MVC

Adding 3rd Party Logins

One of the first things we'd like to do is set up external login authorization using social networks such as Facebook, Twitter, Google, LinkedIn, and even GitHub. The process for each of these is similar, but before we can begin we must modify the MVC project to accept these login procedures.

Set up the MVC Project to accept 3rd Party Logins

1) In the Solution Explorer, click the project name (Blog), then press F4 (Fn+F4 on some laptops) to bring up the properties window
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 social media authentication apps.
    a. Once you publish the project to Azure, you'll need to make note of your project's Azure URL so you can add it to the social media configurations.
6) Now, 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
    a. Add the same attribute to any other Controllers present in your project as well.
8) Run the application.
    a. If running in IE, follow the instructions to trust the self-signed certificate that IIS Express has generated. Then, in the Security Warning dialog, click Yes if you want to install the certificate representing localhost.
    b. If running in Google Chrome, the certificate is accepted and will show HTTPS content after a warning.
    c. If running in Firefox, a warning will. For our application you can safely click I Understand the Risks.

Set up Google Authentication

1) Navigate to the Google Developers Console https://console.developers.google.com/
2) If you have not already done so, register as a developer with Google. Once registered, you can create a new project, as outlined below.

3) 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.
4) Under Social APIs, click Google+ API.
5) Click the Enable API button at the top of the page.
6) In the left tab, under APIs & Auth, click Create Credentials.
7) Click OAuth Client ID. Click the Configure consent screen button.
8) Complete the credentials OAuth consent form
9) In the Create Client ID dialog, select Web application for the application type. The default name can be left.
10) Set the Authorized JavaScript origins to the SSL URL for your project.
11) Set the Authorized redirect URI to:
    <your project SSL URL>/signin-google
12) Click the create button and copy the client id and and client secret.
13) Back in Visual Studio, open App_Start\Startup.Auth.cs.
14) Locate the ConfigureAuth method and uncomment the lines pertaining to Google Authentication at the bottom of the method.
15) Copy and paste the ClientId and ClientSecret from your Google project into the UseGoogleAuthentication method.
16) Save your project. You may test it at this point or continue adding additional external logins.

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) Click the + Add a New App green button.
4) Enter an App Name, email address, and Category, then click Create App ID.
    a. 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) You will be prompted with a security CAPTCHA, fill in the text box and click SUBMIT.
6) Once the app has been created, select Settings for the left menu bar.
7) On the Basic settings section of the page select Add Platform to specify that you are adding a website application.
8) Make a note of your App ID and your App Secret so that you can add both into your MVC application later.
9) Now add the site URL for your MVC application, then add any other required information and select Save Changes.
10) Note that you will only be able to authenticate using the email alias you have registered. Other users and test accounts will not be able to register. You can grant other Facebook accounts access to the application on the Facebook Developer Roles tab.
11) Back in Visual Studio, open App_Start\Startup.Auth.cs.

12) Locate the ConfigureAuth method and uncomment the lines pertaining to Facebook Authentication near the bottom of the method.
13) Copy and paste the appId and appSecret into the UseFacebookAuthentication method.
14) Save your project.

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
    a. Select Tools > NuGet Package Manager > Manage NuGet Packages for Solution
    b. In the left column, select Online > All
    c. In the search bar (top right column), type Owin.Security.Providers
    d. The package should appear in the center column - click Install
    e. 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:
    a. Like Facebook and Google, you need to create a project on the LinkedIn Developer Network
    b. Go to http://developer.linkedin.com/ and login using your existing LinkedIn ID (if you have one), if not, register as a developer
    c. When logged in, click MY APPS followed by the yellow button CREATE APPLICATION.
    d. Complete the new application registration form, using your MVC project's SSL URL for the website URL and <your SSL URL>/signin-linkedin for the OAuth 2.0 Redirect URL.
    e. A 1:1 aspect ratio logo is required (same width to height) to be uploaded
    f. When finished, create the project and wait for the verification page
    g. Once created, a page will display showing the API Key and Secret Key for the application
3) Now go back to your MVC project and open the App_Start\Startup.Auth.cs file and add the following namespace:
    a. using Owin.Security.Providers.LinkedIn;
4) Register the LinkedIn provider in the ConfigureAuth method (you can add this below the Google and Facebook authentications)
    a. app.UseLinkedInAuthentication("<YOUR API KEY>", "<YOUR SECRET KEY>");
    b. Make sure your use the API and Secret Keys for your application

BONUS: Optional Configurations

Twitter and Microsoft

Based upon your experience setting up Facebook, Google, and LinkedIn, see if you can figure out how to set up these 3rd party authentications.

GitHub

This one requires more a little bit more work. Go to https://developer.github.com/v3/oauth/ and read through GitHub's oauth API documentation. See if you can figure out how to add this as an additional login option.

Test the Logins

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).