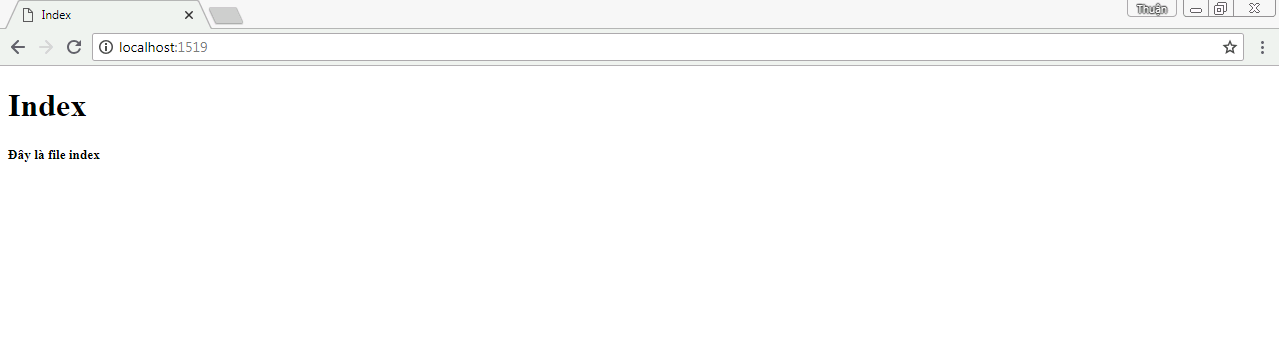
**HƯỚNG DẪN LẬP TRÌNH WEB VỚI ASP.NET MVC**

**( Phần 1 - Các thành phần của ASP.Net MVC)**

1. Controller và router

Trước hết hãy chạy lại kết quả hôm trước với URL như sau: <http://localhost:1519/>. Chú ý đây là địa chỉ Server của tôi, của bạn có thể khác. Mỗi lần chạy IIS cung cấp một cổng khác nhau.

Và kết quả chạy lên như sau:



Dù không có Controller và Action được chỉ định trên URL nhưng server vẫn hiểu và gửi trang Index của HomeController điều này xảy ra vì đâu?

Để trả lời câu hỏi trên quay lại với file RouteConfig.cs

|  |
| --- |
| C# |
| public class RouteConfig  {  public static void RegisterRoutes(RouteCollection routes)  {  routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");  routes.MapRoute(  name: "Default",  url: "{controller}/{action}/{id}",  defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }  );  }  } |

Tạm dịch như sau:

Phương thức static RegisterRoutes được gọi khi ứng dụng khởi chạy. Nó quy định các mẫu URL.

Hàm MapRoute đăng ký một mẫu URL cho server và dĩ nhiên bạn có thể đăng ký nhiều mẫu URL cho một ứng dụng. ASP.Net sẽ lựa chọn mẫu phù hợp nhất với URL của bạn và nếu có nhiều mẫu phù hợp thì mẫu phía trên sẽ được chọn. Giờ ta tiến hành phân tích các tham số của hàm.

Name: đây là tên route, bạn có thể đặt bất kỳ tên gì.

Url: mẫu URL, nếu người dùng gửi một URL phù hợp với mẫu trên thì server sẽ tìm đến đúng controller và Action phù hợp để khởi chạy. Mẫu URL mặc định là {Controller}/{Action}/{Parameter}. Với mẫu này URL phù hợp là Home/Index/1.

Defaults: Tham số này quy định Action mặc định sẽ được thực thi nếu như URL thỏa Home/ , Home/Index, Home/Index/1, và chỉ có Localhost.

* Bây giờ bạn đã hiểu được cách đăng ký một URL cho hệ thống. Việc làm tiếp theo là đăng ký một URL xữ lý việc đăng nhập hệ thống. Cách làm khá đơn giản.

Thêm một MapRoute như sau:

|  |
| --- |
| C# |
| public static void RegisterRoutes(RouteCollection routes)  {  routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");  routes.MapRoute(  name: "Default",  url: "{controller}/{action}/{id}",  defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }  );  routes.MapRoute(  name: "Login",  url: "{controller}/{action}/{id}/{pass}",  defaults: new { controller = "Account", action = "Login", id = UrlParameter.Optional, pass = UrlParameter.Optional}  );  } |

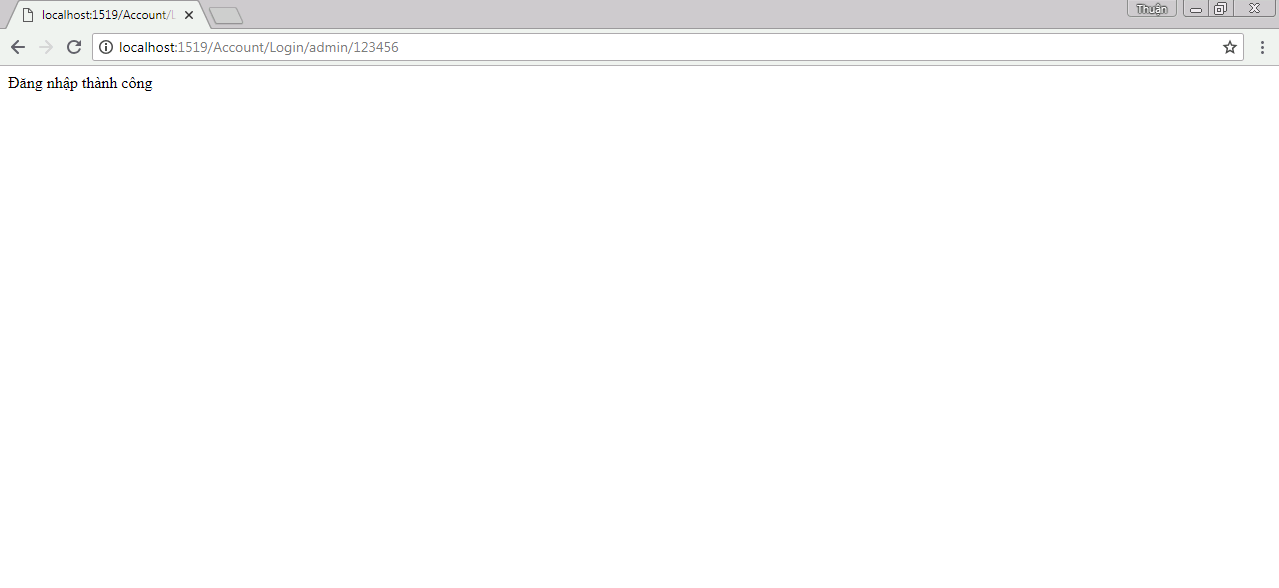
Trong thư mục Controllers thêm một controller với tên Account và hai Action: Index và Login.

|  |
| --- |
| C# |
| public class AccountController : Controller  {  public ActionResult Index()  {  return View();  }  public string Login(string id,string pass)  {  if(id.Equals("admin")&&pass.Equals("123456"))  {  return "Đăng nhập thành công";  }  else  {  return "Sai tên đăng nhập hoặc mật khẩu";  }  } |

Có một điều thú vị là Action Login trả về một chuỗi chứ không phải một View điều này thật thú vị. Thực ra ASP.Net cho phép bạn trả về nhiều kiểu khác nhau trong một Action có thể là một file, một chuỗi, View, Redirect, vv…

Giờ chạy ứng dụng lên và ghõ URL như sau: <http://localhost:1519/Account/Login/admin/123456>

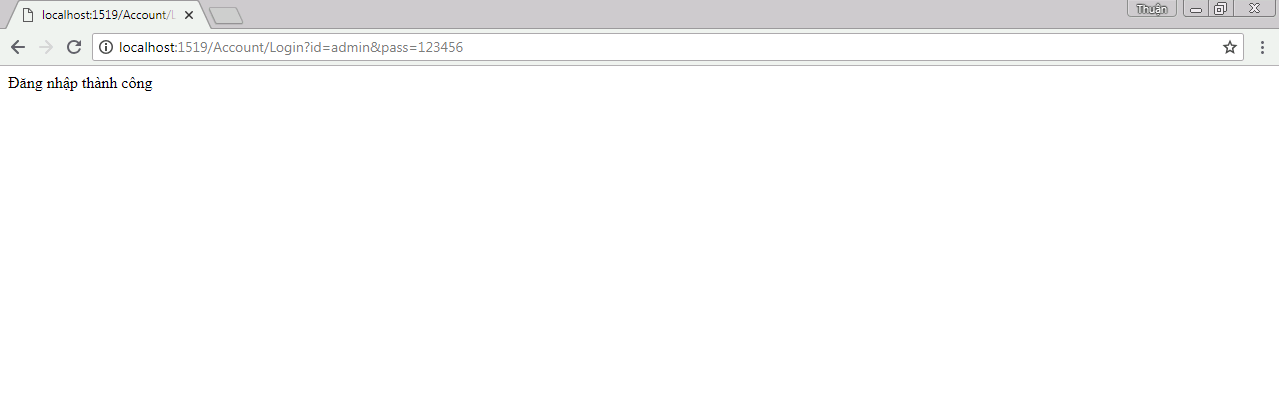
Kết quả:



Theo cơ chế Mapping URL thì URL thứ 2 trong RouteConfig được thực hiện điều này khá dễ hiểu vì mẫu URL đầu tiên không khớp với URL bạn nhập vào.

Action Login khởi chạy và cho ra màn hình một chuỗi là đăng nhập thành công. Bạn cũng có thể chạy với URL như sau và két quả là giống nhau:

<http://localhost:1519/Account/Login?id=admin&pass=123456>



URL này có vẽ quen quen. Đây là URL cho phương thức HTTP GET đã được nhắc đến trong bài học trước. Khi bạn không chỉ định một Action là GET hay POST thì mặc nhiên nó là GET.

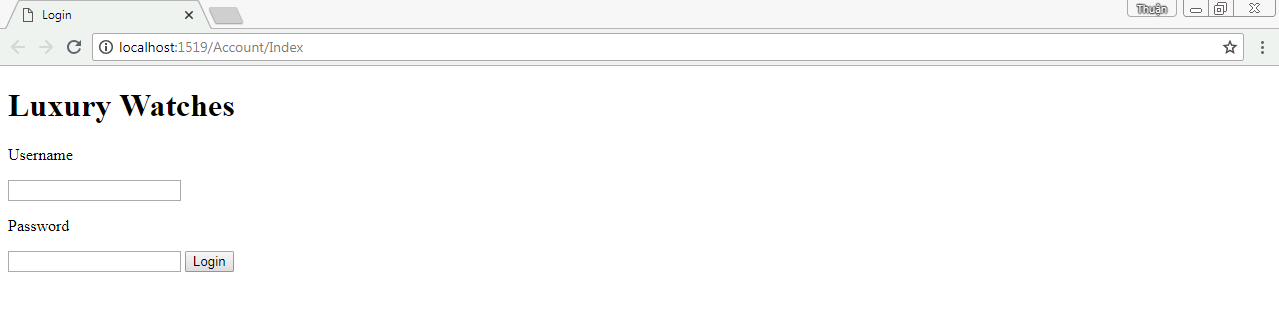
* Đến đây có lẽ bạn đã biết cách tạo và cấu hình một URL phù hợp với một controller. Phần sau chúng ta sẽ tiến hành tạo giao diện cho các action.

1. View và cấu hình resource

Trong phần trước chúng ta đã tìm hiểu qua cách cấu hình các mẫu URL và tạo ra một controller xữ lý việc đăng nhập hệ thống. Trong phần này chúng ta sẽ tạo ra một giao diện đăng nhập. Như đã nói ở trước ASP.NET hổ trợ bạn viết giao diện bằng mã HTML và C#. Bây giờ tạo mới một View của Action Index thuộc Controller Account và viết mã như sau.

|  |
| --- |
| HTML |
| <!DOCTYPE html>  <html>  <head>  <meta name="viewport" content="width=device-width" />  <title>Login</title>  </head>  <body>  <div class="wrap">  <div class="panel"><h1>Luxury Watches</h1></div>  <div class="form">  <form id="login-form" action="~/Account/Login">  <p>Username</p>  <input type="text" name="id"/>  <p>Password</p>  <input type="password" name="pass" />  <button>Login</button>  </form>  </div>  </div>  </body>  </html> |

Chạy mã trên ta được giao diện như sau:



Có vẽ khá xấu vì vẫn chưa có CSS.

Bây giờ tạo một file CSS tên là login.css trong thu mục Contents và nếu chưa có thư mục này thì bạn hoàn toàn có thể thêm vào. Thêm mã sau vào login.css:

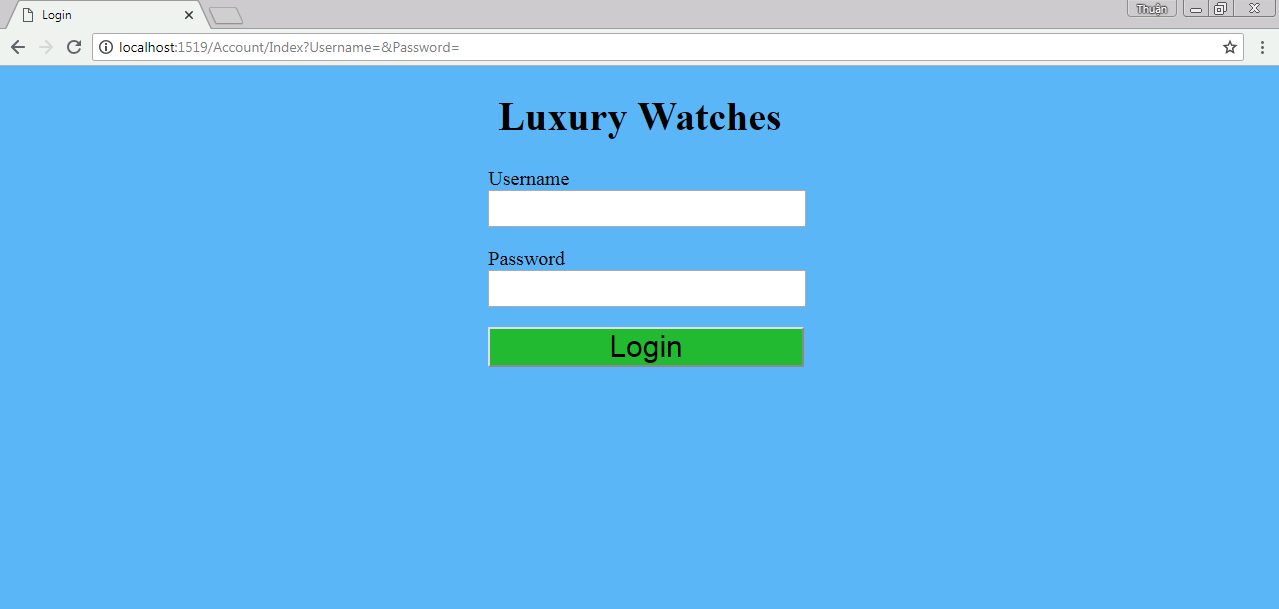
|  |
| --- |
| CSS |
| body{  background:#5ab5f7;  }  .wrap{  padding:0 35%;  }  .panel{  font-size:20px;  text-align:center;  }  #login-form{  width:80%;  padding:0 10%;  }  #login-form input{  width:100%;  font-size:20px;  padding:5px;  }  #login-form p{  font-size:20px;  margin-bottom:0px;  }  #login-form button{  font-size:30px;  width:104%;  background:#1db931;  margin-top:20px;  } |

Bây giờ bạn vẫn chưa thể sử dụng được file CSS này. Để file này chạy bạn cần truyền nó vào View. Cách truyền như sau:

Thêm mã sau vào phần head của View:

<link href="~/Contents/login.css" rel="stylesheet" type="text/css"/>

Sau đó chạy ứng dụng và kết quả:



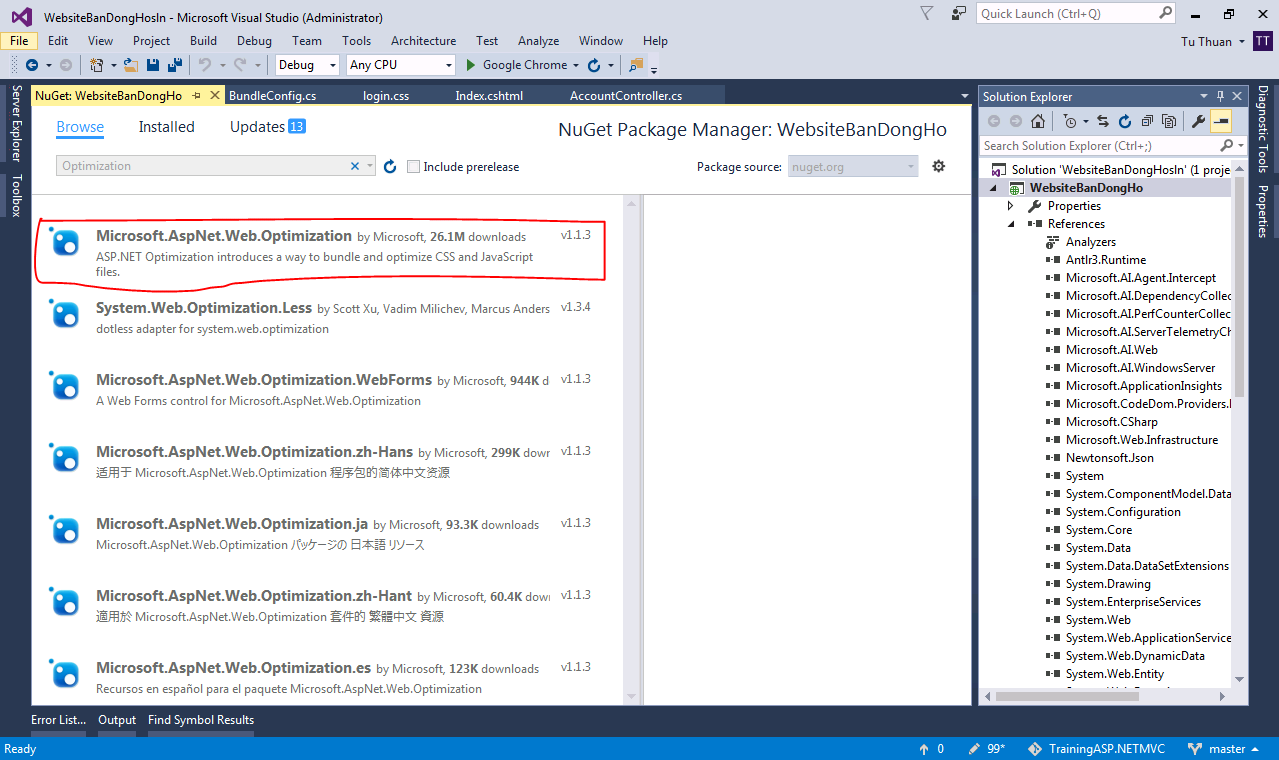
Không phải là tuyệt vời nhưng trông có vẽ khá ổn.

Có một cách khác cấu hình CSS và JavaScript bằng mã C#.

Trong thư mục App\_Start bạn tạo một class có tên là BundleConfig với nội dung như sau:

|  |
| --- |
| C# |
| using System.Web.Optimization;  namespace WebsiteBanDongHo.App\_Start  {  public class BundleConfig  {  public static void RegisterBundles(BundleCollection bundles)  {  bundles.Add(new ScriptBundle("~/bundles/css").Include(  "~/Contents/login.css"));  }  }  } |

Chú ý: Có thể thư viện Optimization chưa có trong project vì vậy bạn có thể chuột phải vào Project và chọn Manage Nuget Packages…, chuyễn sang tab Browse và ghõ Optimization vào ô tìm kiếm. Sau đó màn hình hiện ra như bên dưới.



Tải về thư viện đầu tiên ( khung màu đỏ).

Sau khi cấu hình xong bạn tìm đến file Global.asax bên trong file có một class là MvcApplication bạn tiến hành sửa nội dung class này lại như sau.

|  |
| --- |
| C# |
| namespace BanDongHo  {  public class MvcApplication : System.Web.HttpApplication  {  protected void Application\_Start()  {  AreaRegistration.RegisterAllAreas();  FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);  RouteConfig.RegisterRoutes(RouteTable.Routes);  BundleConfig.RegisterBundles(BundleTable.Bundles);  }  }  } |

Bây giờ thay lệnh

<link href="~/Contents/login.css" rel="stylesheet" type="text/css"/>

Bằng lệnh @Styles.Render("~/Content/css")

Có lẽ khá phức tạp với cách này tuy nhiên nếu đi sâu bạn có thể thấy nó rất tiện lợi.

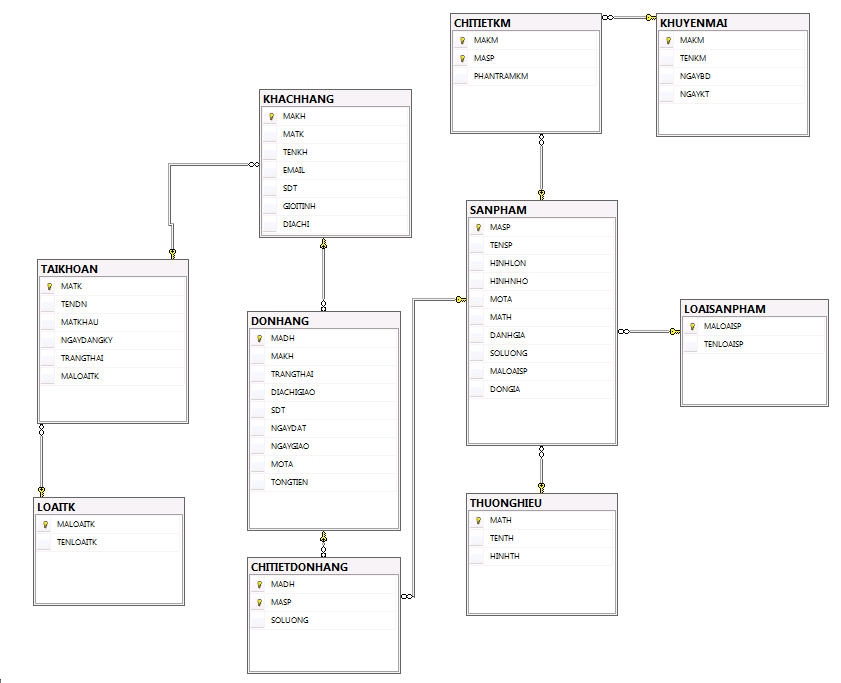
* Bây giờ bạn đã biết cách tạo ra một trang View và cấu hình CSS vào trang view đó. Trong bước sau bạn sẽ tạo kho lưu trữ và xữ lý chức năng đăng nhập.

1. Model và xữ lý login

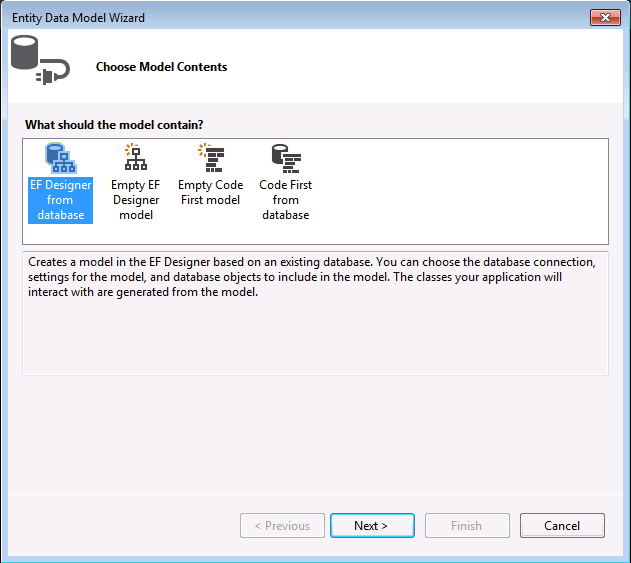
Trong hai phần trước chúng ta tạo được một Controller xữ lý yêu cầu đăng nhập, một View đăng nhập hệ thống. Bây giờ chúng ta tiến hành gắn Database vào hệ thống và truy xuất database.

Một số thành phần của tần này các bạn cần biết là ADO.NET Entity, Interface và Kế thừa interface.

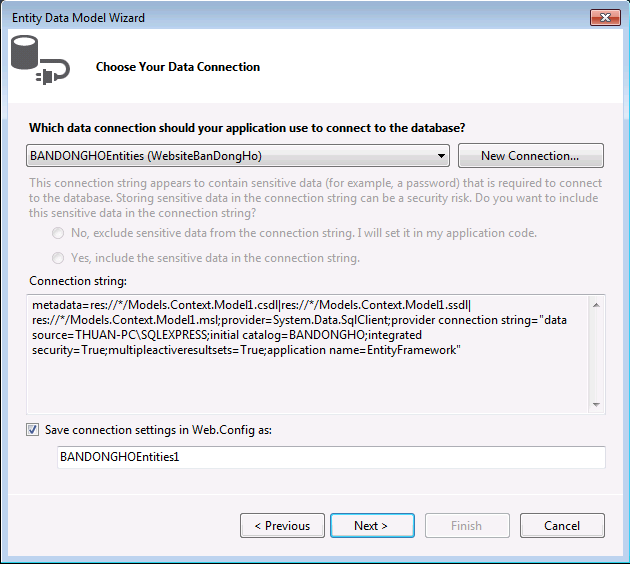
Bây giờ tôi sẽ ngay lập tức mô tả Data base như sau (Data base này được sử dụng cho đến hết ứng dụng) :



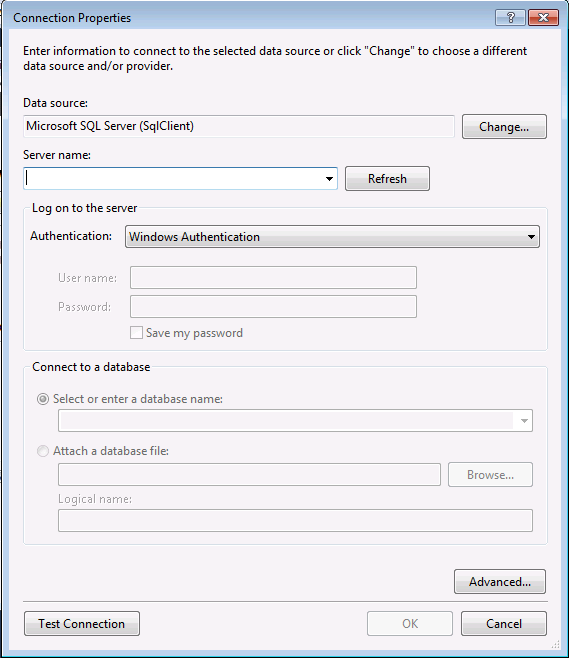
Trong bản Tài khoản tôi đã có sẵn một số dòng dữ liệu. Chúng ta sẽ thực hiện gửi một tài khoản, mật khẩu lên server. Server sẽ kiểm tra xem có trùng với bất kỳ tài khoản nào có sẳn trong CSDL hay không. Nếu có xuất thông báo “Đăng nhập thành công” ngược lại “Đăng nhập thất bại”. Trong lần này tôi sẽ mã hóa mật khẩu bằng MD5 các bạn có thể tìm hiểu thêm về kỹ thuật mã hóa này. Bây giờ tôi tiến hành kết nối CSDL vào hệ thống. Trong thư mục Models/Context bạn thêm một Item ADO.NET Entity Database Model với tên là BanDongHo. Lưu ý bạn nên chọn EF5 thay vì EF6 vì bản EF5 khá ổn định. Quy trình tạo Như sau:



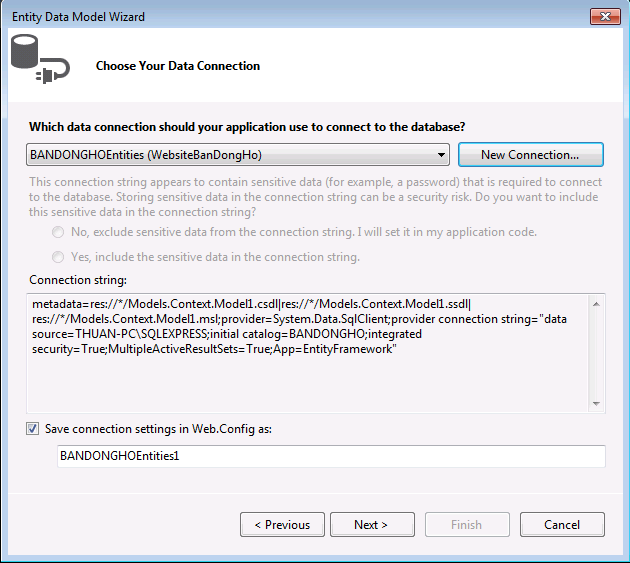
Chọn EF Design from database sau đó chọn next.



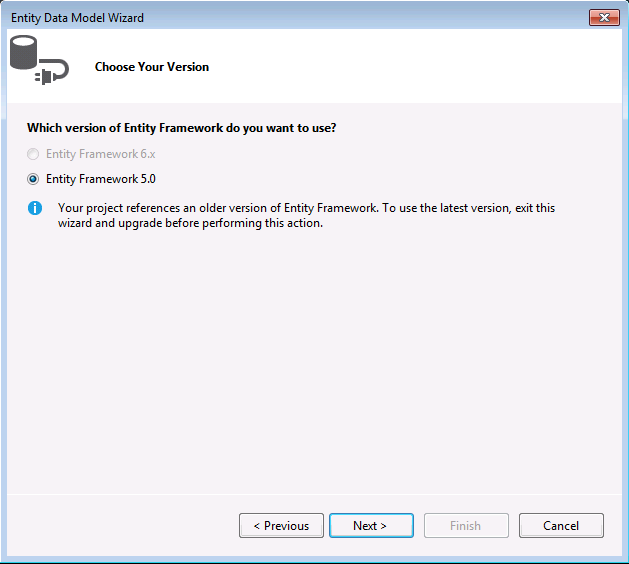
Chọn tiếp vào New Connection.



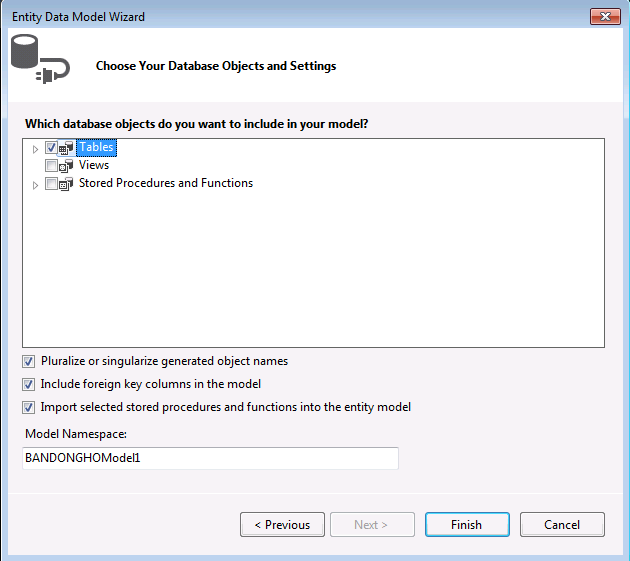
Chọn Server name và Data base nhấn OK.



Một Connection string hiện ra bạn nhấn Next.



Chọn EF5 nhấn Next.



Chọn vào Table và Nhấn Finish.

Sau khi kết nối CSDL và hệ thống chúng ta tiếp tục viết mã model như sau.

Trước hết trong thư mục service bạn tạo một interface có tên LoginService và viết mã như sau:

|  |
| --- |
| C# |
| interface LoginService  {  bool CheckAccount(LoginViewModel login);  } |

Ở đây bạn thấy có một Class là LoginViewModel thực ra nếu bạn chỉ dùng Model để đón dữ liệu từ View thì chưa hẵn đã đón được hết dữ liệu vì vậy ta tạo một Class là LoginViewModel thể hiện hế tất cả các thông tin từ View mà Model cần nhận được. Mã của LoginViewModel như sau:

|  |
| --- |
| C# |
| namespace WebsiteBanDongHo.Models.ViewModel  {  public class LoginViewModel  {  public string Id { get; set; }  public string Pass { get; set; }  }  } |

Dĩ nhiên là Class này nằm trong thư mục ViewModel đã tạo từ trước. Vì trên giao diện chỉ có hai controll cung cấp dữ liệu là id và password nên ViewModel này chỉ có hai thuộc tính là Id và Pass. Bây giờ nền tảng đã tạo xong ta chỉ việc viết mã kiểm tra việc đăng nhập như sau. Trong thư mục Models/Implement tạo Class LoginImplement kế thừa LoginService với mã:

|  |
| --- |
| C# |
| namespace WebsiteBanDongHo.Models.Implement  {  public class LoginImplement : LoginService  {  private String GetMD5(string txt)  {  String str = "";  Byte[] buffer = System.Text.Encoding.UTF8.GetBytes(txt);  System.Security.Cryptography.MD5CryptoServiceProvider md5 =  new System.Security.Cryptography.MD5CryptoServiceProvider();  buffer = md5.ComputeHash(buffer);  foreach (Byte b in buffer)  {  str += b.ToString("X2");  }  return str;  }  public bool CheckAccount(LoginViewModel login)  {  string matKhauMD5 = GetMD5(login.Pass);  // tìm trong CSDL  using (var db = new BANDONGHOEntities())  {  TAIKHOAN taikhoan = (from tk in db.TAIKHOANs  where tk.TENDN.Equals(login.Id) &&  tk.MATKHAU == matKhauMD5  select tk).FirstOrDefault();  if(taikhoan!=null)  {  return true;  }  }  return false;  }  }  } |

Vậy là bạn đã hoàn thành chức năng Đăng Nhập tại Models. Lý do tại sau tôi phải viết Interface rồi Class kế thừa, bởi vì bạn có thể phát triển dự án theo hướng DI. Bây giờ việc còn lại là làm sao trong controller sử dụng nó.

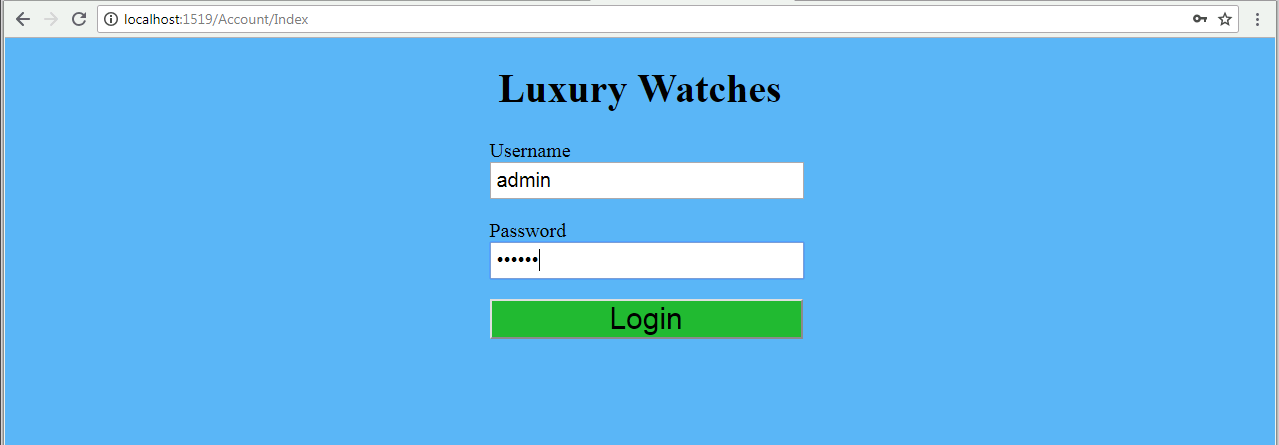
Bạn quay lại với AccountController và chỉnh sửa mã lại như sau:

|  |
| --- |
| C# |
| namespace WebsiteBanDongHo.Controllers  {  public class AccountController : Controller  {  private LoginService loginService;  public ActionResult Index()  {  return View();  }  [HttpPost]  public string Login(LoginViewModel account)  {  loginService = new LoginImplement();  if (loginService.CheckAccount(account))  {  return "Đăng nhập thành công";  }  else  {  return "Đăng nhập thất bại";  }  }  }  } |

Trong phương thức POST – Login tôi sử dụng đối tượng LoginService để kiểm tra một tài khoản lấy từ View. Ở đây bạn sẽ thấy lạ vì tôi không có hai tham số Id, Pass như lúc trước mà hiện giờ nó được thay đổi bằng một đối tượng ViewModel. Và tôi đã thay đổi từ phương thức GET sang phương thức POST. Việc dùng phương thức POST giúp dữ liệu được giấu đi không hiện lên thanh URL điều này đảm bảo bảo mật cho người dùng. Giờ ta lại quay sang sửa trang View cho phù hợp với hai vấn đề. Phương thức Post và đối tượng ViewModel.

|  |
| --- |
| HTML |
| @using System.Web.Optimization  <!DOCTYPE html>  <html>  <head>  <meta name="viewport" content="width=device-width" />  @Styles.Render("~/Content/css")  <title>Login</title>  </head>  <body>  <div class="wrap">  <div class="panel"><h1>Luxury Watches</h1></div>  <div class="form">  <form id="login-form" action="~/Account/Login" method="post">  <p>Username</p>  <input type="text" name="Id"/>  <p>Password</p>  <input type="password" name="Pass" />  <button>Login</button>  </form>  </div>  </div>  </body>  </html> |

Tôi thay đổi hay điều ở trang này. Trong thẻ form tôi dùng method= “post”. Thứ hai tôi điều chỉnh hai Input trong form với hai tên tương ứng với hai thuộc tính trong ViewModel. Tôi có một tài khoản là “admin” với mật khẩu là “123456” giờ tôi sẽ khởi chạy form này và đón chờ kết quả.





Vậy là kết quả mỹ mãn chúng ta đã thực hiện xong chức năng Đăng nhập. Trong phần sau chúng ta sẽ tiến hành load danh sách sản phẩm lên trang Home.