


Azure AD B2C Quickstart with Visual Studio & Blazor

 medium.com/marcus-tee-anytime/azure-ad-b2c-quickstart-with-visual-studio-blazor-563efdff6fdd

October 16, 2021

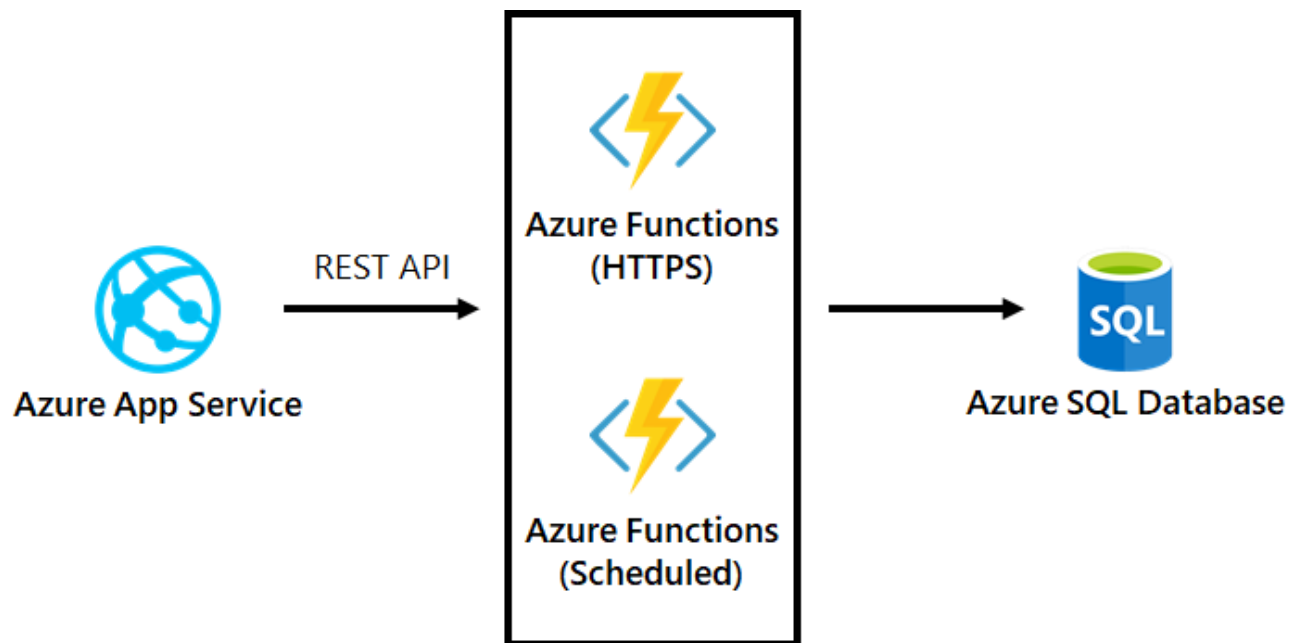


Marcus Tee
Feb 23, 2020

In short, Blazor allows developer to develop web applications in C# (and of course, HTML), without toggling to JavaScript to render frontend, and back to C# for backend. This greatly reduce the complexity of web app development for me, as I can continue to tap on my C# knowledge to develop apps. Blazor supports both server-side and client-side. More details can be found [here](#).

I was developing an application on Blazor, and planning to host it in Azure. The goal here is that I can access my web application anytime, anywhere, yet Azure helps to protect my cloud assets, and of course, minimizing the cost.

Basically, this application will run batch job that retrieves information from public, store in a database, and perform some calculation that is going to be presented in front-end.



Architecture of my application

I need to implement an authorization component in my application as the information is sensitive. The pre-requisite of having authorization is authentication. This is where I have to choose between these 2 options:

1. Form based Authentication
2. External Authentication

While Form based authentication gives more control, but it means that I have to cater for additional efforts to setup an identity store and secure it. While I'm searching for alternatives, I studied Azure AD B2C, and it looks like a viable solution for my application. I

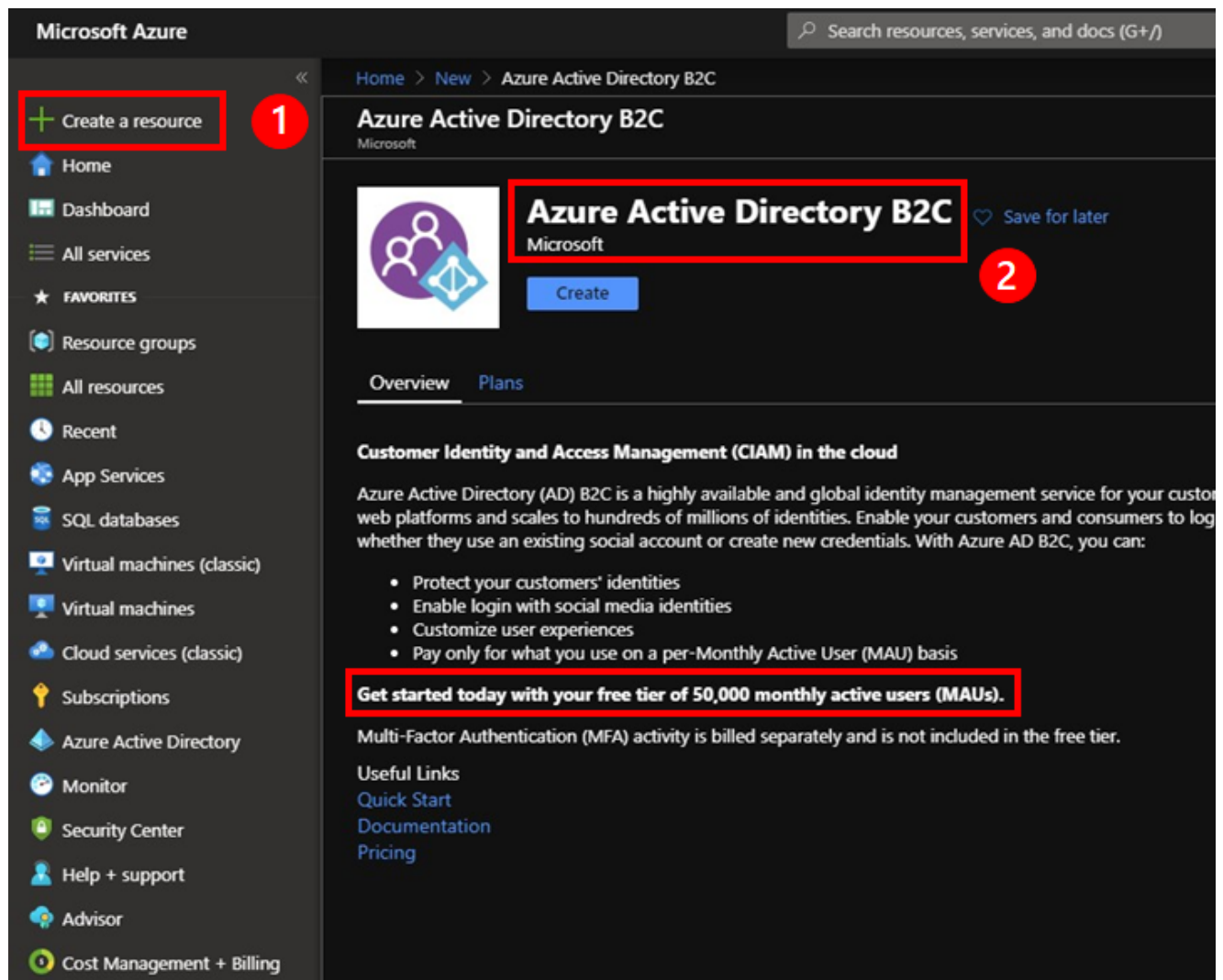
can leverage on other identity providers to secure my identity, in addition to enjoying Single-Sign On without having additional sets of credential.

To my surprise, adding authentication using Azure AD B2C to Blazor is few clicks away, and I'm able to enjoy the benefits mentioned above. The steps can be broken down into 4 sections:

1. Register Azure AD B2C
2. Configure Azure AD B2C Identity Provider & User Flow
3. Register Apps in Azure AD B2C tenant
4. Create new Blazor App
5. Enjoy!

[1] Register Azure AD B2C

This is pretty straightforward. Login to Azure Portal, simply click “**Create a resource**”, search for “*Azure Active Directory B2C*”, and click create. Fill in the domain name, and it takes about a minute to create the tenant.

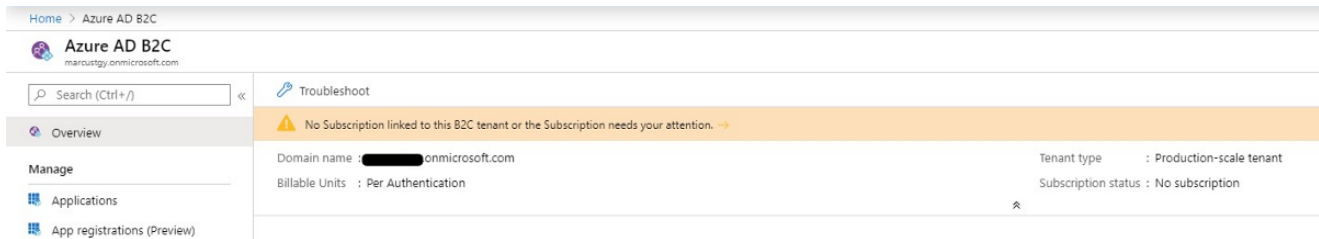


Azure Portal Screen to create Azure AD B2C.

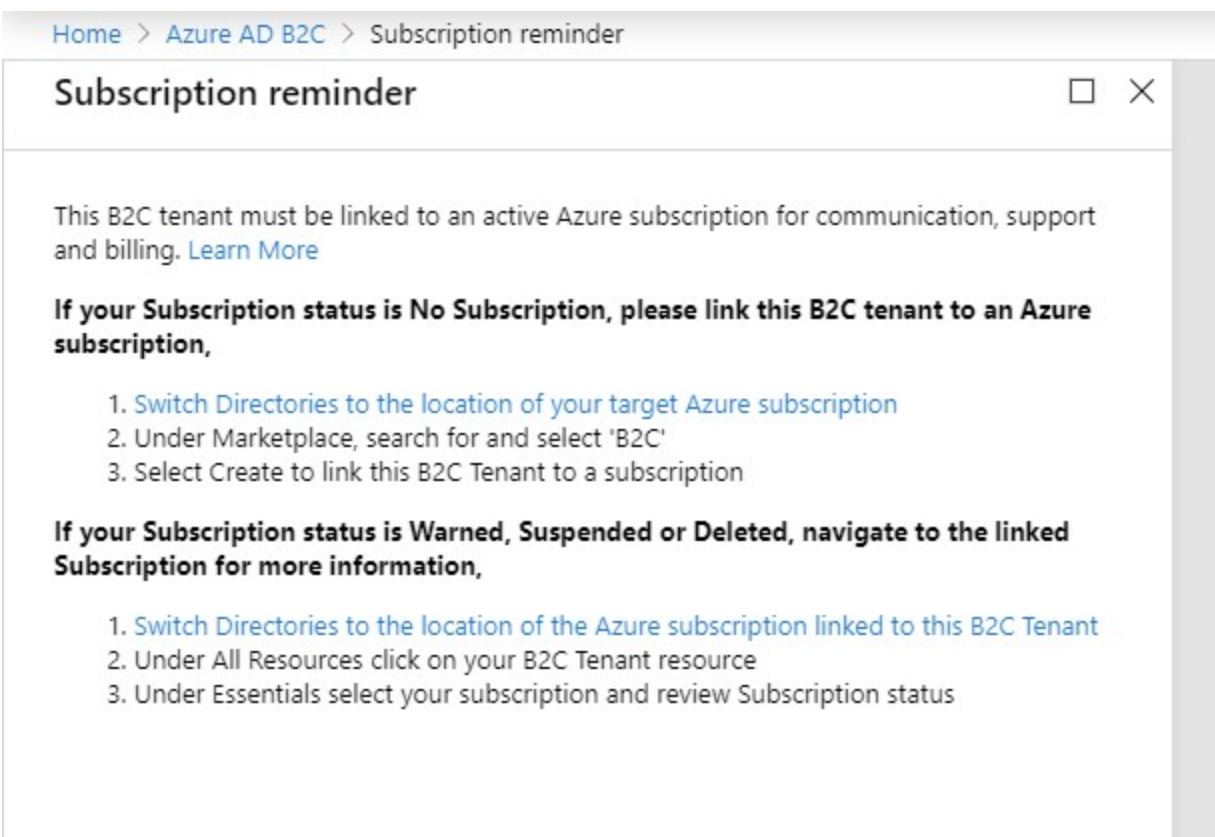
Quick tips: Azure AD B2C pricing has changed. It used to be consumption basis, i.e. number of authentication, with a limited number of authentication under free tier. Recently, it has changed to Monthly Active Users (MAU) basis, and the first MAU is free! Yes, you hear it

right, users. Technically I'm using this service without paying anything. More details on the pricing .

Once done, navigate to the Azure AD B2C portal. If this is the first time you create Azure AD B2C, you will need to associate this tenant to an Azure subscription. Simply click on the yellow bar to associate this tenant to the Azure subscription.

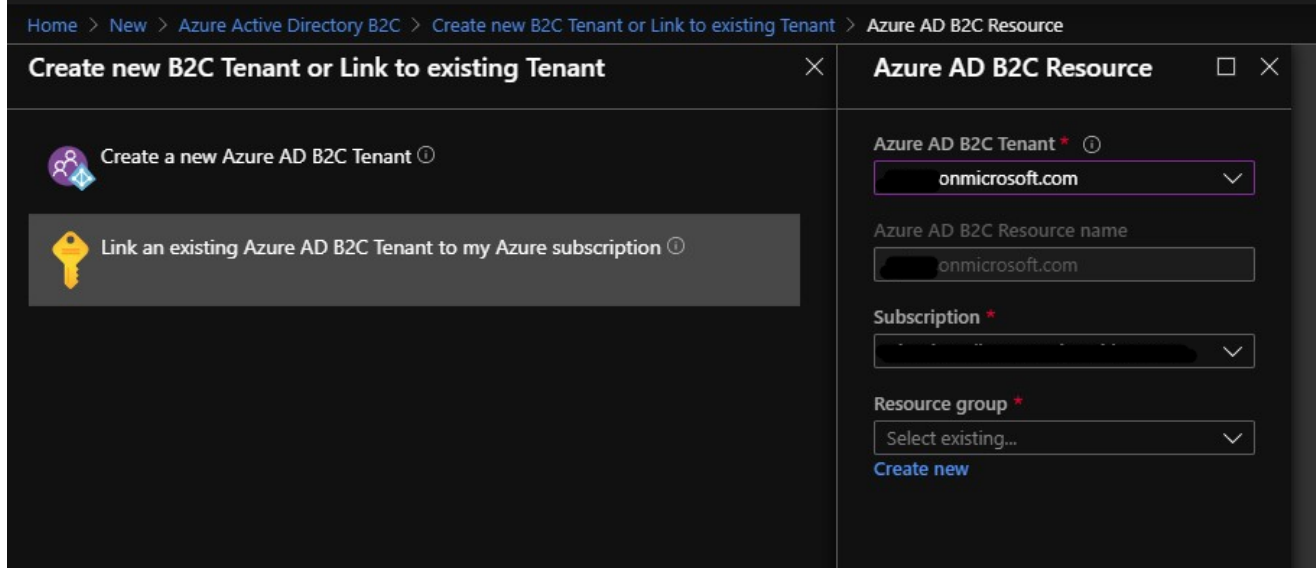


The steps is self-explanatory, we will just need to switch back to our previous tenant who holds the Azure subscription and link this tenant. To switch tenant, navigate to top right corner, click on your account, and select “Switch directory”.



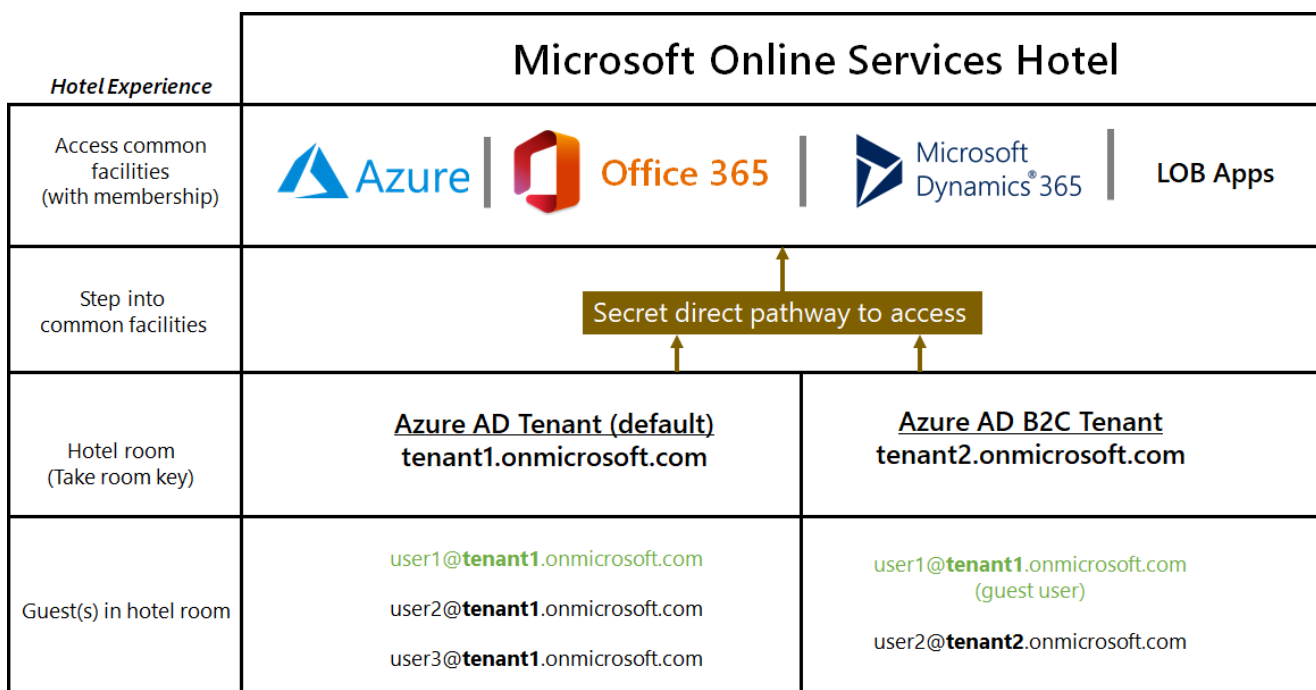
Steps to link Azure AD B2C Tenant to Azure Subscription.

You will go through similar steps as Azure AD B2C Tenant creation, but this time, you will link the tenant to your subscription.



To help you understand the concept of tenant, I use the analogy as follow. Imagine today, Microsoft operates a giant hotel which provides many common facilities such as restaurant, sports center etc. Anyone who stays in this hotel can enjoy these facilities as long as the guest has memberships. In simplest view, one family books a room, and all guests staying in this room can access these common facilities. Of course, it could be the case of two families travelling together, and they book room separately. Family member A can go to family B's room and vice versa.

Let me relate back to the platform. Microsoft operates hyperscale datacenters globally (hotel), and these datacenters host multi-tenanted online services (common facilities). Each tenant (hotel room) can access the same services independently, as long as the tenant has subscriptions (memberships). In Azure AD, there's a context of guest users, where admin can allows users from other tenant to have limited access. Diagram below explains the concepts.



Analogy of tenant and Microsoft online services.

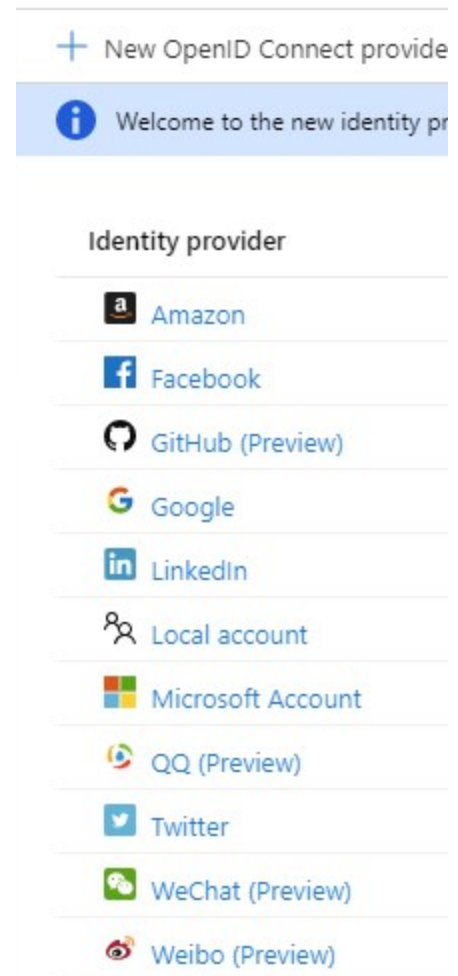
[2] Configure Azure AD B2C Identity Provider & User Flow

Azure AD B2C allows developer to incorporate 3rd party service provider as identity provider. There are two main advantages. From users perspective, there isn't additional sets of credential. As application developer, he/she can focus on the application and offload authentication to the respective identity provider.

To configure this, simply navigate to “*Identity providers*” tab on left panel. There are pre-configured connectors available, as well as custom OpenID Connect provider. Each identity provider has different steps to setup, I'm not going through the details, basically it splits into 2 parts, first you register an application with the respective identity provider, then you configure the connection using client ID and secret. As of today (Feb 2020), it supports the following:

Here's a sample tutorial to configure using Microsoft Account: <https://docs.microsoft.com/en-us/azure/active-directory-b2c/identity-provider-microsoft-account>

There are certain attributes which is retrievable from the identity, such as email, display name etc. Depends on the application needs, you may want to have additional custom attributes that users can enter, for example, nickname, contact number etc.



List of identity provider supported out of the box in Azure AD B2C.

Search (Ctrl+/)	+ Add
Overview	
Manage	
Applications	
App registrations (Preview)	
Identity providers	
Company Branding	
User attributes	
Users	
Roles and administrators	
Policies	
User flows	
Identity Experience Framework	
Security	
Authentication methods (Preview)	
Activities	
Audit logs	

Name	Data Type	Description	Attribute type
City	String	The city in which the user is located.	Built-in
Country/Region	String	The country/region in which the user is located.	Built-in
Display Name	String	Display Name of the User.	Built-in
Email Addresses	StringCollection	Email addresses of the user.	Built-in
Given Name	String	The user's given name (also known as first name).	Built-in
Identity Provider	String	The social identity provider used by the user to access to your application.	Built-in
Job Title	String	The user's job title.	Built-in
Legal Age Group Classification	String	The legal age group that a user falls into based on their country and date of birth	Built-in
Nickname	String	User self-defined nickname that is displayed in application.	Custom
Postal Code	String	The postal code of the user's address.	Built-in
State/Province	String	The state or province in user's address.	Built-in
Street Address	String	The street address where the user is located.	Built-in
Surname	String	The user's surname (also known as family name or last name).	Built-in
User is new	Boolean	True, if the user has just signed-up for your application.	Built-in
User's Object ID	String	Object identifier (ID) of the user object in Azure AD.	Built-in

Steps to create custom attributes.

Once decided, we can proceed to create User flows. In short User flows is a set of policies that describe identity experiences such as sign-up, sign-in and profile editing. More details can be found here. Common user flows in an application are sign-up / sign-in, and also password reset policy. Azure AD B2C provides default policies that application can consume, or advanced policies (not going to details).

Home > Azure AD B2C - User flows > Create a user flow

Azure AD B2C - User flows

marcustgy.onmicrosoft.com

Search (Ctrl+/)

+ New user flow

User flow name

Search using user flow name

Filter by

Name	Type
No user flows found.	

Overview

Manage

Applications

App registrations (Preview)

Identity providers

Company Branding

User attributes

Users

Roles and administrators

Policies

User flows

Identity Experience Framework

Let's setup 2 policies, which is sign-up/sign-in, and password reset. Setup is pretty straight forward, give it a name, select identity provider(s), and finally, select user attributes that you want to collect and return as part of the claim. The claim can then be used in the application to personalize user's experiences. Repeat similar setup of both policies.

Home > Azure AD B2C - User flows > Create a user flow > Create

Create

Sign up and sign in v2 (Preview)

New user flows will now use the Ocean Blue template by default instead of the classic template!

B2C_1_* sign_up_in 1

2. Identity providers *

Identity providers are the different types of accounts your users can use to log into your application. You need to select at least one for a valid user flow and you can add more in the identity providers section for your directory. Learn more about identity providers.

Please select at least one identity provider

☒ Microsoft
 ☒ Facebook
 ☐ Email sign-up

2

3. Multifactor authentication

Enabling multifactor authentication (MFA) requires your users to verify their identity with a second factor before allowing them into your application. Learn more about multifactor authentication.

Multifactor authentication Enabled Disabled

4. User attributes and token claims

User attributes are values collected on sign up. Claims are values about the user returned to the application in the token. You can create custom attributes for use in your directory. Learn more about user attributes and claims.

	Collect attribute	Return claim
Given Name	<input type="checkbox"/>	<input type="checkbox"/>
Surname	<input type="checkbox"/>	<input type="checkbox"/>
City	<input type="checkbox"/>	<input type="checkbox"/>
Country/Region	<input type="checkbox"/>	<input type="checkbox"/>
Email Address	<input type="checkbox"/>	<input type="checkbox"/>
Show more...		

Create

Create

4. User attributes and token claims

User attributes are values collected on sign up. Claims are values about the user returned to the application in the token. You can create custom attributes for use in your directory. Learn more about user attributes and claims.

	Collect attribute	Return claim
City	<input type="checkbox"/>	<input type="checkbox"/>
Country/Region	<input type="checkbox"/>	<input type="checkbox"/>
Display Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Email Address	<input type="checkbox"/>	<input type="checkbox"/>
Email Addresses	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Given Name	<input type="checkbox"/>	<input type="checkbox"/>
Identity Provider	<input type="checkbox"/>	<input type="checkbox"/>
Identity Provider Access Token	<input type="checkbox"/>	<input type="checkbox"/>
Job Title	<input type="checkbox"/>	<input type="checkbox"/>
Legal Age Group Classification	<input type="checkbox"/>	<input type="checkbox"/>
Nickname	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Postal Code	<input type="checkbox"/>	<input type="checkbox"/>
State/Province	<input type="checkbox"/>	<input type="checkbox"/>
Street Address	<input type="checkbox"/>	<input type="checkbox"/>
Surname	<input type="checkbox"/>	<input type="checkbox"/>
User is new	<input type="checkbox"/>	<input checked="" type="checkbox"/>
User's Object ID	<input type="checkbox"/>	<input checked="" type="checkbox"/>

OK

3

Steps to setup policies.

[3] Register Apps in Azure AD B2C tenant

Now we need to register an application to this tenant, so that Azure AD knows where to look for identity. App registration is seamless, simply navigate to “**App registrations (Preview)**”, click “+ **New registration**” on top.

Give it a name, and make sure you select “*Accounts in any organizational directory or identity provider. For authenticating users with Azure AD B2C*” under Support account type.

Register an application

* Name

The display name for this application (this can be changed later).

myApp

1

Supported account types

Who can use this application or access this API?

- ☐ Accounts in this organizational directory only (marcustgy only).
- ☐ Accounts in any organizational directory (Any Azure AD directory – Multitenant).
- ☒ Accounts in any organizational directory or any identity provider. For authenticating users with Azure AD B2C.

2

[Help me choose...](#)

Redirect URI (recommended)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web

e.g. https://myapp.com/auth

Permissions

Azure AD B2C requires this app to be consented for openid and offline_access permissions. You must be an app administrator to grant admin consent

- ☒ Grant admin consent to openid and offline_access permissions

By proceeding, you agree to the [Microsoft Platform Policies](#)

Register

Application registration layout.

We will fill in “redirect URL” later after we created the application. Click “Register”, and we are good to go!

[4] Create new Blazor App

Head on to Visual Studio, and create a Blazor App. You can refer here if this is new to you. Select Blazor Server App, and change the authentication to “Individual User Accounts”. Simply select “Connect to an existing user store in the cloud”, which refers to Azure AD B2C in this context. You will see the following:

Create a new Blazor app

Change Authentication

☐ No Authentication

☒ Individual User Accounts

☐ Work or School Accounts

☐ Windows Authentication

Connect to an existing user store in the cloud [Learn more](#)

Select this option to connect to an existing Azure AD B2C application.

Domain Name

Application ID

Callback Path

/signin-oidc

Reply URL: https://localhost:44304/signin-oidc [Copy](#)

Sign-up or Sign-in Policy

Reset Password Policy

Edit Profile Policy

(optional)

[Learn more about third-party open source authentication options](#)

OK Cancel

Authentication

Individual User Accounts

[Change](#)

Advanced

☒ Configure for HTTPS

☒ Enable Docker Support

(Requires [Docker Desktop](#))

Linux

Author: Microsoft

Source: .NET Core 3.1.1

Back Create

Simply enter the field:

Domain Name: <your AAD B2C tenant name>.onmicrosoft.com

Application ID: <retrieve from application registration in steps above>

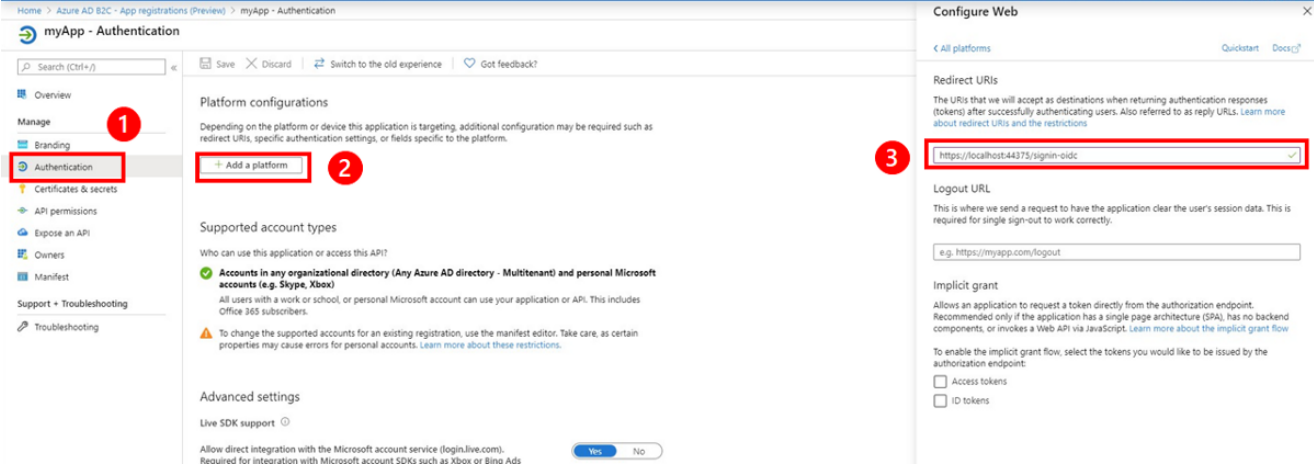
Sign-up or Sign-in Policy: <retrieve from user flow created above. Simply copy the policy name will do. It's in the form of B2C_1_>

Reset Password Policy: <retrieve from user flow created above. Simply copy the policy name will do. It's in the form of B2C_1_>

Edit Profile Policy (Optional): <retrieve from user flow created above. Simply copy the policy name will do. It's in the form of B2C_1_>

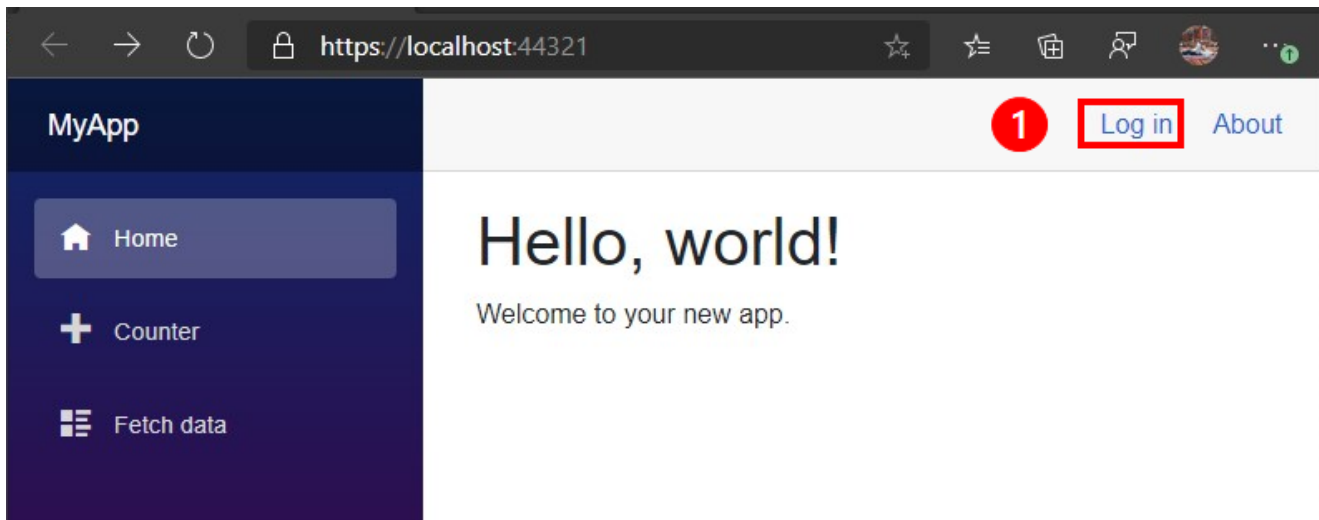
Now, before start creating the application, copy the Callback Path above. We will need to use this URL as application callback URL. Toggle back to Azure AD B2C application registration, include this URL as callback.

Now we are using localhost, in production, remember to update this URL.



Add redirect URLs.

Once done, click create. Here's the magic happening behind the scene. Visual Studio knows that you need authentication in the application, hence created the template which consist of the code to request users to sign in and handle it.



Click on it, it will then re-direct to a nice default UI (by Azure AD B2C) for user to register or sign in. In my example, I configured Microsoft Account, Facebook and email registration. Voila, users can use their social media to access your application!



Sign in with your existing account

[Forgot your password?](#)

Sign in

Don't have an account? [Sign up now](#)

Sign in with your social account

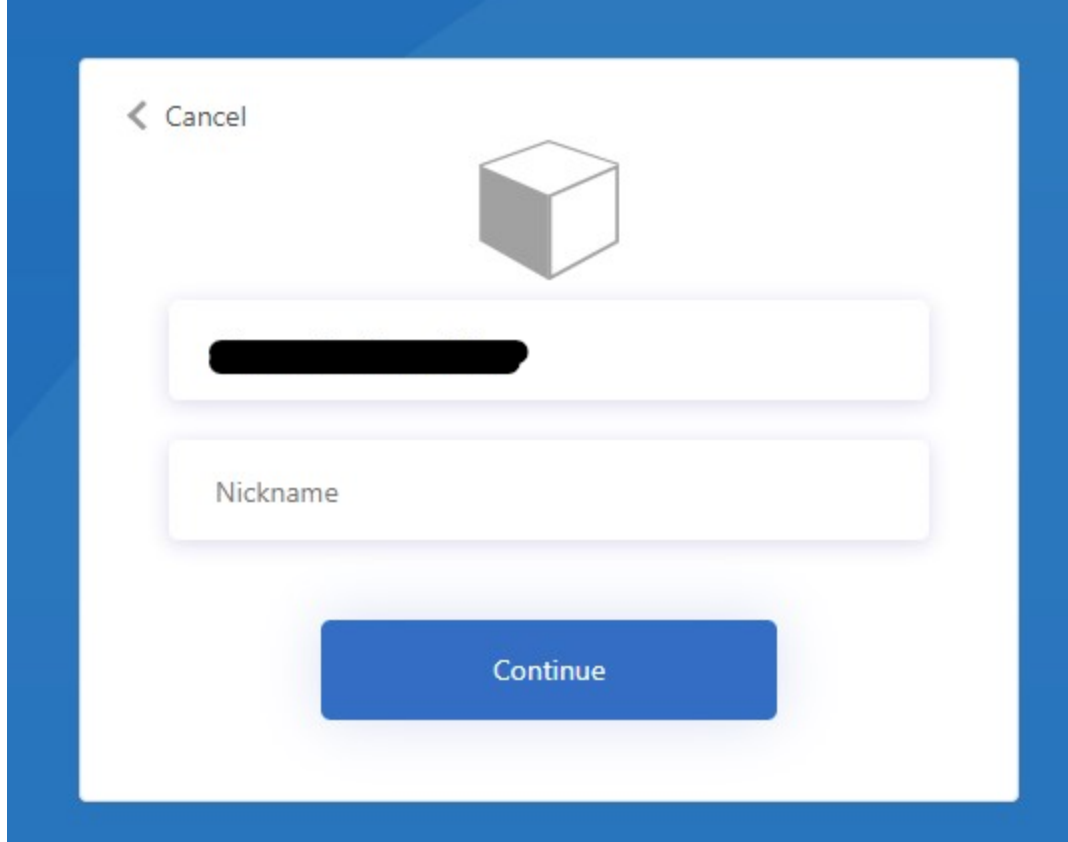


Microsoft



Facebook

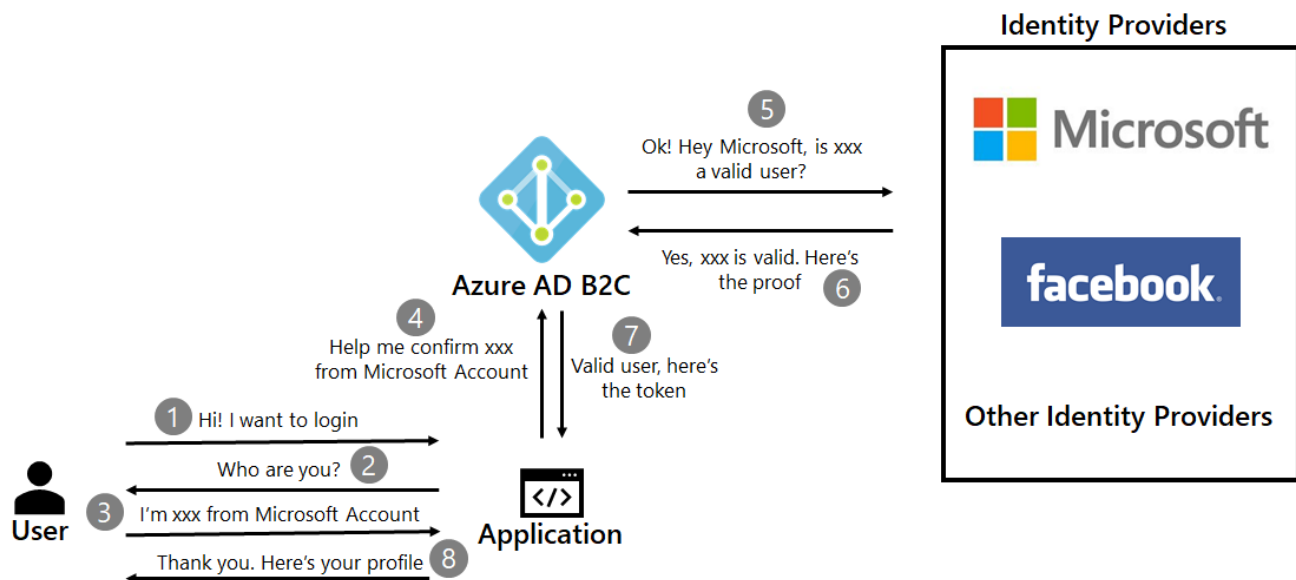
Recalled we included custom user attribute? For first time login, it will then prompt users to provide additional info.



Yes, now you have authentication in your application without writing a single line of code! Developer has the option to customize the sign in page above, which is part of Azure AD B2C. More details [here](#).

Additional Info

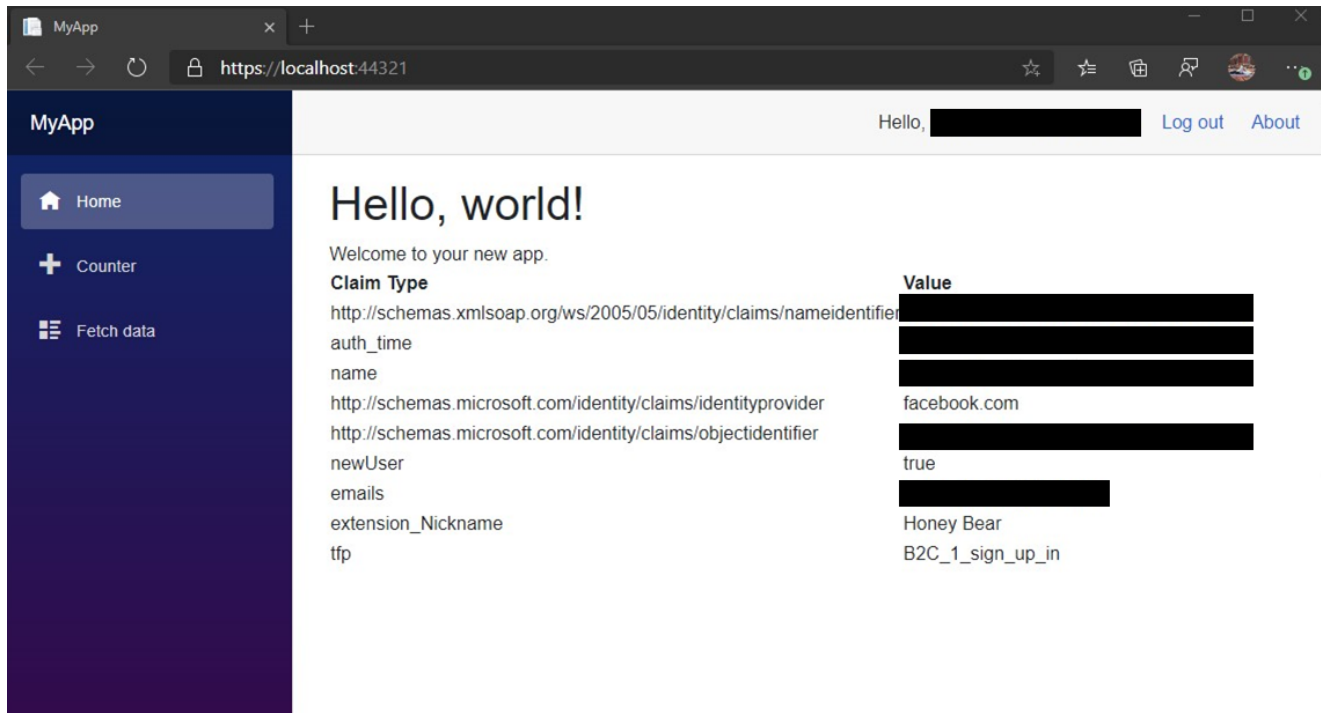
In layman term, Azure AD B2C acts as a broker to validate users for applications. The flow below simplifies authentication flows:



Azure AD B2C will issue claims to the application to understand user context, for instance, pulling information for a particular users. Upon successful login, application can “decipher” claims and retrieve information about this particular user. Recall we configured “return claims” in user policy? This claims will then contains these information in your application.

There are 2 ways to retrieve the claims. First option would be using the authenticated context and list down all information in this claims.

The <AuthorizeView> is only shown when users are authenticated. Here, you can see it iterates all claims and present in a html table. Here's how it looks like.



Second way requires the knowledge of claim type needed, and the info will then be retrievable in the code, and can be stored in a variable for future usage.

There are more advanced authorization capabilities in Blazor, feel free to take reference from here.

That's it! Azure AD B2C greatly reduce the complexity of managing users for both authentication and authorization.

By the way, given the pricing model above, here's the breakdown of my bill:

1. (free tier):
2. (Basic, 5 DTU):
3. :
4. :

Total Bill: ~USD 5 / month. \$5 includes application hosting, serverless processing, database, and even identity provider. That is why cloud gives agility and scalability in cost effective manner.