**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**---------------**



**BÀI TẬP LỚN MÔN LẬP TRÌNH WEB**

**ĐỀ BÀI:**

Building Dynamic Data Applications:

Introducing ASP.NET Dynamic Data.

Building a Dynamic Data Application.

Working with Dynamic Data Templates.

Building ASP.NET MVC Applications:

Introducing the ASP.NET MVC Framework.

Creating Your First ASP.NET MVC Application.

Building an MVC Page.

Accepting Form Input.

Sinh viên thực hiện:

**1551010145 - Nguyễn Thị Bích Tuyền**

Giảng viên hướng dẫn

**ThS. Tô Oai Hùng**

**Mục lục:**

[1. Tổng quan về ASP.NET Dynamic Data 3](#_Toc531608452)

[1.1. Giới thiệu về ASP.NET Dynamic Data 3](#_Toc531608453)

[1.1.1. Mô hình cấu trúc của Dynamic Data 3](#_Toc531608454)

[1.1.2. Yếu tố lớp dữ liệu 6](#_Toc531608455)

[1.2. Cách tạo 1 chương trình Dynamic Data 6](#_Toc531608456)

[1.3. Làm việc với mô hình ASP.NET Dynamic Data 6](#_Toc531608457)

[1.3.1. Kết nối cơ sở dữ liệu 6](#_Toc531608458)

[1.3.2. Register data 10](#_Toc531608459)

[1.3.3. Chạy chương trình ASP.NET dynamic data 10](#_Toc531608460)

[1.3.4. Thao tác trên dữ liệu 11](#_Toc531608461)

[2. Tổng quan về ASP.NET MVC 15](#_Toc531608462)

[2.1. Giới thiệu về ASP.NET MVC Framework 15](#_Toc531608463)

[2.1.1. Tổng quan về mô hình MVC 15](#_Toc531608464)

[2.1.2. Tổng quan về lập trình ASP.NET MVC: 16](#_Toc531608465)

[2.2. Cách tạo 1 chương trình ASP.NET MVC: 16](#_Toc531608466)

[2.3. Chương trình ASP.NET MVC mặc định: 19](#_Toc531608467)

[2.3.1. Cấu trúc chương trình: 19](#_Toc531608468)

[2.3.2. Chạy chương trình: 19](#_Toc531608469)

[2.3.3. Giải thích chương trình: 21](#_Toc531608470)

[2.4. Nhận dữ liệu nhập: 23](#_Toc531608471)

[2.5. Demo chương trình ASP.NET MVC: 23](#_Toc531608472)

Danh sách hình ảnh.

Hình 1: Cấu trúc Dynamic Data

Hình 2: Page template

Hình 3: Entity template

Hình 4: Field template

Hình 5: Filter template

Hình 6: Tạo project ASP.NET Dynamic Data

Hình 7: Thư mục App\_Data

Hình 8: Thư mục Server Explorer

Hình 9: Cấu trúc LinQ to SQL data

Hình 10.1: Sử dụng ADO.NET Entity Framework

Hình 10.2: Sử dụng ADO.NET Entity Framework

Hình 10.3: Sử dụng ADO.NET Entity Framework

Hình 10.4: Sử dụng ADO.NET Entity Framework

Hình 11: Cách chạy chương trình ASP.NET Dynamic Data

Hình 12: Giao diện chương trình ASP.NET Dynamic Data

Hình 13.1: Giao diện xem dữ liệu trong ASP.NET Dynamic Data

Hình 13.2: Giao diện xem dữ liệu trong ASP.NET Dynamic Data

Hình 13.3: Giao diện xem dữ liệu trong ASP.NET Dynamic Data

Hình 14: Giao diện lọc dữ liệu trong ASP.NET Dynamic Data

Hình 15.1: Giao diện thêm dữ liệu trong ASP.NET Dynamic Data

Hình 15.2: Giao diện thêm dữ liệu trong ASP.NET Dynamic Data

Hình 16: Giao diện xem chi tiết, sửa, xóa dữ liệu trong ASP.NET Dynamic Data

Hình 17: Giao diện xem chi tiết, sửa, xóa dữ liệu trong ASP.NET Dynamic Data

Hình 18: Giao diện sửa dữ liệu trong ASP.NET Dynamic Data

Hình 19: Giao diện xem chi tiết dữ liệu trong ASP.NET Dynamic Data

Hình 20: Mô hình MVC

Hình 21: Cơ chế hoạt động của mô hình MVC

Hình 22.1: Tạo 1 chương trình ASP.NET MVC

Hình 22.2: Tạo 1 chương trình ASP.NET MVC

Hình 22.3: Tạo 1 chương trình ASP.NET MVC

Hình 23: Cấu trúc chương trình ASP.NET MVC

Hình 24: Chạy chương trình ASP.NET MVC

Hình 25: Chọn trình duyệt mặc định

Hình 26: Cấu hình duyệt mặc định

Hình 27: Giao diện chương trình ASP.NET MVC

Hình 28: Thư mục controller trong ASP.NET MVC

Hình 29: File HomeController trong ASP.NET MVC

Hình 30: Thư mục View trong ASP.NET MVC

Hình 31: File index.cshtml trong ASP.NET MVC

Hình 32: File RouteConfig.cs trong ASP.NET MVC

Hình 33: File Model.cs

Hình 34: File OurDbContext.cs

Hình 35: Thiết lập cac ActionResult trong Controller

Hình 36: Tạo View

Hình 37: Giao diện Register

# Tổng quan về ASP.NET Dynamic Data

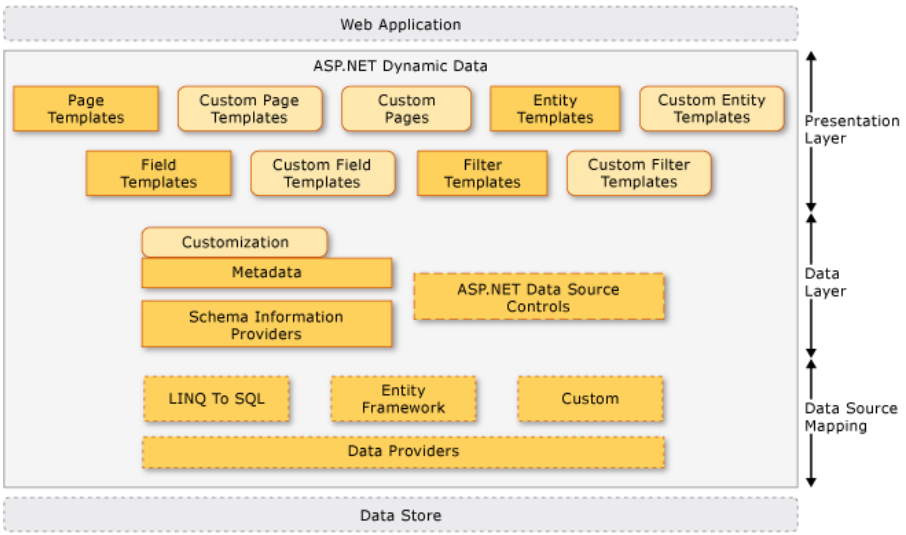
## Giới thiệu về ASP.NET Dynamic Data

ASP.NET Dynamic Data cung cấp một Framework cho phép chúng ta nhanh chóng xây dựng một chức năng ứng dụng driver-data, dựa trên LINQ to SQL hay Entity Framework.

Dựa trên cấu trúc của cơ sở dữ liệu(CSDL) mà Dynamic Data Framework (DDF) sẽ tạo nên các trang web cho phép người dùng xem/chèn/xóa/sửa dữ liệu.

Nhiều tính linh hoạt cho các DetailsView, FormView, GridView, ListView trong kiểm tra tính hợp lệ của dữ liệu, hoặc chỉnh sửa lại các mẫu để thay đổi cách hiển thị dữ liệu.

### Mô hình cấu trúc của Dynamic Data



*Hình 1: Cấu trúc Dynamic Data*

**Presentation Layer**: chứa các phần tử được sử dụng để tạo giao diện người dùng, hiển thị và chỉnh sửa các thực thể dữ liệu.

**Data Layer:** chứa mô hình dữ liệu đại diện cho các thực thể cơ sở dữ liệu như các kiểu CLR.

**Data Source Mapping:** chứa các phần tử là một phần của CLR nhưng được sử dụng bởi dữ liệu động. Các phần tử này là các công nghệ như [LINQ to SQL](http://go.microsoft.com/fwlink/?LinkId=177401) và [ADO.NET Entity Framework](http://go.microsoft.com/fwlink/?LinkId=177400) được sử dụng để tạo ra các mô hình dữ liệu.

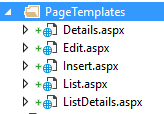
**Thành phần lớp Presentation**

Phần này cung cấp tổng quan về các yếu tố Dynamic Data được sử dụng để tạo giao diện người dùng, hiển thị và chỉnh sửa các thực thể dữ liệu.

**Page Templates (Mẫu trang)**

Các mẫu trang là các trang web hiển thị dữ liệu từ bất kỳ bảng nào trong cơ sở dữ liệu. Các mẫu trang sẽ hiển thị dữ liệu theo các chế độ sau: chế độ xem danh sách, chế độ xem chi tiết và chế độ xem chỉnh sửa. Mặc định, là chế độ xem danh sách.

Bạn có thể tùy chỉnh tại thư mục DynamicData\PageTemplates.



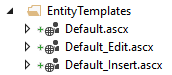
*Hình 2: Page template*

**Entity Templates (Mẫu thực thể)**

Mẫu thực thể cho phép bạn tùy chỉnh giao diện người dùng cho toàn bộ thực thể dữ liệu, chẳng hạn như hàng hoặc bảng. Điều này mang lại sự linh hoạt hơn việc tùy chỉnh các trường dữ liệu riêng lẻ và hữu ích nếu bạn muốn tạo giao diện người dùng tùy chỉnh và áp dụng nó cho các mẫu trang mặc định hoặc tùy chỉnh.

Bạn có thể tùy chỉnh tại thư mục DynamicData\EntityTemplates. Thư mục này chứa các mẫu thực thể mặc định. Có các mẫu để hiển thị, chỉnh sửa và chèn hoạt động. Các mẫu thực thể mặc định hiển thị dữ liệu trong hai cột, sử dụng nhãn cho tên trường và điều khiển thích hợp cho giá trị trường. Bạn có thể sửa đổi các mẫu thực thể mặc định để thay đổi giao diện và hành vi của các hàng dữ liệu cho toàn bộ trang web.

Các mẫu thực thể được bao gồm trong các mẫu trang Details.aspx, Insert.aspx và Update.aspx và bất kỳ trang tùy chỉnh nào. Các mẫu thực thể lần lượt sử dụng các mẫu trường để hiển thị dữ liệu thực tế.



*Hình 3: Entity template*

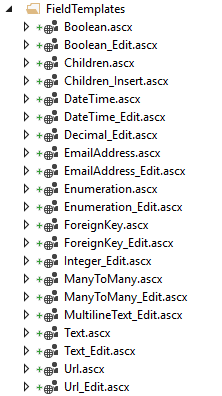
**Field Templates (Mẫu trường)**

Các mẫu trường là các điều khiển dùng để hiển thị giao diện người dùng cho các trường dữ liệu riêng lẻ. Theo mặc định, dữ liệu động chọn mẫu trường dựa trên kiểu dữ liệu của trường đang được hiển thị.

Ví dụ: để hiển thị dữ liệu Boolean, Dữ liệu động sử dụng mẫu trường Boolean; để hiển thị dữ liệu văn bản, Dữ liệu động sử dụng mẫu trường văn bản...

Thường có một mẫu trường để hiển thị dữ liệu và một mẫu khác cho phép người dùng nhập hoặc chỉnh sửa giá trị trường.

Bạn có thể tùy chỉnh tại thư mục DynamicData\FieldTemplates.



*Hình 4: Field template*

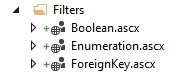
**Filter Templates (Mẫu bộ lọc)**

Các mẫu bộ lọc là các điều khiển hiển thị giao diện người dùng để lọc dữ liệu, cho phép người dùng chọn các hàng trong bảng để hiển thị dựa trên giá trị cột.

Ví dụ: bạn có thể hiển thị tất cả các hàng chứa một danh mục nhất định. Theo mặc định, dữ liệu động cung cấp các mẫu bộ lọc Boolean, khóa ngoài và liệt kê.

Dữ liệu động cho phép bạn sử dụng đánh dấu trang để tạo giao diện người dùng cho lọc dữ liệu trong một cột. Khi bạn tạo một dự án Web dữ liệu động, Visual Studio thêm một thư mục DynamicData \ FilterTemplates vào dự án. Thư mục này chứa các mẫu bộ lọc mặc định.

Nếu mẫu bộ lọc không tồn tại cho loại cột đó, bạn có thể tạo một mẫu tùy chỉnh và hướng dẫn Dữ liệu động sử dụng nó khi tạo giao diện người dùng. Bạn cũng có thể tạo đánh dấu trang để cho phép dữ liệu động tự động tạo giao diện người dùng cho bất kỳ loại cột nào mà mẫu bộ lọc mặc định tồn tại.



*Hình 5: Filter template*

### Yếu tố lớp dữ liệu

ASP.NET Dynamic hỗ trợ các mô hình dữ liệu [LINQ to SQL](http://go.microsoft.com/fwlink/?LinkId=177400) và [ADO.NET Entity Framework](http://go.microsoft.com/fwlink/?LinkId=177400) .

Dữ liệu động sử dụng các loại này để truy vấn cơ sở dữ liệu và tạo, đọc, cập nhật và xóa dữ liệu (nghĩa là, để thực hiện các hoạt động CRUD). Mô hình dữ liệu cung cấp một cách dễ dàng để tích hợp xác thực dữ liệu tùy chỉnh và các quy tắc logic nghiệp vụ.

Vào thời gian chạy, Dữ liệu động sẽ tự động trích xuất thông tin siêu dữ liệu từ mô hình dữ liệu, chẳng hạn như thuộc tính trường dữ liệu. Từ thông tin này, nó sẽ hướng dẫn cách tạo giao diện người dùng để hiển thị và chỉnh sửa dữ liệu.

## Cách tạo 1 chương trình Dynamic Data

Bước 1: Mở Visual Studio hoặc Visual Web Developer.

Bước 2: Trên thanh menu chọn “File”, chọn “Web site”.

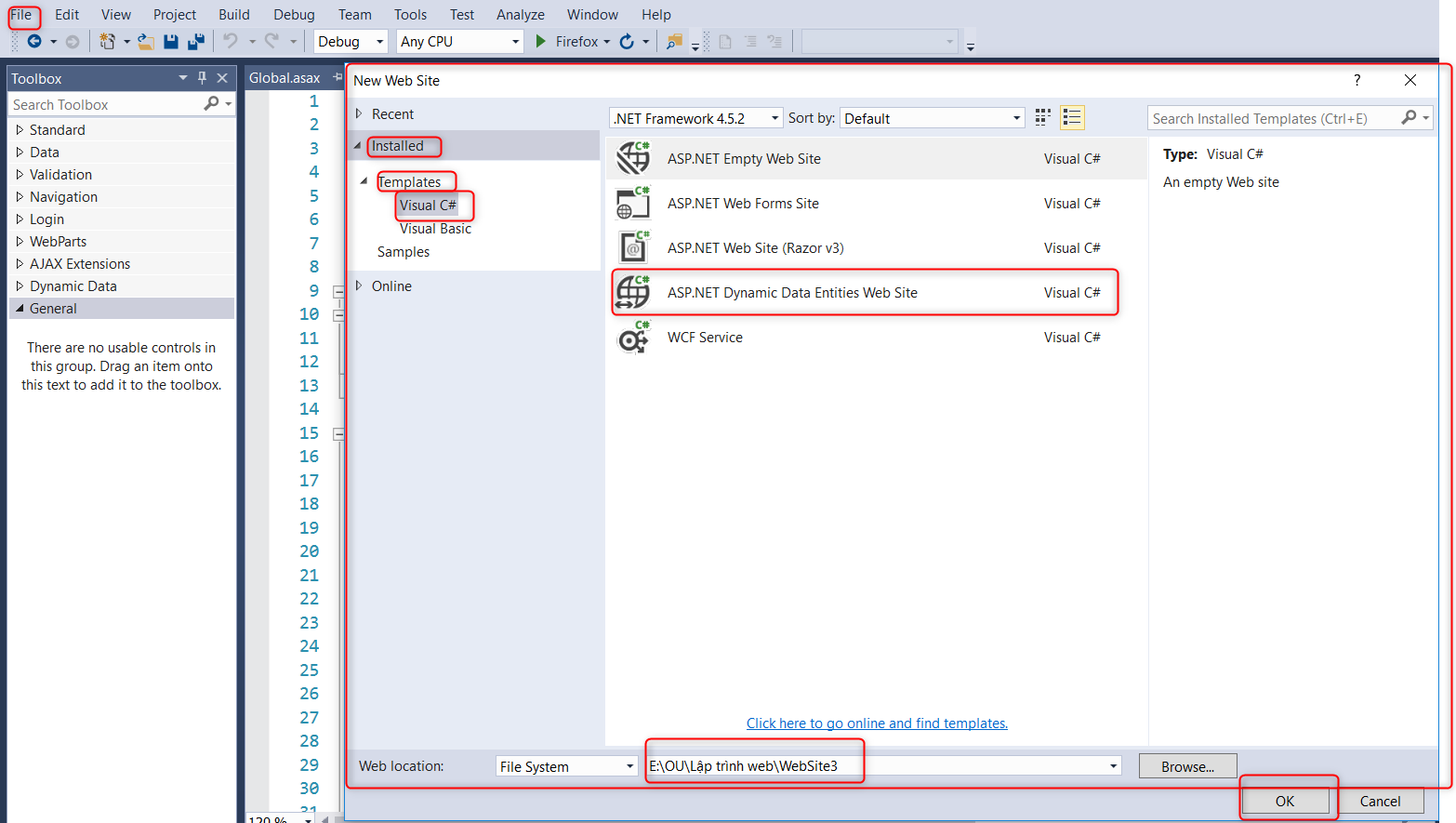
Cửa sổ “New Web Site” sẽ hiện ra.

Bước 3: Dưới mục “Installed/Templates”, chọn Visual C#.

Bước 4: Chọn ASP.NET Dynamic Data Entities Web Site.

Bước 5: Chọn địa chỉ để lưu project.

Bước 6: Click OK.



*Hình 6: Tạo project ASP.NET Dynamic Data*

## Làm việc với mô hình ASP.NET Dynamic Data

### Kết nối cơ sở dữ liệu

Thêm dữ liệu vào thư mục App\_Data (lưu ý nên kéo thả vào):



*Hình 7: Thư mục App\_Data*

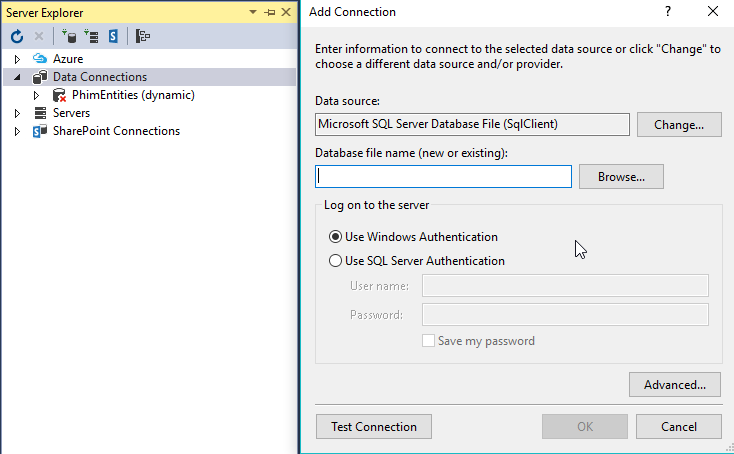
Tiến hành kết nối cơ sở dữ liệu theo các bước:

Mở giao diện Server Explorer.

Chuột phải vào Data Connections, chọn Add Connection…

Nhập vào đường link dẫn đến CSDL trong mục Database file name.

Nhấn Test Connection và chọn OK.



*Hình 8: Thư mục Server Explorer*

Một số trường hợp do file .mdf không tương thích ta vẫn kết nối bình thường. Hệ thống sẽ tự động thay đổi định nghĩa lại file .mdf.

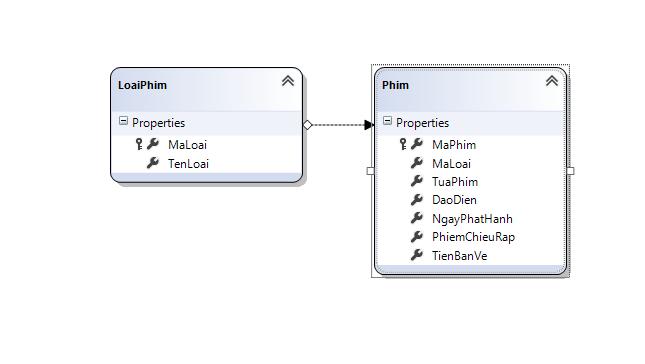
**Sử dụng mô hình dữ liệu**[**LINQ to SQL**](http://go.microsoft.com/fwlink/?LinkId=177400)

Ta thực hiện theo các bước như sau:

Chuột phải vào project chọn Add new item…

Chọn [**LINQ to SQL**](http://go.microsoft.com/fwlink/?LinkId=177400) **classes**

Chọn OK, sau thao tác ta sẽ thấy diagram như sau:



*Hình 9: Cấu trúc LinQ to SQL data*

**Sử dụng mô hình dữ liệu**[**ADO.NET Entity Framework**](http://go.microsoft.com/fwlink/?LinkId=177400)

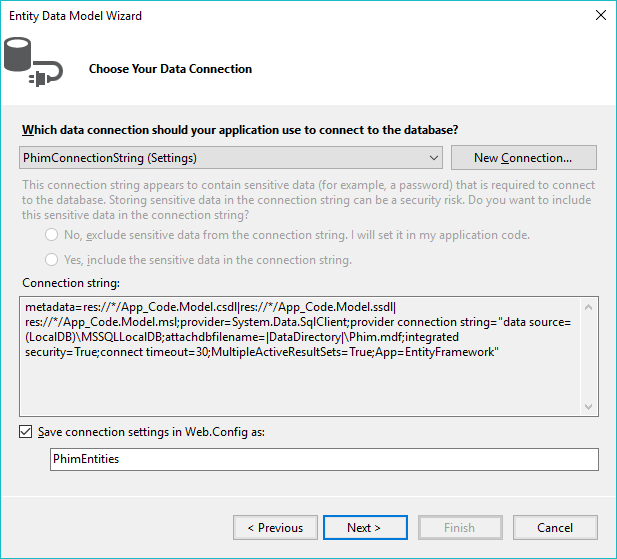
Ta thực hiện theo các bước như sau:

Chuột phải vào project chọn Add new item…

Chọn ADO.NET Entity Data Model

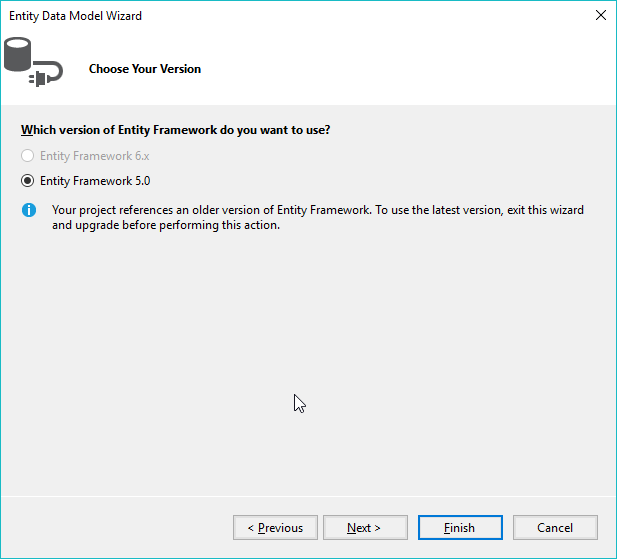
Tùy chọn database first hoặc code first.

Chọn data muốn kết nối



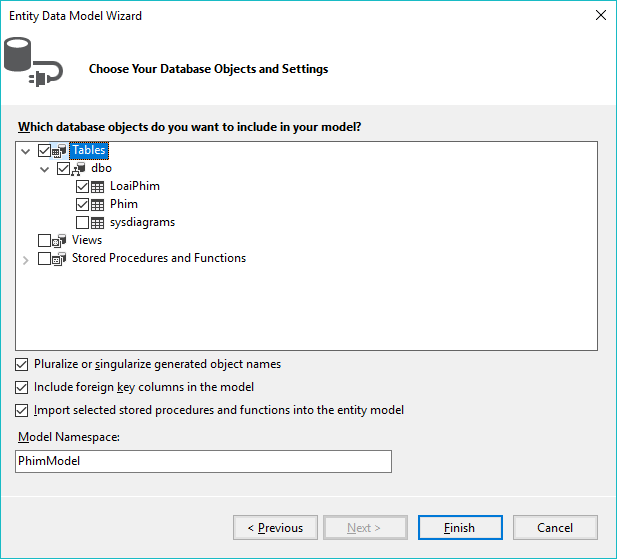
*Hình 10.1: Sử dụng ADO.NET Entity Framework*

Chọn next, chọn version Entity Framwork.



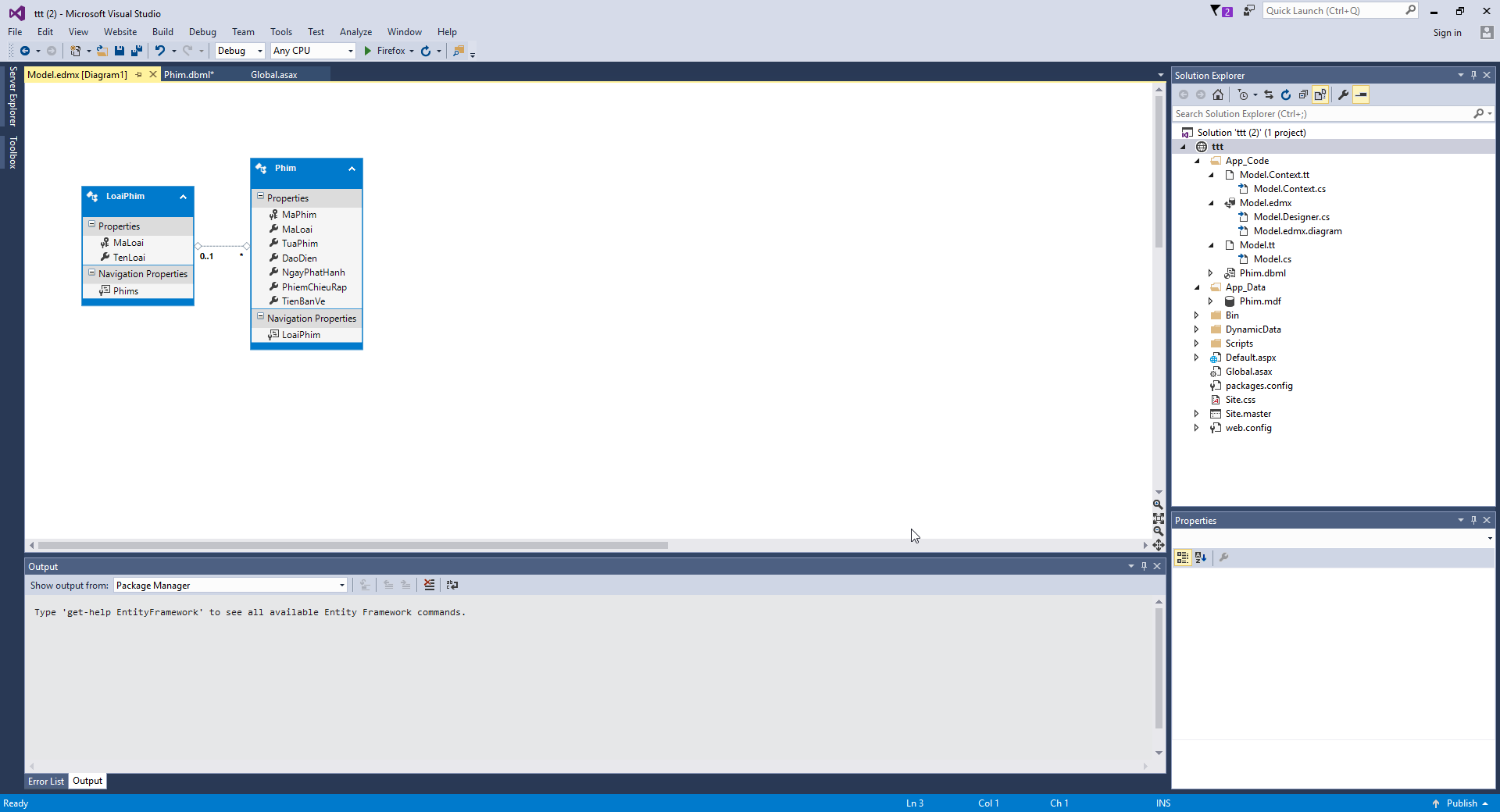
*Hình 10.2: Sử dụng ADO.NET Entity Framework*

Chọn tables, views, procedure, function trong cơ sở dữ liệu:



*Hình 10.3: Sử dụng ADO.NET Entity Framework*

Chọn finish, ta sẽ thấy cửa sổ như hình:



*Hình 10.4: Sử dụng ADO.NET Entity Framework*

### Register data

**Lưu ý**: kể từ visual 2012 trở đi, syntax để đăng kí dữ liệu đã có những thay đổi.

Syntax cũ:

DefaultModel.RegisterContext(typeof(DbEntities), new ContextConfiguration() { ScaffoldAllTables = false });

Syntax mới:

DefaultModel.RegisterContext(() =>

{

return ((IObjectContextAdapter)new YourDataContextType()).ObjectContext;

}, new ContextConfiguration() { ScaffoldAllTables = false });

**Sử dụng mô hình dữ liệu**[**LINQ to SQL**](http://go.microsoft.com/fwlink/?LinkId=177400)

DefaultModel.RegisterContext( new Microsoft.AspNet.DynamicData.ModelProviders.EFDataModelProvider(() =>   
                RendicontazioneEntities()),  
                newContextConfiguration { ScaffoldAllTables = true });

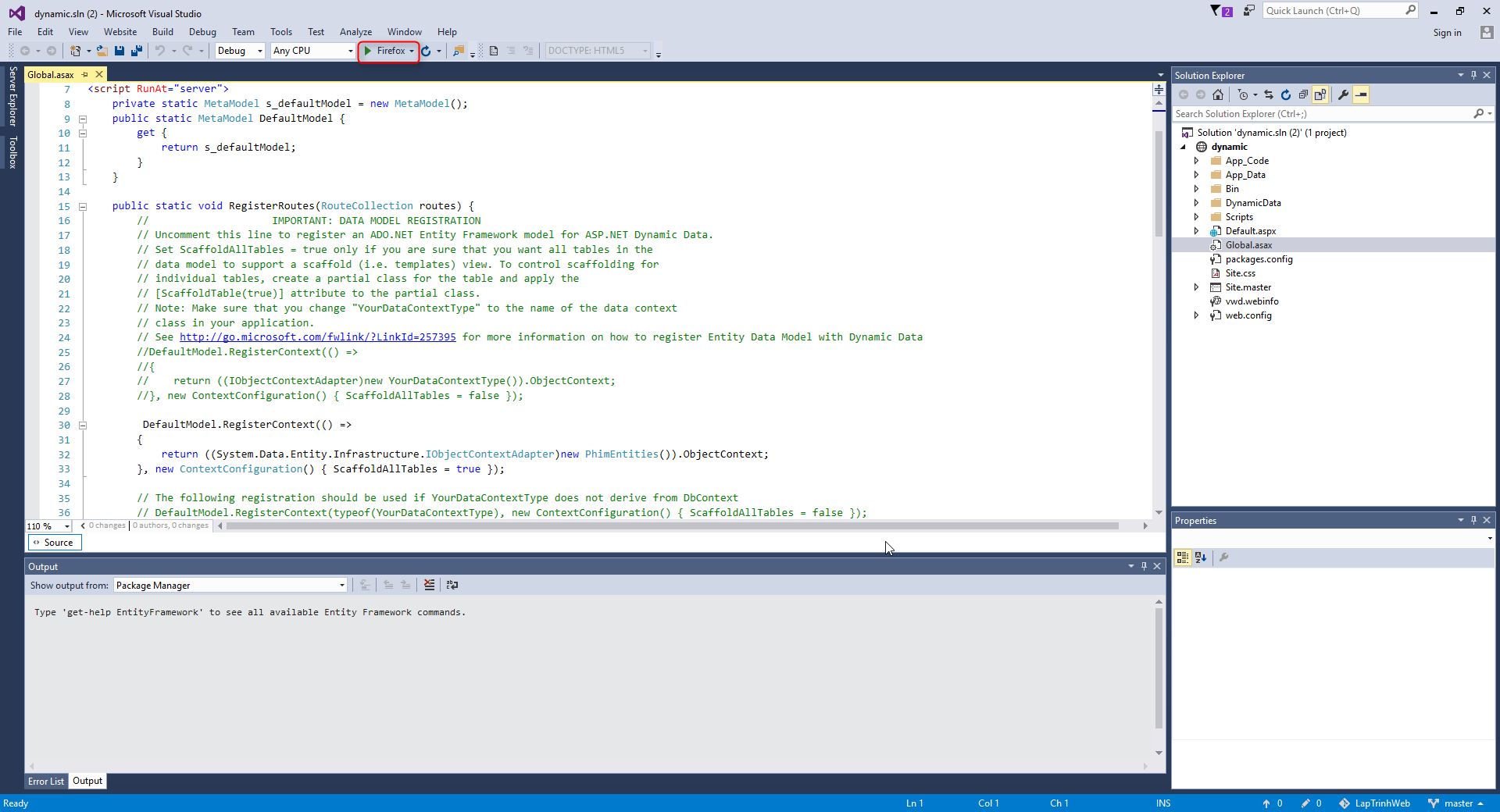
**Sử dụng mô hình dữ liệu**[**ADO.NET Entity Framework**](http://go.microsoft.com/fwlink/?LinkId=177400)

DefaultModel.RegisterContext(() =>  
        {  
            return ((System.Data.Entity.Infrastructure.IObjectContextAdapter)new RendicontazioneEntities()).ObjectContext;  
        }, new ContextConfiguration() { ScaffoldAllTables = true });

### Chạy chương trình ASP.NET dynamic data

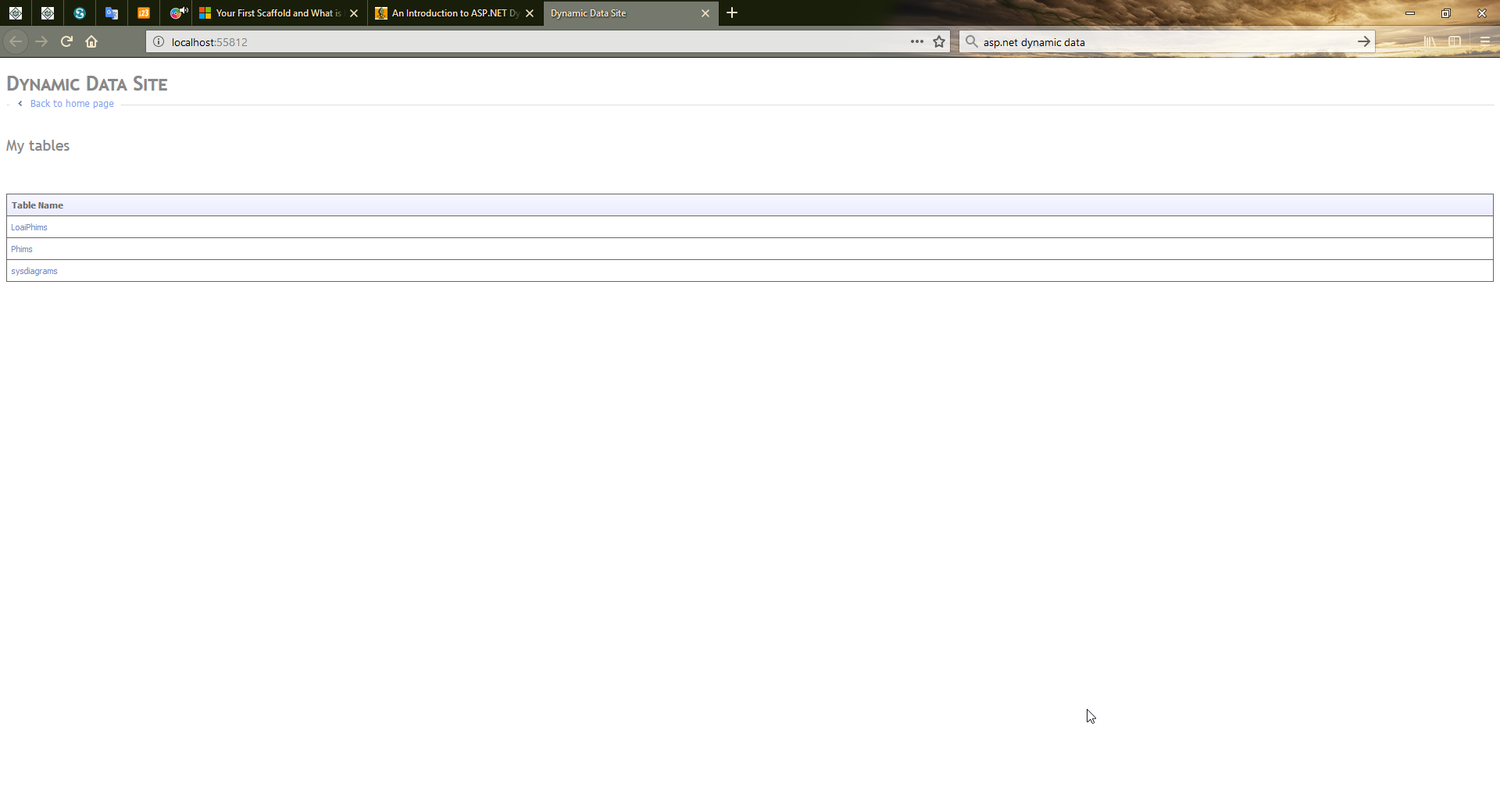
Sau khi đã cài đặt và register data xong, ta tiến hành chạy chương trình.

Nhấn chọn tam giác màu xanh trên thanh công cụ để khởi động website.



*Hình 11: Cách chạy chương trình ASP.NET Dynamic Data*

Sau khi website được thực thi, ta sẽ thấy có giao diện như sau:

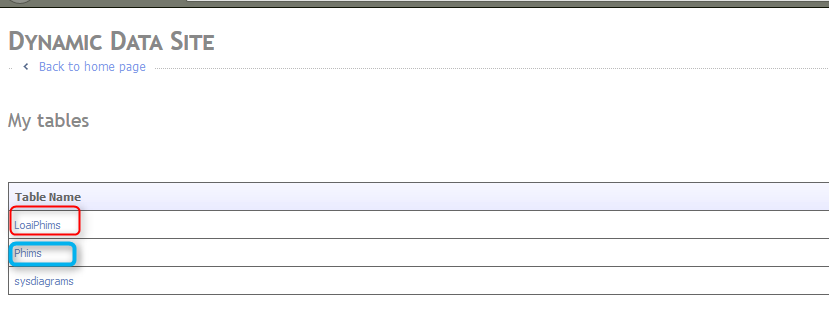


*Hình 12: Giao diện chương trình ASP.NET Dynamic Data*

### Thao tác trên dữ liệu

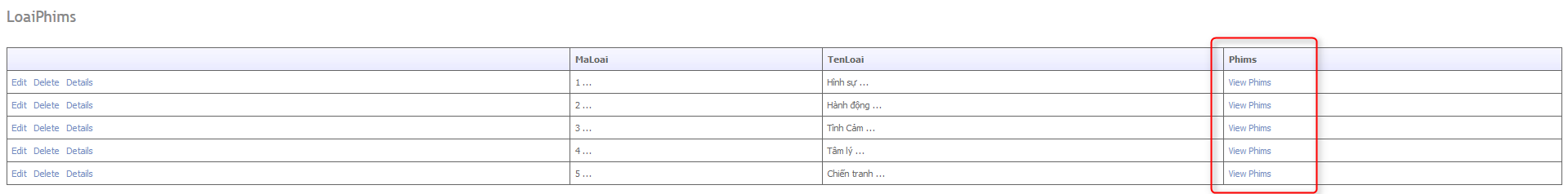
**Xem dữ liệu**

Chọn tên bảng muốn xem dữ liệu:



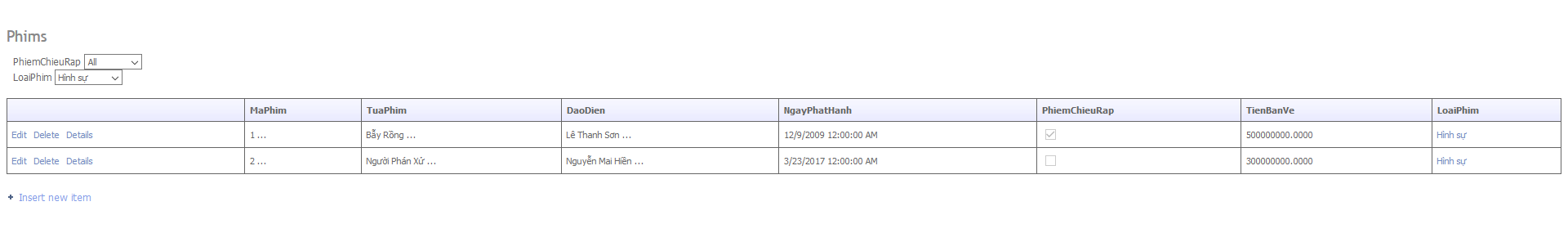
*Hình 13.1: Giao diện xem dữ liệu trong ASP.NET Dynamic Data*

Sau khi chọn bảng muốn xem, ta sẽ thấy đối với những bảng có khóa ngoại sẽ được thêm 1 cột liên kết đến dữ liệu bên bảng khác:



*Hình 13.2: Giao diện xem dữ liệu trong ASP.NET Dynamic Data*

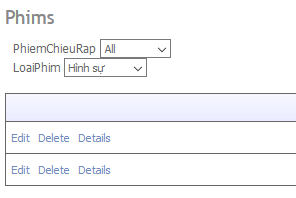
Click chọn ta sẽ xem được chi tiết dữ liệu được liên kết:



*Hình 13.3: Giao diện xem dữ liệu trong ASP.NET Dynamic Data*

**Lọc dữ liệu**

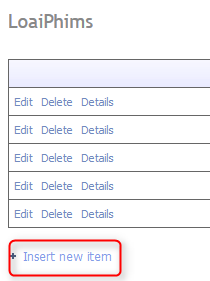
Hệ thống hỗ trợ xem và sắp xếp theo các cột dựa vào những cột đã định nghĩa sẵn.

****

*Hình 14: Giao diện lọc dữ liệu trong ASP.NET Dynamic Data*

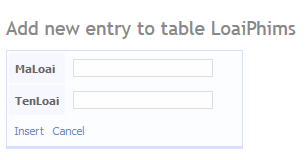
**Thêm dữ liệu**

Nhấn vào nút insert item dưới cùng mỗi bảng:



*Hình 15.1: Giao diện thêm dữ liệu trong ASP.NET Dynamic Data*

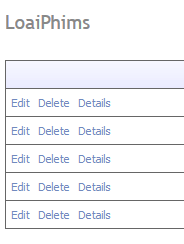
Tùy vào cấu trúc mỗi bảng ta sẽ có các cột tương ứng:



*Hình 15.2: Giao diện thêm dữ liệu trong ASP.NET Dynamic Data*

Nhập các trường tên ứng, chọn Insert, ta đã thêm thành công 1 hàng dữ liệu.

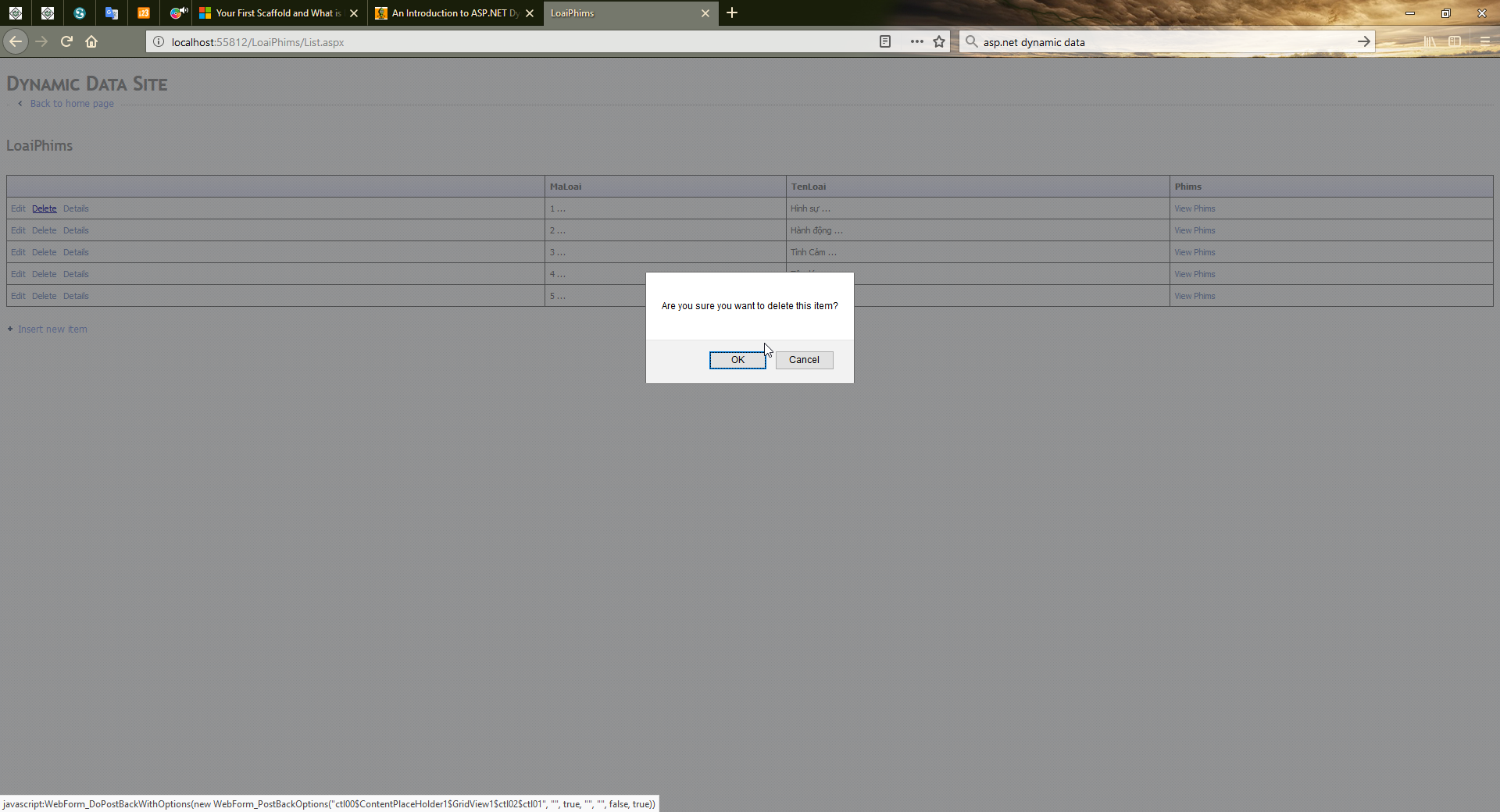
Ở mỗi bảng ASP.NET Dynamic Data sẽ hỗ trợ các chức năng xóa, sửa, xem chi tiết ở mổi dòng như sau:



*Hình 16: Giao diện xem chi tiết, sửa, xóa dữ liệu trong ASP.NET Dynamic Data*

**Xóa dữ liệu**

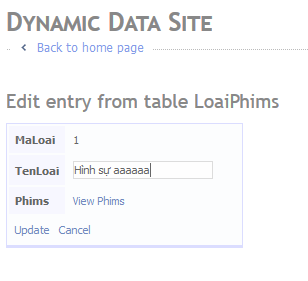
Chọn **Delete,** website sẽ hiện ra giao diện như hình để chắc chắn bạn muốn xóa dòng dữ liệu đã chọn.



*Hình 17: Giao diện xem chi tiết, sửa, xóa dữ liệu trong ASP.NET Dynamic Data*

**Sửa dữ liệu**

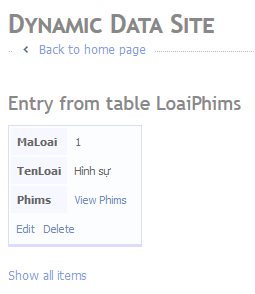
Chọn **Edit,** website sẽ hiện ra giao diện như hình để bạn tùy chỉnh dòng dữ liệu đã chọn. Nhấn update để lưu thay đổi và nhấn cancel để hủy bỏ.

****

*Hình 18: Giao diện sửa dữ liệu trong ASP.NET Dynamic Data*

**Chi tiết dữ liệu**

Chọn **Details,** website sẽ hiện ra giao diện như hình để bạn tùy chỉnh dòng dữ liệu đã chọn. Nhấn Edit để chỉnh sửa và nhấn delete để xóa.



*Hình 19: Giao diện xem chi tiết dữ liệu trong ASP.NET Dynamic Data*

# Tổng quan về ASP.NET MVC

## Giới thiệu về ASP.NET MVC Framework

### Tổng quan về mô hình MVC

*Hình 20: Mô hình MVC*

MVC là một design partern đã tồn tại rất lâu trong ngành công nghệ phần mềm. Một ứng dụng viết theo mô hình MVC sẽ bao gồm 3 thành phần tách biệt nhau đó là Model, View, Controller.

Giống như trong mô hình 3 lớp, mô hình MVC giúp tách biệt 3 tầng trong mô hình lập trình web, vì vậy giúp tối ưu ứng dụng, dễ dàng thêm mới và chỉnh sửa nội dung hoặc giao diện.

**Model**

Mô hình 3 lớp thì trong đó gồm có 2 tầng Data Access Layer và tầng Business Logic Layer. Hai tầng này là hai tầng tương đương với tầng model trong mô hình MVC.

**View**

Là tầng giao diện, hiển thị dữ liệu được truy xuất từ tầng model. Tầng này tương đương với tầng Presentation Layer trong mô hình 3 lớp.

**Controller**

Đây là tầng giúp kết nối giữa tầng model và tầng view trong mô hình MVC, có nghĩa là nếu phía client yêu cầu hiển thị dữ liệu thì controller gọi dữ liệu từ model và trả về cho view vì view tương tác trực tiếp với client.

Ví dụ:

User yêu cầu hiển thị thông tin cá nhân của user.

User gửi một yêu cầu tới controller.

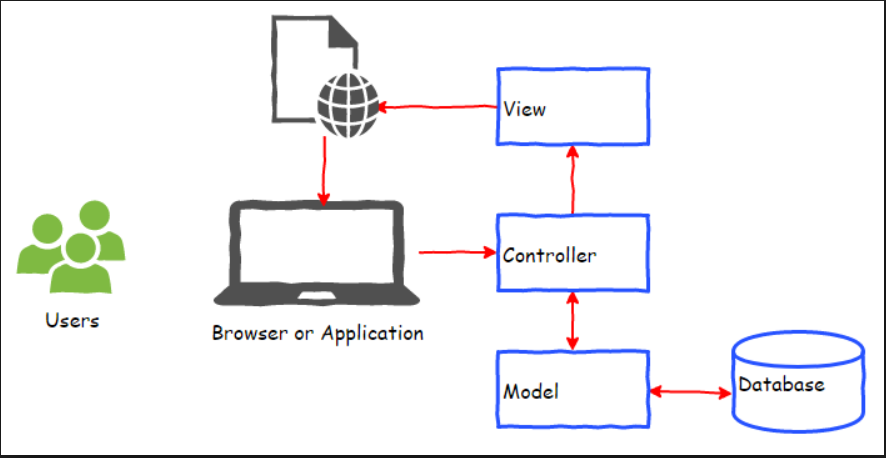
Controller nhận yêu cầu, xử lý yêu cầu, nếu yêu cầu cần truy xuất dữ liệu thì controller sẽ gửi yêu đó xuống tầng model để truy xuất dữ liệu.

Tầng model sẽ lấy dữ liệu từ database sau đó truyền dữ liệu qua tầng view thông qua tầng controller để tầng view hiển thị dữ liệu cho User.

User sẽ thấy thông tin hiển thị ở giao diện và cụ thể ở đây là tầng view.

### Tổng quan về lập trình ASP.NET MVC:

**Cơ chế hoạt động của ASP.Net MVC**



*Hình 21: Cơ chế hoạt động của mô hình MVC*

User gửi 1 yêu cầu tới server bằng cách truyền vào 1 URL trong browser.

Yêu cầu đó được gửi tới controller đầu tiên, controller sẽ xử lý yêu cầu, nếu yêu cầu cần truy xuất dữ liệu thì controller sẽ chuyển qua tầng model.

Tại tầng model, dữ liệu được truy xuất từ database và sau đó truyền qua view thông qua controller.

Controller sẽ giúp dữ liệu được chuyển từ model qua view.

View là tầng cuối cùng giao tiếp với User, mọi dữ liệu sẽ được hiển thị cho User thông qua tầng View.

**Những ưu điểm mà ASP.Net MVC mạng lại cho chúng ta:**

Do sử dụng mô hình MVC nên trong ASP.Net MVC đã tách biệt được các tầng trong mô hình lập trình web vì vậy giúp tối ưu ứng dụng và dễ dàng trong việc viết code, giao diện

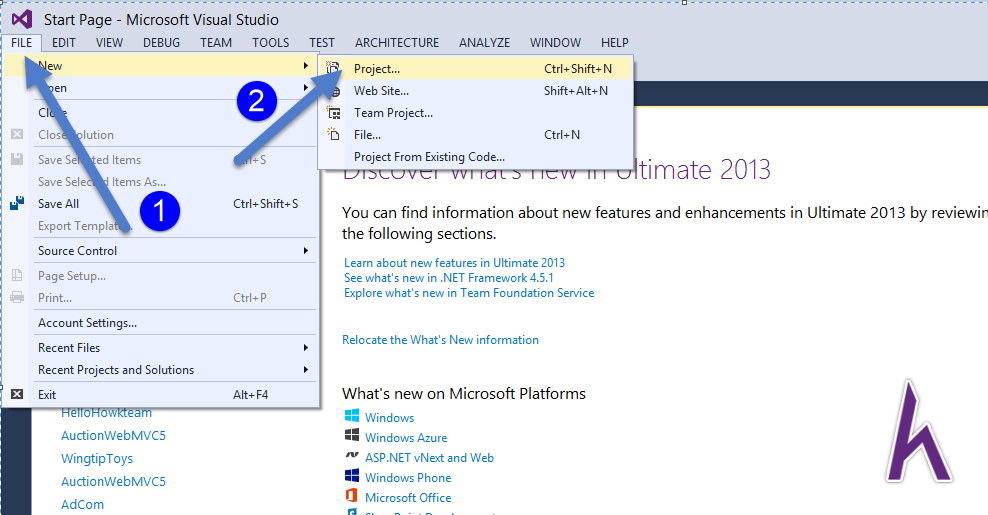
Giao diện trong ASP.Net MVCsử dụng công nghệ thiết kế web HTML, CSS nền việc thiết kế giao diện trở nên dễ dàng và giúp cho designer linh hoạt trong việc thiết kế

ASP.Net MVC không sử dụng view state vì vậy trang web không bị tăng kích thước do đó hiệu năng hoạt động không bị bị giảm

## Cách tạo 1 chương trình ASP.NET MVC:

Bước 1: Đầu tiên các bạn mở Visual Studio lên

Bước 2: Tại thanh menu bạn chọn FILE >New >Project hoặc là cách đơn giản hơn là bạn nhấn tổ hợp phím tắt **Ctrl + Shift + N**

 *Hình 22.1: Tạo 1 chương trình ASP.NET MVC*

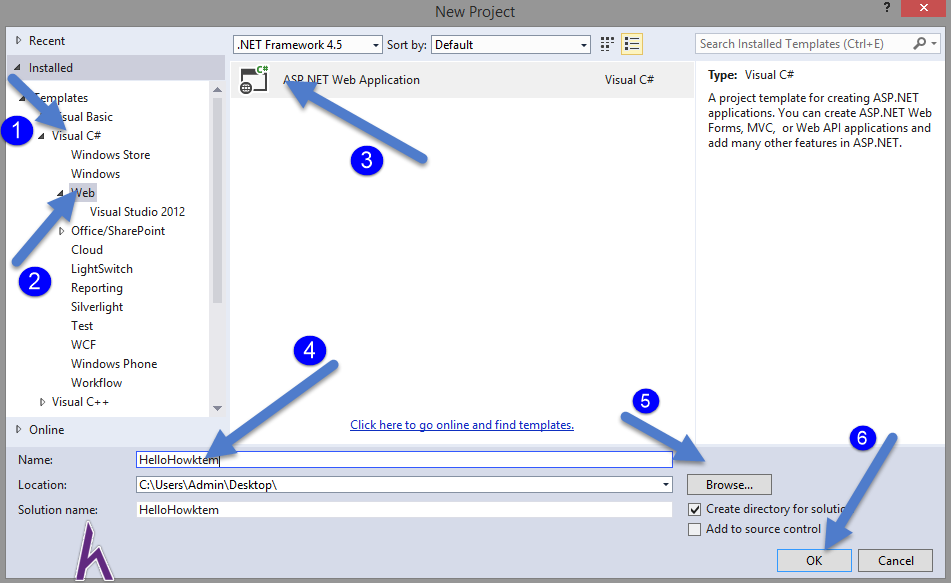
Bước 3: Sau khi bạn làm xong bước trên, chúng ta tiếp tục làm theo như hình:

Installed/ Templates 🡺 Visual C# 🡺 Web

ASP.NET Web Application

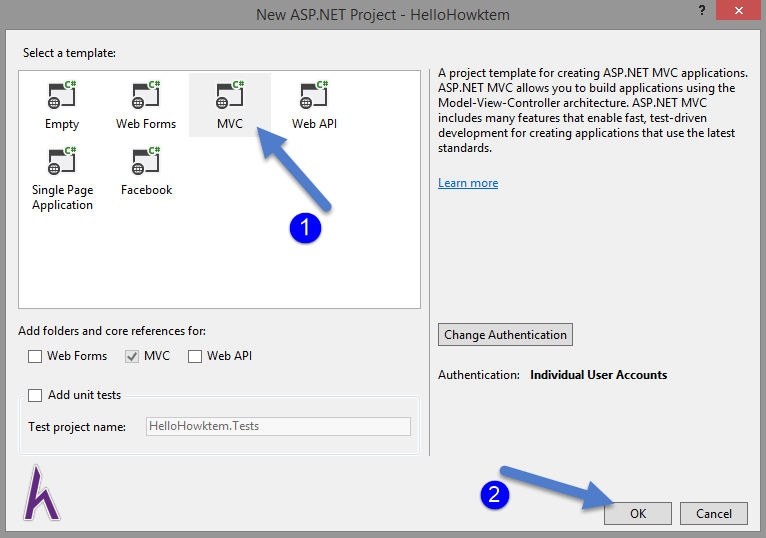
Nhập tên chương trình, địa chỉ lưu trữ

OK



*Hình 22.2: Tạo 1 chương trình ASP.NET MVC*

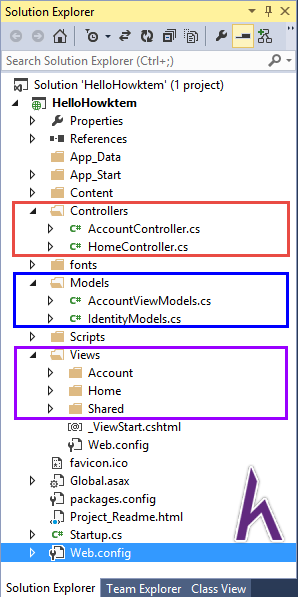
Bước 4: Sau khi bạn nhấn OK ở bước trên thì sẽ có 1 bảng hội thoại xuất hiện: Chọn MVC và nhấn OK.



*Hình 22.3: Tạo 1 chương trình ASP.NET MVC*

## Chương trình ASP.NET MVC mặc định:

### Cấu trúc chương trình:



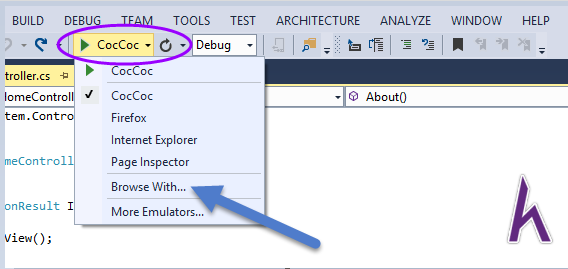
*Hình 23: Cấu trúc chương trình ASP.NET MVC*

Như hình mình show ra cho các bạn ở phía trên, vùng màu đỏ chính là Controller, vùng màu xanh là Models và vùng màu tím chính là Views trong mô hình MVC.

Tương ứng với 1 controller thì sẽ có 1 view đi kèm theo, mặc định project khi tạo gồm có 2 controller đó là Account và Home vì vậy ta sẽ thấy có 2 folder đó là Account và Home trong phần View.

### Chạy chương trình:

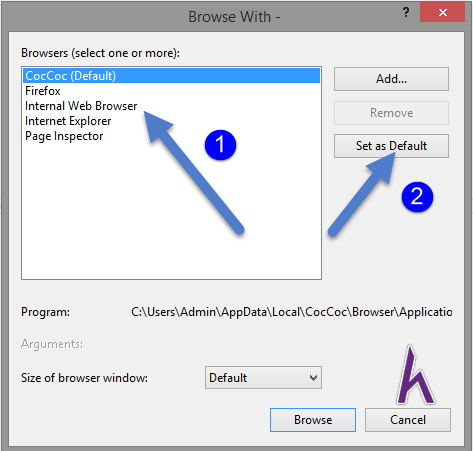
Trên thanh menu sẽ nhìn thấy nút mũi tên màu xanh, đây là nút giúp **Run project** lên, tuy nhiên ban đầu nếu thấy tại nút này không có hiển thị trình duyệt mặc định trong máy mình như cốc cốc, chrome, firefox … thì các ta chọn **Browse With..**



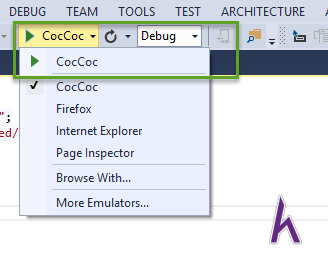
*Hình 24: Chạy chương trình ASP.NET MVC*

Sau khi chọn, 1 hộp thoại hiện lên, đầu tiên các ta chọn trình duyệt mà mình muốn sử dụng mặc định khi chạy project lên, sau đó chọn Set as Default.

Để chạy project bạn click vào nút **Run Project**

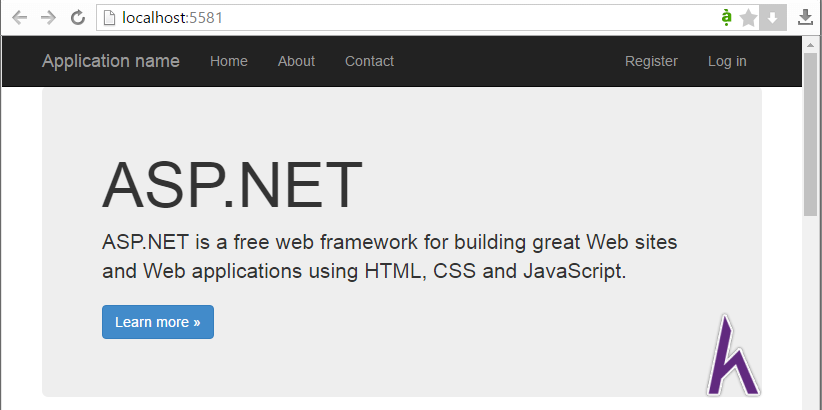


*Hình 25: Chọn trình duyệt mặc định*



*Hình 26: Cấu hình duyệt mặc định*

 Giao diện hiển thị khi chạy thành công như hình :

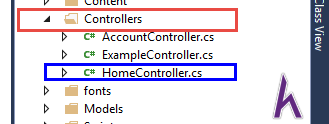


*Hình 27: Giao diện chương trình ASP.NET MVC*

### Giải thích chương trình:

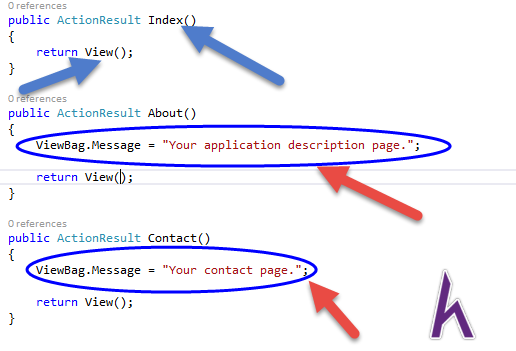
Để hiển thị được giao diện như trên ta sẽ tìm hiểu như sau:

Bước 1: Vào mục **Controller**.



*Hình 28: Thư mục controller trong ASP.NET MVC*

Bước 2: Sau đó chọn và mở **HomeController**:

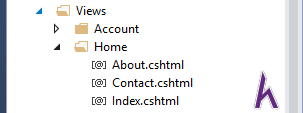


*Hình 29: File HomeController trong ASP.NET MVC*

Controller có cung cấp 1 phương thức có tên gọi là ActionResult. Phương thức này trả về rất nhiều kiểu dữ liệu khác nhau, và trong trường hợp này kiểu trả về là View().

Kiểu trả về View() dùng để hiển thị giao diện mà mình muốn xuất ra cho người dùng khi ActionResult được gọi lên.

 Bước 3: Mở thư mục **View:**



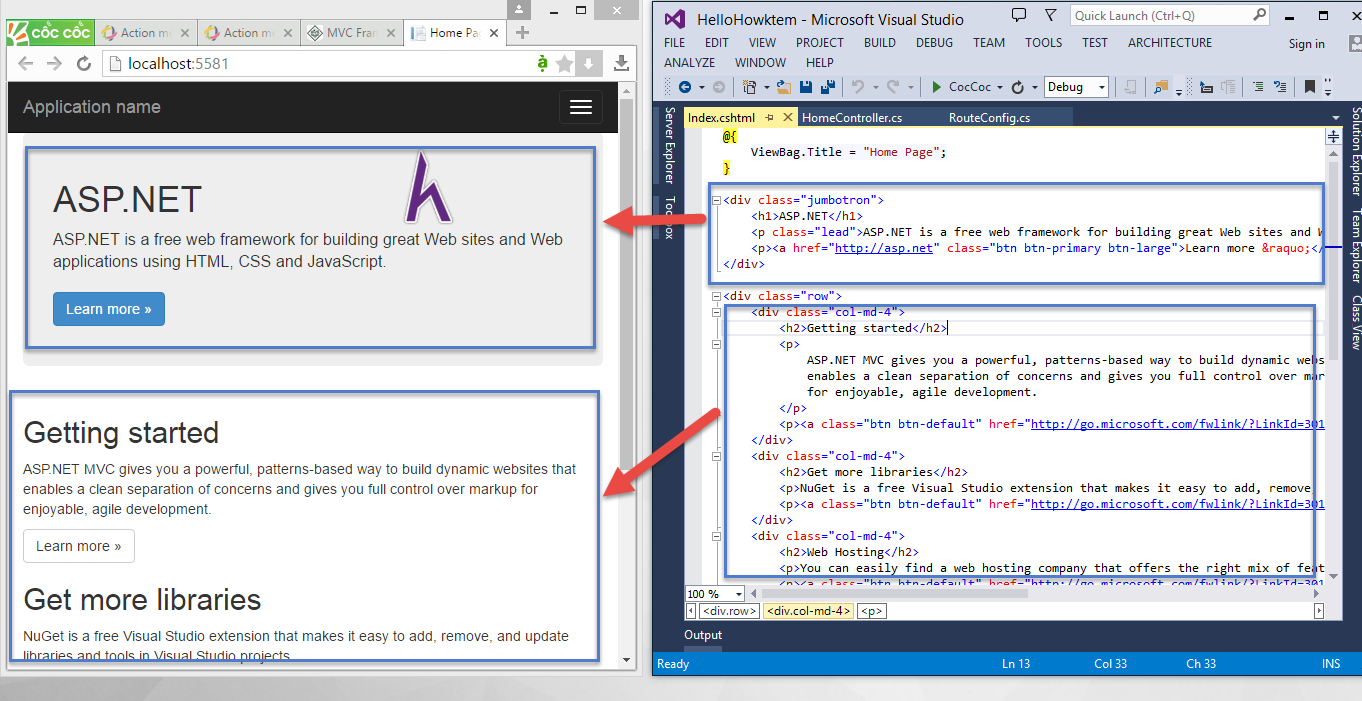
*Hình 30: Thư mục View trong ASP.NET MVC*

Trong HomeController bao gồm 3 ActionResult đó là **About**, **Contact**, **Index**. Mà cả 3 ActionResult này đều trả về kiểu dữ liệu là View() vì vậy trong folder home sẽ cũng có 3 View tương ứng cho 3 ActionResult.

Cụ thể khi ActionResult Index được gọi lên thì Index.cshtml sẽ được hiển thị, và 2 cái còn lại tượng tự như vậy.

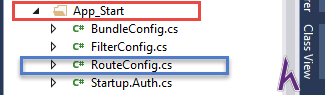
Bước 4: Click vào **Index.cshtml**:

Nội dung của file giống với nội dung được hiển thị khi chúng ta chạy project lên.



*Hình 31: File index.cshtml trong ASP.NET MVC*

Cài đặt file mặc định khi chương trình chạy: **App\_Start 🡺 RouteConfig.**

****

*Hình 32: File RouteConfig.cs trong ASP.NET MVC*

## Nhận dữ liệu nhập:

Việc nhận dữ liệu luôn là điều thường xuyên xảy ra khi xây dựng 1 ứng dụng web, điển hình như: đăng nhập, đăng kí…

Để nhận được dữ liệu từ người dùng ta thực hiện lần lượt các bước như sau:

Cài đặt entityFramework trong Nuget.

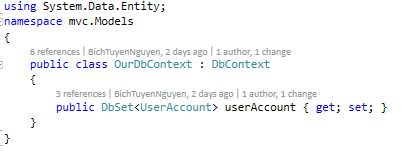
Tạo model:

* Tạo class mô tả các trường trong bảng:



*Hình 33: File Model.cs*

* Tạo 1 class entity:



*Hình 34: File OurDbContext.cs*

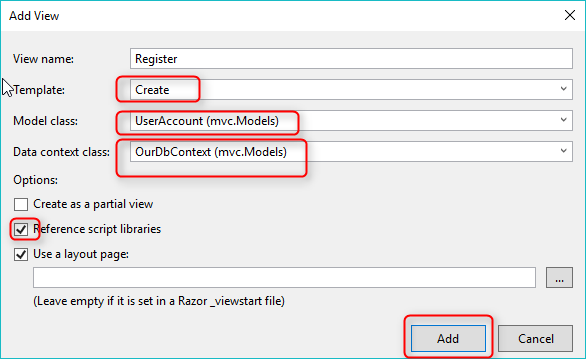
Tạo các hàm ActionResult trong controller:

* Hàm Register() đầu tiên là để website xác định được địa chỉ để chuyển tiếp View.
* Hàm Register() thứ hai để xử lý việc nhận dữ liệu từ người nhập.



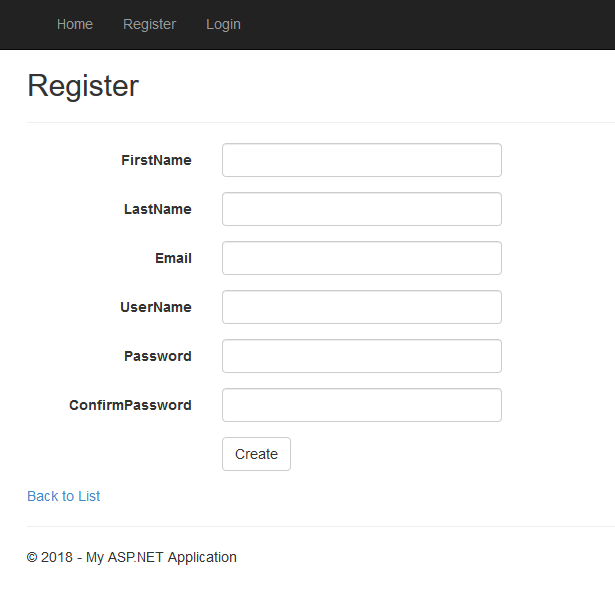
*Hình 35: Thiết lập cac ActionResult trong Controller*

Sau khi nhấn tạo view với tùy chọn như hình:



*Hình 36: Tạo View*

Hệ thống sẽ tạo động tạo giao diện các trường có trong cơ sở dữ liệu ta đã chọn.



*Hình 37: Giao diện Register*

Tài liệu kham khảo:

Stephen Walther, Kevin Scott Hoffman, Nate Scott Dudek-ASP.NET 4 Unleashed-SAMS Publishing (2010)

<https://docs.microsoft.com/en-us/aspnet/web-forms/videos/aspnet-dynamic-data/your-first-scaffold-and-what-is-dynamic-data>

[https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/database-first-development/generating-views?fbclid=IwAR00SNDutsnYk10G\_5CJfcDV6u2jsgnqDnTh9uXVoDzucBKR-ybVokWNiZA](https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/database-first-development/generating-views?fbclid=IwAR33EXMRsI7FDNLY7sURpK9VXflSHMM_F9mkWuxqW0DtsL2WEG3TjjyJMeY)