Homework #4: New Project (more houses)

Task 1) For the sake of making things as easy as possible, while giving you more practice building MVC Web applications, complete the following steps:

1. Create a NEW MVC project in Visual Studio. (Don’t just add it to your existing one. Make this a new project and solution entitled PropertyLister or something of that sort.)
2. Like you did in your previous project, set this one up to accept third-party logins (i.e. Facebook, Google, Linked-In).
3. Again, as in your previous project, modify the Login and Register pages to accept a login with EITHER a username or email.
4. Build a new PropertyListingViewModel that contains all of the elements you might need to list if you were a real estate agent. (Visit Zillow.com for help deciding how to build a model that works for you.)
5. Add the following **using** statement to the top of the file containing your view model:

using System.Data.Entity;

1. Add a new DBContext-based class below PropertyListingViewModel class (such as):

public class PropertyListingsDBContext : DbContext  
 {  
 public DbSet<PropertyListingViewModel> Listings { get; set; }  
 }

1. Add a **Connection** **String** to the **Web.config** file so that your application can create a connection to the database to store your property information.
   1. Open the **Web.config** file at the bottom of the **Solution** **Explorer** and locate the **<connectionStrings>** tag.
   2. Add the following string to this section below any other connection strings that are currently present:

<add name="PropertyListingsDBContext" connectionString="Data   
 Source=(LocalDb)\v11.0;AttachDbFilename=|DataDirectory|\Properties.mdf;  
 Integrated Security=True" providerName="System.Data.SqlClient"

1. Now we need to set up Code First Migrations for this project.
   1. Select Tools > NuGet Package Manager > Package Manager Console
   2. At the PM> prompt in the newly opened console window, type the following:

Enable-Migrations –ContextTypeName  
 PropertyLister.Models.PropertyListingsDBContext   
 -EnableAutomaticMigrations

NOTE: This is the path to the DB model in my project. “PropertyLister” is the name of my solution. Make sure your path includes the name of your solution/project.

* 1. When you receive a message indicating that the migrations are enabled, enter Update-Database at the PM> prompt.
  2. You project should now be ready for database access.

1. Create a new set of views and controllers by SCAFFOLDING your view model (just as you did in the previous project).
2. Now begin enhancing the functionality of your new Web application by making sure you can create, edit and delete property listings, and by creating custom behavior. Use your previous project as a guide. Consider allowing users of your site to be able to view a slideshow of property images, maybe even floor plans if they can be made available.