|  |  |
| --- | --- |
|  | Azure Active Directory B2C  Module 2 Lab – Set Up Lab Environment |
|  |  |
|  |  |

Contents

[Overview 3](#_Toc455401010)

[Learning Objectives 3](#_Toc455401011)

[Pre-requisites 3](#_Toc455401012)

[1. Set up your Azure AD B2C directory 3](#_Toc455401013)

[2. Create a web application 4](#_Toc455401014)

[3. Add Facebook support 7](#_Toc455401015)

[4. Add Google account support (optional) 8](#_Toc455401016)

[5. Add Microsoft account support (optional) 10](#_Toc455401017)

[6. Add MFA support to your sign-up or sign-in policy 11](#_Toc455401018)

[7. Wire up a .NET MVC app in Visual Studio with your policies (optional) 12](#_Toc455401019)

# Overview

Azure Active Directory (Azure AD) B2C is a cloud identity service for your consumer-facing web and mobile apps. It is highly-available, secure and scales to millions of consumer identities. Consumers can use their social accounts (Facebook, Google, Microsoft account, etc.) or create new credentials to access your apps. In this lab you will set up the environment you will be using in subsequent labs, including creating a B2C directory.

**Estimated time to complete this lab: 90 minutes (plus 30 minutes for optional exercises)**.

# Learning Objectives

In this hands-on lab you will set up the environment by creating an Azure Active Directory tenant, register a web application, and then set up policies.

# Pre-requisites

1. Windows 10
2. Visual Studio 2015 (to complete app-side work)
3. Azure subscription – if you have not got one, you will sign up for a free account as your first task

**Note:** At the time of writing, you can use Azure AD B2C preview for free but its use is limited to 50,000 users per tenant. An Azure subscription is required to access the Azure classic portal, and create a B2C tenant.

## Set up your Azure AD B2C directory

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Sign up for an Azure account - **ignore this step if you already have one** | 1. You will need to use an “MSA” account (e.g. [xyz@outlook.com](mailto:xyz@outlook.com)), so if you do not have one, create one [here](https://signup.live.com/signup?cbcxt=mail&wreply=http%3a%2f%2fmail.live.com&id=64855&lic=1&OLrefer=merchsite&uaid=74d9bb8fbfda401c92194b342e5377bd) (or go to <https://www.microsoft.com/en-gb/outlook-com/> and click **Sign up**) 2. Go to <https://azure.microsoft.com/free/> and sign up for a new Azure account using an MSA account (above)   **Note:** You must verify by phone, and credit card. Your credit card details are required, but will not be charged for the demo environments as they all use free trial SKUs (unless you decide to extend your demo environment beyond the trial period). |
| Create an Azure AD B2C directory | **Note:** Currently B2C features can't be turned on in any existing directory, so you will have to add one.   1. Sign in to the Azure classic portal at <https://manage.windowsazure.com/> as the Subscription Administrator (this will be your MSA account if you just created a new Azure subscription, and is generally the sign-up account) 2. Click **+ NEW**, **APP SERVICES**, **ACTIVE DIRECTORY**, **DIRECTORY**, **CUSTOM CREATE** 3. Choose the **Name**, **Domain Name** and **Country or Region** for your directory 4. Select the option that says **This is a B2C directory.** 5. Complete the action (tick)   **Note:** When your directory has been created, it will appear in the list of directories. You are also made a user of the directory, with the organizational role being Global Administrator. You can add other Global Administrators as required.   1. The (new) Azure portal will open in a new browser tab or window, showing the Settings blade (for B2C); **pin this blade to your dashboard** (the pin is at the top of the main blade)   **Note:** It can take a few minutes for your new directory to be accessible on the Azure portal. If it doesn’t seem to be working, retrying these steps after some time may fix it. If not, please contact Support. |
| Create a Global Admin user in your B2C tenant | 1. Switch back to the tab or window that still has the “classic” Azure Portal open   **Note:** You will manage users and groups, self-service password reset configuration, and company branding features of your tenant from the Azure classic portal.   1. Click **USERS** and **ADD USER** 2. The TYPE OF USER is **New user in your organization** 3. USERNAME is **Admin@<TenantName>.onmicrosoft.com** (where <TenantName> is the NAME you used in step 3 above – see note below) 4. DISPLAYNAME is **Admin** 5. ROLE is **Global Admin** 6. Provide a suitable (real) alternate email address 7. On the next page click **create** and be sure to **write down the temporary password**!   **Important Notes – read carefully:** The B2C directory you have just created has a B2C portal, and now it has its own administrator. To all intents and purposes it behaves like a tenant, and we will refer to it as your B2C “tenant”, and **we will use <TenantName> to refer to the NAME you provided** in step 3 above.  **In future exercises it makes sense to sign in to the Azure portal (portal.azure.com) with this new admin account**, because it naturally defaults to your B2C tenant. However, if you want to directly administer users or **anything that requires the classic portal (manage.windowsazure.com), you will have to use the existing admin account** (that you are presumably still signed in with now). Many of the exercises will work perfectly well with this account too, though you will have to make sure that the correct directory is selected – you can select the correct directory in the drop down at the top-right of the Azure portal (remembering that your tenant is really a directory). When uploading files in later labs, you must use your b2c tenant admin account.  **In summary it is probably best that you think of the B2C tenant admin account when accessing the B2C portal (portal.azure.com), and you must use the original admin whenever you are accessing the classic portal (manage.windowsazure.com).**  **At the time of writing B2C is not “GA” (not generally available), and until it is you will need to get your tenant “whitelisted” for premium features. This is needed in some later labs, and it can take a while. Therefore, you should initiate that process right now! Send an email addressed to AADB2CPreview@microsoft.com with the subject “Whitelist tenant for Azure AD B2C Partner Conference” followed by your B2C domain name (<TenantName>.onmicrosoft.com).**   1. Close all instances of your browser |

## Create a web application

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Register a web app | 1. Navigate to **portal.azure.com** and **sign in with your tenant admin account** ([admin@<TenantName>.inmicrosoft.com](mailto:admin@%3cTenantName%3e.inmicrosoft.com)) 2. Click **AZURE AD B2C SETTINGS** (which you pinned, earlier – if it is not there, browse to Azure AD B2C) 3. On the **Settings** blade in the Azure portal, click **Applications** andclick **+Add** at the top of the Applications blade 4. In the New Application blade, enter the name **b2ctestapp**, and set the Web App / Web API to **Yes**   **Note:** This is the name that your consumers/users will see.   1. Enter the Redirect URIs **https://jwt.io** (to use in your Visual Studio .NET MVC app) and **https://localhost:44321/**   **Note:** These two URLs will be used to test your policies by simply revealing what is in the token following a successful login. If you are developing an application locally, following our optional exercise later on, you will use **https://localhost:44321/**. Right now we will use **https://jwt.io** which is a public site which decodes, verifies and displays the contents of the token returned after successful sign-up or sign-in.   1. Click **Create** to register your application 2. Click the **b2ctestapp** application that you just created, and copy the globally unique Application Client ID 3. Open notepad, paste the Application Client ID and save the file as (for example) **b2cdata.txt** on your desktop (you will need this for your code later on) |
| Create a sign-up or sign-in policy | **Note:** A Policy fully describes the consumer identity experience when you sign-up or sign-in, profile editing and password reset. Your app triggers the appropriate experience by invoking the policy (query parameter ‘p’ for the policy name) as part of the authentication request.   1. On the Settings blade, click **Sign-up or sign-in policies** andclick **+Add** at the top of the blade 2. In the Add sign-up or sign-in policy blade enter the name **susi**   **Note:** The name determines the sign-up policy name used by your application, and in the sample application this is set in the web.config file under the key ida:SusiPolicyId.   1. Click **Identity providers**, select **Email signup**, and click **OK**   **Note:** Later you will be able to choose additional identity providers such as Facebook and Google.   1. Click **Sign-up attributes**, click to the left of the name column and select **Country/Region**, **Display Name**, **Postal Code** and click **OK**   **Note**: The attributes you select here are the ones that you want to collect from the consumer during sign-up. You will see later how to add additional attributes.   1. Click **Application claims**,click to the left of the name column and select **Display Name**, **Identity Provider**, **Postal Code**, **User is new**, **User's Object ID** and click **OK**   **Note:** Application claims are the claims that you want returned in the tokens sent back to your application after a successful sign-up or sign-in experience.   1. Click **Create**   **Note:** The policy appears as "**B2C\_1\_susi**" (the **B2C\_1\_** fragment is automatically added) in the **Sign-up or sign-in policies** blade. |
| Test your policy | 1. On the Sign-up or sign-in policies blade click **B2C\_1\_susi** 2. On the Select Redirect URI drop-down ensure **https://jwt.io** is selectedand click **Run now**   **Note:** You will now be taken through the sign-up or sign-in consumer experience as configured, in a new browser tab. As we haven’t created any consumer accounts yet (and only have email sign-up/in) the only rational option is to sign-up with an email account now.   1. In the new browser window, click **Sign up now** 2. Provide an Email Address (a real email address you can access) and click **Send verification code**, when you receive the email, enter the Verification code and click **Verify code** 3. Provide a password and confirm 4. Provide the attributes that you selected when you configured the policy and click **Create**   **Note:** The website displays the contents of your token. You can see the attributes that you specified as claims and a Boolean attribute called newUser.   1. Click **Sign Out** and close the browser tab 2. Click **Run now** again and this time sign-in using the Email Address and Password you just registered   **Note:** Again the website displays the contents of the token and this time there is no newUser Boolean.   1. Click **Sign Out** and close the browser tab |
| Verify account creation | **Note:** You now need to run the classic portal (manage.windowsazure.com), and as explained earlier, you have to do this using your original admin account. So it makes sense to open a different browser (or an inPrivate session), signing in with that account, so that you can switch between the two sessions as necessary.   1. In a different browser (or inPrivate session), navigate to manage.windowsazure.com and then to your B2C **<TenantName>** directory 2. Click **USERS** and see that you have a new user corresponding to the identity that just signed up |
| Create a password reset policy | 1. Back in the Azure portal (signed in with your tenant admin), on the **Settings** blade click **Password reset policies** 2. On the Password reset polices blade, click **+Add** 3. Enter the name **SSPR** 4. Click **Identity providers**, select **Reset password using email address** andclick **OK** (the only one currently available) 5. Click **Application claims** and select **User's Object ID** and click **OK**   **Note:** Here you choose claims that you want returned in the tokens sent back to your application after a successful password reset experience.   1. Click **Create**   **Note:** That the policy just created appears as "**B2C\_1\_SSPR**" (the **B2C\_1\_** fragment is automatically added) in the **Password reset policies** blade.   1. Click **B2C\_1\_SSPR** 2. On the Select Redirect URI drop-down ensure **https://jwt.io** is selectedand click **Run now** |
| Test your password reset policy | 1. Enter your Email Address and click **Send verification code** 2. When you receive the code in email, enter the Verification code and click Continue 3. Enter and confirm your new password and click Continue   **Note:** The website is displayed again with the oid that you specified (but no other attributes). |

## Add Facebook support

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Create a Facebook application | **Note:** To use Facebook as an identity provider in Azure Active Directory (Azure AD) B2C, you need to create a Facebook application and supply it with the right parameters. You need a Facebook account to do this. If you don’t have one, you can get it at, https://www.facebook.com/.   1. Go to the Facebook for developers website, **https://developers.facebook.com/** and sign in with your Facebook credentials 2. If you have not already done so, you need to register as a Facebook developer - Click **Register** (on the upper-right corner of the page), accept Facebook's policies, and complete the registration steps 3. Click **My Apps** and then click **Add a new App** 4. Click **Website** as the platform, and then click **Skip and Create App ID** 5. On the form, enter the Display Name **<TenantName>**, a valid Contact Email, select the **Education** Category (for example), and click **Create App ID**. 6. Accept the Facebook platform policies and complete the online security check 7. Look for the Settings menu (normally in the left hand navigation) and click **+Add Platform** 8. Select **Website** 9. Enter **https://login.microsoftonline.com/** in the Site URL field and then click **Save Changes** 10. Copy the value of **Application ID** and paste it into your **b2cdata.txt** file (saved on your desktop) 11. Click **Show** and copy the value of **App Secret** and paste it into your **b2cdata.txt** file too   **Note:** You will need both the Application ID and the App Secret to configure Facebook as an identity provider in your tenant. **App Secret** is an important security credential.   1. Click **+ Add Product** on the left navigation and then the **Get Started** button next to **Facebook Login** 2. Enter **https://login.microsoftonline.com/te/<TenantName>.onmicrosoft.com/oauth2/authresp** in the Valid OAuth redirect URIs field in the Client OAuth Settings section (making sure that you replace **<TenantName>** with your tenant name) 3. Click **Save Changes** at the bottom of the page 4. Click **App Review** on the left navigation and set the make public switch to **Yes** and click **Confirm**   **Note:**  Making your Facebook application publically available, allows us to use it in Azure AD B2C. |
| Configure Facebook as an identity provider for your B2C environment | 1. On the Setting blade of the Azure portal, click **Identity providers** 2. Click **+Add** at the top of the blade 3. Enter the name **Facebook**   **Note:** The name you choose does not have to be Facebook but it is displayed to the consumer.   1. Click **Identity provider type**, select **Facebook**, and click **OK** 2. Click **Set up this identity provider** and enter the **app ID** and **app secret** (that you just saved in your **b2cdata.txt** file) into the **Client ID** and **Client secret** fields respectively 3. Click **OK** (you may have to click the background of the blade first) and then click **Create** to save your Facebook configuration |
| Add Facebook support to your sign-up or sign-in policy | 1. On the Setting blade, click **Sign-up or sign-in policies** 2. Click the **B2C\_1\_susi** 3. Click **Edit** (at the top) 4. Click **Identity Providers** and select **Facebook** (in addition to Email signup) 5. Click **OK** and then click **Save** |
| Test Facebook sign-up and sign-in for your app | 1. Click **Run now** to test the sign-up or sign-in policy again (which will now show Facebook in addition to email Address sign in 2. Click the **Facebook** button 3. Because you are already logged in to Facebook, just click **OK** (or log in if necessary)   **Note:** Since this is a sign-up (the first time this Facebook identity has been seen by our B2C directory) you are asked to provide the attributes we configured. Note how the display name has already been populated in this case, but can change it if you want to.   1. Click **Continue**   **Note:** The website displays the contents of your token. You can see the attributes that you specified as claims and the newUser Boolean, and also the idp provider facebook.com.   1. Click Sign out and close the browser tab. 2. Click **Run now** again 3. Click the **Facebook** button again (authenticating if necessary)   Note: The website displays the contents of your token as before, but this time without the newUser Boolean. |
| Verify account creation | 1. In manage.windowsazure.com (presumably still open in another browser session), if necessary navigate to your **<TenantName>** directory 2. Click **USERS** and see that you have another new user, corresponding to the identity that just signed up |

## Add Google account support (optional)

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Create a Google+ application | **Note:** To use Google+ as an identity provider in Azure Active Directory (Azure AD) B2C, you need to create a Google+ application and supply it with the right parameters. You need a Google+ account to do this. If you don’t have one, you can get one at https://accounts.google.com/SignUp/.   1. Back in your Azure Portal browser session, go to the Google Developers Console **https://console.developers.google.com** and sign in with your Google+ account credentials 2. From the select a project drop-down, select **Create a project...** 3. Enter the name **<TenantName>** (for example) answer the questions and then click **Create** 4. Under **API Manager** click **Credentials** (top left navigation) 5. Click the **OAuth consent screen** tab at the top 6. Select or specify a valid Email address, provide a Product name **<TenantName>** for example, and click **Save** 7. From the Create credentials drop-down **selectOAuth client ID** 8. Under **Application type**, select **Web application**   Enter **<TenantName>** for example in the Name field, enter **https://login.microsoftonline.com** in the Authorized JavaScript origins field, and **https://login.microsoftonline.com/te/<TenantName>.onmicrosoft.com/oauth2/authresp** in the Authorized redirect URIs field, remembering to replace <TenantName>, and noting that the **<TenantName> is all lower case as this is case-sensitive!**   1. Click **Create** 2. Copy the values of **Client ID** and **Client secret** and save them in your **b2cdata.txt** file (on the desktop) - you will need both of them to configure Google+ as an identity provider in your B2C environment |
| Configure Google+ as an identity provider for your B2C environment | 1. In the Azure portal, in the Settings blade, click **Identity providers** 2. Click **+Add** 3. Enter the name **Google+**, click **Identity provider type**, select **Google**, and click **OK** 4. Click **Set up this identity provider** and enter the client ID and client secret of the Google+ application (which you saved in your b2cdata.txt file) 5. Click **OK** (you will have to click somewhere else in the blade first) and then click **Create** to save your Google+ configuration |
| Add Google+ support to your sign-up or sign-in policy | 1. On the Settings, click **Sign-up or sign-in policies** 2. Click **B2c\_1\_susi** and click **Edit** at the top 3. Click **Identity Providers** and select **Google+** (in addition to the others) 4. Click **OK** and then click **Save** |
| Test Google+ sign-up and sign-in for your app | 1. Click **Run now** to test the sign-up or sign-in policy again (which will now show Google+ in addition to email Address and Facebook sign in) 2. Click the Google**+** button 3. Because you are already logged in to Google+, just click **Allow** (or log in if necessary) 4. Enter values for the additional attributes and click **Save**   **Note:** Since this is a sign-up (the first time this Google+ identity has been seen by our B2C directory) you are asked if you want Google to pass over your name and email address. You also need to provide the attributes we configured.   1. Click **Continue**   **Note:** The website displays the contents of your token. You can see the attributes that you specified as claims and the newUser Boolean, and also the IDP as google.com.   1. Click **Sign out** and close the browser tab 2. Click **Run now** again 3. Click the **Google+** button again (authenticating if necessary)   Note: The website displays the contents of your token as before, but this time without the newUser Boolean. |
| Verify account creation | 1. In manage.windowsazure.com (presumably still open in another browser session), if necessary navigate to your **<TenantName>** directory 2. Click **USERS** and see that you have another new user, corresponding to the identity that just signed up |

## Add Microsoft account support (optional)

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Create a Microsoft account application | **Note:** To use Microsoft account as an identity provider in Azure Active Directory (Azure AD) B2C, you need to create a Microsoft account application and supply it with the right parameters. You need a Microsoft account to do this. If you don’t have one, you can get it at <https://www.live.com/>.   1. Back in your Azure Portal browser session, go to the Microsoft account Developer Center at **https://account.live.com/developers/applications** and sign in with your Microsoft account credentials 2. Click **Add an app** 3. Enter the name **<TenantName>** 4. Copy the App Id and Paste it in your **b2cdata.txt** file 5. Click Generate New Password and paste that into your **b2cdata.txt** 6. Click **Add platform** and choose **Web** 7. Enter  **https://login.microsoftonline.com/te/<TenantName>.onmicrosoft.com/oauth2/authresp** in the **Redirect URLs** field (making sure that you replace **<TenantName>** with your directory name) 8. Click **Save** (at the bottom of the page) |
| Configure the Microsoft account as an identity provider for your B2C environment | 1. In the Azure portal, in the Settings blade, click **Identity providers** 2. Click **+Add** 3. Enter the name **Microsoft Account**, click **Identity provider type**, select **Microsoft Account**, and click **OK** 4. Click **Set up this identity provider** and enter the Application ID and Application secret of the Microsoft application (which you saved in your b2cdata.txt file) 5. Click **OK** (you will have to click somewhere else in the blade first) and then click **Create** to save your Microsoft Account configuration |
| Add Google+ support to your sign-up or sign-in policy | 1. On the Settings, click **Sign-up or sign-in policies** 2. Click **B2c\_1\_susi** and click **Edit** at the top 3. Click **Identity Providers** and select **Microsoft Account** (in addition to the others) 4. Click **OK** and then click **Save** |
| Test Google+ sign-up and sign-in for your app | 1. Click **Run now** to test the sign-up or sign-in policy again (which will now show Microsoft Account in addition to email Address, Facebook and Google+ sign in 2. Click the **Microsoft Account** button 3. Although you are already logged in, you may be asked to verify your password – so do this 4. Click **Yes** when asked for permission   **Note:** Since this is a sign-up (the first time this Microsoft Account identity has been seen by our B2C directory) you are asked if you want Microsoft to sign you in and pass over your some of your details such as your name and email address.   1. Enter values for the additional attributes and click **Save** 2. Click **Continue**   **Note:** The website displays the contents of your token. You can see the attributes that you specified as claims and the newUser Boolean, and also the IDP as live.com.   1. Click **Sign out** and close the browser tab. 2. Click **Run now** again 3. Click the **Microsoft Account** button again (authenticating if necessary)   **Note:** The website displays the contents of your token as before, but this time without the newUser Boolean. |
| Verify account creation | 1. In manage.windowsazure.com (presumably still open in browser session another tab), if necessary navigate to your **<TenantName>** directory 2. Click **USERS** and see that you have another new user, corresponding to the identity that just signed up |

## Add MFA support to your sign-up or sign-in policy

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Enable MFA in your policy | 1. Back in your Azure Portal browser session, on the Settings blade, click **Sign-up or sign-in policies** 2. Click **B2C\_1\_susi** and click **Edit** at the top 3. Click **Multi-factor authentication** and turn the State to **ON** 4. Click **OK** 5. Click **Save** (at the top of the blade) |
| Test your MFA policy | 1. Click **Run now** to test the MFA policy 2. Sign-in by any method – for example Facebook 3. Enter a valid phone number and click **Send Code**   **Note:** Since this is the first sign-in for this account since MFA was enabled you are asked to provide a telephone number. This is stored against the consumer account in the directory and must be used for future authentications.   1. When you receive the code, enter your **Verification code** to complete authentication   **Note:** The website displays the contents of your token as before.   1. Sign out and close the browser tab 2. Click **Run now** 3. Sign-in by the same method as before and establish that it has remembered your phone number |
| Verify the account | 1. In manage.windowsazure.com (presumably still open in another browser session), and if necessary navigate to your **<TenantName>** directory 2. Click **USERS**, select the consumer account that you used for sign-in and click the Work Info tab to verify that the phone number has been stored here   **Note:** Under authentication contact info, you can see that the authentication phone has been stored. You could edit this (as an administrator) if the consumer’s phone number changed. |

## Wire up a .NET MVC app in Visual Studio with your policies (optional)

|  |  |
| --- | --- |
| Task | Detailed Steps |
| Download and configure a .NET MVC app | 1. Download the B2C sample app on GitHub at **https://github.com/azureadquickstarts/b2c-webapp-openidconnect-dotnet-susi** 2. Extract/copy all the files to the **c:\B2C-webapp\** folder 3. Open the **B2C-WebApp-OpenIDConnect-DotNet-SUSI-master folder** 4. Open the **WebApp-B2C-DotNet folder** 5. Open the **WebApp-OpenIDConnect-DotNet-B2C.csproj** project file in Visual Studio 6. In Visual Studio, open the **Web.Config** file (double-click in Solution Explorer) and edit the appSettings node as follows (including values from your notepad file):   <appSettings>  <add key="webpages:Version" value="3.0.0.0" />  <add key="webpages:Enabled" value="false" />  <add key="ClientValidationEnabled" value="true" />  <add key="UnobtrusiveJavaScriptEnabled" value="true" />  <add key="ida:Tenant" value="[Name of your B2C directory]" />  <add key="ida:ClientId" value="[Application Client Id]" />  <add key="ida:AadInstance" value="https://login.microsoftonline.com/{0}{1}{2}" />  <add key="ida:RedirectUri" value="https://localhost:44316/" />  <add key="ida:SusiPolicyId" value="b2c\_1\_susi" />  <add key="ida:PasswordResetPolicyId" value="b2c\_1\_sspr" />  </appSettings>   1. Press **F5** to build and run the application – the application default page is displayed 2. Click **Claims** and your susi policy is called 3. Sign in and complete the MFA and you will see your claims displayed   **Note:** If you had clicked “Forgot your password?” your SSPR policy would have been called. |