dụng sẽ sử dụng các biểu thức truy vấn của LINQ như một khung công tác chung để tính toán và có thể đọc được dữ liệu một cách dễ dàng.

LINQ cũng định nghĩa một tập hợp các tên phương thức (được gọi là toán tử truy vấn), cùng với các quy tắc dịch được trình biên dịch sử dụng để dịch biểu thức truy vấn bằng cách sử dụng các tên phương thức, biểu thức lambda (lambda expressions) và các kiểu ẩn danh (anonymous types).

Các toán tử truy vấn hay dùng: Select, Where, SingleOrDefault, Contains, Count, OrderBy, Join, ... các toán tử này gần giống bên SQL Server.

Anonymous type (kiểu ẩn danh) là một tính năng của C # 3.0 cho phép các kiểu dữ liệu gói gọn một tập hợp các thuộc tính vào một đối tượng đơn lẻ mà không cần phải định nghĩa rõ ràng một kiểu. Đây là một tính năng quan trọng cho tính năng LINQ.

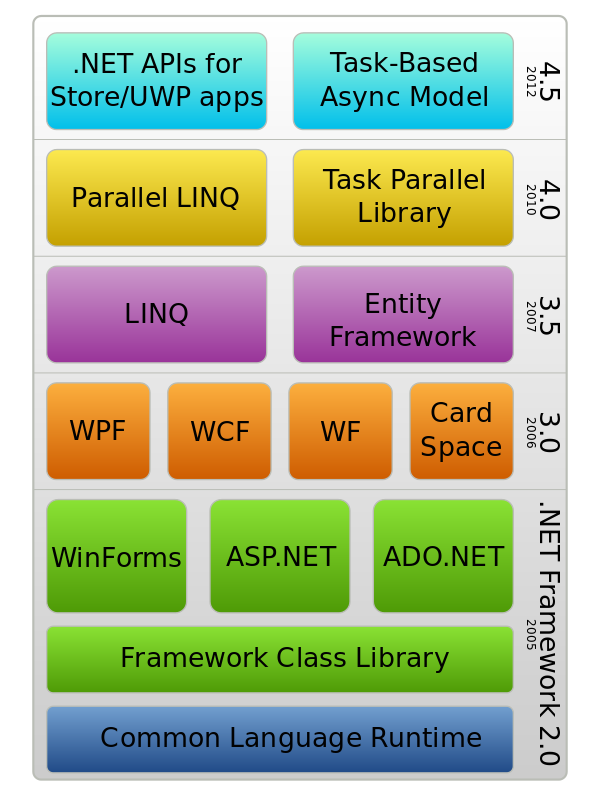
Trong lập trình có lúc ta sẽ cần truy xuất dữ liệu của database và trong C# thì ta phải tạo một class để hứng dữ liệu đó nhưng ta chỉ cần sử dụng dữ liệu đó đúng một lần mà phải tạo một class mới thì rất bất tiện. Vì thế mà ta có thể sử dụng Anonymous type để xử lý việc này. Lưu ý là những Anonymous này chỉ sử dụng một lần và trình biên dịch sẽ xóa nó và chạy lại nó ở lần chạy tiếp theo. Vì thế mà ta hãy tùy chỉnh để sử dụng cho từng trường hợp khác nhau.

Lambda Expressions hay còn gọi một hàm ẩn danh (anonymous function) là một định nghĩa hàm không bị ràng buộc với một định danh. ta có thể tạo các hàm in-line nhằm hạn chế việc khai báo các hàm riêng lẻ không cần thiết, giúp mã lệnh ngắn gọn hơn. Với lambda expression, ta có thể viết ngắn gọn và dễ dàng hơn nhờ việc cung cấp toán tử và cú pháp mới, đồng thời thể hiện sự “thông minh” của compiler bằng cách tự nhận diện kiểu của dữ liệu.

## Tổng Quan Về .Net Framework

.NET Framework là một nền tảng lập trình và cũng là một nền tảng thực thi ứng dụng chủ yếu trên hệ điều hành [Microsoft Windows](https://vi.wikipedia.org/wiki/Microsoft_Windows) được phát triển bởi [Microsoft](https://vi.wikipedia.org/wiki/Microsoft). Các chương trình được viết trên nền tảng .NET Framework sẽ được triển khai trong môi trường [phần mềm](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m) (ngược lại với *môi trường phần cứng*) được biết đến với tên [Common Language Runtime](https://vi.wikipedia.org/w/index.php?title=Common_Language_Runtime&action=edit&redlink=1) (CLR). Môi trường phần mềm này là một [máy ảo](https://vi.wikipedia.org/wiki/M%C3%A1y_%E1%BA%A3o) trong đó cung cấp các dịch vụ như [an ninh phần mềm](https://vi.wikipedia.org/w/index.php?title=An_ninh_ph%E1%BA%A7n_m%E1%BB%81m&action=edit&redlink=1) (*security*), [quản lý bộ nhớ](https://vi.wikipedia.org/wiki/Qu%E1%BA%A3n_l%C3%BD_b%E1%BB%99_nh%E1%BB%9B) (*memory management*), và các xử lý lỗi ngoại lệ (*exception handling*).

.NET Framework bao gồm các thư viện lập trình lớn và những thư viện này hỗ trợ việc xây dựng các chương trình phần mềm như: lập trình giao diện, truy cập, kết nối [cơ sở dữ liệu](https://vi.wikipedia.org/wiki/C%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u), [ứng dụng web](https://vi.wikipedia.org/wiki/%E1%BB%A8ng_d%E1%BB%A5ng_web), các [giải thuật](https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n), [cấu trúc dữ liệu](https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u), giao tiếp mạng... CLR cùng với bộ thư viện FCL (Flow Chart Language) là hai thành phần chính của .NET Framework.

.NET Framework đơn giản hóa việc viết ứng dụng bằng cách cung cấp nhiều thành phần được thiết kế sẵn, người lập trình chỉ cần học cách sử dụng và tùy theo sự sáng tạo mà gắn kết các thành phần đó lại với nhau. Nhiều công cụ được tạo ra để hỗ trợ xây dựng ứng dụng .NET và IDE (*Integrated Developement Environment*) đó là Visual Studio, phần mềm này được phát triển và hỗ trợ bởi chính Microsoft.

Hình

### **ADO.NET:**

ADO.NET là một bộ các thư viện hướng đối tượng (OOP) cho phép bạn tương tác với dữ liệu nguồn. Thông thường thì dữ liệu nguồn là một cơ sở dữ liệu (database), nhưng nó cũng có thể là file text, exel hoặc XML. Theo những mục tiêu của hướng dẫn này, chúng ta sẽ chỉ xem xét tới cách ADO.NET làm việc với database.

Có rất nhiều loại database hiện nay như Microsoft SQL Server, Microsoft Access, Oracle, Borland Interbase, và IBM DB2, … Để làm rõ hơn phạm vi của loạt bài này, tất cả ví dụ sẽ sử dụng SQL Server.

ADO.NET ánh xạ đối tượng quan hệ thông qua Entity Framework (EF). Entity  
Framework (EF) là một khung công tác (ORM – Object Relational Mapping) ánh xạ  
đối tượng - quan hệ mã nguồn mở cho ADO.NET và EF cũng là một phần của .NET  
Framework. Nó là một bộ công nghệ trong ADO.NET hỗ trợ phát triển các ứng dụng  
phần mềm theo định hướng dữ liệu. Entity Framework cho phép các nhà phát triển  
làm việc với dữ liệu dưới dạng các đối tượng và thuộc tính cụ thể. Với Entity  
Framework, các nhà phát triển có thể làm việc ở mức trừu tượng cao hơn khi họ xử lý  
dữ liệu, có thể tạo và duy trì các ứng dụng hướng dữ liệu với ít mã hơn trong các ứng  
dụng truyền thống.

ADO.NET truy xuất với CSDL thông qua Language-Integrated Query (LINQ).  
LINQ to SQL (trước đây gọi là DLINQ) cho phép LINQ được sử dụng để truy vấn cơ  
sở dữ liệu Microsoft SQL Server, bao gồm cơ sở dữ liệu SQL Server Compact. Vì dữ  
liệu SQL Server có thể nằm trên một máy chủ từ xa, và bởi vì SQL Server có công cụ  
truy vấn riêng của nó, nó không sử dụng công cụ truy vấn của LINQ. Thay vào đó, nó  
chuyển đổi một truy vấn LINQ thành một truy vấn SQL sau đó được gửi đến SQL  
Server để xử lý. Tuy nhiên, vì SQL Server lưu trữ dữ liệu dưới dạng dữ liệu quan hệ  
và LINQ làm việc với dữ liệu được đóng gói trong các đối tượng, hai biểu diễn phải  
được ánh xạ với nhau. Vì lý do này, LINQ to SQL cũng định nghĩa một khung công  
tác lập bản đồ. Việc ánh xạ được thực hiện bằng cách định nghĩa các lớp tương ứng  
với các bảng trong cơ sở dữ liệu và chứa tất cả hoặc một tập con nhất định của các cột  
trong bảng dưới dạng các thành phần dữ liệu.

Data Provider: Chúng ta biết rằng ADO.NET cho phép tương tác với các loại dữ liệu và kiểu database. Mỗi loại dữ liệu cần một cách thức khác nhau để có thể truy xuất. Các loại  
dữ liệu cũ sử dụng giao thức ODBC, các loại dữ liệu mới hơn sử dụng giao thức  
OleDb. Vì vậy cần có một thư viện thống nhất để làm việc với chúng, đây chính là lý  
do mà ADO.NET được tạo ra.

ADO.NET cung cấp một cách thức chung để tương tác với nguồn dữ liệu, nhưng  
với mỗi loại dữ liệu ta phải sử dụng một thư viện khác nhau. Các thư viện này được  
gọi là Data Provider và thường được đặt tên theo giao thức hoặc loại dữ liệu mà chúng cho phép ta truy xuất. Bảng bên dưới liệt kê các data provider phổ biến, tiền tố (prefix)  
API mà chúng sử dụng và cơ sở dữ liệu mà ta có thể làm việc.

Các đối tượng của ADO.NET: bao gồm nhiều đối tượng bạn có thể dùng với dữ liệu. Phần này giới thiệu một vài đối tượng chính bạn sẽ sử dụng. Theo tiến độ trong các bài viết sau, bạn sẽ khám phá thêm nhiều đối tượng ADO.NET và cách chúng được sử dụng trong mỗi lesson riêng. Các đối tượng dưới đây bắt buộc bạn phải biết để làm việc với dữ liệu khi dùng ADO.NET.

SqlConnection: Để tương tác với database, bạn phải có một kết nối tới nó. Kết nối giúp xác định database server, database name, user name, password, và các tham số cần thiết để kết nối tới database. Một đối tượng connection được dùng bởi đối tượng command vì thế chúng sẽ biết database nào để thực thi lệnh.

SqlCommand: Quá trình tương tác với database cần phải biết hành động nào bạn muốn xảy ra. Điều này được thực hiện bởi đối tượng command. Bạn dùng đối tượng command để gửi một câu lệnh SQL tới database. Một đối tượng command dùng một đối tượng connection để xác định database nào sẽ được truy xuất. Bạn có thể dùng một đối tượng command riêng lẻ để thực thi lệnh trực tiếp, hoặc để gắn một tham chiếu của đối tượng command cho một SqlDataAdapter – đối tượng giữ các command sẽ làm việc trên một nhóm dữ liệu như sẽ đề cập tới trong phần dưới.

SqlDataReader: Nhiều thao tác dữ liệu đòi hỏi bạn chỉ lấy một luồng dữ liệu để đọc. Đối tượng data reader cho phép bạn lấy được kết quả của một câu lệnh SELECT từ một đối tượng command. Để tăng hiệu suất, dữ liệu trả về từ một data reader là một luồng dữ liệu fast forward-only. Có nghĩa là bạn chỉ có thể lấy dữ liệu từ luồng theo một thứ tự nhất định. Mặc dù điều này có lợi về mặt tốc độ, nhưng nếu bạn cần phải thao tác dữ liệu, thì một DataSet sẽ là một đối tượng tốt hơn để làm việc.

DataSet: Đối tượng DataSet là một thể hiện của dữ liệu trong bộ nhớ. Chúng chứa nhiều đối tượng DataTable, bên trong DataTable lại có nhiều column và row, giống như các database table thông thường. Bạn thậm chí có thể định nghĩa dữ liệu giữa các table để tạo các quan hệ parent-child. DataSet được thiết kế đặc biệt để giúp quản lý dữ liệu trong bộ nhớ và để hỗ trợ các thao tác không cần kết nối (disconnected) trên dữ liệu. DataSet là một đối tượng được dùng bởi tất cả Data Provider, đó là lý do tại sao nó không có một Data Provider prefix trong tên gọi.

SqlData adapter: Đôi lúc dữ liệu mà bạn làm việc là read-only và bạn ít khi cần thay đổi dữ liệu nguồn. Vài trường hợp cần lưu trữ tạm dữ liệu trong bộ nhớ để hạn chế truy xuất đến database. Data adapter làm điều này dễ dàng bằng cách giúp bạn quản lý dữ liệu trong chế độ ngắt kết nối. Data adapter sẽ đổ vào DataSet khi đọc dữ liệu và thực hiện thay đổi dữ liệu một lượt vào database.

Data adapter chứa một tham chiếu đến đối tượng connection và mở/đóng kết nối tự động khi đọc và ghi dữ liệu vào database. Hơn nữa, data adapter chứa đối tượng command cho những thao tác SELECT, INSERT, UPDATE và DELETE trên dữ liệu. Bạn sẽ có một data adapter được định nghĩa cho mỗi table trong một DataSet và nó sẽ quản lý các giao tiếp với database cho bạn. Tất cả những gì bạn cần làm là chỉ cho data adapter khi nào nạp hoặc ghi vào database.

Điểm mạnh của ADO.NET

* Được thiết kế làm việc với cả dữ liệu phi kết nối trong môi trường đa tầng  
  (Multi-Tier). Sử dụng XML để trao đổi dữ liệu phi kết nối, do vậy dễ dàng  
  khi giao tiếp giữa các ứng dụng không phải trên nền tảng window.
* Thiết kế hoàn toàn bằng hướng đối tượng, đây là điểm chi phối toàn bộ  
  sản phẩm Microsoft .NET.
* Là thành phần nội tại (có sẵn) trong .NET Framework, do vậy dễ dàng khi  
  phát triển bằng nhiều ngôn ngữ khác nhau.
* Hỗ trợ XML hoàn toàn (ADO thì không), nghĩa là chúng ta có thể nạp dữ  
  liệu từ một tệp XML và thao tác như một CSDL, sau đó cũng có thể lưu  
  kết quả ngược trở lại tệp XML.
* Hỗ trợ XML hoàn toàn (ADO thì không), nghĩa là chúng ta có thể nạp dữ  
  liệu từ một tệp XML và thao tác như một CSDL, sau đó cũng có thể lưu  
  kết quả ngược trở lại tệp XML

### **Tổng Quan Về Entity Framework**

Entity Framework([EF](https://www.dammio.com/glossary/ef)) là một framework ánh xạ quan hệ đối tượng (ORM) dành cho ADO.NET, là 1 phần của .NET Framework. EF cho phép các nhà phát triển Web tương tác với dữ liệu quan hệ theo phương pháp hướng đối tượng đặc trưng. Lợi ích lớn nhất của EF là giúp lập trình viên giảm thiểu việc lập trình mã nguồn cần thiết để truy cập và tương tác với cơ sở dữ liệu. EF được Microsoft hỗ trợ phát triển lâu dài và bền vững, vì vậy EF là 1 framework mạnh nhất hiện nay để phát triển ứng dụng Web với sự hỗ trợ đông đảo của các nhà phát triển Web.

Entity Framework ra đời từ 2008 cùng với .NET 3.5. Phiên bản hiện nay là 6.0. Có một số mốc đặc biệt trong lịch sử framework này.

Phiên bản đầu tiên chỉ hỗ trợ hướng tiếp cận Database-first. Nghĩa là EF lúc đó chỉ có thể làm việc với một cơ sở dữ liệu sẵn có. EF giúp sinh code các lớp model bằng C# hoặc VB.NET.

Đến phiên bản 4 xuất hiện hướng tiếp cận Model-first, giúp thiết kế các lớp thực thể trước sử dụng giao diện đồ họa.

Phiên bản 4.1 đưa thêm hướng tiếp cận Code-first, nghĩa là từ giờ mọi thứ được viết bằng code VB.NET hoặc C# ngay từ đầu, một hướng tiếp cận được lập trình viên hoan nghênh đặc biệt.

Đặc điểm quang trong của Entity Framework thể hiện cơ sở dữ liệu dưới dạng tập hợp của các object quen thuộc.

Trước hết, cả cơ sở dữ liệu được thể hiện bằng một lớp con của lớp DbContext. Mỗi bảng dữ liệu sẽ thể hiện bằng một object của DbSet<T>. Mỗi hàng trong bảng thể hiện bằng một object của lớp thực thể bạn xây dựng. Mỗi cột thể hiện bằng một thuộc tính của object. Tất cả những thao tác ánh xạ này có thể được thực hiện hoàn toàn tự động hoặc bạn có thể can thiệp vào để thực hiện theo ý mình.

Thứ hai, bạn có thể truy vấn dữ liệu sử dụng LINQ, thay vì dùng SQL. Bạn có thể dễ dàng thực hiện các truy vấn CRUD hoàn toàn từ code C# với các class quen thuộc mà không cần viết dòng SQL nào.

Thứ ba, bạn có thể thực hiện các thao tác với cấu trúc dữ liệu như tạo cơ sở dữ liệu, tạo bảng, thay đổi cấu trúc bảng một cách dễ dàng với công cụ Migration mà không làm mất dữ liệu.

Entity Framework Core là một phiên bản mã nguồn mở, nhỏ, nhẹ, có thể mở rộng và đa nền tảng nằm trong bộ Entity Framework. EF Core là bộ ánh xạ đối tượng – quan hệ (Object Relation Mapping) cho phép các lập trình .NET làm việc với CSDL quan hệ thông qua các đối tượng (object), giúp các lập trình viên không cần viết mã cho những gì liên quan tới dữ liệu

Giai đoạn phát triển của EF Core:

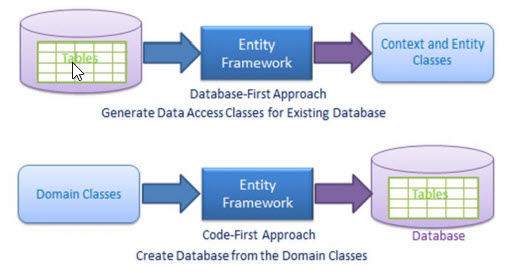
* EF Core 1.0 ra đời vào tháng 2016.
* EF Core 1.1 ra đời vào tháng 11 năm 2016.
* EF Core 2.0 ra đời vào tháng 11 năm 2017.
* EF Core 2.1 ra đời vào tháng 05 năm 2018.
* EF Core 2.2 ra đời vào tháng 12 năm 2018.
* EF Core 3.0 ra đời vào tháng 09 năm 2019.
* EF Core 3.1 ra đời vào tháng 12 năm 2019.

Trong EF Core có 2 cách để tiếp cận khi làm việc với database:

* Cách 1: Code First tạo model các đối tượng dữ liệu, sau đó Migration vào database.
* Cách 2: Database First sử dụng các database có sản và sau đó phát sinh ra các model dữ liệu tương ứng.

Để làm việc với Entity Framework Core, cần sử Nuget package để cài database provider tương ứng. Trong giáo trình này sử dụng database là SQL Server nên cần cài 2 Nuget:

* Microsoft. EntityFrameworkCore. SqlServer.
* Microsoft. EntityFrameworkCore. Tools.



Hình:

### **Tổng Quan Về ASP.NET CORE**

#### **Khái niệm và lịch sử hình thành**

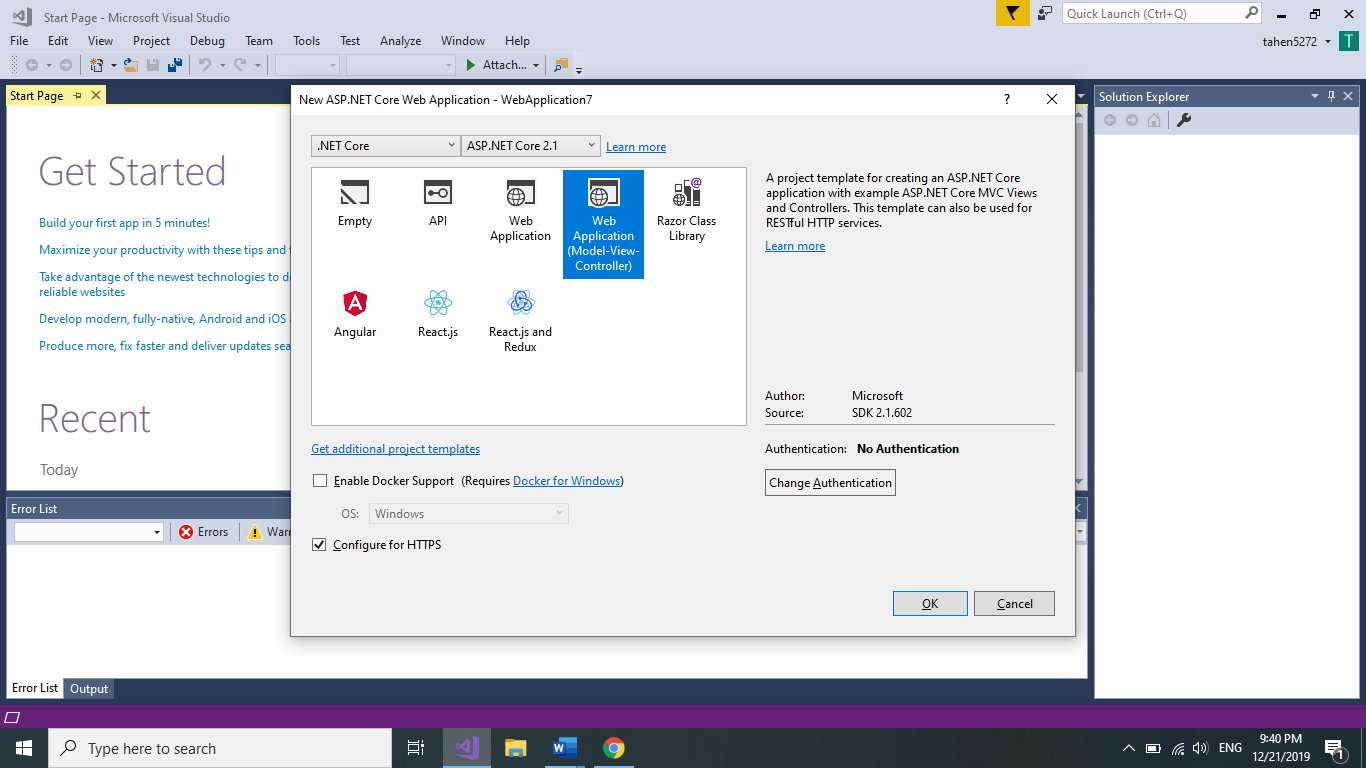
ASP.NET Core là một Open-source mới và là Cross-Platform framework giúp xây dựng các ứng dụng web hiện đại dựa trên đám mây như là web apps, IoT apps, mobile backends. Các ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên .NET framewrok. ASP.NET Core cung cấp một kiến trúc để tối ưu hóa việc xây dựng các ứng dụng đám mây (Microsoft Azure) hoặc các ứng dụng độc lập. Nó bao gồm các Module thành phần với chi phí tối thiểu. Do đó bạn có thể giữ lại tính linh hoạt khi xây dựng ứng dụng của mình. Giai đoạn phát triển của Asp.net Core:

ASP.NET Core được module hóa, không còn dựa vào System.Web.dll mà dựa vào nhu cầu người dùng cần cài đặt các gói trên Nuget và các thành phần khác. Điều này giúp bạn optimize apps, chỉ lấy các gói Nuget mà bạn cần. Lợi ích của ứng dụng nhỏ bao gồm bảo mật chặt chẽ hơn, giảm dịch vụ không cần thiết, hiệu suất được cải thiện và giảm chi phí tương ứng với các module mà bạn sử dụng

* ASP 1.0 được phát hành vào tháng 12 năm 1996 là một phần của IIS 3.0.
* ASP 2.0 được phát hành vào tháng 09 năm 1997 là một phần của 4.0.
* ASP 3.0 được phát hành vào tháng 11 năm 2000 là một phần của 5.0.
* ASP.NET được phát hành vào tháng 1 năm 2002 với phiên bản 1.0 của .NET Framework.
* Ngày 24 tháng 4 năm 2003 với phiên bản 1.1 của .NET Framework.
  + - Tích hợp hỗ trợ mobile ASP.NET (trước đây chỉ là phần mở rộng tùy chọn)
    - Thay đổi về kiến trúc an ninh - sử dụng [sandbox](https://vi.wikipedia.org/w/index.php?title=Sandbox&action=edit&redlink=1) khi thực thi các ứng dụng từ Internet.
    - Tích hợp hỗ trợ [ODBC](https://vi.wikipedia.org/wiki/ODBC) và [cơ sở dữ liệu Oracle](https://vi.wikipedia.org/w/index.php?title=C%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u_Oracle&action=edit&redlink=1).
    - [.NET Compact Framework](https://vi.wikipedia.org/w/index.php?title=.NET_Compact_Framework&action=edit&redlink=1)
    - Hỗ trợ [IPv6](https://vi.wikipedia.org/wiki/IPv6) (*Internet Protocol version 6*)
    - Vài thay đổi khác trong [API](https://vi.wikipedia.org/wiki/Giao_di%E1%BB%87n_l%E1%BA%ADp_tr%C3%ACnh_%E1%BB%A9ng_d%E1%BB%A5ng)
* Ngày 7 tháng 11 năm 2005 với phiên bản 2.0 của .NET Framework
* .NET framework hỗ trợ đầy đủ nền tảng 64-bit.
* .NET framwork có quan hệ với *Smart Personal Objects Technology*.
* Ngày 6 tháng 11 năm 2006 với phiên bản 3.0 của .NET Framework
* Ngày 6 tháng 11 năm 2006 với phiên bản 3.0 của .NET Framework
* Windows Communication Foundation (WCF - tên mã là Indigo): Một nền tảng mới cho phép xây dựng các ứng dụng hướng dịch vụ (service-oriented).
* [Windows Workflow Foundation](https://vi.wikipedia.org/w/index.php?title=Windows_Workflow_Foundation&action=edit&redlink=1) (WF): Một kiến trúc hỗ trợ xây dựng các ứng dụng workflow (luồng công việc) một cách dễ dàng hơn. WF cho phép định nghĩa, thực thi và quản lý các workflow từ cả cách nhìn theo hướng kĩ thuật và hướng thương mại.
* [Windows CardSpace](https://vi.wikipedia.org/w/index.php?title=Windows_CardSpace&action=edit&redlink=1) (tên mã là *InfoCard*): một kiến trúc để [quản lý định danh](https://vi.wikipedia.org/w/index.php?title=Qu%E1%BA%A3n_l%C3%BD_%C4%91%E1%BB%8Bnh_danh&action=edit&redlink=1) (*identity management*) cho các ứng dụng được phân phối.
* Có thể minh họa.NET 3.0 bằng một công thức đơn giản:
* NET 3.0 =.NET 2.0 + WPF + WCF + WF + WCS
* Ngày 19 tháng 11 năm 2007 với phiên bản 3.5 của .NET Framework
* Các tính năng mới cho ngôn ngữ [C#](https://vi.wikipedia.org/wiki/C) 3.0 và [VB.NET](https://vi.wikipedia.org/wiki/Visual_Basic_2005) 9.0
* Hỗ trợ [Expression Tree](https://vi.wikipedia.org/w/index.php?title=Expression_Tree&action=edit&redlink=1) và [Lambda](https://vi.wikipedia.org/wiki/Lamda)
* Các phương thức mở rộng (*Extension methods*)
* Các kiểu ẩn danh (*Anonymous types*)
* LINQ
* Phân trang (*paging*) cho ADO.NET
* API cho nhập xuất mạng không đồng bộ (*asynchronous network I/O*)
* Peer Name Resolution Protocol resolver
* Cải thiện WCF và WF
* Tích hợp ASP.NET AJAX
* Namespace mới System.CodeDom
* Microsoft ADO.NET Entity Framework 1.0
* Cũng như phiên bản 3.0, có thể minh họa sự thay đổi của.NET 3.5 bằng công thức:
  + - * .NET 3.5 =.NET 3.0 + LINQ + ASP.NET 3.5 + REST
* Ngày 12 tháng 4 năm 2010 với phiên bản 4.0 của .NET Framework
* Dynamic Language Runtime
* Code Contracts
* Managed Extensibility Framework
* Hỗ trợ các tập tin ánh xạ bộ nhớ (*memory-mapped files*)
* Mô hình lập trình mới cho các ứng dụng đa luồng (*multithreaded*) và bất đồng bộ (*asynchronous*)
* Cải thiện hiệu năng, các mô hình workflow.
* Ngày 15 tháng 8 năm 2012 với phiên bản 4.5 của .NET Framework
* Ngày 17 tháng 10 năm 2013 với phiên bản 4.5.1 của .NET Framework
  + [Bootstrap](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)) 3.0
  + Web API 2: [OAuth](https://en.wikipedia.org/wiki/OAuth) 2.0, [OData](https://en.wikipedia.org/wiki/Open_Data_Protocol) improvements, [CORS](https://en.wikipedia.org/wiki/Cross-origin_resource_sharing)
  + [MVC](https://en.wikipedia.org/wiki/ASP.NET_MVC) 5: Attribute routing, authentication filters and filter overrides
  + [EF](https://en.wikipedia.org/wiki/Entity_Framework) 6
  + [SignalR](https://en.wikipedia.org/wiki/SignalR)
  + [OWIN](https://en.wikipedia.org/wiki/Open_Web_Interface_for_.NET)
* Ngày 29 tháng 7 năm 2015 với phiên bản 4.6 của .NET Framework
* Ngày 29 tháng 7 năm 2015 với phiên bản 4.6 của .NET Framework
  + Hỗ trợ [HTTP / 2](https://en.wikipedia.org/wiki/HTTP/2) khi chạy trên Windows 10
  + Các API trả về tác vụ không đồng bộ khác
* Ngày 27 tháng 6 năm 2016 với phiên bản ASP.NET Core 1.0
* ASP.NET Core 1.0 đa nền tảng (cross platform) và cloud ready
* **MVC + Web API + Web Pages = ASP.NET Core MVC 1.0.** Vì vậy người dùng chỉ có thể sử dụng Web forms với ASP.NET 4.6 về trước.
* ASP.NET Core MVC 1.0 có nhiều biến đổi cách cấu tạo chương trình. Tiêu biểu là App\_Start, App\_Data, Global.asax và root tệp web.config biến mất. App\_Start được thay thế bằng Startup.cs, web.config được thay bằng appsetting.json Trong quá trình cài đặt config, định dạng JSON được sử dụng phổ biến hơn, thay vì XML.
* Trong ASP.NET Core MVC 1.0, một project dạng class library khi được compile sẽ trở thành một **Nuget package**, thay vì dạng **.dll package**
* Ngày 18 tháng 11 năm 2016 với phiên bản ASP.NET Core 1.1
* Ngày 14 tháng 08 năm 2017 với phiên bản ASP.NET Core 2.0
* Xây dựng các ứng dụng và dịch vụ web, ứng dụng [IoT](https://www.microsoft.com/internet-of-things/) và phụ trợ di động.
* Sử dụng các công cụ phát triển yêu thích của bạn trên Windows, macOS và Linux.
* Triển khai lên đám mây hoặc tại chỗ.
* Chạy trên [.NET Core hoặc .NET Framework](https://docs.microsoft.com/en-us/dotnet/articles/standard/choosing-core-framework-server) .
* Ngày 30 tháng 05 năm 2018 với phiên bản ASP.NET Core 2.1
* Một mô hình quy mô đơn giản hóa.
* Một máy khách JavaScript mới không có phụ thuộc jQuery.
* Một giao thức nhị phân nhỏ gọn mới dựa trên MessagePack.
* Hỗ trợ cho các giao thức tùy chỉnh.
* Một mô hình phản ứng phát trực tuyến mới.
* Hỗ trợ cho khách hàng dựa trên WebSockets trần
* Thời gian khởi động ứng dụng nhanh hơn đáng kể.
* Các bản cập nhật nhanh cho các chế độ xem và trang dao động trong thời gian chạy vẫn có sẵn như là một phần của quy trình phát triển lặp.
* Là một phần của trải nghiệm chạy .NET Core SDK lần đầu tiên, khi bạn sử dụng SDK lần đầu tiên.
* Sử dụng thủ dev-certscông công cụ mới.
* Nhiều điểm cuối bao gồm các URL. Để biết thêm thông tin, hãy xem [triển khai máy chủ web Kestrel: Cấu hình điểm cuối](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/servers/kestrel?view=aspnetcore-2.1#endpoint-configuration) .
* Chứng chỉ sử dụng cho HTTPS từ tệp trên đĩa hoặc từ kho lưu trữ chứng chỉ.
* Ngày 04 tháng 12 năm 2018 với phiên bản ASP.NET Core 2.2
  + [Sử d](https://docs.microsoft.com/vi-vn/aspnet/core/web-api/advanced/analyzers?view=aspnetcore-2.1)[[Giảm thiểu xung đột UseIIS](https://docs.microsoft.com/vi-vn/aspnet/core/web-api/advanced/analyzers?view=aspnetcore-2.1)](https://github.com/aspnet/KestrelHttpServer/issues/2760)
  + [[Định cấu hình tùy chọn máy chủ Kestrel với ConfigureKestrel](https://docs.microsoft.com/vi-vn/aspnet/core/web-api/advanced/analyzers?view=aspnetcore-2.1)](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/servers/kestrel?view=aspnetcore-2.1#how-to-use-kestrel-in-aspnet-core-apps)
  + [ụng máy phân tích API web](https://docs.microsoft.com/vi-vn/aspnet/core/web-api/advanced/analyzers?view=aspnetcore-2.1)
  + [Sử dụng các quy ước API web](https://docs.microsoft.com/vi-vn/aspnet/core/web-api/advanced/conventions?view=aspnetcore-2.1)
  + [ASP.NET Core 2.2.0-preview1: Phân tích & quy ước OpenAPI](https://blogs.msdn.microsoft.com/webdev/2018/08/23/asp-net-core-2-20-preview1-open-api-analyzers-conventions/)
  + [Định tuyến điểm cuối trong 2.2](https://blogs.msdn.microsoft.com/webdev/2018/08/27/asp-net-core-2-2-0-preview1-endpoint-routing/)
  + [Biến áp tham số tuyến đường](https://www.hanselman.com/blog/ASPNETCore22ParameterTransformersForCleanURLGenerationAndSlugsInRazorPagesOrMVC.aspx) (xem phần **Định tuyến**)
  + [Sự khác nhau giữa định tuyến dựa trên IRouter và điểm cuối](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/routing?view=aspnetcore-2.1#differences-from-earlier-versions-of-routing)
* Ngày 23 tháng 09 năm 2019 với phiên bản ASP.NET Core 3.0
  + Tạo UI tương tác phong phú bằng C # thay vì JavaScript.
  + Chia sẻ logic ứng dụng phía máy chủ và phía máy khách được viết bằng .NET.
  + Kết xuất giao diện người dùng dưới dạng HTML và CSS để hỗ trợ trình duyệt rộng, bao gồm cả trình duyệt di động.
  + Blazor framework hỗ trợ các kịch bản:
  + Thành phần UI tái sử dụng (Thành phần dao cạo)
  + Định tuyến phía máy khách
  + Bố cục thành phần
  + Hỗ trợ tiêm phụ thuộc
  + Biểu mẫu và xác nhận
  + Xây dựng thư viện thành phần với thư viện lớp Dao cạo
  + JavaScript xen kẽ
  + Là một khung RPC (gọi thủ tục từ xa) phổ biến, hiệu suất cao.
  + Cung cấp một cách tiếp cận hợp đồng đầu tiên để phát triển API.
  + Dòng hai chiều và kiểm soát dòng chảy.
  + Hủy bỏ và thời gian chờ.

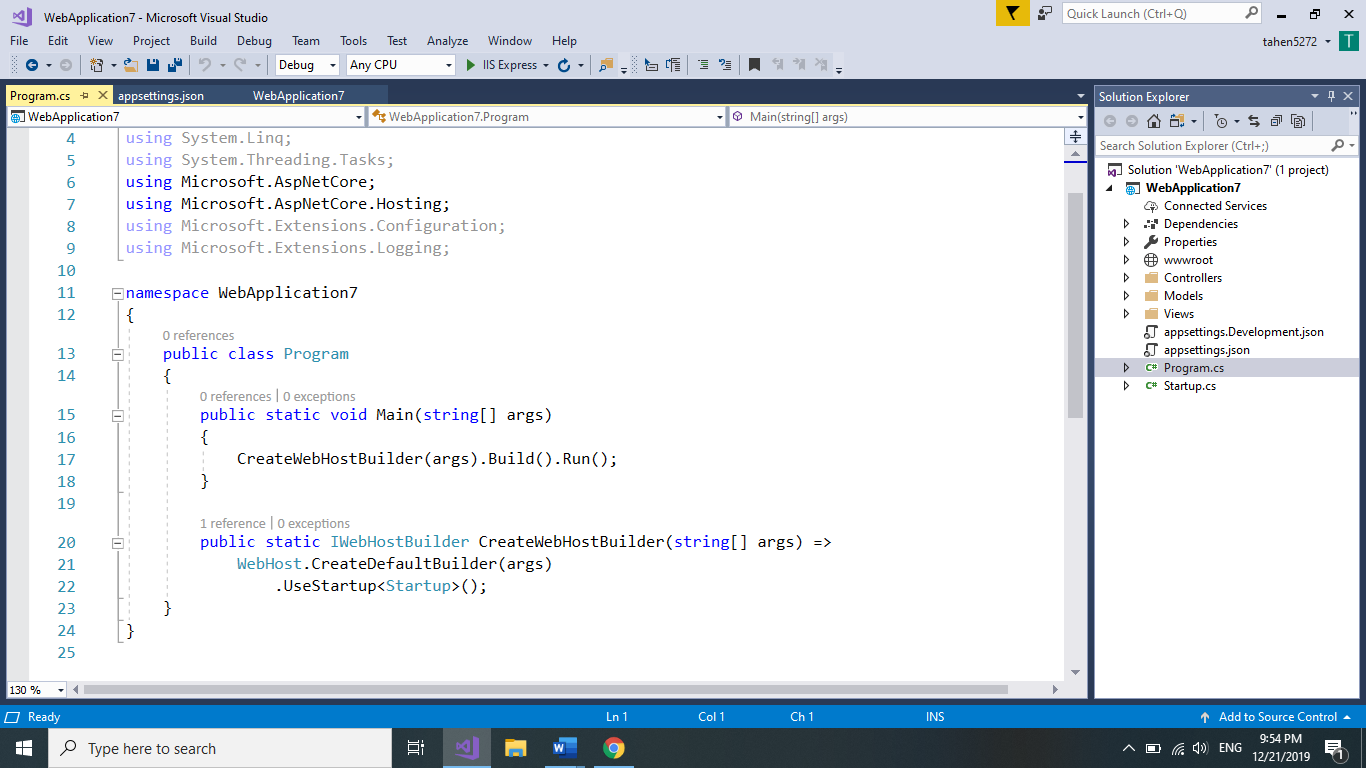
#### **Các thành phần trong Asp.Net Core**

Cách tạo ứng dụng Asp.Net Core: vào menu File chọn New Project. Trong cửa sổ tạo mới chọn template Web và chọn ASP.NET Core Web Application. Điền thông tin project và bấm OK.



Hình:

* Application Startup: trong project có 2 file đặc biệt là Program.cs, class program khởi tạo webservice trong hàm main (), còn class StartUp cấu hình yêu cầu pipeline từ ứng dụng.



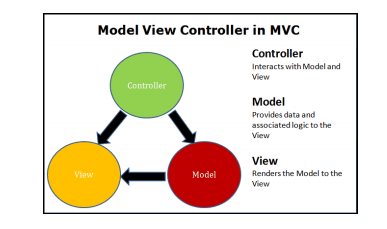
**Hình:**

* Dependency Injection: Đây là một kỹ thuật nhằm loại bỏ sự liên kết lẫn nhau giữa các object và các object dùng trong nó, loại bỏ các phụ thuộc, ASP.NET Core được thiết kế để hỗ trợ dependency injection, ASP.NET Core dùng DI để dẫn các service được xây dựng sẵn hoặc service tự tạo vào trong các phương thức, được config trong class Startup. Các đăng ký và quản lý lifetime của DI trong ASP.NET Core:
  + Transient
  + Scoped
  + Singleton

Lợi ích của DI đó là quản lý resource, test dễ dàng và khả năng mở rộng cao.

Service là các dịch vụ được tạo ra để xử lý một việc cụ thể (database, log, cache...), được quản lý bởi DI trong ASP.NET Core và được đăng ký bên trong StartUp của ứng dụng. ASP.NET Core được xây dựng theo cơ chế service, tất cả đều là service và quản lý bởi DI.

* Mô hình Model – View – Controller



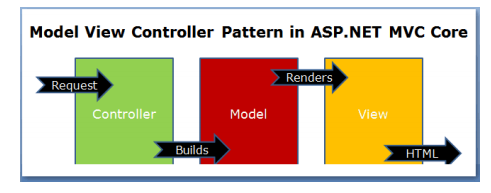
Hình:

Model-View-Controller (MVC) là mô hình kiến trúc xây dựng ứng dụng tách ra làm ba phần chính riêng biệt Model, View và Controller. MVC giúp bạn xây dựng ứng dụng dễ dàng bảo trì, cập nhật hơn các ứng dụng truyền thống.

* Model: là các class đại diện cho dữ liệu và logic để thực thi nghiệp vụ của dữ liệu đó. Thông thường, mỗi model sẽ đại diện cho một table trong database.
* View: là thành phần hiển thị giao diện người dùng (UI) của ứng dụng. Nhìn chung, view sẽ hiển thị dữ liệu từ model.
* Controller: Có nhiệm vụ xử lý các request từ trình duyệt (GET, POST, PUT ...) và sau đó trả về các data tương ứng cho view.

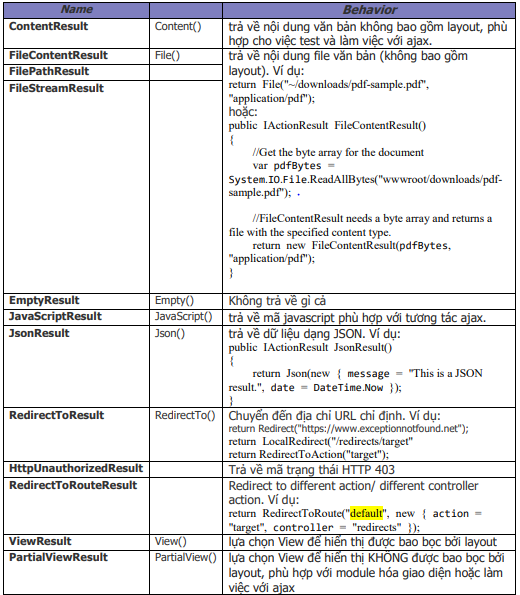
MVC giúp bạn tách biệt các phần của ứng dụng (input logic, business logic, UI logic) và cung cấp kết nối giữa các lớp này. Sự tách biệt này giúp bạn dễ dàng quản lý những ứng dụng phức tạp vì nó cho phép bạn làm việc trên một phần và không ảnh hưởng đến những phần khác.

Trong mô hình MVC, cái yêu cầu gửi đến (incoming request) được xử lý bởi các bộ điều khiển (controller). Mỗi phương thức công khai (public method) trong controller được gọi là một phương thức hành động (action method), nghĩa là bạn có thể gọi nó từ Web thông qua địa chỉ URL để thực hiện một action. Các controller được đặt trong thư mục Controllers của project.



Hình:

* Action Method: bất kỳ method public nào được gọi từ Controller thì phải gọi phương thức Action, cụ thể là thông qua đường dẫn URL trên trình duyệt. Action Method thường gọi service layer để phản hồi yêu cầu người dùng. Service layer thường truy vấn hoặc thay đổi cơ sở dữ liệu bằng cách sử dụng Data Access layer và sau đó chuyển kết quả trả về cho Model và chuyển ngược lại cho phương thức Action. Một số lưu ý khi tạo Action method:
* Luôn là public method, không thể là static method hay extension method.
* Các phương thức kế thừa trong Controller không được xem là action method.
* Tham số của action method không là ref, out.
* Không chứa thuộc tính [NonAction].
* Không thể overload các phương thức action.
* ActionResult: kết quả trả về của action là ActionResult, một kiểu dữ liệu chung chung từ kiểu đơn giản string, int, … cho đến kiểu phức tạp như JSON, html, file (dùng để download). IActionResult là một Interface, định nghĩa các hợp đồng kết quả của ActionResult. ActionResult là một lớp cơ sở trừu tượng cài đặt cho IActionResult. Các ViewResult, PartialViewResult, JsonResult, … đều kế thừa lớp ActionResult này. Các loại IactionResult:



Hình:

***Hình 3‑2***: Mô hình cấu trúc trong Spring framework

* Spring Core: Core package là phần lõi của framework, cung cấp những đặc tính IoC (Inversion of Control) và DI (Dependency Injection). Đây chính là nền tảng để xây dựng nên các thành phần khác trong hệ sinh thái của Spring Framework.

- BeanFactory đảm nhận việc sản sinh và móc nối sự phụ thuộc giữa các đối tượng trong file cấu hình. Là trung tâm của Spring Core là trái tim của một ứng dụng Spring, Spring framework được thiết kế từ lõi bằng cách sử dụng các POJO hay các Spring Bean. Spring Bean có thể được hiểu là các đối tượng Java đơn giản.

- DI (Dependency Injection): có thể hiểu là “tiêm” các thành phần được quản lý bởi IoC container vào một thành phần khác, là một sức mạnh nổi bật của Spring Framework.

- IoC (Inversion of Control): là một nguyên tắc lập trình, IoC đảo ngược điều khiển của chương trình nó giúp thay đổi luồng điều khiển chương trình một cách linh hoạt.

* Spring Context/ Application Context: phía trên core package là context package-cung cấp cách để truy cập đối tượng, kế thừa các tính năng của Spring Bean và bổ sung các tính năng hỗ trợ them như đa ngôn ngữ (i18n), truyền sự kiện, resource-loading, …
* Spring AOP (Aspect Oriented Programming): Spring AOP module tích hợp chức năng lập trình hướng đối tượng 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 dựng 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 them annotation (chú thích) vào source code để hướng dẫn Spring và làm thế nào để thực hiện các phương thức sự kiện đã được cái đặt sẵn.
* Spring DAO (Data Access Object): DAO package cung cấp cho tầng JDBC, bỏ bớt những coding dài dòng của JDBC và chuyển đổi mã lỗi được xác định bởi database vendor. JDBC package cung cấp cách lập trình tốt như declarative transaction management. Spring DAO hỗ trợ kết nối cơ sở dữ liệu Java không đồng nhất và ánh xạ O / R, giúp Spring hoạt động với một số công nghệ truy cập dữ liệu. Để truy cập dễ dàng và nhanh chóng vào tài nguyên cơ sở dữ liệu, khung công tác Spring cung cấp các lớp cơ sở DAO trừu tượng.
* Spring ORM (Object Relational Mapping): ORM package cung cấp tích hợp với object-relational mapping API bao gồm: JDO, Hibernate. Sử dụng ORM package có thể sử dụng tất cả các object-relational mapping đó kết hợp với các đặc tính trong Spring như declarative transaction management, resource management, DAO support classes.
* Spring Web module: Spring web package cung cấp đặc tính của web như: chức năng up file, khởi tạo IoC container sử dụng các tính năng lắng nghe servlet và web-oriented application context. Web module làm giảm bớt các công việc điều khiển nhiều request và gắn các tham số của request vào các đối tượng domain.

### **Các dự án trong Spring framework**

Thực ra Spring framework là tập hợp các dự án nhỏ lại với nhau, dựa trên các nguyên tắc thiết kế của Spring core. Gồm có các dự án như sau:

-Spring MVC được sử dụng trong việc thiết kế các ứng dụng chạy trên nền tảng web.

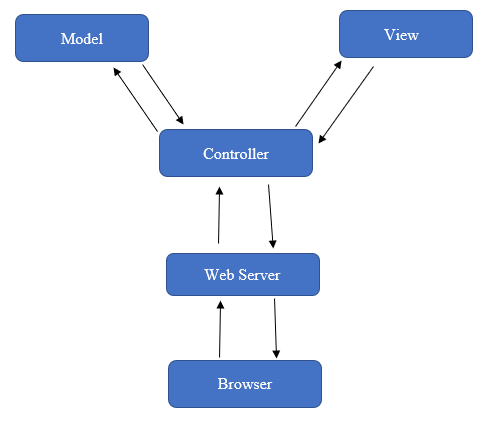
-Spring Security: Cung cấp các cơ chế xác thực (authentication) và phân quyền (authorization) cho ứng dụng của bạn.

-Spring Boot: là một framework giúp chúng ta phát triển cũng như chạy ứng dụng một cách nhanh chóng.

-Spring Batch.

-Spring Social.

-Spring Cloud.

- Spring IO.

### **Mô hình hoạt động Spring MVC**

***Hình 3‑3***: Mô hình MVC

Model gồm các lớp java có nhiệm vụ:

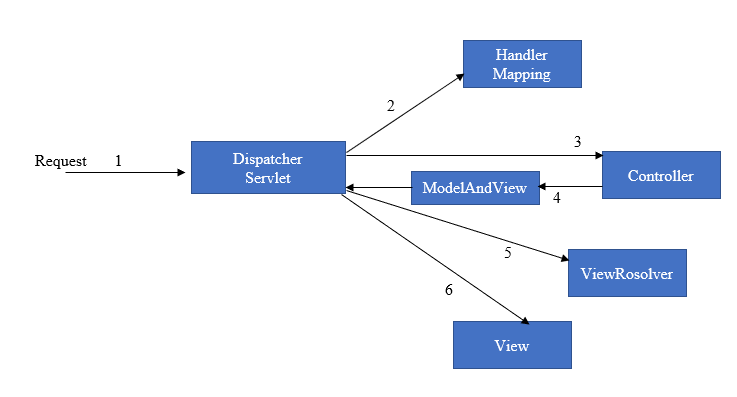
* Biểu diễn data và cho phép đọc/ghi data thông qua các phương thức getter/setter theo qui ước JavaBean.
* Buộc dữ liệu form giao diện tức là nhận dữ liệu từ tham số và cung cấp dữ liệu để trình bày len giao diện.
* Thi hành các yêu cầu: kết nối CSDL, tính toán, …
* Trả về các giá trị tính toán theo yêu cầu từ Controller.

View: bao gồn các mã tương tự như JSP, HTML, CSS, XML, … để hiển thị giao diện người dung.

Controller: đồng bộ giữa View và Model, một trang JSP thì tương ứng với lớp java nào để xử lý nó và ngược lại, kết quả sẽ được trả về trang jsp nào đó. Nó đóng vai trò điều tiết giữa model và view.

### **Sơ đồ xử lý request trong Spring MVC:**

Hình ‑: Sơ đồ xử lý request

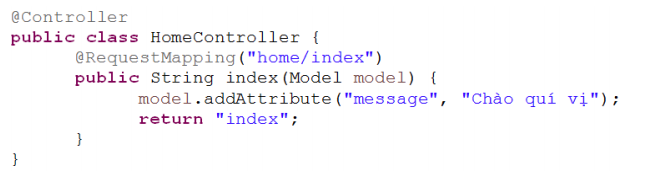
******Một Request được gửi bởi DispatcherServlet đến điều khiển (được chọn thông qua một bản đồ xử lý). Một khi điều khiển kết thúc, yêu cầu sau đó được gửi để xem (đó là lựa chọn thông qua ViewResolver) để làm cho đầu ra.

Trình tự xử lý yêu cầu:

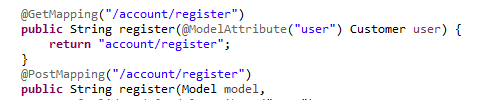
1. DispatcherServlet tiếp nhận Request từ trình duyệt.
2. DispatcherServlet gửi yêu cầu đến Handler Mapping (bộ phận quản lý các ánh xạ) để xác nhận controller nào sẽ xử lý yêu cầu này.
3. DispatcherServlet gửi yêu cầu đến Controller sau khi biết được Controller nào sẽ xử lý yêu cầu. Nếu yêu cầu đó cần truy xuất cơ sở dữ liệu thì Controller sẽ ủy nhiệm cho một business login hay nhiều hơn một service Object (Model) để lấy thông tin và gửi dữ liệu về cho Controller lúc này Controller đóng gói mô hình dữ liệu và tên của một view sẽ được tải lên thành đối tượng ModelAndView.
4. Gói ModelAndView được gửi trả về DispatcherServlet.
5. DispatcherServlet gửi gói ModelAndView cho ViewResolver để tìm xem trang web (JSP) nào sẽ được load lên.
6. DispatcherServlet load trang web đó lên cùng với dữ liệu của nó và trả kết quả về cho trình duyệt

### **Implementing Controllers**

Controller cung cấp truy cập vào các ứng dụng mà bạn thường xác định thông qua môt service interface. Controller điều khiển người dụng nhập vào và chuyển qua model, cuối cùng người dùng sẽ vào view để xem lại. Spring thực hiện controller một cách trừu tượng và cho phép tạo ra một loạt các controller mới.

Spring 2.5 giới thiệu một mô hình lập trình model từ MVC controllers có sử dụng các chú thích như @RequetMapping, @RequestParm, @ModelAttribute. @Controller là một lớp service phục vụ cho vai trò Controller, để có thể một controller thì cần thêm chú thích @Controller trên lớp đó. Các @Controllers hoạt động như một khuôn mẫu cho lớp, nó chỉ ra vai trò của mình. Dispartcher sẽ quét các lớp cho các method và detects @RequestMapping annotations.

***Hình 3‑5***: RequestMapping ánh xạ đường dẫn

Bạn có thể sử dụng các chú thích như là một đường dẫn. Thông thường thì các controller sẽ có một chú thích cụ thể, một đường dẫn cụ thể cho một form thực hiện chức năng với các method như GET/POST.

***Hình 3‑6***: Phương thức GET/POST

### **Handler Mapping**

Một trong những điều quan trọng trong một ứng dựng là việc xử lý web một cách thích hợp, xử lý các đường dẫn và truy cập vào các đường dẫn. DispathcherServlet cho phép DefaultAnnotationsHandlerMapping tìm kiếm các @RequestMapping. Để có thể xử lý các controller trong web một cách hợp lý thì có thể sử dụng các thuộc tính của HandlerMapping như:

* interceptors: (HandlerInterceptors), chặn các yêu cầu chức năng từ giao diện người dung, danh sách các chức năng, các controller chặn sử dụng.
* Cơ chế xử lý Mapping của Spring bao gồm điều khiển intercepter, nó hữu ích khi áp dụng vào các chức năng cụ thể với các yêu cầu nhất định. Có 3 phương thức xác định khi sử dụng: một là gọi trước khi xử lý thực tế được thực thi (preHandler ()), hai là được gọi sau khi xử lý được thực thi (postHandler ()), ba là gọi khi yêu cầu hoàn thành đã hoàn tất (afterCompletion ()). Ba phương thức cung cấp sự linh hoạt, đủ để hoàn tất các tiền xử lý và sau xử lý preHandler () phương thức trả về giá trị Boolean. Có thể sử dụng phương thức này để toát hoặc tiếp tục thực hiện chuỗi điều khiển. khi phương thức này trả về true, chuỗi đang thực thi sẽ tiếp tục xử lý, ngược lại khi phương thức trả về false DispatcherServlet giả sử đã chặn và trả về một view chỉ định và không thực hiện chuỗi interceptors.

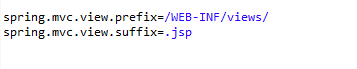
### **ViewResolver**

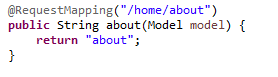
***Hình 3‑7***: Phương thức trong HandlerInterceptors

Views trong Spring được gán “view name” và được phân giải bởi ViewResolver. Các ViewResolver trong Spring:

* BeanNameViewResolver.
* FreeMarkerViewResolver.
* InternalResourceViewResolver.
* JasperReportsViewResolver.
* ResourceBundleViewResolver.
* UrlBasedViewResolver.
* VelocityLayoutViewResolver.
* VelocityViewResolver.
* XmlViewResolver.
* XsltViewResolver.

Ở phần này xin trình bày về InternalResourceViewResolver, vì đây là phần được sử dụng phổ biến nhất. View Resource sẽ ánh xạ Logical name của resource (Model và view được trả về bởi controller) đến một Physical view ví dụ như trang jsp.

Chúng ta có một Controller với đường dẫn là home/about và trả về trang about.

Chúng ta cần phải cấu hình đường dẫn chứa trang about.jsp là /WEB-INF/views và định dạng cho đuôi của trang about được trả về là .jsp. Khi đó nếu request của người dẫn đến controller about thì trên phía người dùng sẽ hiển thị trang giao diện “about.jsp”.

### **Các ưu điểm và khuyết điểm của Spring MVC Web Framework**

Ưu điểm:

* Spring là một framework Java mạnh mẽ được sử dụng trong những ứng dụng Java có phạm vi lớn. Nó cung cấp những dịch vụ Enterprise đến những Plain Old Java Objects (POJOs). Cơ chế IoC giúp ứng dụng đạt được sự đơn giản hoá và tăng khả năng kiểm tra lỗi.
* Spring MVC cung cấp một sự phân chia rất rõ ràng, rành mạch giữa những Controller, Java Bean models và Views.
* Spring MVC rất linh hoạt, toàn bộ Spring MVC được xây dựng dựa trên những interfaces. Mọi phần của Spring MVC framework được cấu hình thông qua việc lắp ghép những interface, class tiện ích sẵn có, và thậm chí được tạo bởi người dùng.
* Spring không chỉ sử dụng công nghệ JSP mà còn có thể dễ dàng tích hợp các công nghệ view khác như Velocity, XSLT, FreeMarker, XL, …
* Cung cấp cơ chế che dấu nền công nghệ bên dưới, trang web khi hiển thị chỉ có extension là .html, không thể biết được bên dưới ta sử dụng công nghệ, kỹ thuật gì, JSP hay Velocity, XLST, … thậm chí là những công nghệ view được custom bởi người dùng.
* Spring Controller được cấu hình thông qua IoC như mọi đối tượng khác. Điều này làm chúng dễ dàng được test, và được tích hợp dễ dàng với những đối tượng khác được quản lý bởi Spring.
* Kết buộc trực tiếp các input từ view với những đối tượng domains.

Khuyết điểm:

* Cấu hình phức tạp và cồng kềnh => không phát huy được sức mạnh khi áp dụng cho các ứng dụng quy mô nhỏ mà có thể ngược lại còn làm cho ứng dụng phức tạp.

# THÀNH PHẦN TÁC NHÂN

## ĐẶC TẢ ACTOR

Khách hàng: Khách hàng xem và mua hàng tại Website. Website cung cấp nhiều chức năng cho cả khách hàng vãng lai và khách hàng là hội viên. Các chức năng phục vụ cho khách hàng nhằm giúp khách hàng có nhiều tiện ích khi xem và mua hàng như: xem tin tức, xem chi tiết sản phẩm, chon sản phẩm bỏ vào giỏ hàng, tạo đơn đặt hàng. Khách hàng vãng lai có thể đăng ký tài khoản và mật khẩu để tiện cho lần giao dịch tiếp theo.

Bộ phận quản trị: Người quản trị website đăng nhập vào hệ thống nhằm mục đích quản lý thông tin, có toàn quyền sử dụng cấu hình và thêm xóa sửa cơ sở dữ liệu, xem các chi tiết đơn hàng, quản lý danh mục, quản lý sản phẩm…

Xác định các chức năng của Actor:

Chức năng đối với khách hàng:

* Chức năng đăng ký tài khoản khách hàng
* Chức năng đăng nhập hệ thống
* Chức năng thoát khỏi hệ thống
* Chức năng sửa thông tin thành viên
* Chức năng tìm kiếm sản phẩm
* Chức năng thêm hàng vào giỏ hàng
* Chức năng xem giỏ hàng

+ Chức năng xóa sản phẩm trong giỏ hàng

+ Chức năng cập nhật giỏ hàng

* Chức năng tạo đơn hàng
* Chức năng xem sản phẩm
* Chức năng xem chi tiết sản phẩm
* Chức năng xem tin tức

Chức năng đối với khách hàng vãng lai:

* Chức năng đăng ký tài khoản khách hàng
* Chức năng đăng nhập hệ thống
* Chức năng tìm kiếm sản phẩm
* Chức năng xem chi tiết sản phẩm
* Chức năng xem tin tức

Chức năng đối với người quản trị hệ thống:

* Chức năng đăng nhập hệ thống
* Chức năng thoát khỏi hệ thống
* Chức năng quản lý danh mục sản phẩm

+ Thêm danh mục

+ Xoá danh mục

+ Cập nhật danh mục

* + - Chức năng quản lý sản phẩm

+ Thêm sản phẩm

+ Xoá sản phẩm

+ Cập nhật sản phẩm

+ Tìm kiếm sản phẩm

* + - Chức năng quản lý đơn đặt hàng

+ Duyệt đơn đặt hàng

+ Xoá đơn đặt hàng

+ Xem chi tiết đơn đặt hàng

+ Tìm kiếm đơn đặt hàng

* + - Chức năng quản lý tin tức

+ Tạo mới tin tức

+ Xoá tin tức

+ Cập nhật tin tức

* + - Chức năng xem sản phẩm
    - Chức năng quản lý nhà cung cấp.

+ Thêm nhà cung cấp.

+ Xoá nhà cung cấp.

+ Cập nhật nhà cung cấp.

* Chức năng xem chi tiết sản phẩm.
* Chức năng xem tin tức.
* Chức năng quản lý tổng hợp thống kê

+ Thống kê doanh số bán hàng theo loại.

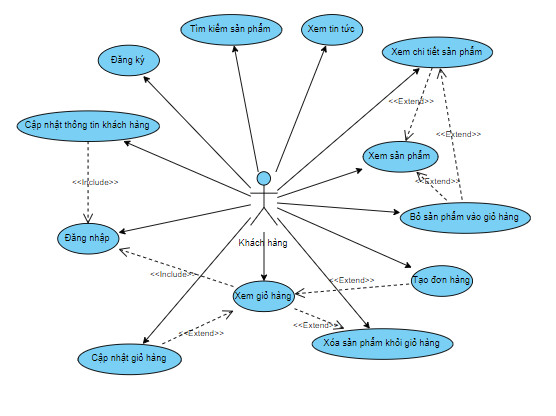
+ Thống kê doanh số theo tháng.

+ Thống kê doanh số theo năm.

+ Thống kê doanh số theo quý.

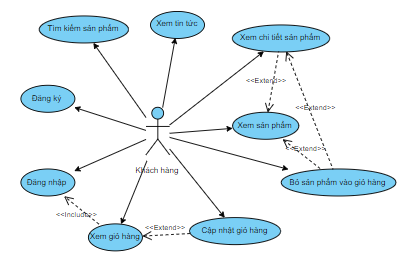
## SƠ ĐỒ USE-CASE TỔNG QUÁT

* Use-Case Diagram cho khách hàng



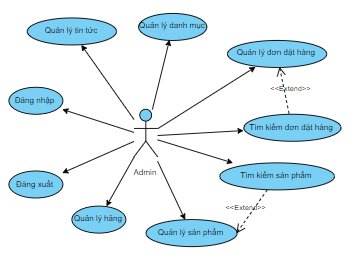
***Hình 4‑1***: Use case Khách hàng

* Use-Case Diagram cho khách vãng lai



***Hình 4‑2***: Use case Khách vãng lai

* Use-Case Diagram cho bộ phận quản trị hệ thống

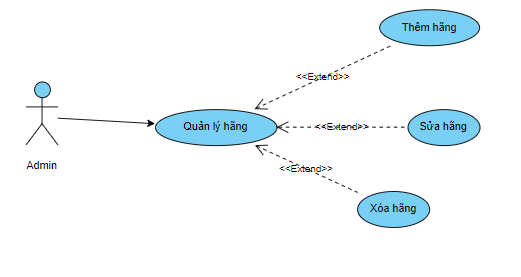


***Hình 4‑3***: Use case quản lí hệ thống

## ĐẶC TẢ USE CASE CHI TIẾT

### **Sơ đồ Use Case quản lý hang sản phẩm**:

##### Sơ đồ chi tiết:



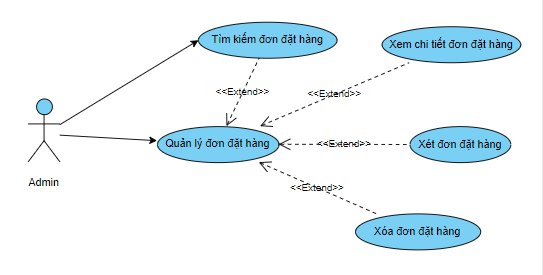
***Hình 4‑4***: Sơ đồ usecase quản lý hãng

##### Đặc tả Use Case quản lý hãng sản phẩm:

|  |  |
| --- | --- |
| **Tên Use case** | **Quản lý hãng sản phẩm** |
| **Admin** | **Quản trị viên** |
| **Mô tả** | **Quản trị viên xem thông tin các hang mà cửa hàng bán đăng ký trên website và thực hiện các chức năng: Thêm, Xóa, Tra cứu và Cập nhật thông tin của hãng.** |
| Luồng sự kiện chính | 1. Admin chọn chức năng Quản lý hang sản phẩm. 2. Hệ thống hiển thị màn hình Quản lý hang sản phẩm.  * Extend Use Case Themhang. * Extend Use Case XoaHang. * Extend Use Case CapNhatthongtinHang. |
| Luồng sự kiện phụ | 1. Admin nhấn nút Thoát 2. Hệ thống hủy màn hình Quản lý hãng. |
| <Extend Use Case> | **ThemKhachHang**   1. Admin nhập thông tin Hãng. 2. Kiểm tra thông tin không rỗng. 3. Admin nhấn nút thêm. 4. Cập nhật CSDL. 5. Hiển thị thông tin cập nhật. 6. Admin nhấn nút thoát. 7. Hủy màn hình cập nhật Quản lý Hãng.   **Rẽ nhánh 1:**  2.1. Kiểm tra thông tin bị rỗng.  3.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 2:**  4.1. Thông tin bị trùng.  5.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 3:**  4.2. Kiểm tra thông tin khách hàng hợp lệ.  4.3. Cập nhật CSDL. |
| <Extend Use Case> | **XoaHang**   1. Admin chọn thông tin hang từ danh sách hãng. 2. Hiển thị thông tin Hãng lên Form. 3. Admin nhấn nút xóa. 4. Cập nhật CSDL. 5. Hiển thị thông tin cập nhật. 6. Admin nhấn nút thoát.   Hủy màn hình Quản Lý Hãng. |
| <Extend Use Case> | **CapNhatthongtinHang**   1. Admin chọn thông tin hãng từ danh sách hãng. 2. Hiển thị thông tin hãng lên Form. 3. Admin thay đổi thông tin hãng. 4. Kiểm tra thông tin không rỗng. 5. Admin nhấn nút sửa. 6. Cập nhật CSDL. 7. Hiển thị thông tin cập nhật. 8. Admin nhấn nút thoát. 9. Hủy màn hình Quản Lý hãng.   **Rẽ nhánh**:  4.1. Kiểm tra thông tin bị rỗng.  5.1. Lập lại bước 1 luồng sự kiện chính. |

### Sơ đồ Use Case quản lý hóa đơn

##### Sơ đồ chi tiết:



***Hình 4‑5***: Sơ đồ usecase quản lý hóa đơn

##### Đặc tả Use Case quản lý đơn hàng:

|  |  |
| --- | --- |
| **Tên Use case** | **Quản lý đơn hàng** |
| **Admin** | **Quản trị viên** |
| **Mô tả** | **Quản trị viên xem thông tin các hóa đơn trên website và thực hiện các chức năng: xem, tìm kiếm, xử lý đơn hàng.** |
| Luồng sự kiện chính | 1. Admin chọn chức năng Quản lý Hóa Đơn. 2. Hệ thống hiển thị màn hình Quản lý Hóa Đơn.  * Extend Use Case ThemHoaDon. * Extend Use Case SuaHoaDon. * Extend Use Case XoaHoaDon. |
| Luồng sự kiện phụ | 1. Admin nhấn nút Thoát 2. Hệ thống hủy màn hình Quản lý Hóa Đơn. |
| Extend Use Case>  **Tìm kiếm đơn hàng** | **Tìm kiếm đơn hàng**  **Actor nhập thông tin đơn đặt hàng cần tìm**  **Actor nhấn nút tìm**  **Kiểm tra thông tin**  **Hiển thị thông tin đơn hàng lên Form**  **Rẽ nhánh 1:**  1.1. Kiểm tra thông tin không hợp lệ.  ->Thông báo lỗi “Không tìm thấy đơn hàng” |
| <Extend Use Case>  **Xem chi tiết đơn hàng** | **Xem chi tiết đơn hàng**  Actor chọn đơn hàng cần xem chi tiết đơn hàng.  Actor nhấn nút xem.  Hiển thị chi tiết đơn hàng.  **Rẽ nhánh 1:**  1.1. Kiểm tra đơn hàng không tồn tại.   1. -> Thông báo lỗi “Đơn hàng không còn tồn tại”. |
| <Extend Use Case>  **Xử lý đơn hàng** | **Xử lý đơn hàng**  Actor chọn đơn hàng cần xử lý  Actor nhấn nút thao tác xử lý.  Cập nhật CSDL.  Hiển thị thông tin cập nhật.  **Rẽ nhánh 1:**  1.1. Kiểm tra đơn hàng đã ở trạng thái thành công.  ->Thông báo lỗi “Không thể chuyển trạng Thành Công sang trạng thái khác!”. |

### **Sơ đồ Use Case Quản lí sản phẩm**:

***Hình 4‑6***: Sơ đồ usecase quản lý sản phẩm

##### Sơ đồ chi tiết:

##### Đặc tả Use Case quản lý sản phẩm:

|  |  |
| --- | --- |
| **Tên Use case** | **Quản lý sản phẩm** |
| **Admin** | **Quản trị viên** |
| **Mô tả** | **Quản trị viên xem thông tin các sản phẩm trên cơ sở dữ liệu và thực hiện các chức năng: Thêm, Xóa và Sửa sản phẩm.** |
| Luồng sự kiện chính | 1. Admin chọn chức năng Quản lý sản phẩm. 2. Hệ thống hiển thị màn hình Quản lý sản phẩm.  * Extend Use Case Themsanpham. * Extend Use Case Suasanpham. * Extend Use Case Xoasanpham. * *Extend Use Case Timsanpham* |
| Luồng sự kiện phụ | 1. Admin nhấn nút Thoát 2. Hệ thống hủy màn hình Quản lý sản phẩm |
| <Extend Use Case> | **Themsanpham**   1. Admin nhập thông tin sản phẩm. 2. Kiểm tra thông tin không rỗng. 3. Admin nhấn nút thêm. 4. Cập nhật CSDL. 5. Hiển thị thông tin cập nhật. 6. Admin nhấn nút thoát. 7. Hủy màn hình cập nhật Quản lý sản phẩm   **Rẽ nhánh 1:**  2.1. Kiểm tra thông tin bị rỗng.  3.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 2:**  4.1. Thông tin bị trùng.  5.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 3:**  4.2. Kiểm tra thông tin dịch vụ hợp lệ.  4.3. Cập nhật CSDL. |
| <Extend Use Case> | **Xoasanpham**   1. Admin chọn thông tin sản phẩm từ danh sách sản phẩm. 2. Hiển thị thông tin sản phẩm lên Form. 3. Admin nhấn nút xóa. 4. Cập nhật CSDL. 5. Hiển thị thông tin cập nhật. 6. Admin nhấn nút thoát. 7. Hủy màn hình Quản Lý sản phẩm. |
| <Extend Use Case> | **Suasanpham**   1. Admin chọn thông tin sản phẩm từ danh sách sản phẩm. 2. Hiển thị thông tin sản phẩm lên Form. 3. Admin nhấn nút sửa. 4. Cập nhật CSDL. 5. Hiển thị thông tin cập nhật. 6. Admin nhấn nút thoát. 7. Hủy màn hình Quản Lý sản phẩm.   **Rẽ nhánh 1:**  2.1. Kiểm tra thông tin bị rỗng.  3.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 2:**  4.1. Thông tin bị trùng.  5.1. Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 3:**  4.2. Kiểm tra thông tin dịch vụ hợp lệ.  4.3. Cập nhật CSDL. |
| <Extend Use Case> | **Timsanpham**   1. Admin nhập thông tin sản phẩm cần tìm. 2. Kiểm tra thông tin không rỗng. 3. Admin nhấn nút tìm kiếm. 4. Hiển thị thông tin sản phẩm muốn tìm. 5. Admin nhấn nút thoát. 6. Huỷ màn hình màn hình Quản lý sản phẩm   **Rẽ nhánh 1:**  3.1. Tìm gần đúng.  4.1. Hiển thông tin sản phẩm lên Form.  **Rẽ nhánh 2:**  3.2. Tìm chính xác.  4.2 Hiển thị thông tin sản phẩm lên Form. |

### **Sơ đồ Use Case tìm kiếm sản phẩm**:

***Hình 4‑7***: Use case tìm kiếm sản phẩm

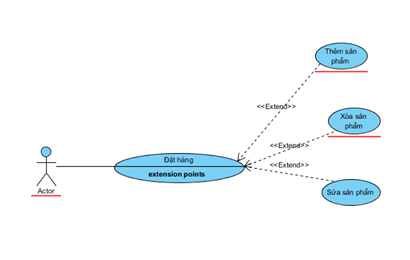
##### Sơ đồ chi tiết:

##### Đặc tả Use Case tìm kiếm sản phẩm:

|  |  |
| --- | --- |
| **Tên Use case** | **Tìm kiếm sản phẩm** |
| **Actor** | **Khách hàng** |
| **Mô tả** | **Khách hàng xem sản phẩm, tìm kiếm sản phẩm có trong cơ sở dữ liệu và xem giỏ hàng** |
| Luồng sự kiện chính | 1. Actor chọn chức năng Tìm kiếm sản phẩm. 2. Hệ thống hiển thị màn hình Tìm kiếm sản phẩm  * Extend Use Case tìm theo giá * Extend Use Case tìm theo loại |
| Luồng sự kiện phụ | 1. Actor nhấn nút Thoát 2. Hệ thống hủy màn hình Tìm kiếm sản phẩm. |
| <Extend Use Case> | **Xem theo giá**   1. Actor chọn giá cần xem 2. Kiểm tra danh sách sản phẩm có trong tầm giá 3. Hiển thị danh sách theo giá 4. Actor nhấn nút thoát. 5. Hủy màn hình xem sản phẩm theo giá   **Rẽ nhánh 1:**  1.1. Kiểm tra thông tin bị rỗng.  1.2. Xuất ra màn hình không nhìn thấy  **Rẽ nhánh 2:**  2.1. Kiểm tra không hợp lệ  2.2. Hiển thị danh sách theo giá |
| <Extend Use Case> | **Xem theo loại**   1. Actor chọn loại sản phẩm cần xem 2. Kiểm tra danh sách sản phẩm theo loại sản phẩm 3. Hiển thị danh sách theo loại sản phẩm 4. Actor ấn nút thoát   **Rẽ nhánh** 1:  1.1. Kiểm tra thông tin bị rỗng.  1.2. Xuất ra màn hình không nhìn thấy  **Rẽ nhánh 2:**  2.1. Kiểm tra không hợp lệ  2.2. Hiển thị danh sách theo loại |

### **Sơ đồ Use Case Đặt hàng**:

##### Sơ đồ chi tiết:



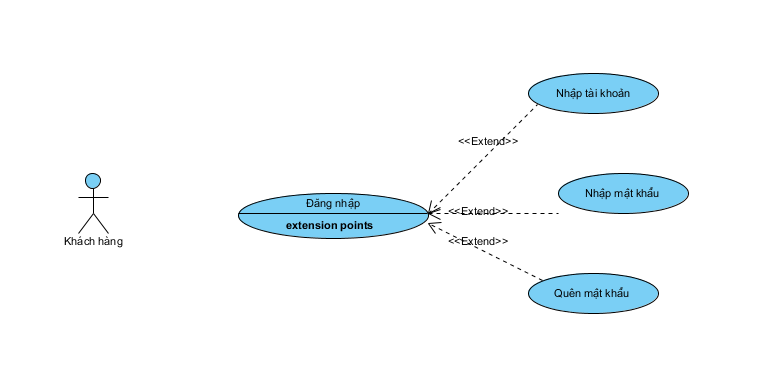
***Hình 4‑8***: Use case đặt hàng

##### Đặc tả Use Case dặt hàng:

|  |  |
| --- | --- |
| **Tên Use case** | **Đặt hàng** |
| **Actor** | **Khách hàng** |
| **Mô tả** | **Khách hàng xem sản phẩm, đặt sản phẩm có trong cơ sở dữ liệu.** |
| Luồng sự kiện chính | 1. Actor chọn chức năng Đặt hàng. 2. Hệ thống hiển thị màn hình Đặt hàng  * Extend Use Case thêm sản phẩm * Extend Use Case sửa sản phẩm * Extend Use Case xóa sản phẩm |
| Luồng sự kiện phụ | 1. Actor nhấn nút Thoát 2. Hệ thống hủy màn hình Đặt hàng. |
| <Extend Use Case> | **Thêm sản phẩm**   1. Actor chọn sản phẩm cần mua 2. Thêm sản phẩm vào giỏ hàng 3. Actor nhấn nút thoát. 4. Hủy màn hình them sản phẩm.   **Rẽ nhánh 1:**  1.1. Kiểm tra giỏ hàng có bị rỗng.  1.2. Nếu rỗng quay lại chọn sản phẩm  **Rẽ nhánh 2:**  2.1. Kiểm tra sản phẩm trùng  2.2. Tăng sản phẩm lên 1 |
| <Extend Use Case> | **Sửa sản phẩm**   1. Actor chọn loại sản phẩm cần sửa 2. Sửa sản phẩm 3. Actor nhấn nút thoát. 4. Actor ấn nút thoát   **Rẽ nhánh** 1:  1.1. Kiểm tra giỏ hàng có bị rỗng.  1.2. Nếu rỗng quay lại chọn sản phẩm  **Rẽ nhánh 2:**  2.1. Kiểm tra sản phẩm trùng  2.2. Nếu trùng quay lại sửa |
| <Extend Use Case> | **Xóa sản phẩm**   1. Actor chọn loại sản phẩm cần sửa 2. Sửa sản phẩm 3. Actor nhấn nút thoát. 4. Actor ấn nút thoát |

### **Sơ đồ Use Case đăng nhập**:

##### Sơ đồ chi tiết:



***Hình 4‑9***: Use case đăng nhập

##### Đặc tả Use Case đăng nhập:

|  |  |
| --- | --- |
| **Tên Use case** | **Đăng nhập** |
| **Actor** | **Khách hàng** |
| **Mô tả** | **Khách hàng Đăng nhập** |
| Luồng sự kiện chính | 1. Actor chọn chức năng Đăng nhập 2. Hệ thống hiển thị màn hình Đăng nhập  * Extend Use Case Nhập tài khoản * Extend Use Case Nhập mật khẩu * Extend Use Case Quên mật khẩu |
| Luồng sự kiện phụ | 1. Actor nhấn nút Thoát 2. Hệ thống hủy màn hình Đăng nhập. |
| <Extend Use Case> | **Nhập tài khoản**   1. Actor chọn đăng nhập 2. Actor Nhập tài khoản vào mục đăng nhập 3. Actor nhấn nút đăng nhập 4. Actor nhấn nút thoát. 5. Hủy màn hình Đăng nhập   **Rẽ nhánh 1:**  1.1. Kiểm tra thông tin tài khoản rỗng  1.2. Quay lại đăng nhập  **Rẽ nhánh 2:**  2.1. Kiểm tra thông tin tài khoản hợp lệ  2.2. Hoạt động đăng nhập vào tài khoản |
| <Extend Use Case> | **Nhập mật khẩu**   1. Actor chọn đăng nhập 2. Nhập mật khẩu vào mục đăng nhập 3. Actor nhấn nút đăng nhập 4. Hiển thị danh sách theo loại sản phẩm 5. Actor ấn nút thoát   **Rẽ nhánh** 1:  1.1. Kiểm tra mật khẩu rổng  1.2. Quay lại đăng nhập mật khẩu  **Rẽ nhánh 2:**  2.1. Kiểm tra mật khẩu hợp lệ  2.2. Hoạt động đăng nhập vào tài khoản |
| <Extend Use Case> | **Quên mật khẩu**   1. Actor chọn nút quên mật khẩu 2. Hiển thị thông tin quên mật khẩu lên Form 3. Actor nhập email 4. Actor kiểm trả email để lấy lại mật khẩu 5. Actor nhấn nút thoát 6. Hủy màn hình quên mật khẩu   **Rẽ nhánh** 1:  1.1. Kiểm tra mật khẩu rổng  1.2. Quay lại quên mật khẩu |

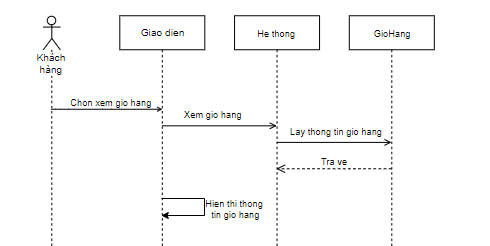
# THÀNH PHẦN XỬ LÝ

## SƠ ĐỒ TUẦN TỰ

***Hình 5‑1***: Sơ đồ thêm sản phẩm vào giỏ hàng

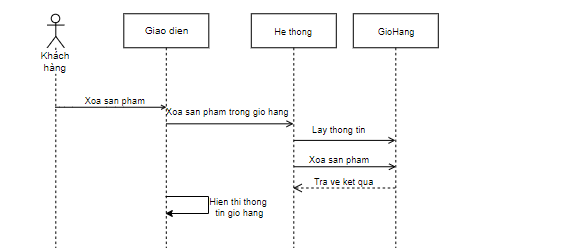
### **Sơ đồ thêm sản phẩm vào giỏ hàng**

### **Sơ đồ xem giỏ hàng**



***Hình 5‑2***: Xem giỏ hàng

### **Sơ đồ xóa sản phẩm khỏi giỏ hàng**

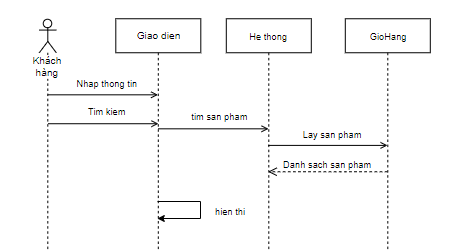


***Hình 5‑3***: Xóa sản phẩm khỏi giỏ hàng

### **Sơ đồ chỉnh sửa thông tin trong giỏ hàng**

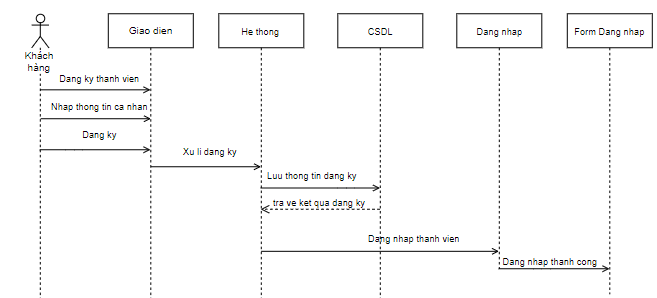
***Hình 5‑4***: Chỉnh sửa thông tin giỏ hàng

### **Sơ đồ tìm kiếm sản phẩm**



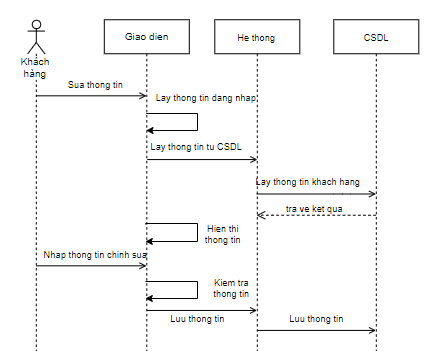
***Hình 5‑5***: Tìm kiếm sản phẩm

### **Sơ đồ đăng ký thành viên của khách hàng**



***Hình 5‑6***: Sơ đồ đăng ký thành viên của Khách hàng

### **Sơ đồ chỉnh sửa thông tin cá nhân của khách hàng**

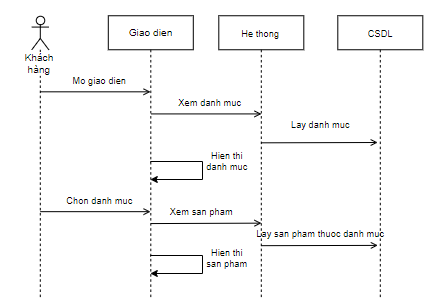


***Hình 5‑7***: Chỉnh sửa thông tin cá nhân của Khách hàng

### **Sơ đồ đăng nhập thành viên của khách hàng**

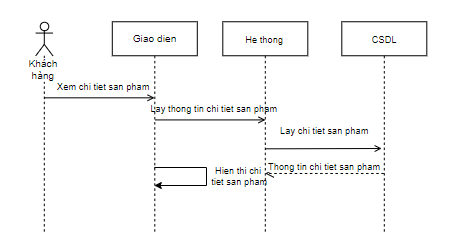
***Hình 5‑8***: Đăng nhập thành viên của Khách hàng

### **Sơ đồ xem sản phẩm của khách hàng**



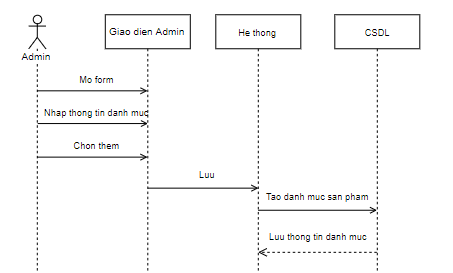
***Hình 5‑9:* Xem sản phẩm của Khách hàng**

### **Sơ đồ xem chi tiết sản phẩm**



***Hình 5‑10***: Xem chi tiết sản phẩm

### **Sơ đồ thêm danh mục của người quản lý**

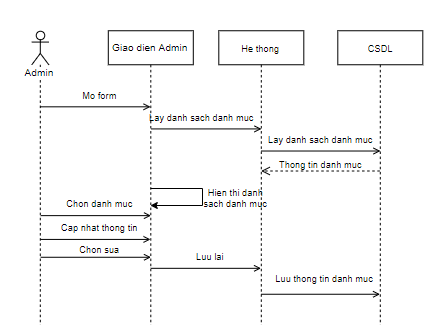


***Hình 5‑11***: Thêm danh mục của người quản lý

### **Sơ đồ xóa danh mục của người quản lý**

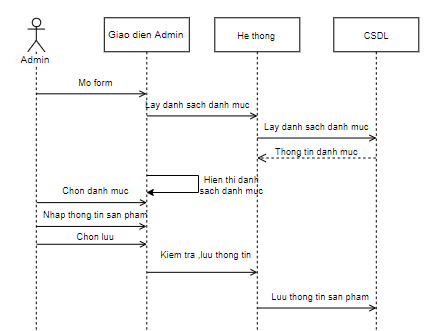
***Hình 5‑12***: Xóa danh mục của người quản lý

### **Sơ đồ sửa danh mục**



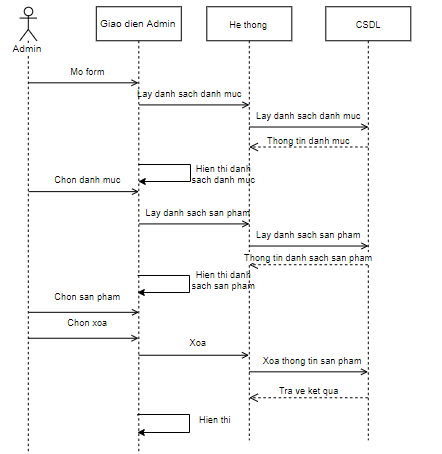
***Hình 5‑13***: Sửa danh mục của người quản lý

### **Sơ đồ thêm sản phẩm mới**



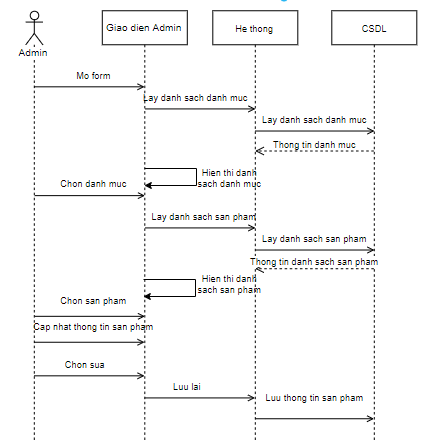
***Hình 5‑14***: Thêm sản phẩm mới

### **Sơ đồ xóa sản phẩm trong trang quản lý**



***Hình 5‑15:*** Xóa sản phẩm trong trang quản lý

### **Sơ đồ sửa sản phẩm của người quản lý**



***Hình 5‑16***: Sửa sản phẩm của người quản lý

# THÀNH PHẦN DỮ LIỆU

## PHÂN TÍCH Ở MỨC QUAN NIỆM

***Hình 6‑1***: Sơ đồ Cơ sở dữ liệu

### **Sơ đồ ER**

.

### **Mô tả các loại thực thể/lớp**

* Loại thực thể Khách Hàng**[[1]](#footnote-1)**

| **Mô tả:** Loại thực thể Khách Hàng gồm những khách hàng đã đăng ký vào hệ thống | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** |
| Mã khách hàng | Nvarchar(50) | x | x | x | mã khách hàng |
| Tên khách hàng | Nvarchar(50) |  |  | x | tên khách hàng |
| Số điện thoại | Nvarchar(50) |  |  | x | Số điện thoại khách hàng |
| email | Nvarchar(50) |  |  | x | Email khách hàng |

* Loại thực thể Hóa Đơn**[[2]](#footnote-2)**

| **Mô tả:** Loại thực thể Hóa Đơn gồm những hóa đơn mà khách hàng đã mua trên hệ thống | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** |
| Số hóa đơn | int | x | x | x | mã hóa đơn |
| Ngày lập hóa đơn | datetime |  |  | x | Ngày đặt |
| Địa chỉ giao hàng | Nvarchar(50) |  |  | x | Địa chỉ giao hàng |

* Loại thực thể chi tiết hóa đơn**[[3]](#footnote-3)**

| **Mô tả:** Loại thực thể Chi Tiết thức ăn là chi tiết của thực thể hóa đơn | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** | |
| Tên sản phẩm | Nvarchar(50) |  |  | x | Tên sản phẩm | |
| Số lượng sản phẩm | int |  |  | x | Số lương | |
| Số tiền | float |  |  | x | Giá mua | |

Loại thực thể thức ăn**[[4]](#footnote-4)**

| **Mô tả:** Loại thực thể thức ăn gồm những sản phẩm mà cửa hàng cung cấp | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** | |
| Mã sản phẩm | int | x | x | x | Mã sản phẩm | |
| Tên sản phẩm | Nvarchar(50) |  |  | x | Tên sản phẩm | |
| Số lượng sản phẩm | int |  |  | x | Số lượng của sản phẩm | |
| Ngày nhập sản phẩm | datetime |  |  | x | Ngày nhập sản phẩm | |
| Giá | float |  |  | x | Giá của sản phẩm | |

* Loại thực thể loại sản phẩm**[[5]](#footnote-5)**

| **Mô tả:** Loại thực thể danh mục đồ ăn gồm nhóm sản phẩm mà cửa hàng cung cấp | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** |
| Mã loại | NVarchar(50) | x |  | x | Mã của loại |
| Tên loại tiếng Việt | NVarchar(50) |  |  | x | Tên loại theo tiếng Việt |
| Tên loại tiếng Anh | NVarchar(50) |  |  | x | Tên loại theo tiếng Anh |

* Loại thực thể nhà sản xuất**6**

| **Mô tả:** Loại thực thể chi tiết kích thước là chi tiết của thực thể hóa đơn | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu** | **K** | **U** | **M** | **Diễn giải** |
| Mã nhà sản xuất | NVarchar(50) | x | x | x | Mã của nhà sản xuất sản phẩm |
| Tên của nhà sản xuất | NVarchar(50) |  |  | x | Tên của nhà sản xuất sản phẩm |

## MÔ TẢ CÁC RÀNG BUỘC DỮ LIỆU

1. Mô tả ràng buộc RBPT1: thực thể sản phẩm ràng buộc với thực thể loại sản phẩm.
2. Mô tả ràng buộc RBPT2: thực thể sản phẩm ràng buộc với thực thể hãng sản xuất.
3. Mô tả ràng buộc RBPT3: thực thể khách hàng ràng buộc với thực thể hóa đơn.
4. Mô tả ràng buộc RBPT4: thực thể sản phẩm ràng buộc với thực thể chi tiết hóa đơn.
5. Mô tả ràng buộc RBPT5: thực thể chi tiết hóa đơn ràng buộc với thực thể hóa đơn.

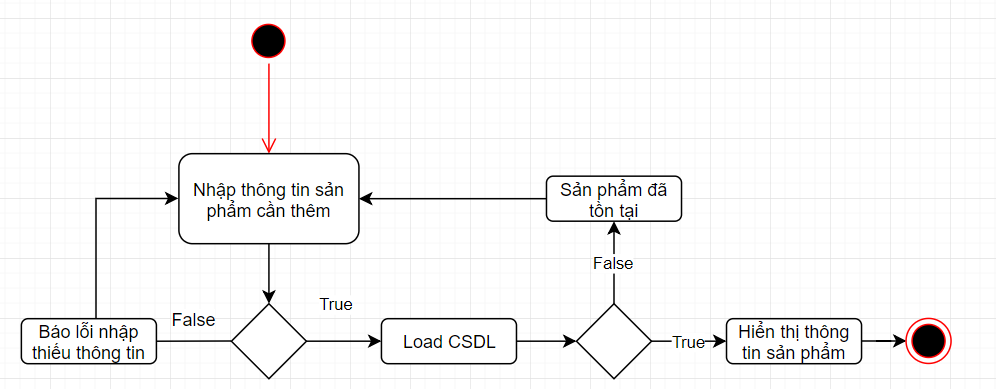
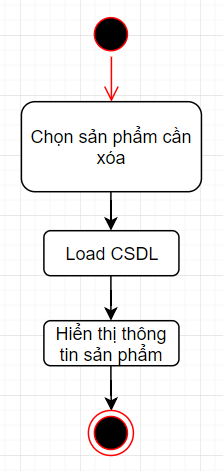
## SƠ ĐỒ HOẠT ĐỘNG

### **Sơ đồ hoạt động đăng nhập**

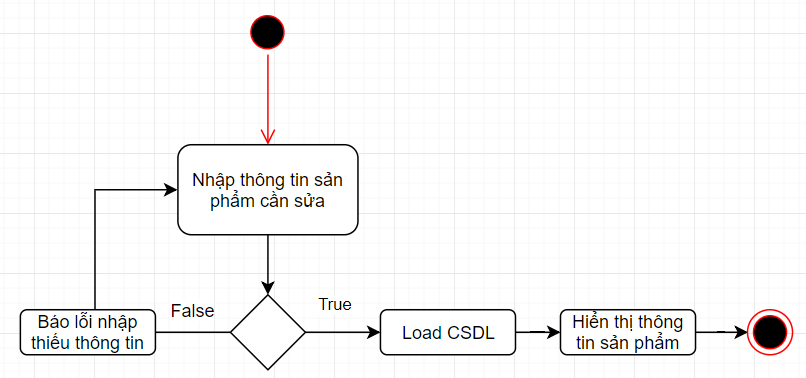
***Hình 6‑2***: Mô tả hoạt động đăng nhập

### **Sơ đồ hoạt động quản lý sản phẩm**

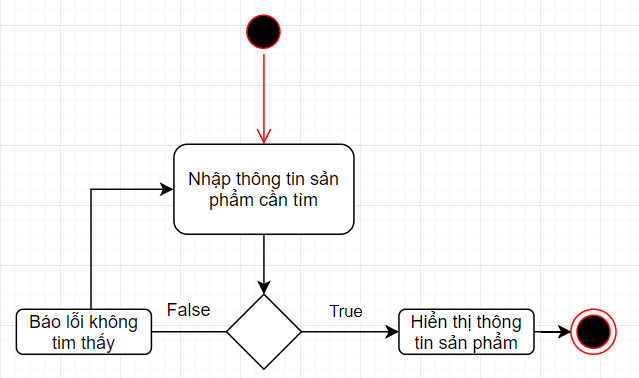
***Hình 6‑3***: Mô tả hoạt động xóa sản phẩm



***Hình 6‑4***: Mô tả hoạt động thêm sản phẩm

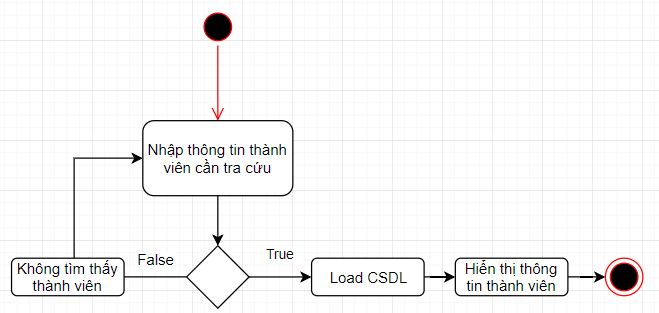


***Hình 6‑5***: Mô tả hoạt động sửa sản phẩm

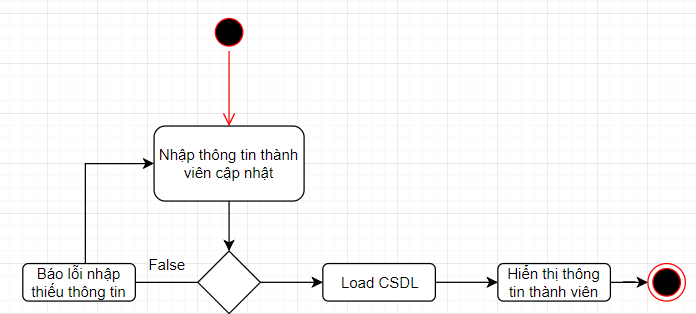


***Hình 6‑6***: Mô tả hoạt động tìm sản phẩm

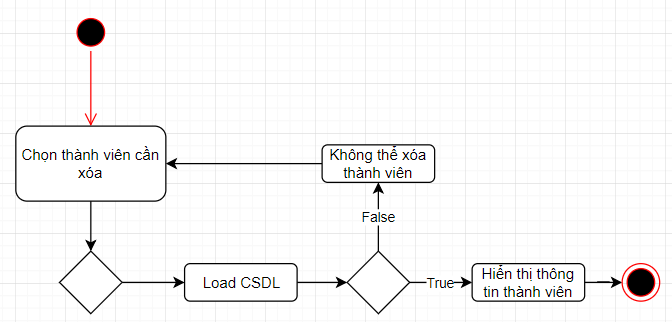
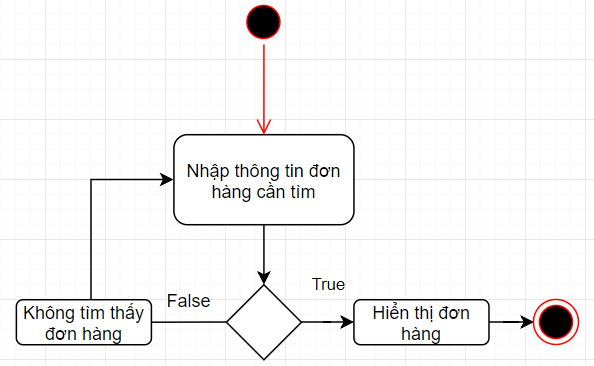
### **Sơ đồ hoạt động quản lý thành viên**



***Hình 6‑7***: Mô tả hoạt động tra cứu thành viên



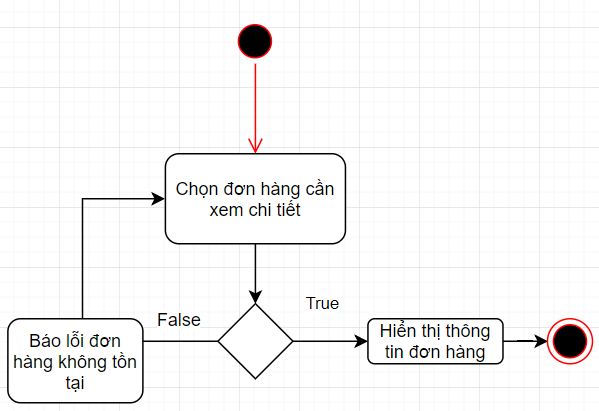
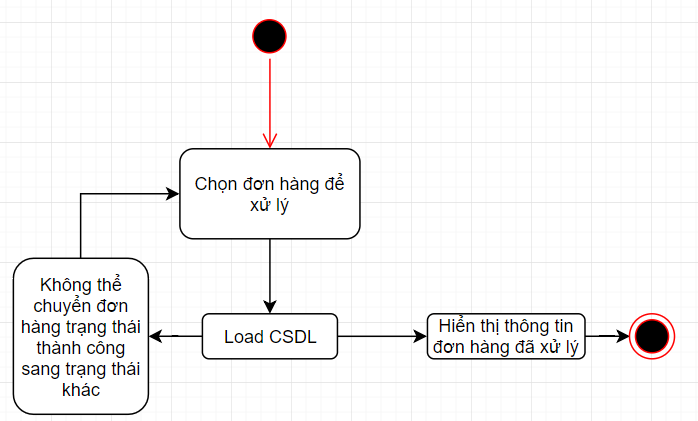
***Hình 6‑8***: Mô tả hoạt động cập nhật thành viên



***Hình 6‑9***: Mô tả hoạt động tìm đơn đặt hàng

***Hình 6‑10***: Mô tả hoạt động xóa thành viên

### **Sơ đồ hoạt động quản lý đơn hàng**



***Hình 6‑11***: Mô tả hoạt động xử lý đơn đặt hàng

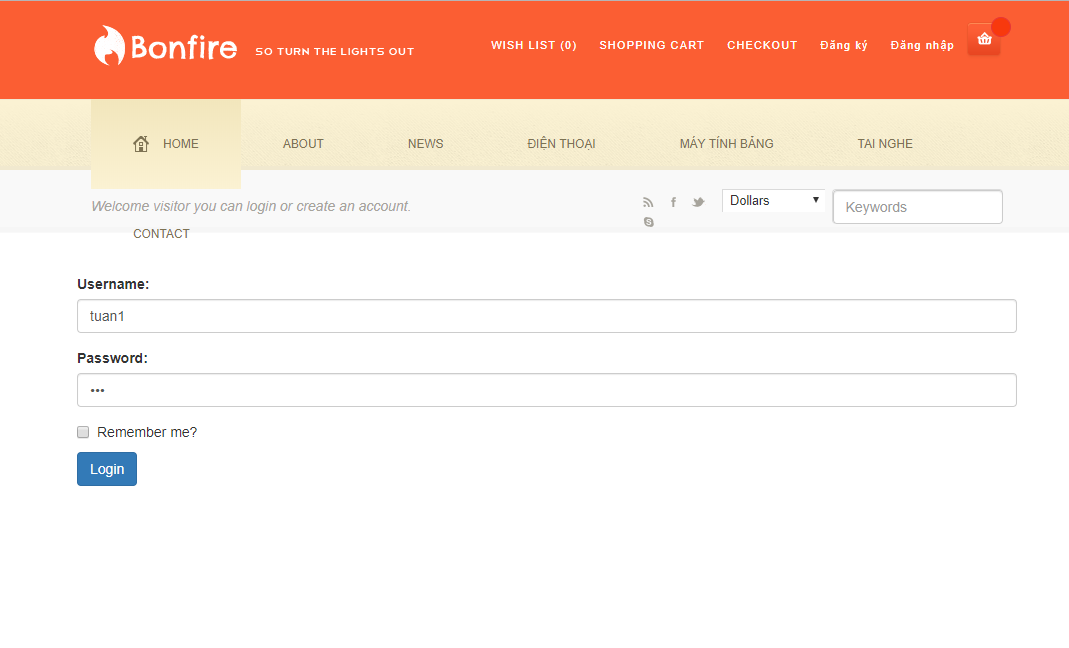
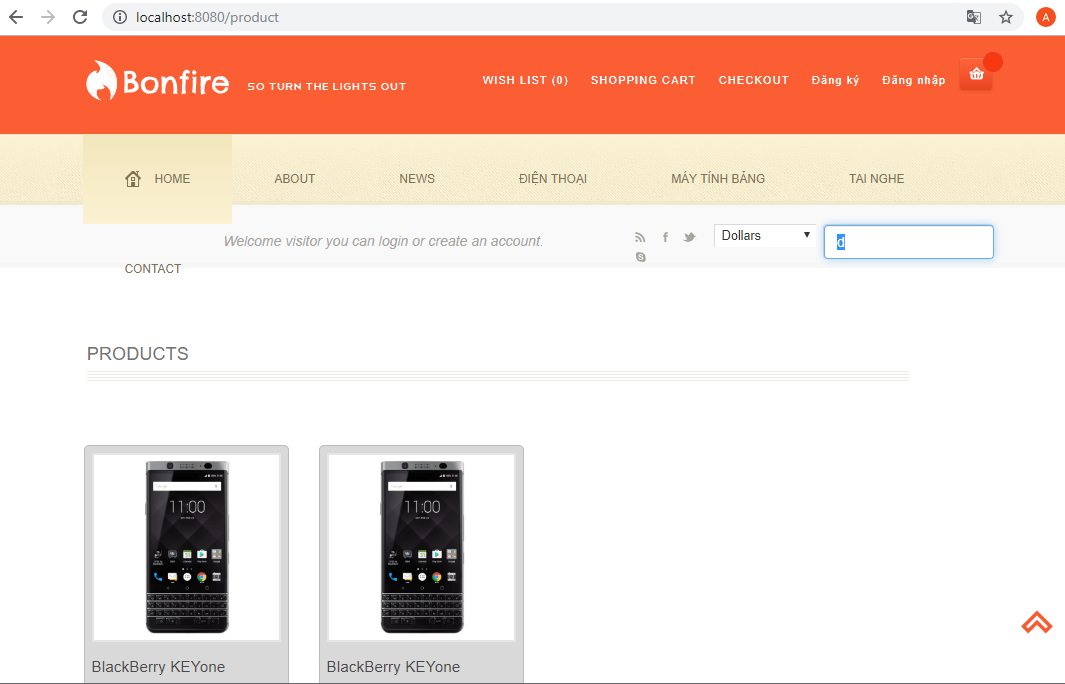
***Hình 6‑12***: Mô tả hoạt động xem chi tiết đơn đặt hàng

# THÀNH PHẦN GIAO DIỆN

## CÁC MÀN HÌNH NHẬP LIỆU

### **Màn hình đăng nhập và đăng ký**

***Hình 7‑1***: Màn hình đăng ký

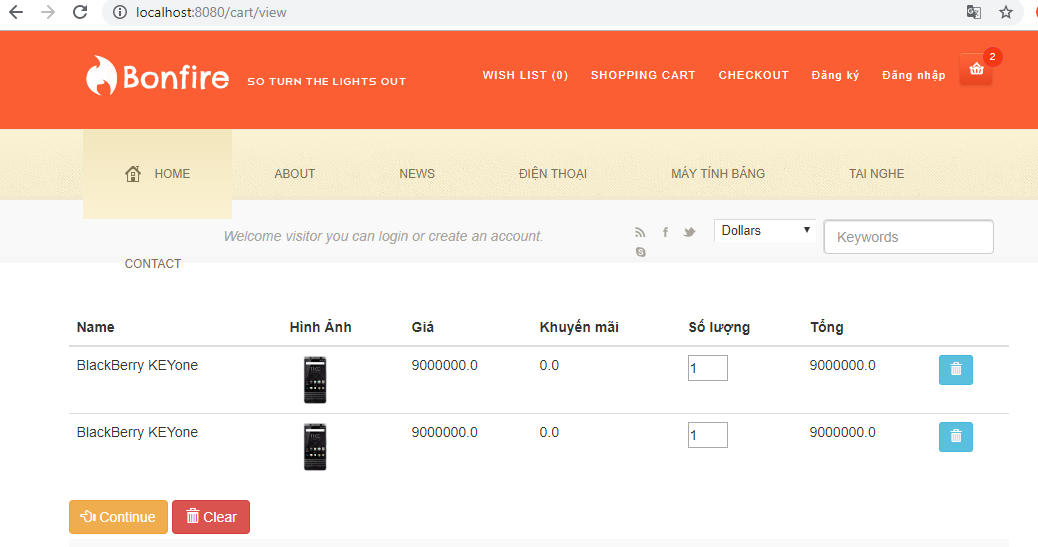


***Hình 7‑2***: Màn hình tìm kiếm sản phẩm

***Hình 7‑3***: Màn hình đăng nhập

### **Màn hình tìm kiếm sản phẩm**

### **Màn hình giỏ hàng**



***Hình 7‑4***: Màn hình hiển thị những sản phẩm trong giỏ hàng

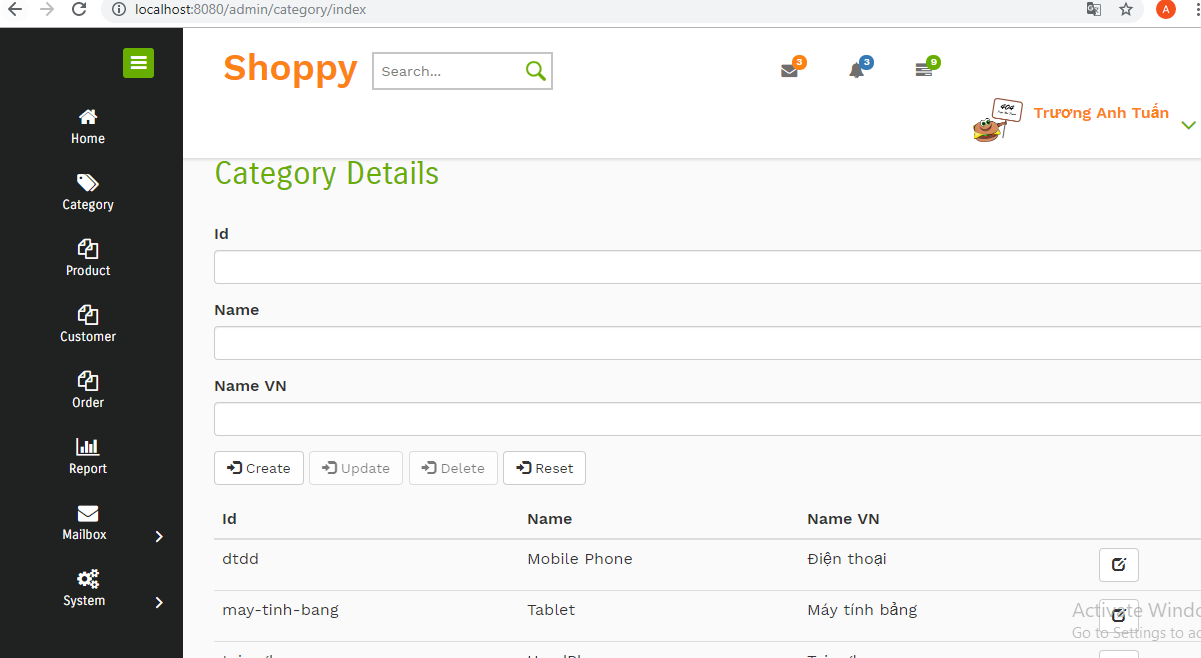
### **Màn hình nhập thông tin đặt hàng**

***Hình 7‑5***: Màn hình hiển thị đặt hàng sau khi mua hàng

### **Màn hình đăng nhập trang quản trị**

***Hình 7‑6***: Màn hình đăng nhập trang quản trị

### **Màn hình nhập liệu thêm loại sản phẩm**

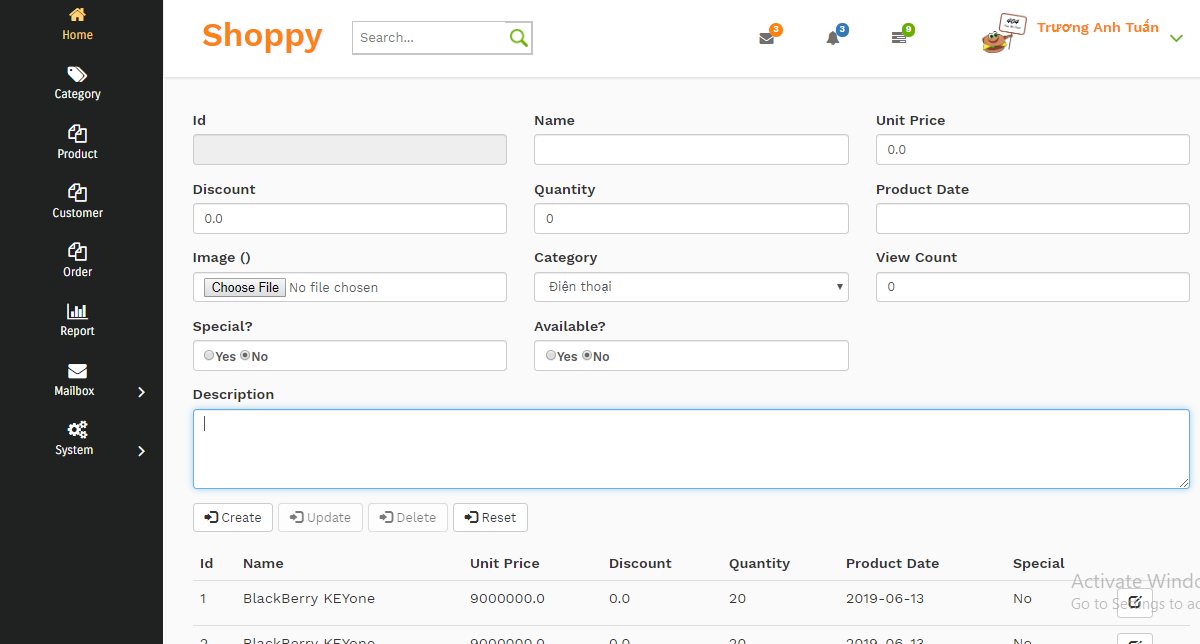


***Hình 7‑7***: Màn hình hiển thị quản lý danh mục

### **Màn hình nhập liệu sửa loại sản phẩm**

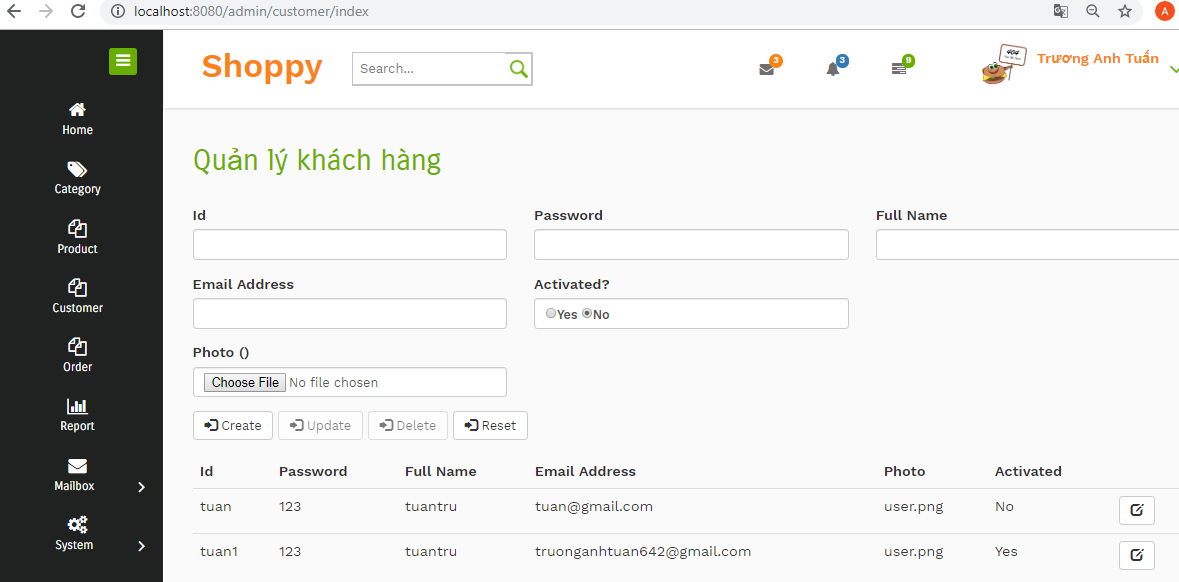
***Hình 7‑8***: Màn hình hiển thị trang sửa loại sản phẩm

### **Màn hình nhập liệu thêm sản phẩm**



***Hình 7‑9***: Màn hình hiển thị trang quản lý sản phẩm

### **Màn hình nhập liệu Khách hàng**



***Hình 7‑10***: Màn hình nhập liệu Khách hàng

### **Màn hình nhập liệu sửa sản phẩm**

***Hình 7‑11***: Màn hình chỉnh sửa sản phẩm

### **Màn hình nhập liệu sửa Khách hàng**

***Hình 7‑12***: Màn hình chỉnh sửa Khách hàng

## TẠO BÁO CÁO

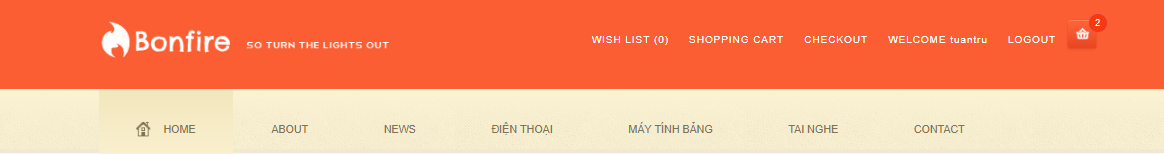
### **Tạo báo cáo doanh thu**

***Hình 7‑13***: Báo cáo doanh thu theo loại

### Tạo báo cáo doanh thu theo từng tháng

***Hình 7‑14***:Báo cáo doanh thu theo tháng

## TẠO MENU SẢN PHẨM



***Hình 7‑15***: Menu sản phẩm trang Khách hàng

# 

Tài liệu tham khảo

1. Introduction to spring framework, www.geeksforgeeks.org
2. Giới thiệu Spring framework, kipalog.com
3. Spring Framework, vi.wikipedia.org
4. Spring Core – Spring IoC Inversion of Control of Spring, 16/12/2017, stackjava.com
5. <https://spring.io/>

1. **K**: key; **U**: unique; **M**: mandatory (not null) [↑](#footnote-ref-1)
2. **K**: key; **U**: unique; **M**: mandatory (not null) [↑](#footnote-ref-2)
3. **K**: key; **U**: unique; **M**: mandatory (not null) [↑](#footnote-ref-3)
4. **K**: key; **U**: unique; **M**: mandatory (not null) [↑](#footnote-ref-4)
5. **K**: key; **U**: unique; **M**: mandatory (not null) [↑](#footnote-ref-5)