# Adding External Logins to an Existing MVC Project

1. Create a new MVC Web project for C#
   1. File > New Project
   2. Templates > C# > Web > ASP.NET Web Application
   3. Give the project a name, click OK
   4. Choose MVC, verify access as Individual User Accounts, no cloud hosting, click OK
2. You currently have a semi-functioning web application. It has a basic data source with user login, but we want to expand that. One of the first things we’d like to do is set up external login authorization using Facebook, Google, and LinkedIn:
   1. In the Solution Explorer, click the project name, then press F4 to view the properties in the window below the Solution Explorer
   2. Change SSL Enabled to True
   3. Copy the SSL URL
   4. In Solution Explorer, right click the project name and select Properties.
   5. Select the Web tab, and then paste the SSL URL into the Project URL box. Save your project. Remember where to find this URL, or write it down. You will need it to configure external authentication apps.
   6. In the Solution Explorer, open the Controllers folder and select the HomeController.cs file
   7. Add the [RequireHttps] attribute to the Home controller (by placing it above the class definition) to require all requests must use HTTPS
3. Set up Google Authentication
   1. Navigate to the Google Developers Console (you may need to answer a few questions to register as a Google developer)  
      https://console.developers.google.com/
   2. Click the Create Project button and enter a project name and ID (you can use the default values). In a few seconds the new project will be created and your browser will display the new projects page.
   3. In the left tab, click APIs & auth, and then click Credentials.
   4. Click the Create New Client ID under OAuth.
   5. In the Create Client ID dialog, keep the default Web application for the application type.
   6. Set the Authorized JavaScript origins to the SSL URL you used above:  
      https://localhost:44300/
   7. Set the Authorized redirect URI to:   
      https://localhost:44300/signin-google
   8. Under APIs & Auth, click Consent Screen
   9. Scroll down and enter a valid email and product name (these are required), then click Save.
   10. Back in Visual Studio, open App\_Start\Startup.Auth.cs.
   11. Locate the ConfigureAuth method and uncomment the lines pertaining to Google Authentication at the bottom of the method.
   12. Copy and paste the ClientId and ClientSecret into the UseGoogleAuthentication method.
   13. Save your project. You may test it at this point or continue adding additional external logins.
4. Set up Facebook Authentication
   1. In your browser, navigate to https://developers.facebook.com/apps and log in by entering your Facebook credentials.
   2. If you aren’t already registered as a Facebook developer, click Register as a Developer and follow the directions to register.
   3. On the Apps tab, click Create New App
   4. Enter an App Name and Category, then click Create App.
      1. This must be unique across Facebook. The App Namespace is the part of the URL that your App will use to access the Facebook application for authentication (for example, https://apps.facebook.com/{App Namespace}). If you don't specify an App Namespace, the App ID will be used for the URL. The App ID is a long system-generated number that you will see in the next step.
   5. Once the app has been created, select Settings for the left menu bar.
   6. On the Basic settings section of the page select Add Platform to specify that you are adding a website application.
   7. Make a note of your App ID and your App Secret so that you can add both into your MVC application later.
   8. Now add your Site URL (https://localhost:44300/) for your MVC application, then add a Contact Email and select Save Changes.
   9. Click the Advanced tab. Make sure Client OAuth Login is set to Yes, then add the URL of your web application (https://localhost:44300) to the box under Valid OAuth redirect URIs. Scroll down and click Save Changes.
   10. Back in Visual Studio, open App\_Start\Startup.Auth.cs.
   11. Locate the ConfigureAuth method and uncomment the lines pertaining to Facebook Authentication near the bottom of the method.
   12. Copy and paste the appId and appSecret into the UseFacebookAuthentication method.
   13. Save your project.
5. Set up LinkedIn authentication (just a little bit more work)
   1. First, we need to install the Owin.Security.Providers package via NuGet – this package provides the framework necessary for additional external authorizations such as LinkedIn and Yahoo
      1. Select Tools > NuGet Package Manager > Manage NuGet Packages for Solution
      2. In the left column, select Online > All
      3. In the search bar (top right column), type Owin.Security.Providers
      4. The package should appear in the center column - click Install
      5. While you are in the package manager, update the NuGet packages for your project by selecting Updates > All from the left column, then click the Update All button and follow the prompts – click Close when finished
   2. Next, register the LinkedIn provider
      1. Like Facebook and Google, you need to create a project on the LinkedIn Developer Network
      2. Go to http://developer.linkedin.com/ and login using your existing LinkedIn ID (if you have one), if not, register as a developer
      3. When logged in, click your name in the upper right corner and choose API Keys
      4. Complete the new application registration form, using your MVC project’s SSL URL (which will be https://localhost:44300/ unless you've created other SSL projects) for the website URL and https://localhost:44300/signin-linkedin for the OAuth 2.0 Redirect URL
      5. When finished, create the project and wait for the verification page
      6. Once created, a page will display showing the API Key and Secret Key for the application
      7. Now go back to your MVC project and open the App\_Start\Startup.Auth.cs file and add the following namespace:
         1. using Owin.Security.Providers.LinkedIn;
      8. Register the LinkedIn provider in the ConfigureAuth method (you can add this below the Google and Facebook authentications)
         1. app.UseLinkedInAuthentication("YOUR API KEY", "YOUR SECRET KEY");
         2. Make sure your use the API and Secret Keys for your application
   3. Now test your project by running it and navigating to the Log in page – you should see three additional service buttons on the right, one for each of the services you registered. Try logging in with all three services (make sure you log out of one before logging in with another).