Trường ĐH Công Nghệ Sài gòn

**KHOA CÔNG NGHỆ THÔNG TIN**

LUẬN VĂN TỐT NGHIỆP

*Tên đề tài:*

**TÌM HIỂU VỀ ASP.NET CORE FRAMEWORK ỨNG DỤNG VÀO WEBSITE BÁN ĐIỆN THOẠI**

TPHCM – Năm 2019

LỜI CẢM TẠ

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. ĐẶT VẤN ĐỀ
  2. MỤC TIÊU ĐỀ TÀI
  3. PHƯƠNG PHÁP NGHIÊM CỨU

Chương 2. MÔ TẢ NGHIỆP VỤ

2.1 TỔNG QUAN

2.2 QUY TRÌNH

Chương 3. GIẢI PHÁP ĐỀ XUẤT

3.1 KIẾN TRÚC TỔNG THỂ

3.2 GIẢI PHÁP CÔNG NGHỆ

3.3 TỔNG QUAN VỀ .NET FRAMEWORK CORE

Chương 4. THÀNH PHẦN TÁC NHÂN

4.1 ĐẶT TẢ ACTER

4.2 SƠ ĐỒ USE- CASE TỔNG QUÁT

4.3 ĐẶT TẢ USE- CASE CHI TIẾT

**MỤC LỤC CÁC HÌNH VẼ**

**Chương 1. GIỚI THIỆU**

* 1. **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’.

* 1. **MỤC TIÊU ĐỀ TÀI**

Mục tiêu:

* Tìm hiểu tổng quan về C#.
* Tìm hiểu ADO.NET.
* Tìm hiểu về Entity Framework Core.
* Tìm hiểu về SQL Server.
* Tìm hiểu về LinQ.
* Tìm hiểu về Jquery.
* Tìm hiểu về ASP.NET Core
* Tìm hiểu về giao thức API.
* Phân tích điển mạnh điểm yếu trong ASP.NET Core.
* Ứng dụng vào website bán điện thoại cho cửa hàng.
  1. **PHƯƠNG PHÁP NGHIÊM CỨU**
* Về lý thuyết:
* Nắm vững kiến thức hệ thống thông tin.
* Nắm vững mô hình MVC.
* Nắm vững các phương pháp tổ chức, phân tích và thiết kế về cơ sở dữ liệu.
* Tham khảo một số website bán điện thoại di động, đang hoạt động trên internet như: thế giới di động, di động thông minh, cellphones…
* Về kỹ thuật:
* Sử dụng ASP.NET Core và Entity Framework để cài đặt và thực hiện các chương trình ứng dụng.
* Sử dụng hệ quản trị cơ sở 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.
* Đùng bootstrap, linq, jquery để xây dựng và lấy dữ liệu.
* Dùng trình duyệt Google Chrome để chạy localhoast.
* Quản lý source code trên githup.

**Chương 2. MÔ TẢ NGHIỆP VỤ**

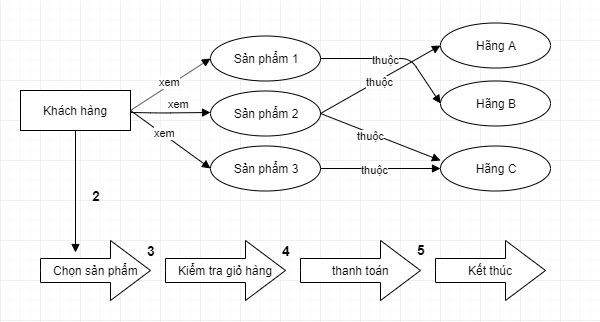
**2.1 TỔNG QUAN**

* Admin (Nhà quản trị): admin là người quản lý 1 trang website nhất định và là người nắm các quyền cao nhất của 1 website ngoài ra còn có (admin1, admin2 …các tài khoản này do chính admin cung cấp) và các cấp người dùng thấp hơn (user) để cùng hoạt động trên 1 trang web.
* Customer (khách hàng): là khách hàng hay các công ty khác cần mua sản phẩm trên mạng. Họ có thể chọn từng hãng điện thoại ưa thích. 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 cho 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: là hiển thị tất cả các sản phẩm mà khách hàng mún mua, ở đây khách hàng có thể thêm, giảm hoặc xóa số lượng sản phẩm có trong giỏ hàng.
* Chức năng tìm kiếm sản phẩm: khách hàng có thể tìm kiếm bằng tên sản phẩm mà không cần phải lướt qua từng sản phẩm, chức năng timg kiếm này giúp khách hàng tìm kiếm sản phẩm nhanh tránh mất thời gian.
* 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.

**2.2 QUY TRÌNH**

**2.2.1 Quy trình đặt hàng**

- Quy trình khách hàng xem sản phẩm và đặt sản phẩm:

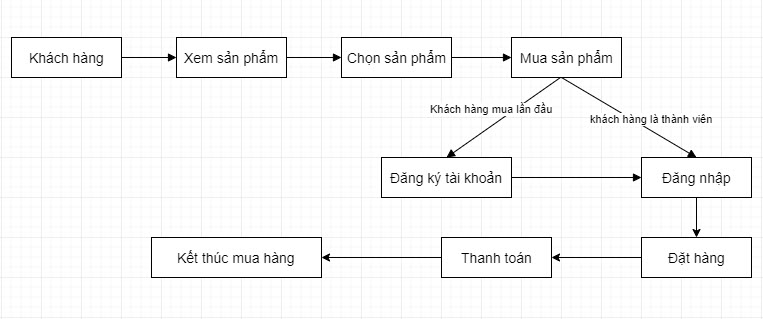


***Hình 2-1:*** Quy trình xem và đặt hàng của khách hàng

Mô tả quy trình như sau:

* Quy trình 1: khách hàng có thể xem nhiều sản phẩm có trong trang website và xem được nhà cung cấp của sản phẩm đó.
* Quy trình 2: sau khi khách hàng xem sản phẩm của mình nếu ưng ý sản phẩm đó thì khách hàng có thể chọn thêm sản phẩm mà khách hàng muốn chọn vào giỏ hàng.
* Quy trình 3: trong khi khách hàng chọn xong khách hàng vào kiểm tra sản phẩm của mình qua việc khách hành click vào giỏ hàng để kiểm tra ở đây khách hàng có thể thêm, sửa, xóa số lượng sản phẩm có trong giỏ hàng.
* Quy trình 4: sau khi khách hàng kiểm tra giỏ hàng xong khách hàng sẽ tiến hành chọn hình thức thanh toán ở đấy khách hàng có thể kiểm tra tổng tiền và tổng sản phẩm của mình.
* Quy trình 5: kết thúc quy trình sau khi khách hàng đăt thành công sản phẩm hệ thống sẽ gửi mail trạng thái đơn hàng.

Mô hinh mua hàng:



***Hinh 2.2*** Mô hình mua hàng của khách hàng

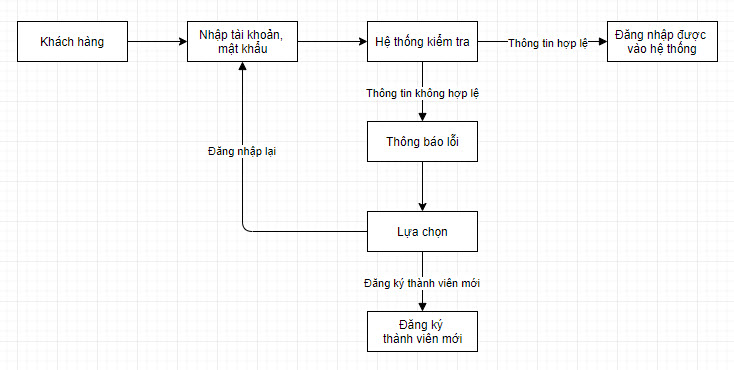
Mô tả mô hình mua hàng:

* Quy trình mua hàng của của hàng như sau: có 2 loại khách hàng

Nếu khách hàng đã là thành viên của của hàng thì khi đó khách hàng chỉ cần đăng nhập tài khoản của mình vào hệ thống (tên đăng nhập, mật khẩu). Nếu đăng nhập thành công hệ thống sẽ báo đăng nhập thành công. Sau đó khách hàng tham quan và dạo cửa hàng trực tuyến, xem thông tin của của sản phẩm xem, xem thông báo khuyến mai của của hàng. Nếu khách hàng chọn sản phẩm nào thì khách hàng chỉ có việc nhấn nút “Thêm vào giỏ hàng”. Trong giỏ hàng khách hàng có thể chọn nhiều sản phẩm hoặc là khách hàng có thể thêm, xóa, sửa thông tin sản phẩm có trong giỏ hàng. Sau khi hoàn thành phần khách hàng chọn sản phẩm xong khách hàng sẽ nhấn vào “Đặt hàng” hệ thống sẽ báo hóa đơn cho khách hàng và lưu thông tin đơn hàng của khách hàng vừa đặt vào database, và hệ thống sẽ gửi mail cho khách hàng thông tin trạng thái của đơn hàng.

Nếu khách hàng lần đầu tiên mua hàng của của hàng thì phải đăng ký khách hàng chỉ nhấn nút đăng ký thành viên hệ thống sẽ hiển thị thông tin form đăng ký cho khách hàng, khách hàng điền đầy đủ thông tin đăng ký vào lưu lại. Sau khi đăng ký thành công khách hàng sẽ có 1 tên đăng nhập và mật khẩu để đăng nhập cho những lần giao dịch tiếp theo. Sau đó quy trình mua hàng diễn ra giống như quy trình trên.

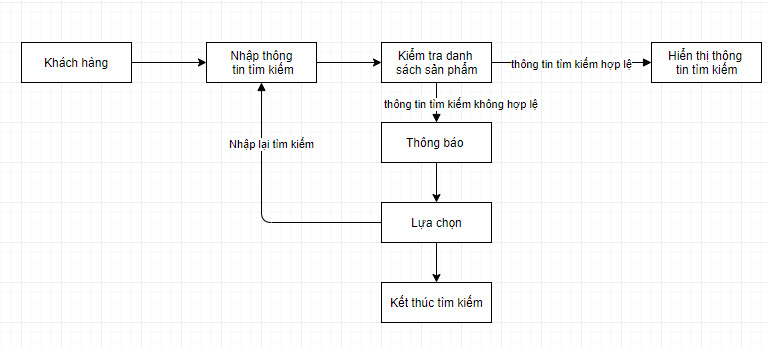
**2.2.2 Quy trình đăng nhập**

****

***Hinh 2-3*** Mô hình đăng nhập của khách hàng

Mô tả mô hình mua hàng: cho khách hàng đăng nhập vào hệ thống, khách hàng nhập thông tin vào hệ thống, hệ thống kiểm tra thống nêu thông tin hợp lệ thì khách hàng có thể mua hàng tại của hàng. Nếu hệ thống kiểm tra bị lỗi thì hệ thống cho khách hàng lựa chọn đăng nhập lại nếu khách hàng chưa có tài thì khách hàng có thể tạo mới tài để đăng nhập.

**2.2.3 Quy trình tìm kiếm sản phẩm**

%3CmxGraphModel%3E%3Croot%3E%3CmxCell%20id%3D%220%22%2F%3E%3CmxCell%20id%3D%221%22%20parent%3D%220%22%2F%3E%3CmxCell%20id%3D%222%22%20value%3D%22Admin%22%20style%3D%22rounded%3D0%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%2240%22%20y%3D%2280%22%20width%3D%2280%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3C%2Froot%3E%3C%2FmxGraphModel%3E****

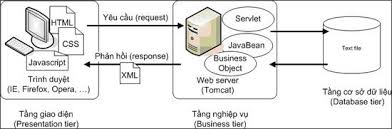
***Hinh 2-4*** Mô hình tìm kiếm sản phẩm của khách hàng

Mô tả mô hình tìm kiếm sản phẩm: để không mất quá nhiều thời gian để có thể tìm sản phẩm mà mình muốn, khách hàng có thể biết được thông tin về sản phẩm mà mình cần bằng cách nhập thông tin vào ô tìm kiếm trên trang web.

Khi khách hàng nhập tên sản phẩm 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”.

**Chương 3. GIẢI PHÁP ĐỀ XUẤT**

**3.1 KIẾN TRÚC TỔNG THỂ**

****

Các đôi tượng tham gia hệ thống trên bao gồm:

Khách hàng và khách hàng vãng lai có các chức năng như: đăng ký, đăng nhập, tìm kiếm, xem sản phẩm, xem giỏ hàng, đặt hàng…

Admin là thành viên quản trị hệ thống có các quyền và chức năng như: tạo các tài khoản, quản lý sản phẩm, quản lý người dùng, quản lý hóa đơn, thống kê…

**3.2 GIẢI PHÁP CÔNG NGHỆ**

**3.2.1 Tổng quan về HTML, CSS, JQUERY, BOOTSTRAP**

**3.2.1.1 HTML**

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: <html></html>

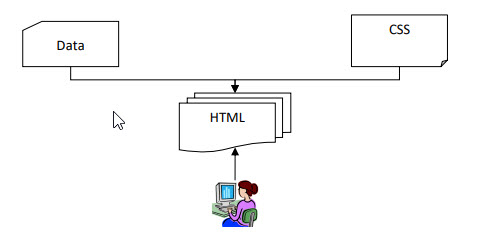
**3.2.1.2 CSS**

- CSS là 1 kỹ thuật dùng để định dạng các tag trong trang web. CSS giúp định dạng trang web rất nhanh nhờ nhiều kiểu định dạng tag, class, element… Bạn không thể định dạng 1 trang web cho đẹp khi không có sự am hiểu về CSS. Nếu làm được điều này, chúng ta có được các lợi điểm sau:

✓ Dễ quản lý, bảo trì.

✓ Tái sử dụng. Một qui luật kiểu dáng có thể áp dụng cho nhiều thành phần web khác nhau.

✓ Cải thiện tốc độ. o Giảm lượng thông tin truyền tải. o Cách hiển thị của trình duyệt

****

***Hinh 3-1*:** Hình biểu điễn của css

Style: Là 1 tập hợp các đặc điểm định dạng cho các thành phần trong trang. Để định dạng, ta chuyển sang chế độ code, rồi định nghĩa các style bên trong tag.

**3.2.1.3** **Jquery**

Jquery là một thư viện được tạo ra từ javascript, ajax hay nói cách khác đó là một bước phát triển mới của JavaScript. Jquery làm cho chúng ta dễ dàng thao tác hơn và đỡ mất thời gian hơn đối với khi thao tác trên Ajax cũng như JavaScript.

jQuery được phát minh bởi John Resig năm 2006, là một trong những thư viện JavaScript được dùng rộng rãi nhất hiện nay giúp viết code JavaScript dễ hơn.

- jQuery có nhiều hàm giúp định dạng, thay đổi nội dung trang web, tạo nhiều hiệu ứng như mờ dần, chạy dọc chạy ngang, tạo request ajax v.v…

- jQuery cho phép tạo ra các Plugin.

**3.2.1.4 bootstrap**

Bootstrap là một framework cho phép thiết kế website reponsive nhanh hơn và dễ dàng hơn. Bootstrap là bao gồm các HTML templates, CSS templates và Javascript tạo ra những cái cơ bản có sẵn như: typography, forms, buttons, tables, navigation, modals, image carousels và nhiều thứ khác. Trong bootstrap có thêm các plugin Javascript trong nó. Giúp cho việc thiết kế reponsive của bạn dễ dàng hơn và nhanh chóng hơn.  
Lịch sử Bootstrap: Bootstrap là dược phát triển bởi Mark Otto và Jacob Thornton tại Twitter. Nó được xuất bản như là một mã nguồn mở vào tháng 8 năm 2011 trên GitHub. Có 2 bản bootstrap: bootstrap 3 và bootstrap 4.

**3.2.2 Tổng quan về C#**

C# (đọc là Cee Sharp) là ngôn ngữ lập trình cho 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 7.3. C# được phát triển bởi đội ngũ kỹ sư của 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.

Trong ngôn ngữ C#, mọi thứ liên quan đến khai báo lớp điều được tìm thấy trong phần khai báo của nó. Định nghĩa một lớp trong ngôn ngữ C# không đòi hỏi phải chia ra tập tin header và tập tin nguồn giống như trong ngôn ngữ C++. Hơn thế nữa, ngôn ngữ C# hỗ trợ kiểu XML, cho phép chèn các tag XML để phát sinh tự động các document cho lớp.

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.

C# cung cấp những đặc tính hướng thành phần (component-oriented): những thuộc tính, những sự kiện. Lập trình hướng thành phần được hỗ trợ bởi CLR cho phép lưu trữ metadata với mã nguồn cho một lớp. Metadata mô tả cho một lớp, bao gồm những phương thức và những thuộc tính của nó, cũng như những bảo mật cần thiết và những thuộc tính khác. Mã nguồn chứa đựng những logic cần thiết để thực hiện những chức năng của nó… Do vậy, một lớp được biên dịch như là một khối self-contained, môi trường hosting biết được cách đọc metadata của một lớp và mã nguồn cần thiết mà không cần những thông tin khác để sử dụng nó.

**Ngôn ngữ ra đời cùng với .NET:**

* Kết hợp C++ và Java.
* Hướng đối tượng.
* Hướng thành phần.
* Mạnh mẽ (robust) và bền vững (durable).
* Mọi thứ trong C# đều Object oriented.
  + Kể cả kiểu dữ liệu cơ bản.
* Chỉ cho phép đơn kế thừa.
  + Dùng interface để khắc phục.
* Lớp **Object** là cha của tất cả các lớp.
  + Mọi lớp đều dẫn xuất từ Object.
* Cho phép chia chương trình thành các thành phần nhỏ độc lập nhau.
* Mỗi lớp gói gọn trong một file, không cần file header như C/C++.
* Bổ sung khái niệm namespace để gom nhóm các lớp.
* Bổ sung khái niệm “*property*” cho các lớp.
* Khái niệm delegate & event.

C# – mạnh mẽ & bền vững:

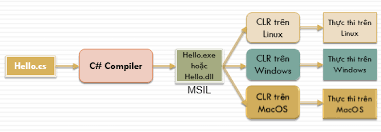
* Garbage Collector
* Tự động thu hồi vùng nhớ không dùng.
* Kiểm soát và xử lý ngoại lệ exception
* Đoạn mã bị lỗi sẽ không được thực thi.
* Type – safe
* Không cho gán các kiểu dữ liệu khác nhau.
* Versioning
* Đảm bảo sự tương thích giữa lớp con và lớp cha.

Vai trò C# trong .NET Framework:

* .NET runtime sẽ phổ biến và được cài trong máy client.
  + Việc cài đặt App C# như là tái phân phối các thành phần .NET
  + Nhiều App thương mại sẽ được cài đặt bằng C#.
* C# tạo cơ hội cho tổ chức xây dựng các App Client/Server n-tier.
* Kết nối ADO.NET cho phép truy cập nhanh chóng & dễ dàng với SQL Server, Oracle…
* Cách tổ chức .NET cho phép hạn chế những vấn đề phiên bản.
  + Loại bỏ “*DLL Hel*” …
* ASP.NET viết bằng C#.
  + GUI thông minh.
  + Chạy nhanh hơn (đặc tính của .NET)
  + Mã ASP.NET ko còn là mới hỗn độn.
  + Khả năng bẫy lỗi tốt, hỗ trợ mạnh trong quá trình xây dựng App Web.

Quá trình dịch CT C#:

* Mã nguồn C# (tập tin \*.cs) được biên dịch qua MSIL.
  + MSIL: tập tin .exe hoặc .dll
* MSIL được CLR thông dịch qua mã máy.
  + Dùng kỹ thuật JIT (just-in-time) để tăng tốc độ.



***Hình 3-2:*** Mô hình biên dịch của chương trình

Các loại ứng dụng C#: Sử dụng C#, ta có thể tạo ra rất nhiều kiểu ứng dụng, ở đây ta quan tâm đến ba kiểu ứng dụng chính: Console, Window và ứng dụng Web

### **Ứng dụng Console:**

* Giao tiếp với người dùng bằng bàn phím.
* Không có giao diện đồ họa (GUI).

Ứng dụng Console là ứng dụng có giao diện text, chỉ xử lý nhập xuất trên màn hình Console, tương tự với các ứng dụng DOS trước đây.

Ứng dụng Console thường đơn giản, ta có thể nhanh chóng tạo chương trình hiển thị kết xuất trên màn hình. Do đó, các minh hoạ, ví dụ ngắn gọn ta thường sử dụng dạng chương trình Console để thể hiện.

**Để tạo ứng dụng Console ta làm như sau:** rong Visual Studio, chọn File → New → Project. Visual Studio sẽ trình bày hộp thoại New Project.

Trong hộp thoại New Project, kích biểu tượng ứng dụng ConSole (Console Application). Trong ô name, gõ tên chương trình (dự án). Trong ô Location, gõ tên của thư mục mà ta muốn Visual Studio lưu dự án. Nhấn OK.

**Ứng dụng Windows Form:** Là ứng dụng được hiển thị với giao diện cửa sổ đồ họa. Chúng ta chỉ cần kéo và thả các điều khiển (control) lên cửa sổ Form. Visual Studio sẽ sinh mã trong chương trình để tạo ra, hiển thị các thành phần trên cửa sổ.

* Giao tiếp với người dùng bằng bàn phím và mouse.
* Có giao diện đồ họa và xử lý sự kiện.

**Để tạo ứng dụng Window ta làm như sau:**

File → New → Project. Visual Studio sẽ trình bày hộp thoại New Project.

Trong hộp thoại New Project, kích biểu tượng ứng dụng Windows (Windows Application). Trong ô Name, gõ tên mô tả chương trình mà ta dự định tạo (tên dự án). Tiếp theo, trong ô Location, gõ tên của thư mục mà ta muốn Visual Studio lưu dự án. Nhấn OK. Visual Studio sẽ hiển thị cửa sổ thiết kế. Ta có thể kéo và thả các thành phần giao diện (control) lên Form.

Để hiển thị cửa sổ Toolbox chứa những điều khiển mà ta có thể kéo và thả lên Form, ta chọn View → Toolbox từ menu.

Ứng dụng Web**:** Môi trường .NET cung cấp công nghệ ASP.NET giúp xây dựng những trang Web động. Để tạo ra một trang ASP.NET, người lập trình sử dụng ngôn ngữ biên dịch như C# hoặc C# để viết mã. Để đơn giản hóa quá trình xây dựng giao diện người dùng cho trang Web, .NET giới thiệu công nghệ Webform. Cách thức tạo ra các Web control tương tự như khi ta xây dựng ứng dụng trên Window Form.

* Kết hợp với [ASP .NET,](https://sinhvientot.net/category/lap-trinh/asp-mvc/) C# đóng vài trò xử lý bên dưới (underlying code).
* Có giao diện đồ họa và xử lý sự kiện.

**Để tạo ứng dụng Web ta làm như sau:**

File → New → Project → Visual Basic Projects → ASP.NET Web Application

Ưu điểm và nhược điểm khi sử dụng c#:

* Đễ học hơn c++
* Đễ đọc mã hơn c++
* Phát triển nhanh hơn và có khả năng ít lỗi so với c++ và java (bạn có các loại không dấu, bạn có các loại ref/out, bạn có thể tạo các giá trị riêng của mình, bạn có những hưu ích khách mà java bỏ qua, có nghĩa là bạn ít nhảy qua vòng, có nghĩa là độ phức tập mã ít cần thiết hơn).
* 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.
* Lock (someObject){code}, giống như java được đồng bộ hóa nhưng có thể được sử dụng ở bất cứ đâu và yêu cầu 1 đối tượng.
* Bạn có thể tạo các loại giá trị, theo mặc định là truyền theo giá trị thay vì truyền theo tham chiếu, bằng cách tạo các cấu trúc thay vì một lớp.
* Có số nguyên không dấu (c++, java không).
* 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).
* Chương trình được biên dịch thành tập tin .exe và không cần phải biên dịch lại cho các HĐH khác, môn có thể chạy trực tiếp exes .NET.

Nhược điểm:

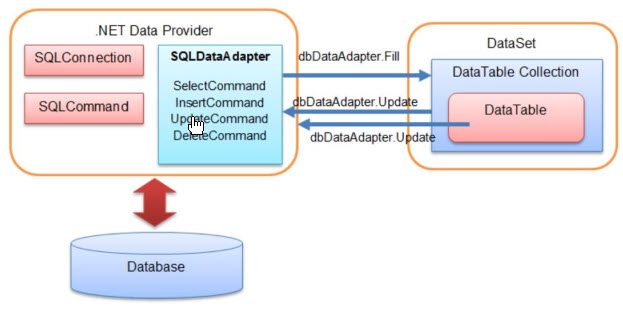
* Sử dụng nhiều bộ nhớ hơn c++.
* Trình thu gom rác sử dụng chu kỳ vào bộ nhớ (nhưng hầu như không có, ít nhất là thời CPU)
* Một số thứ không hoạt động trong các chức năng gọi đơn trong một số của sổ sẽ không tồn tại trên linux, việc triển khai các biểu mẫu đơn của windows cần có cách giải quyết trong mã (nhưng có lẽ chúng ta không sử dụng chúng).
* Mono chưa hoàn hảo.
* 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 IDE tốt trên các cửa sổ không. E clipse có plugin ac # nhưng nó có thể không hỗ trợ hầu hết các tính năng của Eclipse.
* Không có con trỏ, nhưng chúng được thay thế bởi các tham số ref và out, vì vạy chúng không cần nhiều.
* Phương thức gọi thông qua các đại biểu chậm hơn đáng kể so với cách gọi phương thức bình thường (hoặc con trỏ hàm trong c++).
* Yêu cầu (phiên bản chính xác của) .NET framework phải được cài đặt để chạy chương trình.

**3.3 TỔNG QUAN VỀ .NET FRAMEWORK CORE**

**3.3.1 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.



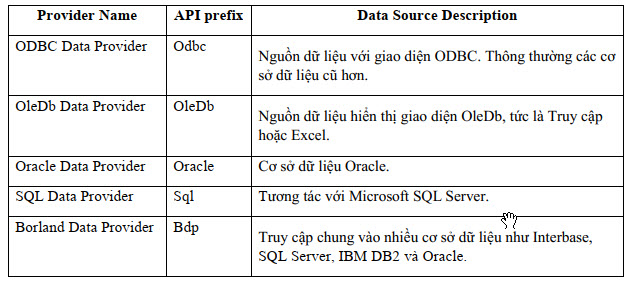
***Hình 3-4:*** Ảnh minh họa ADO.NET của tầng Provider

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.



***Hình 3-5:*** Danh sách các Data provider

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.

**3.3.2 Tìm hiểu 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.

Ưu điểm của sử dụng Entity Framework:

+ Thứ nhất là tăng năng suất thông qua giảm số lượng code.

+ Thứ hai là giảm sự phức tạp khi bảo trì phần mềm.

+ Thứ ba, Entity Framework được đặt trong .NET Framework và tích hợp trong Visual Studio.

Entity Framework Core:

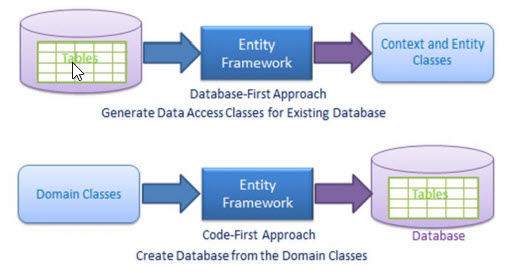
Entity Framework Core là một phiên bản mã nguồn mở, nhỏ, nhẹ, có thể mở rộng và đa nền tảng nằm trong bộ Entity Framework. EF Core là bộ ánh xạ đối tượng – quan hệ (Object Relation Mapping) cho phép các lập trình .NET làm việc với CSDL quan hệ thông qua các đối tượng (object), giúp các lập trình viên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

+ 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

Trong EF Core có 2 hướng tiếp cận khi làm việc với CSDL:



***Hinh 3-5:*** Hình Entity framework core.

Code First: Tạo model các đối tượng dữ liệu, sau đó migration vào Database.

Dadatabase First: sử dụng khi database có sẵn, sau đó phát sinh các model dữ liệu tương ứng.

✓ Microsoft. EntityFrameworkCore.SqlServer

✓ Microsoft. EntityFrameworkCore.Tools

**3.3.3 Tìm hiểu về ASP.NET CORE**

ASP.NET Core là một khung công tác mã nguồn mở, đa nền tảng, hiệu suất cao để xây dựng các ứng dụng kết nối Internet hiện đại, dựa trên đám mây. Với ASP.NET Core, bạn có thể:

* Xây dựng các ứng dụng và dịch vụ web, ứng dụng IoT 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 Framework core

ASP.NET Core cung cấp các lợi ích sau:

* Một câu chuyện thống nhất để xây dựng giao diện người dùng web và API web.
* Kiến trúc cho khả năng kiểm tra.
* Razor pages giúp mã hóa các kịch bản tập trung vào trang dễ dàng hơn và hiệu quả hơn.
* Blazor cho phép bạn sử dụng C # trong trình duyệt cùng với JavaScript. Chia sẻ logic ứng dụng phía máy chủ và phía máy khách, tất cả được viết bằng .NET.
* Có khả năng phát triển và chạy trên Windows, macOS và Linux.
* Nguồn mở và tập trung vào cộng đồng.
* Tích hợp các modern, client-side framework và quy trình phát triển.
* Hỗ trợ lưu trữ các dịch vụ Gọi thủ tục từ xa (RPC) bằng [gRPC](https://docs.microsoft.com/vi-vn/aspnet/core/grpc/index?view=aspnetcore-3.0) .
* Một hệ thống cấu hình dựa trên môi trường, sẵn sàng cho đám mây.
* Tích hợp [dependency injection](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/dependency-injection?view=aspnetcore-3.0) .
* Một đường ống yêu cầu HTTP nhẹ, [hiệu suất cao](https://github.com/aspnet/benchmarks) và mô-đun.
* Khả năng lưu trữ trên các mục sau:
  + [Kestrel](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/servers/kestrel?view=aspnetcore-3.0)
  + [IIS](https://docs.microsoft.com/vi-vn/aspnet/core/host-and-deploy/iis/index?view=aspnetcore-3.0)
  + [HTTP.sys](https://docs.microsoft.com/vi-vn/aspnet/core/fundamentals/servers/httpsys?view=aspnetcore-3.0)
  + [Nginx](https://docs.microsoft.com/vi-vn/aspnet/core/host-and-deploy/linux-nginx?view=aspnetcore-3.0)
  + [Apache](https://docs.microsoft.com/vi-vn/aspnet/core/host-and-deploy/linux-apache?view=aspnetcore-3.0)
  + [Docker](https://docs.microsoft.com/vi-vn/aspnet/core/host-and-deploy/docker/index?view=aspnetcore-3.0)
* Công cụ đơn giản hóa phát triển web hiện đại.

**Giai đoạn phát triển của ASP.NET CORE:**

Active Server Pages (viết tắt là ASP, hay là ClassicASP hoặc ASP Classic) là [kịch bản kích hoạt](https://vi.wikipedia.org/w/index.php?title=K%E1%BB%8Bch_b%E1%BA%A3n_k%C3%ADch_ho%E1%BA%A1t&action=edit&redlink=1) [phía server](https://vi.wikipedia.org/w/index.php?title=K%E1%BB%8Bch_b%E1%BA%A3n_ph%C3%ADa_server&action=edit&redlink=1) đầu tiên của Microsoft cho [trang web động](https://vi.wikipedia.org/w/index.php?title=Trang_web_%C4%91%E1%BB%99ng&action=edit&redlink=1). [ASP.NET](https://vi.wikipedia.org/wiki/ASP.NET), ra mắt lần đầu vào tháng 01 năm 2002, đã thay thế cho ASP.

ASP được phát hành với dạng tích hợp vào server [IIS](https://vi.wikipedia.org/w/index.php?title=Internet_Information_Services&action=edit&redlink=1) thông qua [Windows NT 4.0#Option Pack](https://vi.wikipedia.org/wiki/Windows_NT_4.0#Option_Pack) (ca. 1996), nó bao gồm một thành phần miễn phí của [Windows Server](https://vi.wikipedia.org/wiki/Windows_Server) (từ lúc phát hành lần đầu của [Windows 2000](https://vi.wikipedia.org/wiki/Windows_2000)). Có 3 phiên bản ASP, mỗi phiên bản được giới thiệu trên mỗi phiên bản IIS khác nhau:

* 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" \o ".NET Compact Framework (trang chưa được viết))
* 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
  + [Windows Presentation Foundation](https://vi.wikipedia.org/wiki/Windows_Presentation_Foundation" \o "Windows Presentation Foundation) (WPF - tên mã là *Avalon*): Đây là một công nghệ mới, và là một nỗ lực của Microsoft nhằm thay đổi phương pháp hay cách tiếp cận việc lập trình một ứng dụng sử dụng giao diện đồ họa trên Windows với sự hỗ trợ của ngôn ngữ [XAML](https://vi.wikipedia.org/w/index.php?title=XAML&action=edit&redlink=1).

* + [Windows Communication Foundation](https://vi.wikipedia.org/w/index.php?title=Windows_Communication_Foundation&action=edit&redlink=1" \o "Windows Communication Foundation (trang chưa được viết)) (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" \o "Windows Workflow Foundation (trang chưa được viết)) (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" \o "Windows CardSpace (trang chưa được viết)) (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)" \o "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
  + 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.

Giới thiệu về ứng dụng web

Hàng ngày chúng ta mở máy tính, vào các trang web nổi tiếng như Google, Yahoo để tìm kiếm, đọc và gửi email. Chúng ta cũng thường vào các trang báo để đọc tin tức; các trang đào tạo trực tuyến để học hành; các trang bán hàng để tìm và mua hàng; tham gia vào diễn đàn để tranh luận. Tất cả rất tuyệt vời và thật sự có ý nghĩa với cuộc sống mỗi một con người trong thời đại internet của chúng ta.

Sự phát triển mạnh mẽ về công nghệ thông tin, đặc biệt là internet, nhiều lĩnh vực ngày nay như : thương mại, y tế, giáo dục..., nhu cầu trao đổi thông tin thực sự là cần thiết, giúp cho công việc được triển khai nhanh chóng, chính xác, dễ dàng và tiết kiệm chi phí, thông tin được cập nhật kịp thời. Do đó vấn đề đặt ra là chúng ta cần phải có một ứng dụng cho phép trao đổi thông tin mọi lúc, mọi nơi, dễ sử dụng, … thông qua mạng. Ứng dụng Web đáp ứng được các yêu cầu đặt ra và sau đây là các lý do tại sao chúng ta phải sử dụng Web:

✓ Dễ dàng trao đổi và chia sẽ thông tin thông tin qua mạng.

✓ Sử dụng giao diện đồ họa giúp cho người dùng dễ sử dụng.

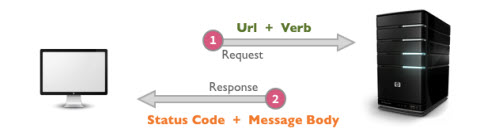
✓ Hỗ trợ về multimedia như: hình ảnh, âm thanh, phim ảnh, …

✓ Hỗ trợ nhiều chương trình(web-browser) để truy cập Web.

✓ Hỗ trợ truy cập web trên các thiết bị đi động: Tablet, SmartPhone, …

✓ Hỗ trợ nhiều ngôn ngữ để phát triển Web: ASP.NET, JSP, PHP, Python, …

Giới thiệu về nguên lý hoạt động của web



Để có được kết quả hiển thị của trang web yêu cầu, các bước thực hiện truyền thông xảy ra ở phía hậu cảnh bao gồm:

✓ Chuyển đổi “url” thành “ip”

✓ Gửi request đến Web Server

✓ Web Server thực hiện các xử lý cần thiết theo request

✓ Kết quả được response đến Browser

✓ Web Browser trình bày dữ liệu trên kết quả trả về và các thẻ markup

3.1.3 Các khái niệm

3.1.3.1Web client (Browser)

Máy khách (Client) thường là Web Browser sẽ sử dụng chương trình để truy cập đến các trang web gọi là trình duyệt web hay browser. Hiện rất nay có nhiều trình duyệt web như: Chrome, Edge, Opera, Mozila FireFox, ...

3.1.3.2Web server

Các máy chủ (Server) chứa các ứng dụng Web, sẳn sàng truy xuất các trang web hay các tài liệu và gửi về cho client khi nhận được yêu cầu từ phía Client. Hiện nay có rất nhiều Web server và chạy trên nhiều hệ thống như: Apache, Nginx, Microsoft IIS, …

3.1.3.3Giao thức HTTP

Quá trình giao tiếp giữa client và server được thực hiện thông qua giao thức chuẩn HTTP (HyperText Transfer Protocol). Hình minh họa sau mô tả việc truy cập ứng dụng Web.

3.1.3.4Client Scripting và Server Scripting

Các ngôn ngữ dùng để viết mã cho trang web. Một trang web được xử lý ở Server và trả kết quả về cho Client. Do đó các ngôn ngữ viết mã cho trang web được chia thanh hai dạng:

✓ Client Scripting: được xử lý tại trình duyệt (Browser) trên máy Client. Các ngôn ngữ dùng để viết là: VBScript, JavaScript, …

o JavaScript là ngôn ngữ phổ biến sử dụng nhiều nhất hiện nay. JavaScript được dùng để kiểm tra việc nhập liệu, kiểm tra trình duyệt, …

o DTHML: là sự kết hợp của HTML, Style Sheet (CSS) và JavaScript nhằm làm cho trang web dễ tương tác, điều khiển và giảm bớt việc xử lý phía Server.

o VBScript là ngôn ngữ script của Microsoft. Chức năng của VBScript cũng giống như JavaScript.

✓ Server Scripting: được xử lý tại Web server trên máy Server. Các ngôn ngữ dùng để viết là: ASP, ASP.NET, PHP, JSP, Python, … Trong giáo trình này chúng ta sẽ khảo sát ngôn ngữ ASP.NET C#.

Mô hình mvc

**Chương 4. CÁC THÀNH PHẦN TÁC NHÂN**

**4.1 ĐẶT 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.

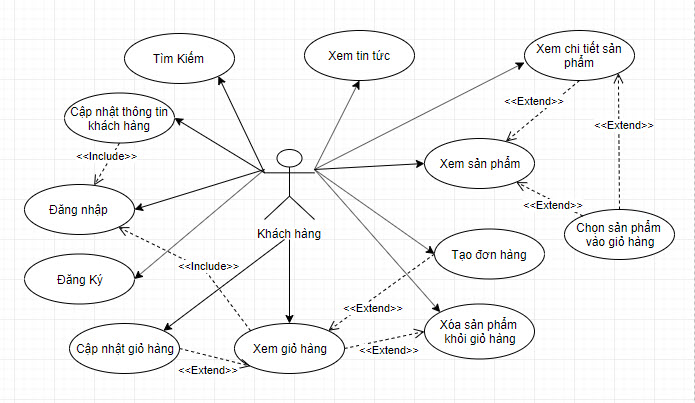
+ Thống kê doanh số theo tháng.

+ Thống kê doanh số theo năm.

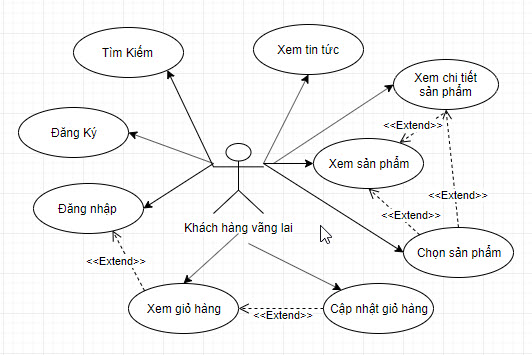
+ Thống kê doanh số theo quý.

**4.2 SƠ ĐỒ USE CASE TỔNG QUÁT**

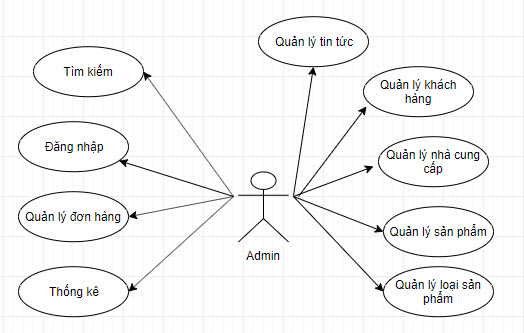
Use-Case Diagram cho khách hàng:

****

Use-Case Diagram cho khách hàng vãng lai:

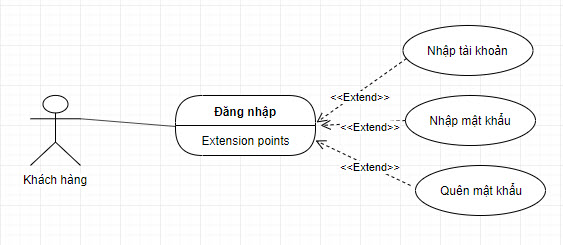
****

Use-Case Diagram cho Admin:



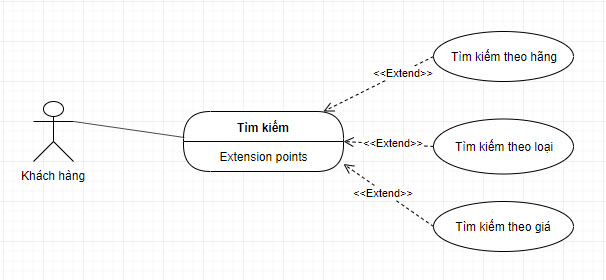
**4.3 ĐẶT TẢ USE-CASE CHI TIẾT**

Sơ đồ chi tiết đăng nhập



|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Đăng nhập |
| Actor | Khách hàng |
| Mô tả | Khách hàng dùng tài khoản đã đăng ký để đăng nhập. |
| Luồng sự kiện chính | 1. Actor chọn chức năng đăng nhập 2. Hệ thống hiển thị màng 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 chọn nút thoát 2. Hệ thống hủy màng hình đăng nhập |
| <Extend Use Case> | **Nhập tài khoản**   1. Actor chọn chức năng hệ thống đăng nhập 2. Actor nhập tên đăng nhập 3. Actor nhấn nút đăng nhập 4. Actor nhấn nút thoát 5. Hủy màng hình đăng nhập   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin tài khoản rỗng   2. Quay lại đăng nhập   **Rẽ nhánh 2:**   * 1. Kiểm tra thông tin hợp lệ   2. Hoạt động đăng nhập vào tài khoản |
| <Extend Use Case> | **Nhập mật khẩu**   1. Actor chọn nút đăng nhập 2. Actor nhập mật khẩu 3. Actor nhấn nút đăng nhập 4. Actor nhấn nút thoát 5. Hệ thống hủy màng hình đăng nhập   **Rẽ nhánh 1:**   * 1. Kiểm tra mật khẩu rổng   2. Quay lại đăng nhập lại mật khẩu   **Rẽ nhánh 2:**   * 1. Kiểm tra thông tin hợp lệ   2. Hoạt động đăng nhập vào tài khoản |
| <Extend Use Case> | **Quên mật khẩu**   1. Actor nhấn nút quên mật khẩu 2. Hiển thị thông tin quên mật khẩu lên Form 3. Actor nhập gmail 4. Kiểm tra mail để lấy mật khẩu 5. Actor nhấn nút thoát 6. Hủy màng hình quên mật khẩu   **Rẽ nhánh 1:**   * 1. Kiểm tra mật khẩu rổng   2. Quay lại màng hình quên mật khẩu |

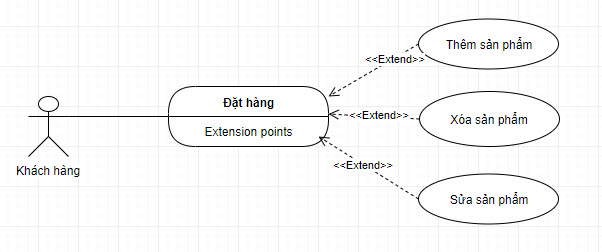
**Sơ đồ tìm kiếm sản phẩm**

****

|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Tìm kiếm |
| Actor | Khách hàng |
| Mô tả | Khách hàng dùng chức năng tìm kiếm sản phẩm để tìm sản phẩm có trong của hàng. |
| Luồng sự kiện chính | 1. Actor chọn chức năng tìm kiếm 2. Hệ thống hiển thị màng hình tìm kiếm  * Extend use case *Tìm kiếm theo hãng* * Extend use case *Tìm kiếm theo giá* * Extend use case *Tìm kiếm theo tên* |
| Luồng sự kiện phụ | 1. Actor chọn nút thoát 2. Hệ thống hủy màng hình tìm kiếm |
| <Extend Use Case> | **Tìm kiếm theo hãng**   1. Actor chọn chức năng tìm kiếm 2. Actor nhập tên hãng cần tìm kiếm 3. Actor nhấn nút tìm kiếm 4. Hệ thống hiển thị thông tin tìm kiếm 5. Actor nhấn nút thoát   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rỗng   1.2 Thông báo không tìm thấy  **Rẽ nhánh 2:**   * 1. Kiểm tra thông tin hợp lệ   2. Hiển thị danh sách theo hãng |
| <Extend Use Case> | **Tìm kiếm theo giá**   1. Actor chọn chức năng tìm kiếm 2. Actor chọn giá cần tìm kiếm 3. Actor nhấn nút tìm kiếm 4. Hiển thị màng hình sản phẩn tìm kiếm 5. Actor nhấn nút thoát 6. Hủy màng hình tìm kiếm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rổng   2. Xuất ra màng hình thông báo không tìm thấy   **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin hợp lệ  2.2 Xuất ra màng hình thông báo không tìm thấy |
| <Extend Use Case> | **Tìm kiếm theo tên**   1. Actor chọn chức năng tìm kiếm 2. Actor nhập tên sản phẩm cần tiềm kiếm 3. Actor nhấn nút tìm kiếm 4. Hiển thị màng hình sản phẩn tìm kiếm 5. Actor nhấn nút thoát 6. Hủy màng hình tìm kiếm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rổng   2. Xuất ra màng hình không tìm thấy   **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin hợp lệ  2.2 Xuất ra danh sách sản phẩm theo tên |

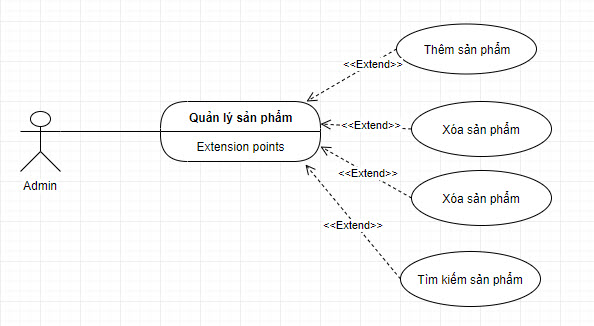
**Sơ đồ use case đặt hàng**

**Sơ đò chi tiết:**

****

|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Đặt hàng |
| Actor | Khách hàng |
| Mô tả | Khách hàng dùng chức năng đặt sản phẩm có trong cửa hàng |
| Luồng sự kiện chính | 1. Actor chọn chức năng đặt hàng 2. Hệ thống hiển thị màng hình đặt hàng  * Extend use case *Thêm sản phẩm* * Extend use case *Xóa sản phẩm* * Extend use case *Sửa sản phẩm* |
| Luồng sự kiện phụ | 1. Actor chọn nút thoát 2. Hệ thống hủy màng đặt hàng |
| <Extend Use Case> | **Thêm sản phẩm**   1. Actor chọn sản phẩm cần mua 2. Thêm sản phẩm vào cửa hàng 3. Actor nhấn nút thoát 4. Hệ thống hủy màng hình thêm sản phẩm   **Rẽ nhánh 1:**   * 1. Kiểm tra giỏ hàng có bị rỗng   1.2 Nêu rỗng quay lại chọn sản phẩm  **Rẽ nhánh 2:**   * 1. Kiểm tra sản phẩm trùng   2.2 Tăng sản phẩm lên 1 |
| <Extend Use Case> | **Sửa sản phẩm**   1. Actor chọn sản phẩm cần sửa 2. Actor nhấn nút sửa sản phẩm 3. Actor nhấn nút thoát   **Rẽ nhánh 1:**   * 1. Kiểm tra giỏ hàng có bị rỗng   2. Nếu rỗng quay về chọn thêm sản phẩm   **Rẽ nhánh 2:**  2.1 Kiểm tra sản phẩm có bị trùng  2.2 Nếu trùng quay lại sủa |
| <Extend Use Case> | **Xóa sản phẩm**   1. Actor chọn sản phẩm cân xóa 2. Actor nhấn nút xóa sản phẩm 3. Actor nhấn nút thoát   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rổng   2. Xuất ra màng hình không tìm thấy   **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin hợp lệ  2.2 Suất ra màng hình thông báo thành công |

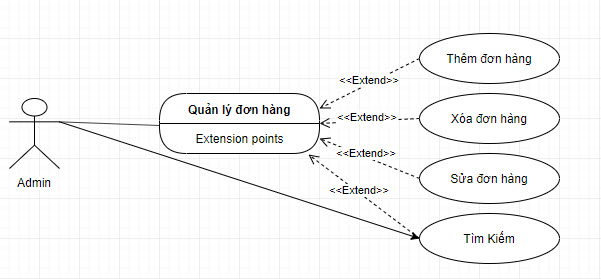
**Sơ đồ use case quản lý sản phẩm**

****

|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Quản lý sản phẩm |
| Actor | Admin |
| 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 chọn nút thoát 2. Hệ thống hủy màng quản lý sản phẩm |
| <Extend Use Case> | **Themsanpham**   1. Admin nhập thông tin sản phẩm cần thêm 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 màng hình thêm sản phẩm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rỗng   1.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 2:**   * 1. Kiểm tra sản phẩm trùng   2. Lập bước 1 luồng sự kiện chính   Rẽ nhánh 3:   * 1. Kiểm tra thông tin hợp lệ   2. Cập nhật vào CSDL |
| <Extend Use Case> | **Sửa sản phẩm**   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 màng hình sửa sản phẩm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rỗng   2. Lập lại bước 1 luồng sự kiệm chính   **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin trùng  2.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 3:**   * 1. Kiểm tra thông tin hợp lệ   2. Cập nhật vào CSDL |
| <Extend Use Case> | **Xóa sản phẩm**   1. Admin chọn sản phẩm từ danh sách sản phẩm 2. Hiển thị thông tin sản phẩm lên form 3. Admin nhấn nút xóa 4. Cập nhật vào CSDL 5. Hiển thị thông tin cập nhật 6. Admin nhấn nút thoát 7. Hủy màng hình xóa sản phẩm |
| <Extend Use Case> | **Timkiemsanpham**   1. Admin nhập thông tin sản phẩm cần tìm kiếm 2. Kiểm tra thông tin không rỗng 3. Admin nhấn nút tìm kiếm 4. Hiển thị thông tin tìm kiếm 5. Admin nhấn nút thoát 6. Hủy màng hình tìm kiếm   Rẽ nhánh 1:   * 1. Tìm gần đúng   2. Hiển thị thông tin lên from   Rẽ nhánh 2:   * 1. Tìm chính xác   2. Hiển thị thông tin sản phẩm lên. |

**Quản lý hóa đơn**

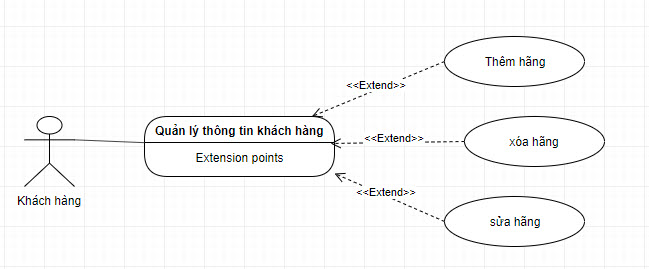
**Sơ đồ chi tiết:**

****

|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Quản lý đơn hàng |
| Actor | Admin |
| 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 đơn hàng. 2. Hệ thống hiển thị màn hình Quản lý đơn hàng.  * Extend Use Case *Themhaodon.* * Extend Use Case *Suahoadon.* * Extend Use Case *Xoahoadon.* |
| Luồng sự kiện phụ | 1. Admin chọn nút thoát 2. Hệ thống hủy màng quản lý hóa đơn |
| <Extend Use Case> | **Themhoadon**   1. Admin chọn chức năng tạo đơn hàng. 2. Nhập thông tin sản phẩm. 3. Kiểm tra thông tin. 4. Xuât kết quả kiểm tra. 5. Nhấn nút tạo. 6. Cập nhật CSDL. 7. Nhấn nút thoát. 8. Hủy màng hình tạo đơn hàng.   **Rẽ nhánh 1:**  1.1 Kiểm tra thông tin bị rỗng.  1.2 Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 2:**  2.1 Thông tin bị trùng.  2.2 Lập lại bước 3 luồng sự kiện chính.  **Rẽ nhánh 3:**   * 1. Kiểm tra thông tin sản phẩm hợp lệ.   2. Cập nhật CSDL. |
| <Extend Use Case> | **Suadonhang**   1. Admin chọn chức năng sửa đơn hàng 2. Chọn đơn hàng. 3. Hiển thị thông tin sản phẩm lên form 4. Admin nhấn nút sửa 5. Cập nhật CSDL 6. Hiển thị thông tin cập nhật 7. Admin nhấn nút thoát 8. Hủy màng hình sửa sản phẩm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rỗng   1.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin trùng  2.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 3:**   * 1. Kiểm tra thông tin hợp lệ   2. Cập nhật vào CSDL |
| <Extend Use Case> | **Xóa sản phẩm**   * 1. Admin chọn chức năng xóa đơn hàng   2. Hiển thị thông tin đơn lên form   3. Admin nhấn nút xóa   4. Cập nhật vào CSDL   5. Hiển thị thông tin cập nhật   6. Admin nhấn nút thoát   7. Hủy màng hình xóa sản phẩm |

**Quản lý hãng sản phẩm**

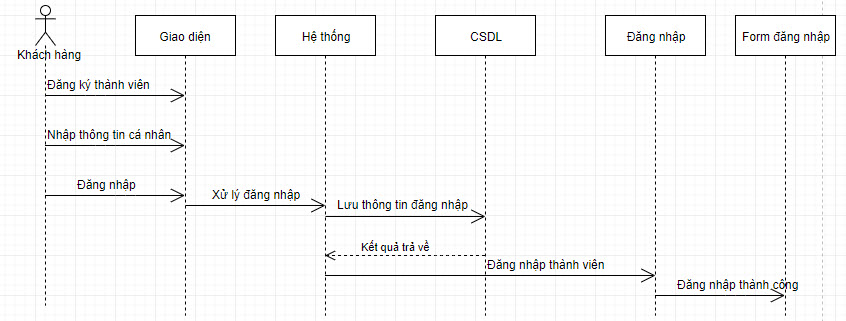
**Sơ đồ chi tiết quản lý hãng sản phẩm**

****

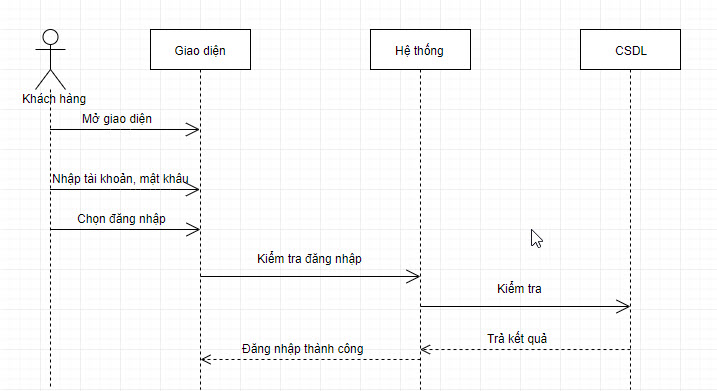
|  |  |
| --- | --- |
| **Các chức năng** | **Ghi chú** |
| Tên use case | Quản lý hãng |
| Actor | Admin |
| Mô tả | Quản trị viên xem thông tin các hang mà cửa hàng bánđăng ký trên website và thực hiện các chức năng: Thêm, Xóa, Tra cứu và Cập nhật thông tin của hãng. |
| Luồng sự kiện chính | 1. Admin chọn chức năng Quản lý hãng. 2. Hệ thống hiển thị màn hình Quản lý hãng.  * Extend Use Case *Themhang.* * Extend Use Case *Suahang.* * Extend Use Case *Xoahang.* |
| Luồng sự kiện phụ | 1. Admin chọn nút thoát 2. Hệ thống hủy màng quản lý hóa đơn |
| <Extend Use Case> | **Themhoadon**   1. Admin chọn chức năng thêm hãng 2. Nhập thông tin hãng. 3. Kiểm tra thông tin. 4. Xuât kết quả kiểm tra. 5. Nhấn nút tạo. 6. Cập nhật CSDL. 7. Nhấn nút thoát. 8. Hủy màng hình tạo đơn hàng.   **Rẽ nhánh 1:**  1.1 Kiểm tra thông tin bị rỗng.  1.2 Lập lại bước 1 luồng sự kiện chính.  **Rẽ nhánh 2:**  2.1 Thông tin bị trùng.  2.2 Lập lại bước 3 luồng sự kiện chính.  **Rẽ nhánh 3:**   * 1. Kiểm tra thông tin sản phẩm hợp lệ.   2. Cập nhật CSDL. |
| <Extend Use Case> | **Suadonhang**   1. Admin chọn chức năng sửa hãng 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 màng hình sửa sản phẩm   **Rẽ nhánh 1:**   * 1. Kiểm tra thông tin bị rỗng   1.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 2:**  2.1 Kiểm tra thông tin trùng  2.2 Lập lại bước 1 luồng sự kiệm chính  **Rẽ nhánh 3:**   * 1. Kiểm tra thông tin hợp lệ   2. Cập nhật vào CSDL |
| <Extend Use Case> | **Xóa sản phẩm**   * 1. Admin chọn chức năng xóa đơn hàng   2. Hiển thị thông tin đơn lên form   3. Admin nhấn nút xóa   4. Cập nhật vào CSDL   5. Hiển thị thông tin cập nhật   6. Admin nhấn nút thoát   7. Hủy màng hình xóa sản phẩm |

**Chương 5: THÀNH PHẦN XỬ LÝ**

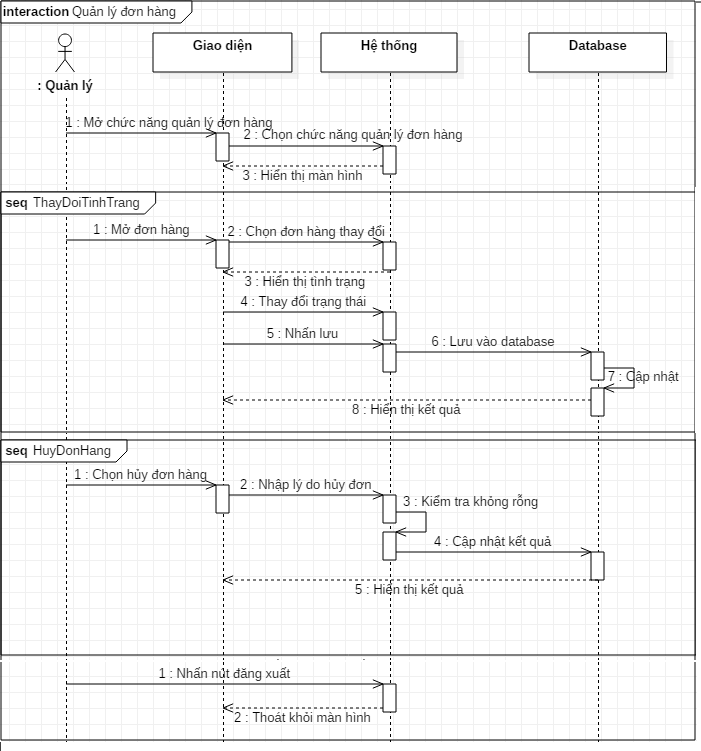
**Sơ đồ tuần tự đăng ký thành viên**

****

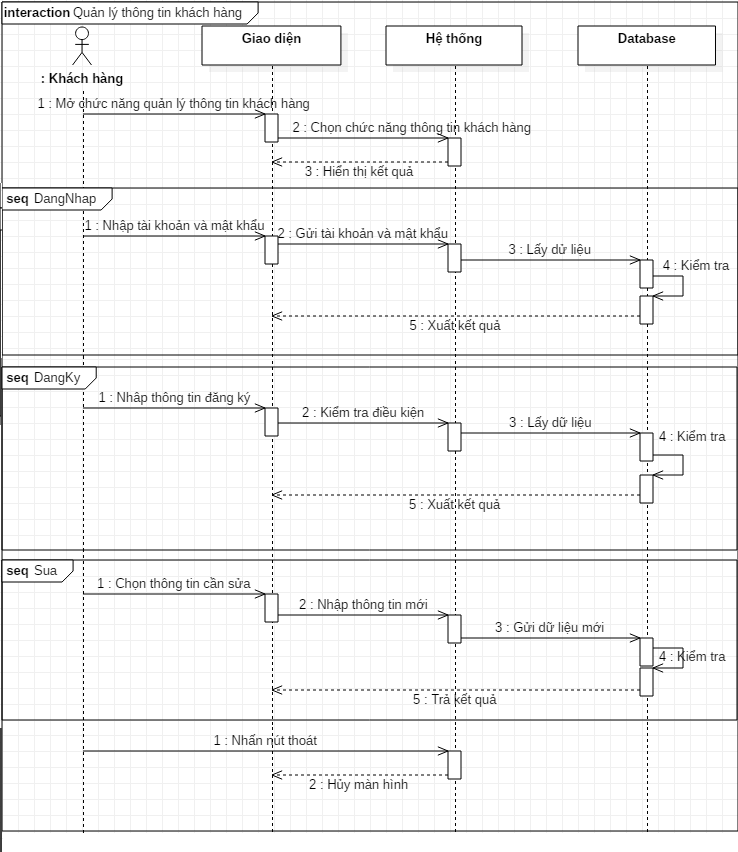
**Sơ đồ tuần tự đăng nhập**

****

**Sơ đồ tuần tự quản lý đơn hàng**



**Sơ đồ tuần tự quản lý thông tin khách hàng**



# **Chương 7:** **Thành phần dữ liệu**