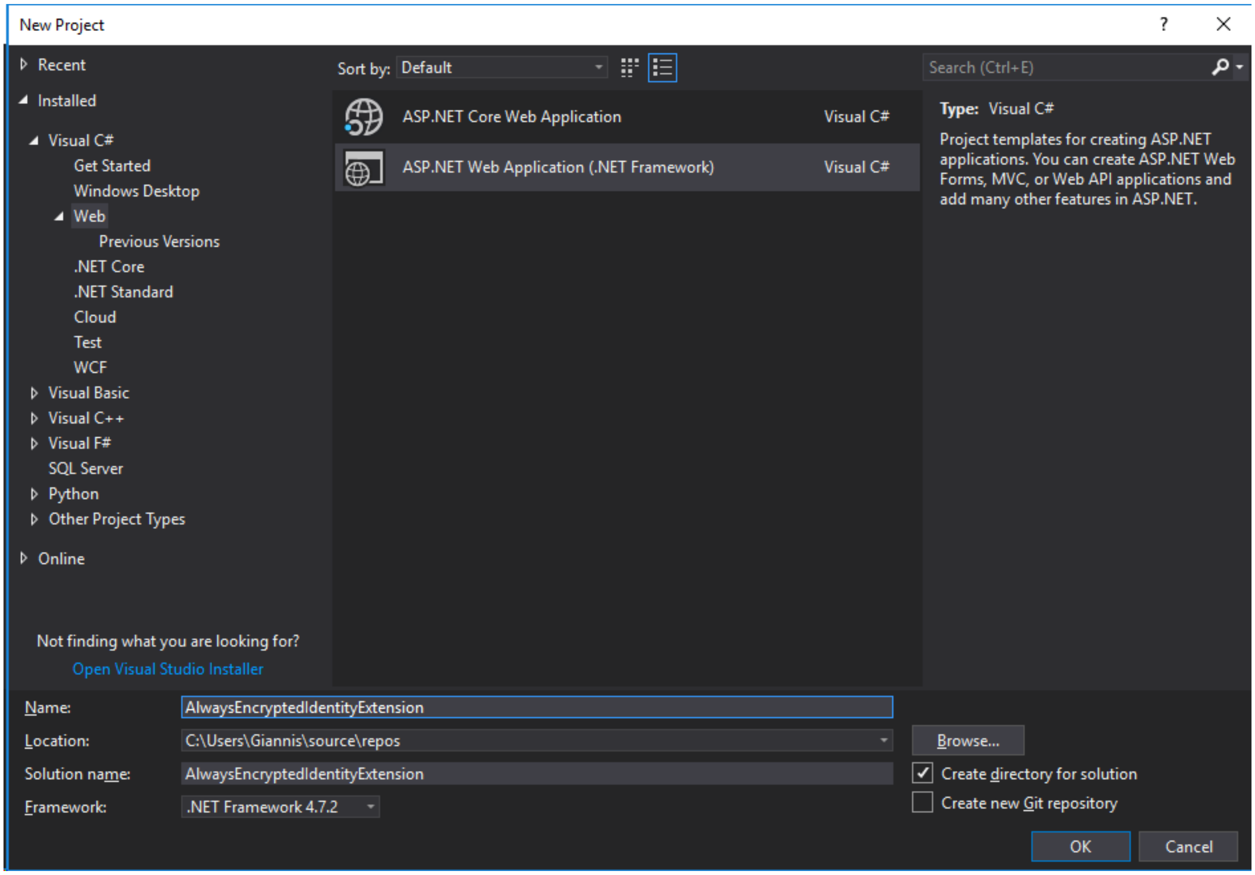
How To Use

From Visual Studio browse “File” > “New” > “Project…”
Give your project a desired name. In the example below I use “AlwaysEncryptedIdentityExtension”.



Type: Visual C#

Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

Name: AlwaysEncryptedIdentityExtension

Location: C:\Users\Giannis\source\repos

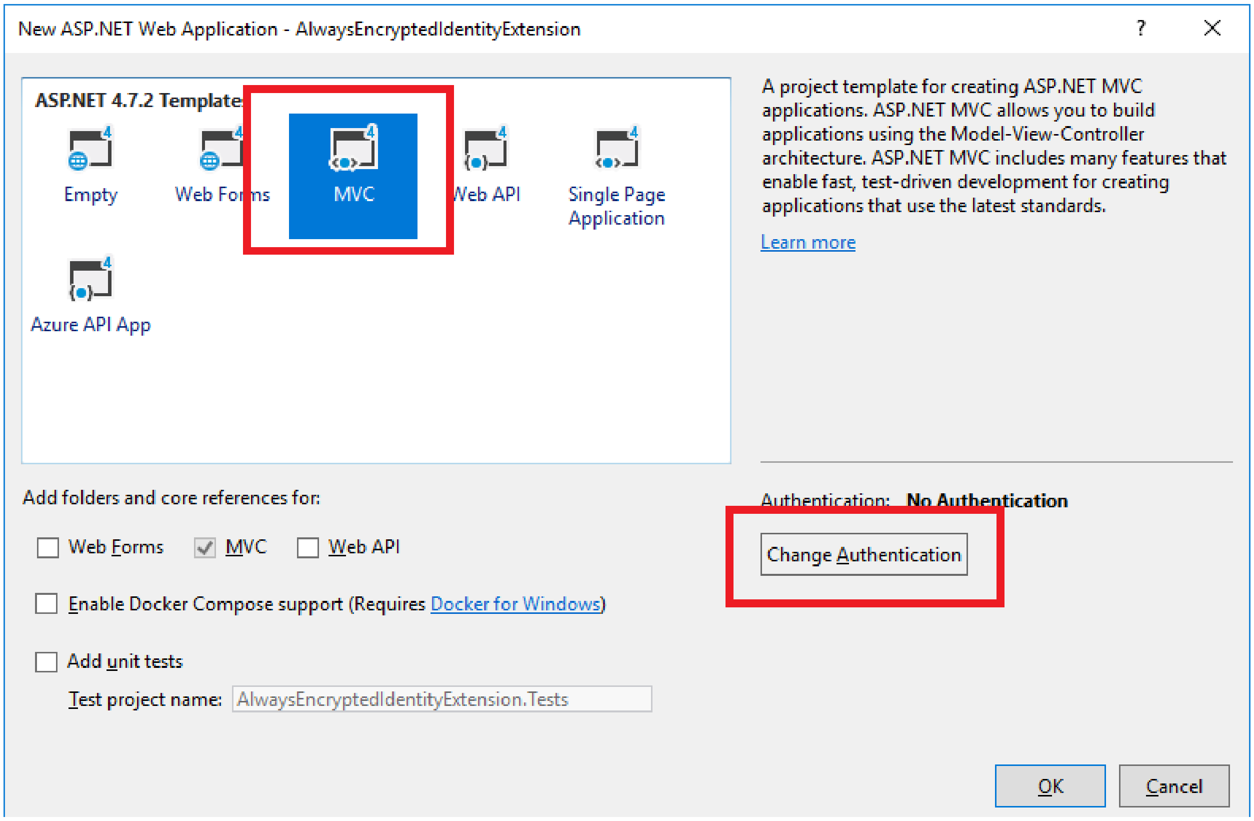
Solution name: AlwaysEncryptedIdentityExtension

Framework: .NET Framework 4.7.2

☑ Create directory for solution

☐ Create new Git repository

Select “MVC” template and “Change Authentication” to “Individual User Accounts”.



A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Authentication: **No Authentication**

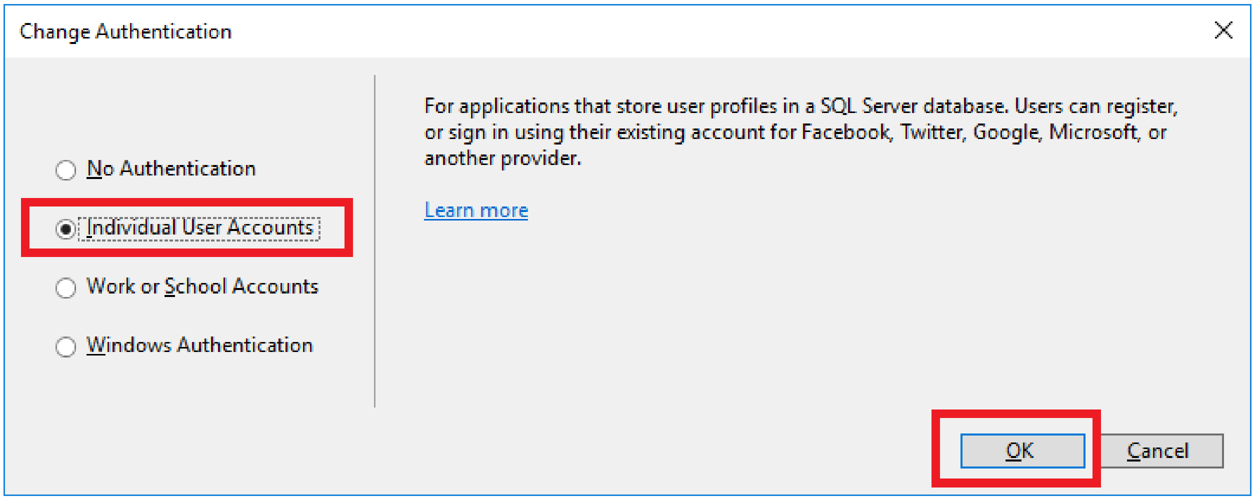
Add folders and core references for:

□ Web Forms ☑ MVC □ Web API

□ Enable Docker Compose support (Requires Docker for Windows)

□ Add unit tests

Test project name: AlwaysEncryptedIdentityExtension.Tests



○ No Authentication

● Individual User Accounts

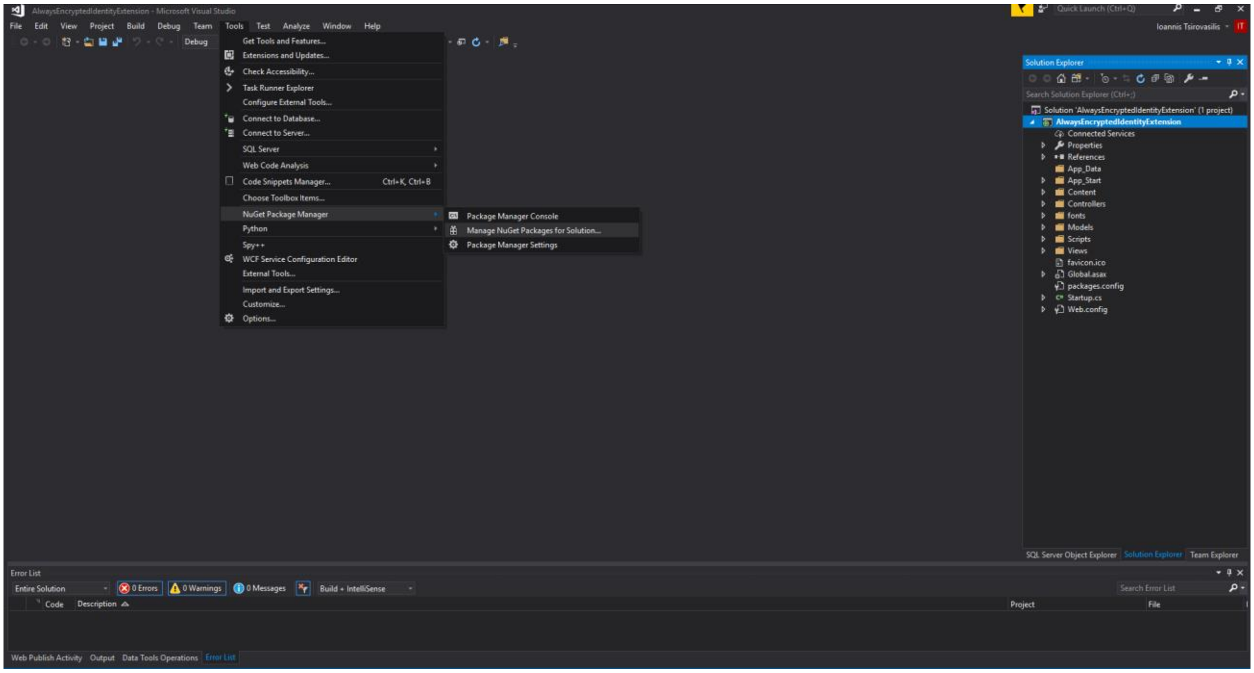
○ Work or School Accounts

○ Windows Authentication

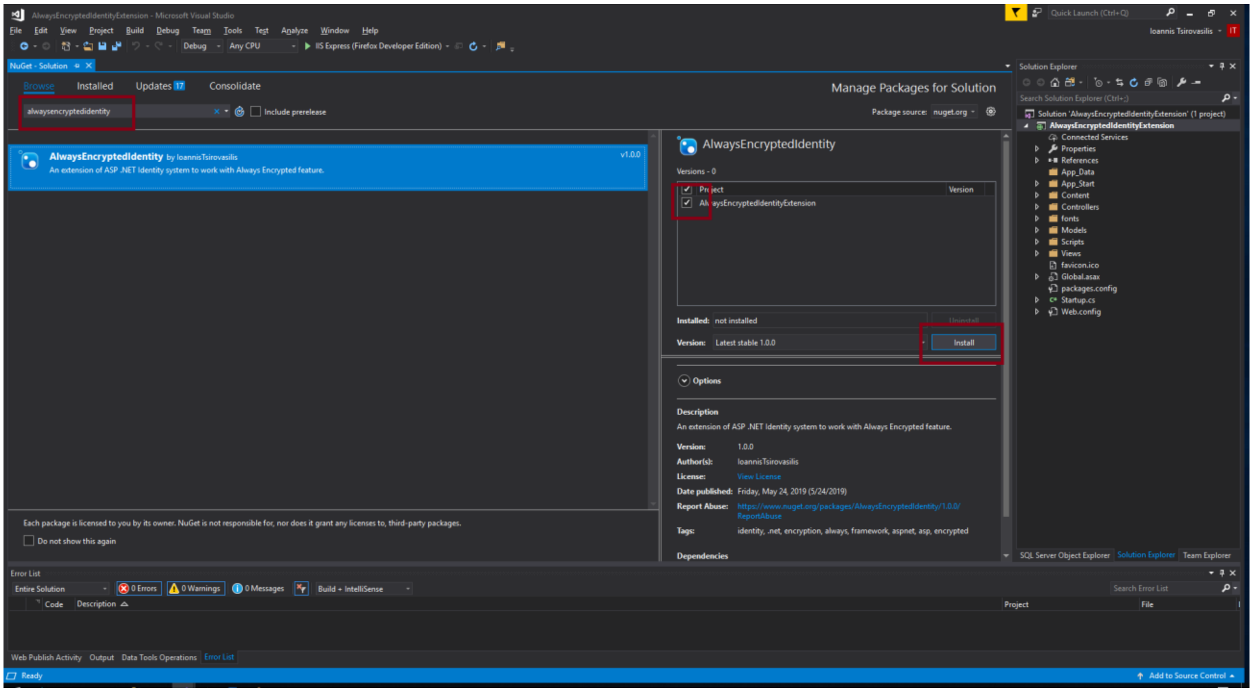
For applications that store user profiles in a SQL Server database. Users can register, or sign in using their existing account for Facebook, Twitter, Google, Microsoft, or another provider.

Wait for the project to load…

Download the nuget by browsing “Tools” > “NuGet Package Manager” > “Manage NuGet Packages for Solution…”



From "Browse" tab type "AlwaysEncryptedIdentity" and install the nuget.


AlwaysEncryptedIdentityExtension - Microsoft Visual Studio
NuGet - Solution
Browse
Installed
Updates 17
Consolidate
Manage Packages for Solution
alwaysencryptedidentity
Include prerelease
Package source: nuget.org
AlwaysEncryptedIdentity by IoannisTsirovasilis
An extension of ASP .NET Identity system to work with Always Encrypted feature.
v1.0.0
AlwaysEncryptedIdentity
Versions - 0
Project
Version
AlwaysEncryptedIdentityExtension
Installed: not installed
Uninstall
Version: Latest stable 1.0.0
Install
Options
Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.
Do not show this again
Solution Explorer
Solution 'AlwaysEncryptedIdentityExtension' (1 project)
Error List
Ready
Add to Source Control

Description
An extension of ASP .NET Identity system to work with Always Encrypted feature.

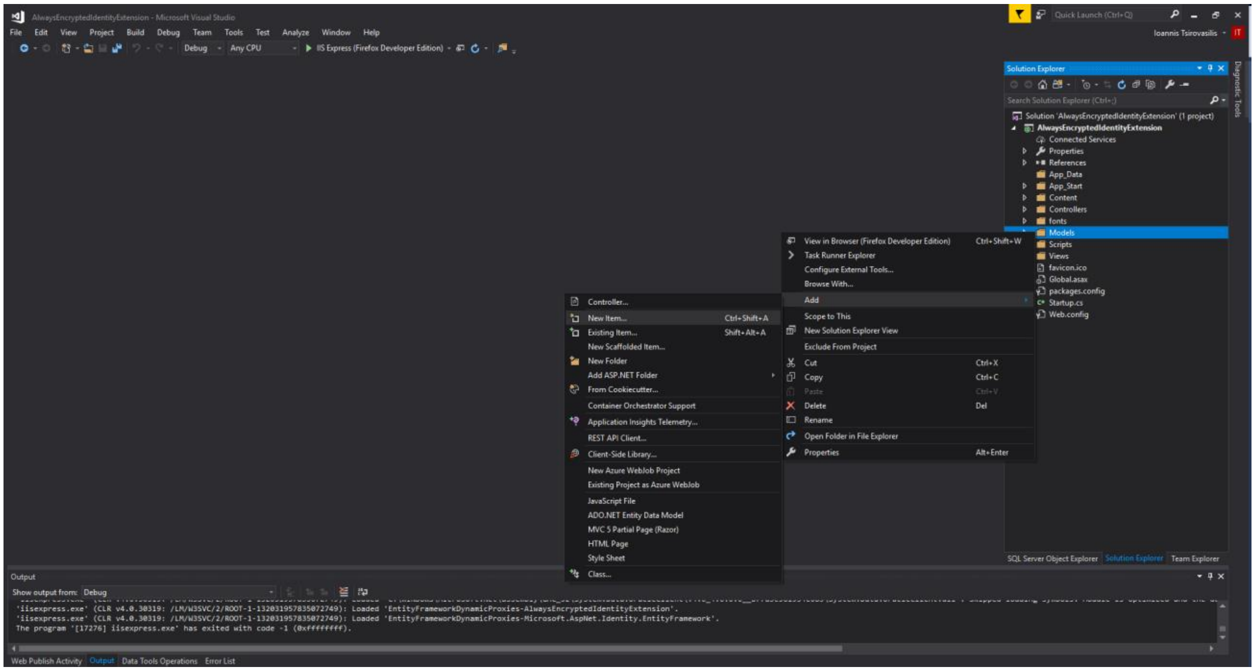
Version: 1.0.0
Author(s): IoannisTsirovasilis
License: View License
Date published: Friday, May 24, 2019 (5/24/2019)
Report Abuse: https://www.nuget.org/packages/AlwaysEncryptedIdentity/1.0.0/ReportAbuse
Tags: identity, .net, encryption, always, framework, aspnet, asp, encrypted
Dependencies

After the installation completes there are a couple of necessary steps to be done in order for the nuget to work.

First of all, create a Model using Entity Framework 6 that contains the tables

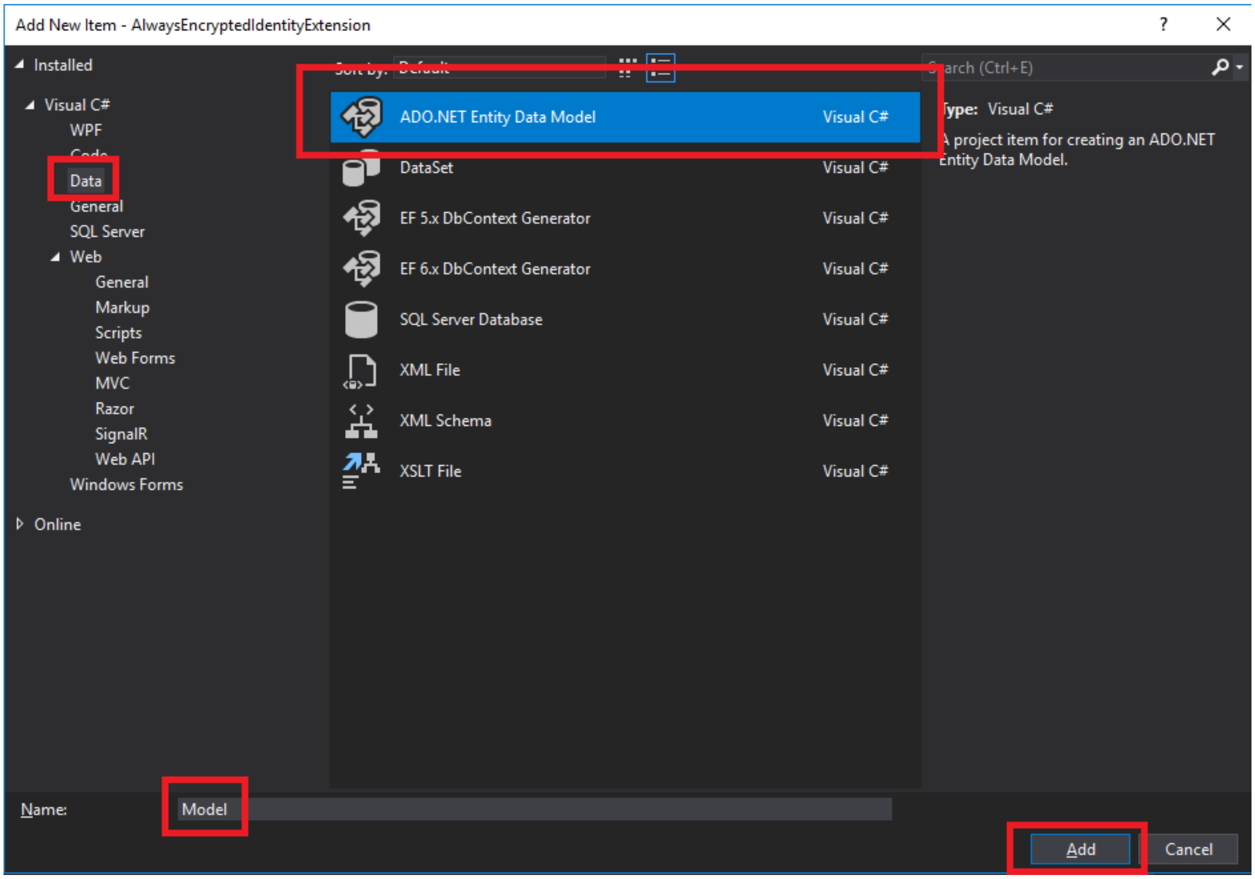
dbo.AspNetRoles
dbo.AspNetUserRoles
dbo.AspNetUsers

Do so by right-clicking the “Models” folder from “Solution Explorer”, browse “Add” > “New Item…”



```
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132031957835072749): Loaded 'EntityFrameworkDynamicProxies-AlwaysEncryptedIdentityExtension'.
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/2/ROOT-1-132031957835072749): Loaded 'EntityFrameworkDynamicProxies-Microsoft.AspNet.Identity.EntityFramework'.
The program '[17276] iisexpress.exe' has exited with code -1 (0xffffffff).
```

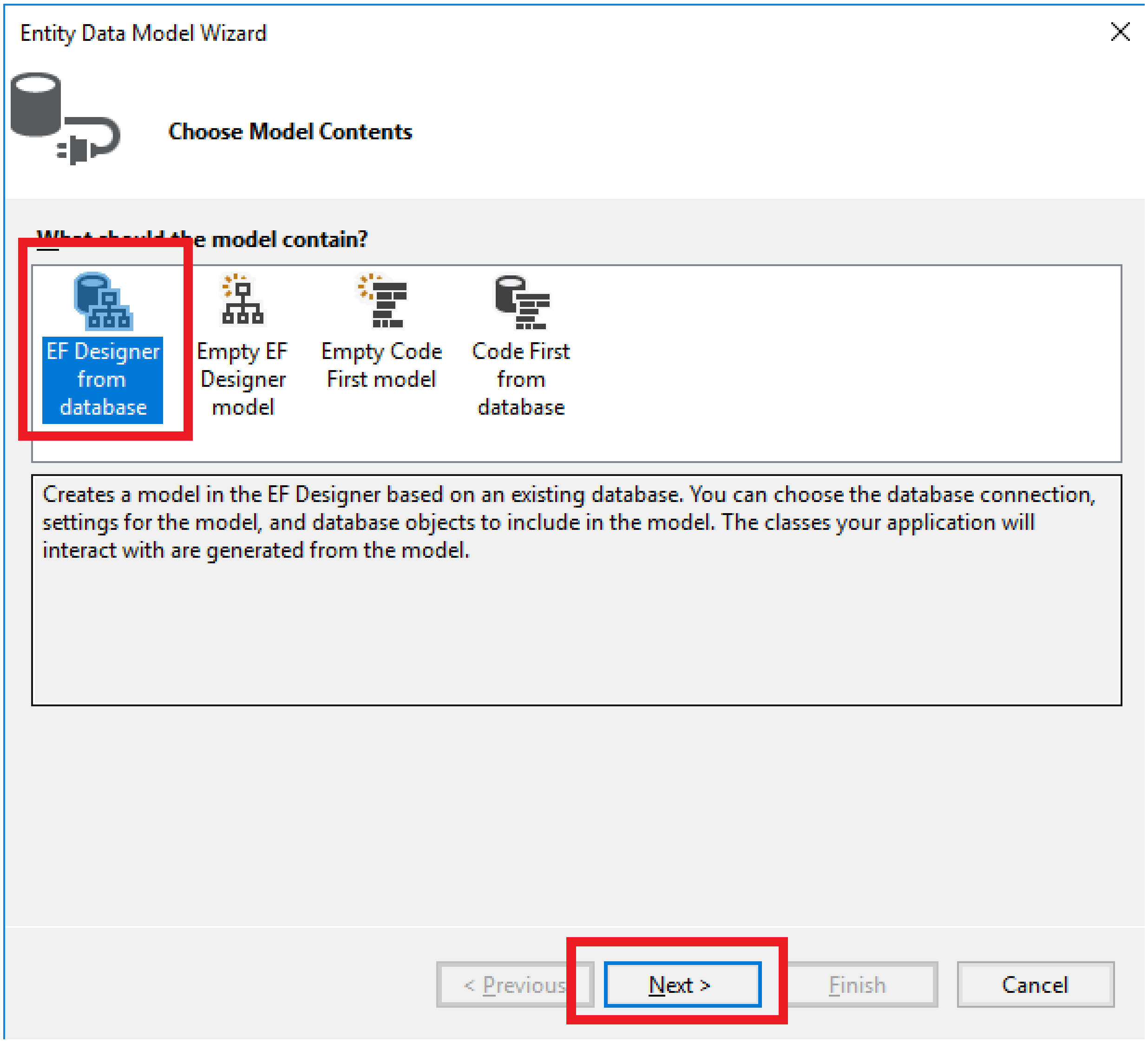
Select “Data” tab from left panel, click “ADO.NET Entity Data Model”, give your model a nice name and click “Add”.



Type: Visual C#

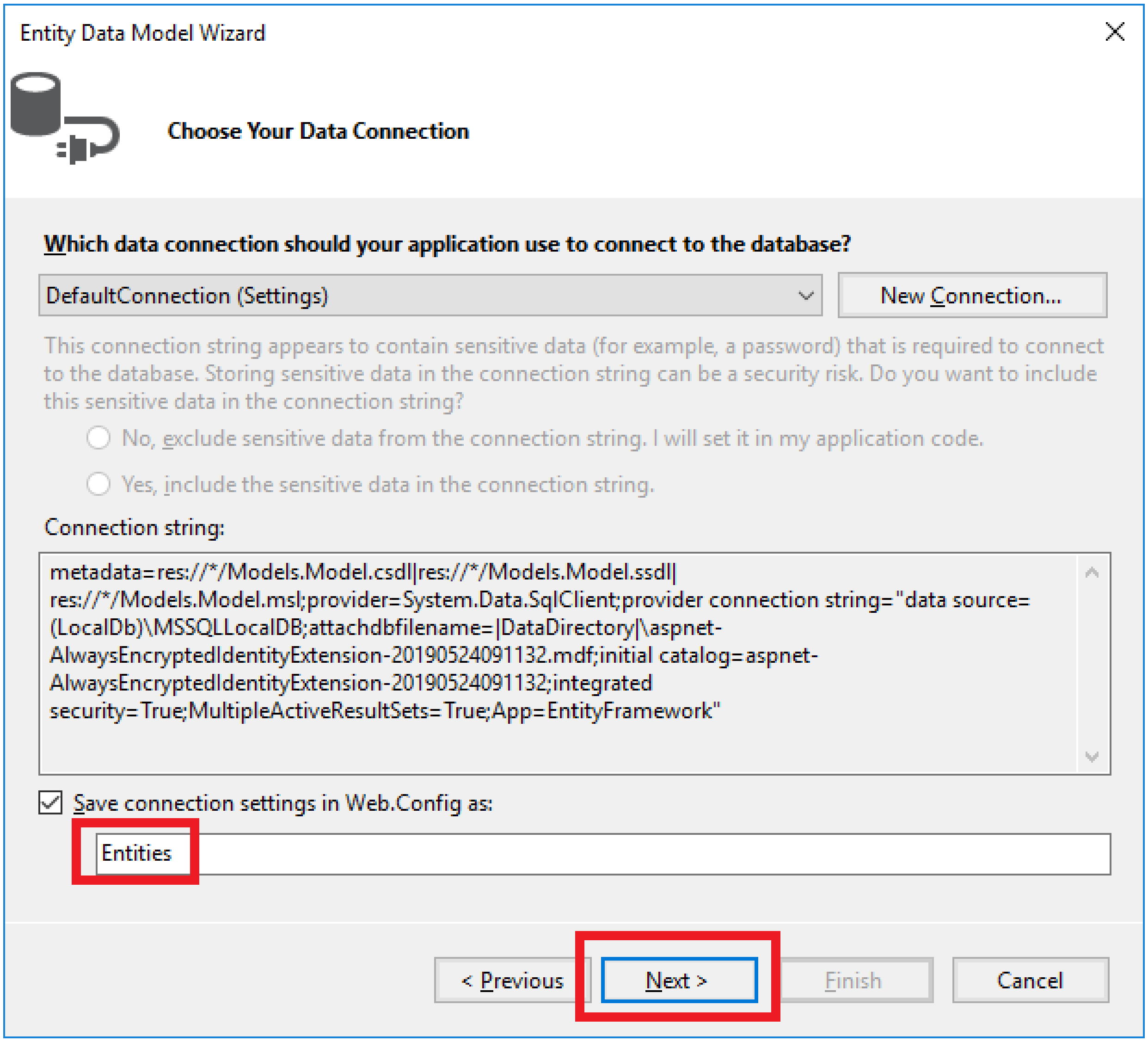
A project item for creating an ADO.NET Entity Data Model.

Name: Model



Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.

Give a name for the generated database context and click "Next". In this example the name "Entities" is used. (This is important for using the nuget)



Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

DefaultConnection (Settings)

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

○ No, exclude sensitive data from the connection string. I will set it in my application code.

○ Yes, include the sensitive data in the connection string.

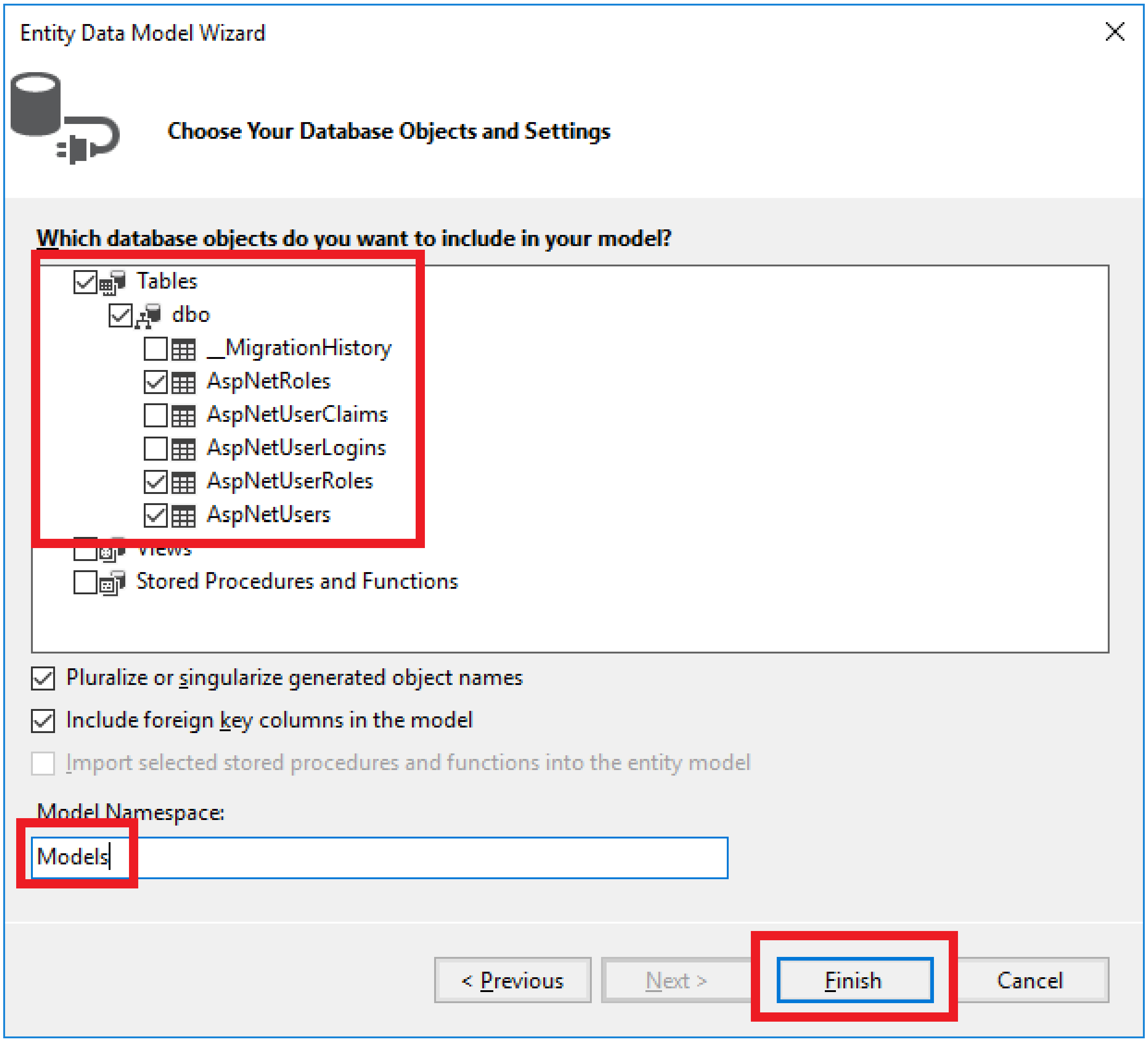
Connection string:

```
metadata=res://*/Models.Model.csdl|res://*/Models.Model.ssdl|
res://*/Models.Model.msl;provider=System.Data.SqlClient;provider connection string="data source=
(LocalDb)\MSSQLLocalDB;attachdbfilename=|DataDirectory|\aspnet-
AlwaysEncryptedIdentityExtension-20190524091132.mdf;initial catalog=aspnet-
AlwaysEncryptedIdentityExtension-20190524091132;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"
```

☑ Save connection settings in Web.Config as:

Entities

Select the tables referenced earlier to include them in the model, give it a name and click “Finish”.



Entity Data Model Wizard

Choose Your Database Objects and Settings

Which database objects do you want to include in your model?

- ☑ Tables
  - ☑ dbo
    - ☐ __MigrationHistory
    - ☑ AspNetRoles
    - ☐ AspNetUserClaims
    - ☐ AspNetUserLogins
    - ☑ AspNetUserRoles
    - ☑ AspNetUsers
- ☐ Views
- ☐ Stored Procedures and Functions

☑ Pluralize or singularize generated object names

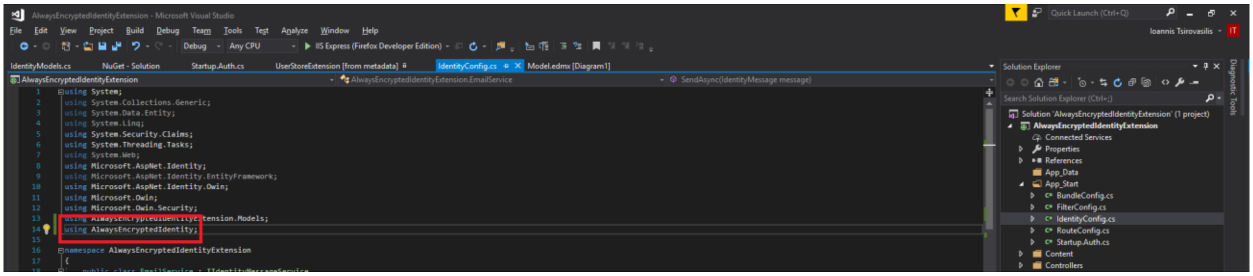
☑ Include foreign key columns in the model

☐ Import selected stored procedures and functions into the entity model

Model Namespace: Models

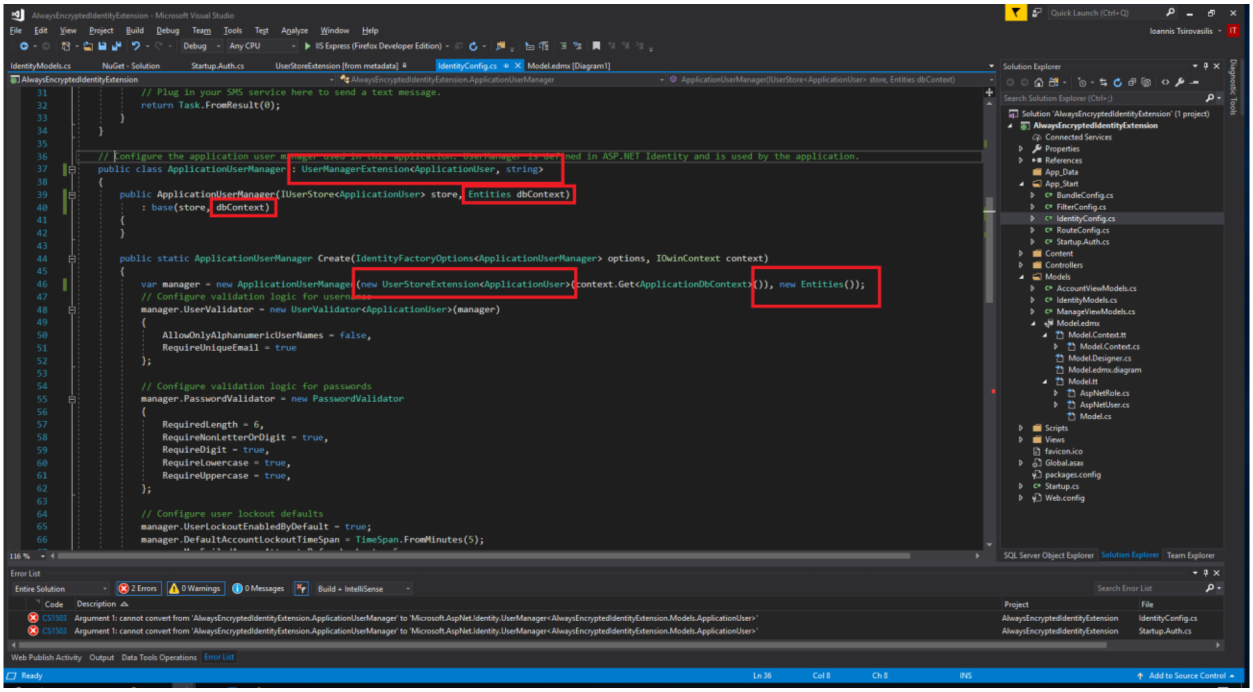
From "Solution Explorer", browse "App_Start" folder and open "IdentityConfig.cs".

Add "using AlwaysEncryptedIdentity;" at the top of the file



```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Security.Claims;
using System.Threading.Tasks;
using System.Web;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using Microsoft.AspNet.Identity.Owin;
using Microsoft.Owin;
using Microsoft.Owin.Security;
using AlwaysEncryptedIdentityExtension.Models;
using AlwaysEncryptedIdentity;

namespace AlwaysEncryptedIdentityExtension
{
    public class EmailService : IIdentityMessageService
```

Move further down and change the following lines as depicted below:

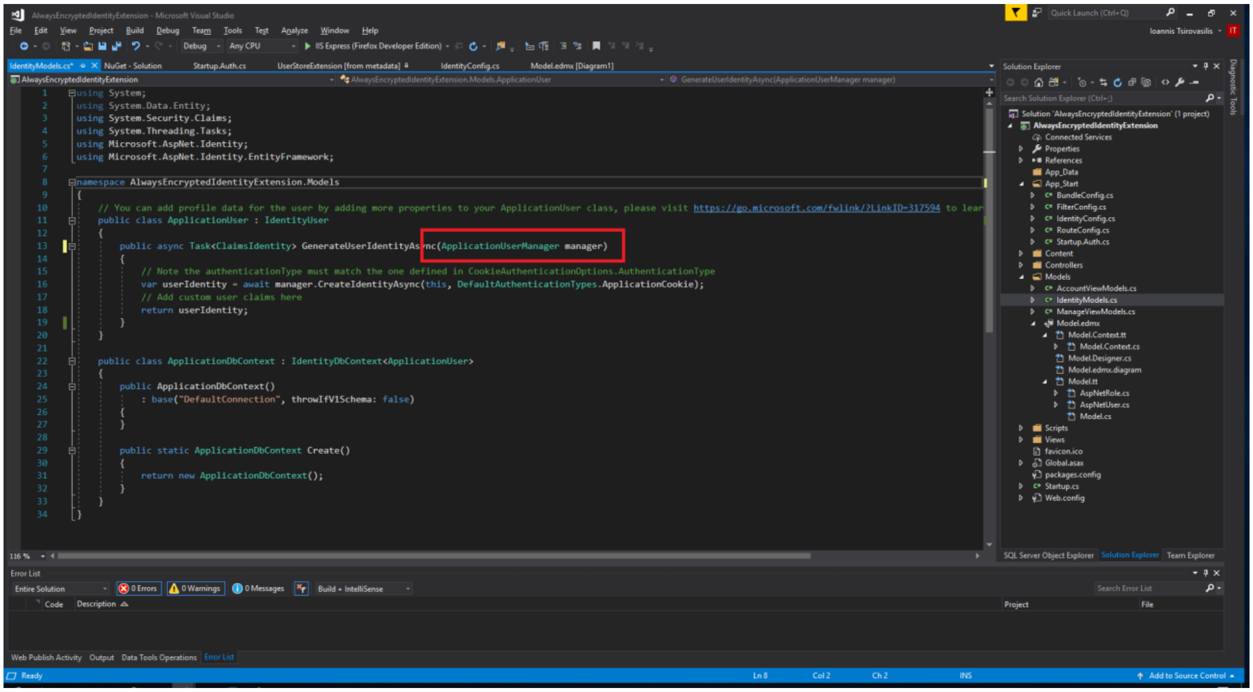


```
            // Plug in your SMS service here to send a text message.
            return Task.FromResult(0);
        }
    }

    // Configure the application user manager used in this application. UserManager is defined in ASP.NET Identity and is used by the application.
    public class ApplicationUserManager : UserManagerExtension<ApplicationUser, string>
    {
        public ApplicationUserManager(IUserStore<ApplicationUser> store, Entities dbContext)
            : base(store, dbContext)
        {
        }

        public static ApplicationUserManager Create(IdentityFactoryOptions<ApplicationUserManager> options, IOwinContext context)
        {
            var manager = new ApplicationUserManager(new UserStoreExtension<ApplicationUser>(context.Get<ApplicationDbContext>()), new Entities());
            // Configure validation logic for usernames
            manager.UserValidator = new UserValidator<ApplicationUser>(manager)
            {
                AllowOnlyAlphanumericUserNames = false,
                RequireUniqueEmail = true
            };

            // Configure validation logic for passwords
            manager.PasswordValidator = new PasswordValidator
            {
                RequiredLength = 6,
                RequireNonLetterOrDigit = true,
                RequireDigit = true,
                RequireLowercase = true,
                RequireUppercase = true,
            };

            // Configure user lockout defaults
            manager.UserLockoutEnabledByDefault = true;
            manager.DefaultAccountLockoutTimeSpan = TimeSpan.FromMinutes(5);
```

Error List: 2 Errors, 0 Warnings, 0 Messages

```
CS1503  Argument 1: cannot convert from 'AlwaysEncryptedIdentityExtension.ApplicationUserManager' to 'Microsoft.AspNet.Identity.UserManager<AlwaysEncryptedIdentityExtension.Models.ApplicationUser>'  AlwaysEncryptedIdentityExtension  IdentityConfig.cs
CS1503  Argument 1: cannot convert from 'AlwaysEncryptedIdentityExtension.ApplicationUserManager' to 'Microsoft.AspNet.Identity.UserManager<AlwaysEncryptedIdentityExtension.Models.ApplicationUser>'  AlwaysEncryptedIdentityExtension  Startup.Auth.cs
```

Finally browse “Models” folder from “Solution Explorer”, open “IdentityModels.cs” and change the line shown below:



```
using System;
using System.Data.Entity;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;

namespace AlwaysEncryptedIdentityExtension.Models
{
    // You can add profile data for the user by adding more properties to your ApplicationUser class, please visit https://go.microsoft.com/fwlink/?LinkID=317594 to lear
    public class ApplicationUser : IdentityUser
    {
        public async Task<ClaimsIdentity> GenerateUserIdentityAsync(ApplicationUserManager manager)
        {
            // Note the authenticationType must match the one defined in CookieAuthenticationOptions.AuthenticationType
            var userIdentity = await manager.CreateIdentityAsync(this, DefaultAuthenticationTypes.ApplicationCookie);
            // Add custom user claims here
            return userIdentity;
        }
    }

    public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
    {
        public ApplicationDbContext()
            : base("DefaultConnection", throwIfV1Schema: false)
        {
        }

        public static ApplicationDbContext Create()
        {
            return new ApplicationDbContext();
        }
    }
}
```

Error List: 0 Errors, 0 Warnings, 0 Messages

The application is ready to support Always Encrypted feature.