​Azure AD, B2C and B2B Labs

# Lab 1 Basics

In this Lab you will learn how to create Azure Active Directory, Create your first users, and create App

## Part 1: Creating Azure Active Directory

Creating Azure AD: in this section you will learn how to provision an Azure AD

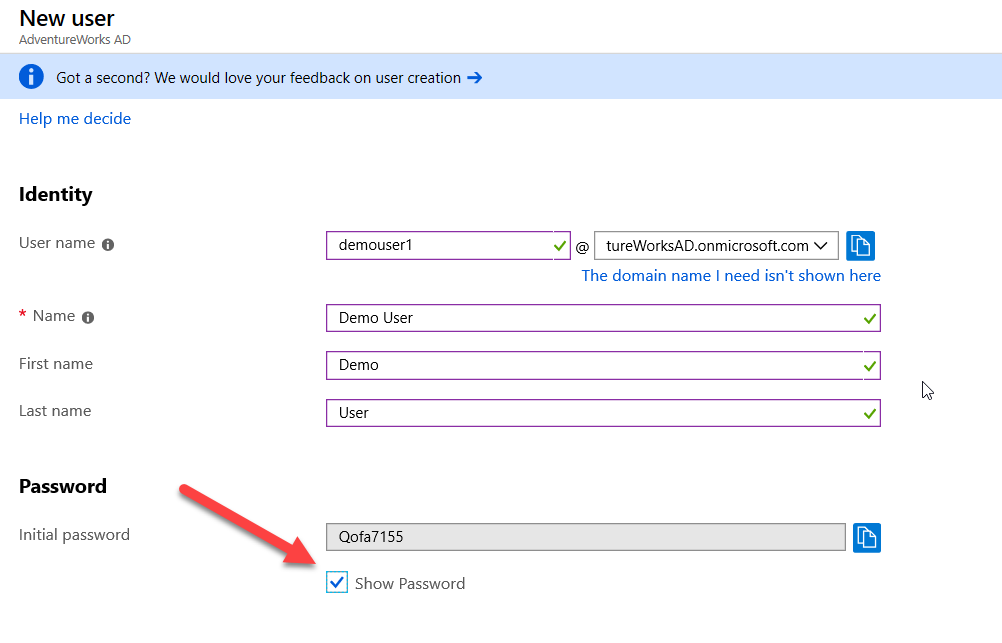
* Navigate to portal.azure.com
* Click on **+ Create a resource** from the left nav
* Search for and Select **Azure Active directory**
* Click on **Create**
* Provide required details as the folllowing
  + Organization Name:  az{your\_id} AD
  + Initial domain name: az{your\_id}AD
  + Country: United Kingdom
* Then click **Create**
* Click on the **here** link to access the created Azure AD  
  Machine generated alternative text:
  Home > New > Azure Active Directory > Create directory 
  Create directory 
  * Organization name O 
  AdventureWorks AD 
  * Initial domain name 
  O 
  AdventureWorksAD 
  AdventureWorksAD.onmicrosoft.com 
  Country or region 
  o 
  United Kingdom 
  Click here to manage your new directory 

Notice that the name of your Azure AD tenant matches the name you provided during the creation.

## Part 2: Creating your first user

**Creating you first user:** In this section you will practice creating a user, and then logging in to portal.azure.com using the user you have just created

* From your Azure AD tenant's blade:
* Click on **Users** link from the left nav
* Click on **+ New user** in the top menu

* Fill user information requested:
  + Username:  az{your\_id}-demouser1
  + Name: az{your\_id}-demouser1
  + First name:  az{your\_id}-demo
  + Last name: User
* Check the show password checkbox.. And note down the password.  
  
* Click on **Create**
* Now try to login to portal.azure.com using the user you just created (make sure you login using an InPrivate window).

## Part 3: Registering your first app

In this section you will practice registering an application with Azure AD.

* Click on **App registrations** from the left hand-menu of the Azure AD tenant's blade.
* Click on **+ New registration**
* Fill the requested details
  + Name: az{your\_id}-app1
  + Select supported account types: Accounts in this organizational directory only ({your\_tenant} only - Single tenant)
  + Redirect URL: For dropdown select Web and in text box provide: <https://localhost:44376/signin-oidc>
* Click **Register**

This will create an app registration in your Azure AD directory tenant.

Make note of the "Application (client) ID" and "Directory (tenant) ID", as you will utilize these in the next lab.

# Lab 2: Application Scenarios

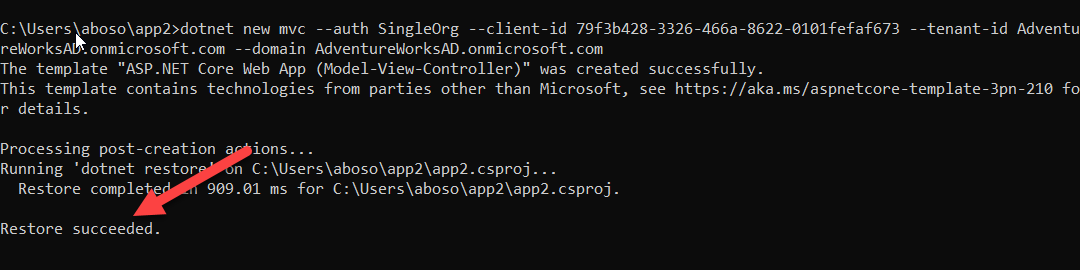
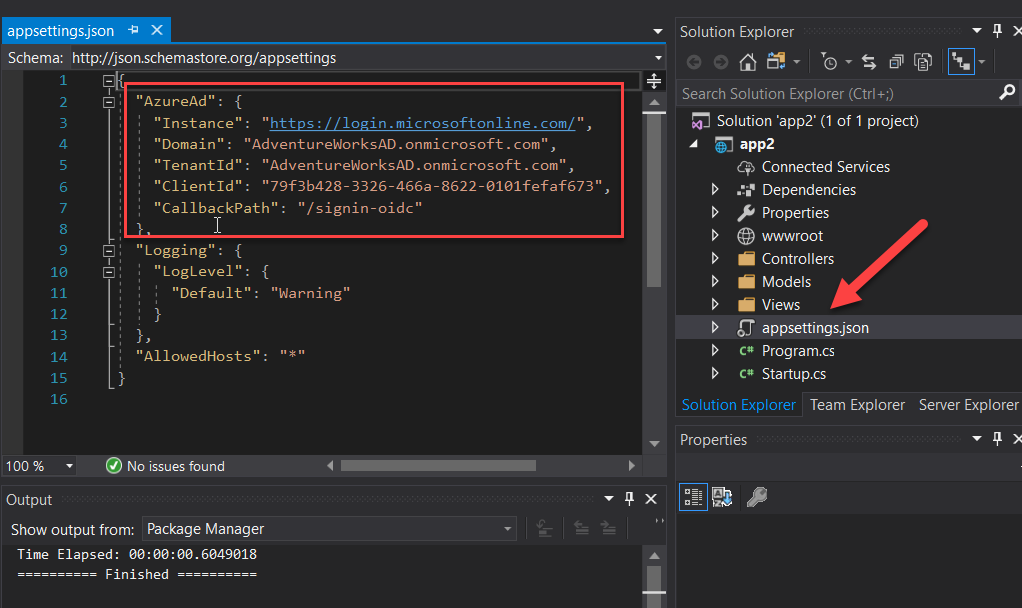
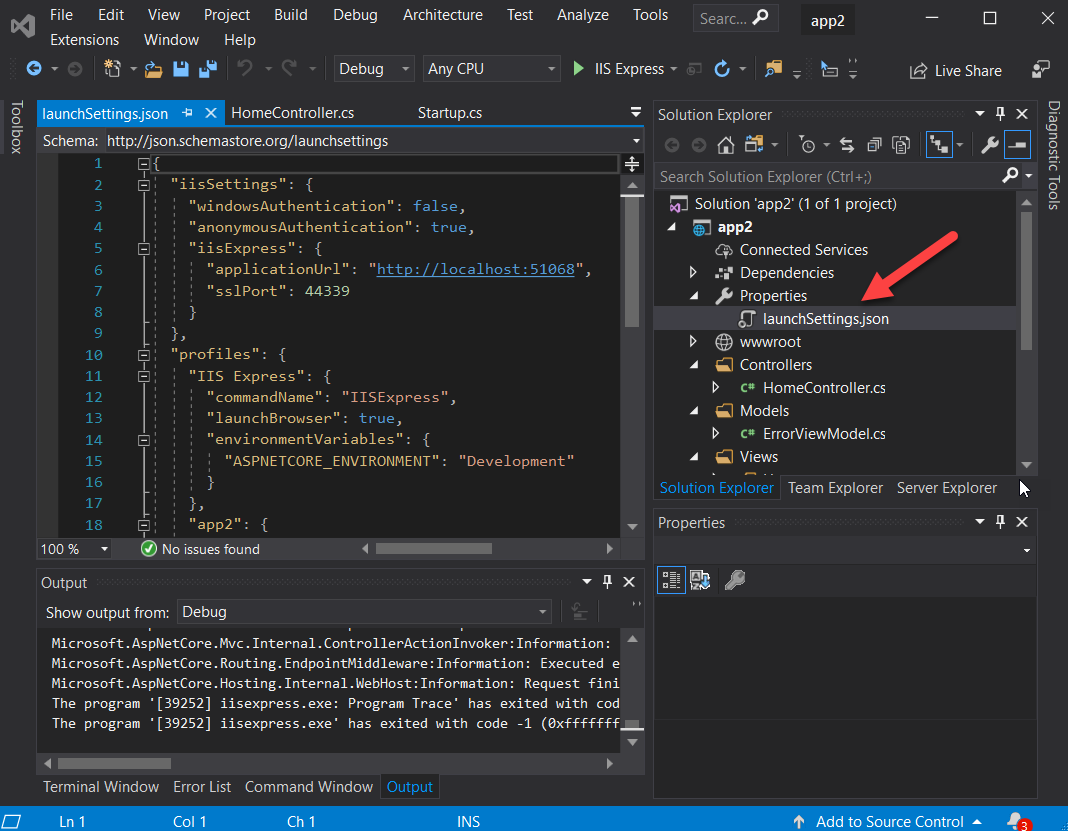
In this lab you will learn how Applications use the various authentication flows to sign in users and get tokens to call protected APIs.

Before proceeding, we will need to setup our app registration to allow it to supply ID Tokens. In the portal, navigate to your app registration in AAD

* In the left-hand menu select **Manage -> Authentication**
* Scroll down and ensure the **Implicit grant** - **ID tokens** check box is checked.
* Click **Save**.

## Part 1: WebApp - User Login Using Azure AD

In this scenario, the web application directs the user’s browser to sign them in to Azure AD. Azure AD returns a sign-in response through the user’s browser, which contains claims about the user in a security token. Signing-in users leverage the Open ID Connect standard protocol itself simplified by the use of middleware libraries.

* Open Windows Command Prompt
* Create Folder for your App
  + Type md az{your\_id}-app1 and hit Enter
  + Type cd az{your\_id}-app1 and hit Enter
* Type the following command (replace place holder with the values from your app registration created in Lab 1 - Part 3.)
  + dotnet new mvc --auth SingleOrg --client-id {client-id for the app you just registered} --tenant-id {yourtenantID} --domain {domain}
  + **Example:**dotnet new mvc --auth SingleOrg --client-id 79f3b428-3326-466a-8622-0101fefaf673 --tenant-id AdventureWorksAD.onmicrosoft.com --domain AdventureWorksAD.onmicrosoft.com
* Notice the domain is the name of your tenant, and will be in the form: {name}.onmicrosoft.com
* Make sure you see message Restore succeeded  
  
* Open the created project using Visual Studio
* Open **appsettings.json** and verify the values for Domain, TenantId, ClientId, and CallbackPath are matching the values you provided earlier  
  
* Open **Properties** -> **launchSettings.json**  
  In the Azure portal, the reply URIs that you need to register in the Authentication page for your application needs to match these URLs; that is, for the two configuration files above, they would be <https://localhost:44321/signin-oidc> as the applicationUrl is <http://localhost:3110> but the sslPort is specified (44321), and the CallbackPath is /signin-oidc as defined in the **appsettings.json**.
* In the same way, the sign-out URI would be set to <https://localhost:44321/signout-callback-oidc>   
  
* Open **Startup.cs** and notice the code facilitating using Azure AD OpenID Connect authentication  
  vis:

public void ConfigureServices(IServiceCollection services)

{

services.Configure<CookiePolicyOptions>(options =>

{

// This lambda determines whether user consent for non-essential cookies is needed for a given request.

options.CheckConsentNeeded = context => true;

options.MinimumSameSitePolicy = SameSiteMode.None;

});

services.AddAuthentication(AzureADDefaults.AuthenticationScheme)

.AddAzureAD(options => Configuration.Bind("AzureAd", options));

services.AddMvc(options =>

{

var policy = new AuthorizationPolicyBuilder()

.RequireAuthenticatedUser()

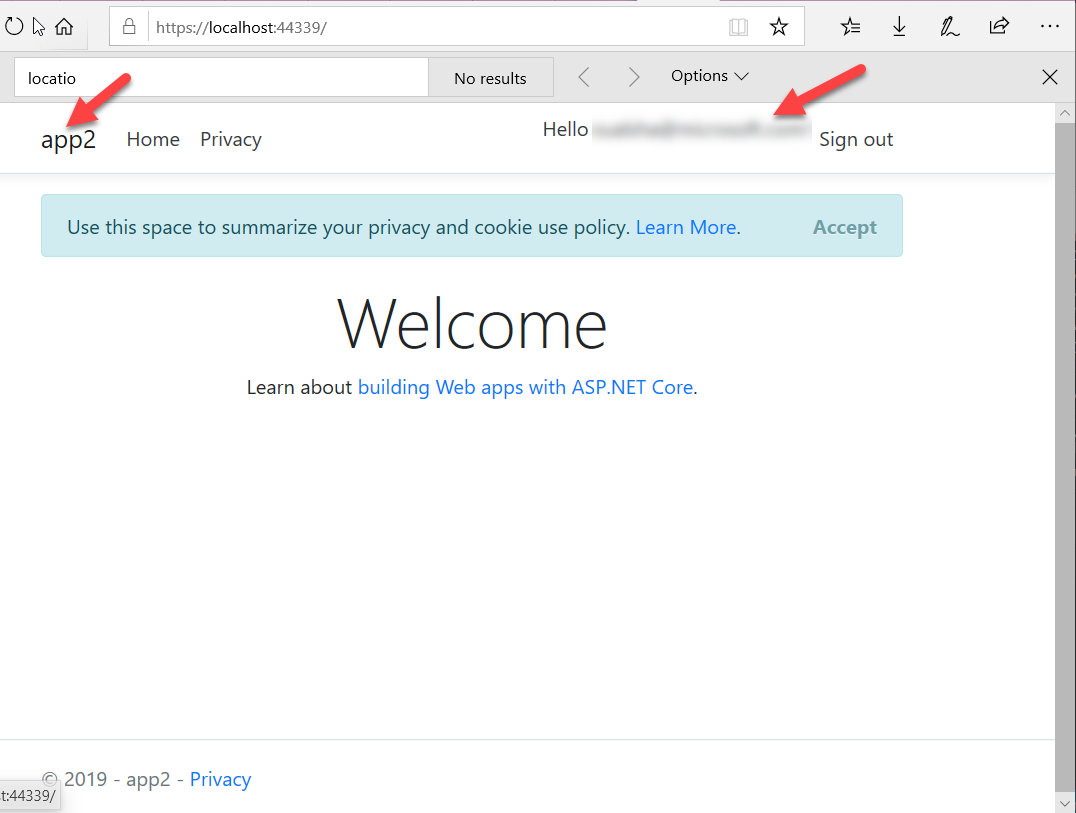
.Build();

options.Filters.Add(new AuthorizeFilter(policy));

})

.SetCompatibilityVersion(CompatibilityVersion.Version\_2\_2);

}

* Run the application by hitting the Run button (or F5)
* In the case you are automatically signed-in, copy the application URL and open a New InPrivate window to experience the login procedure.
* Notice the sign-in user and app name.  
  

Once signed in you have successfully created a WebApp that facilitate user login using Azure AD.

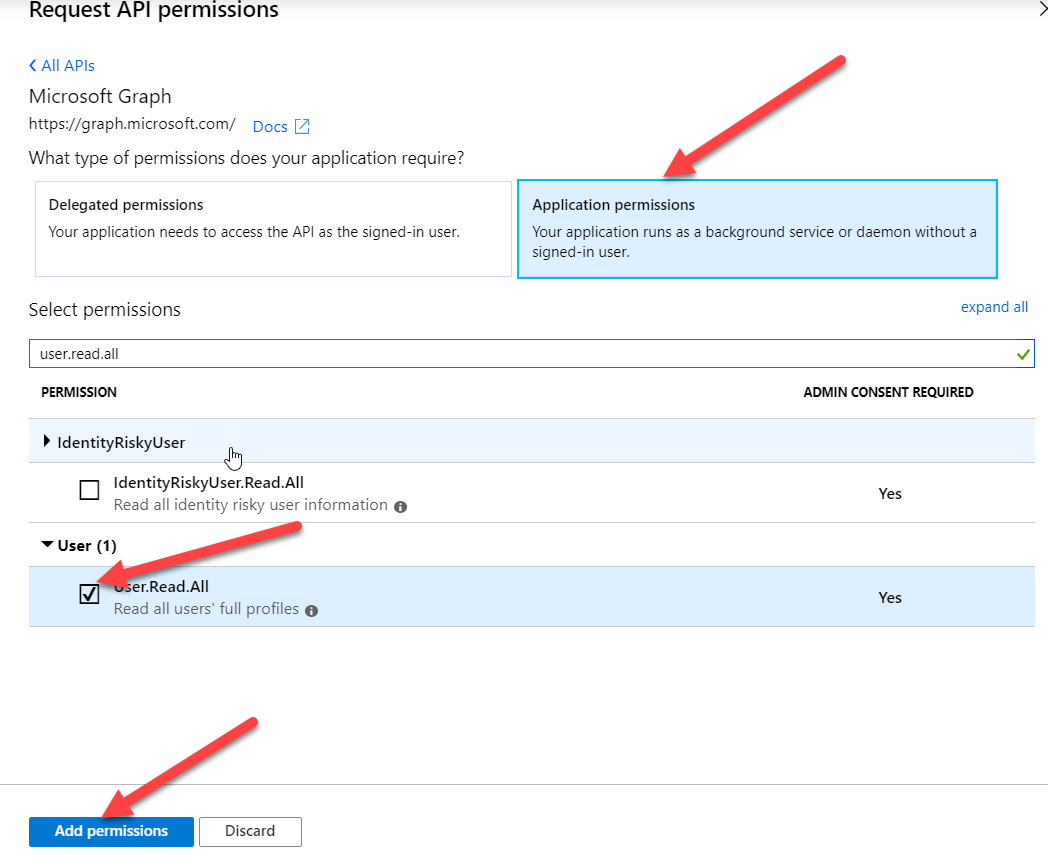
## Part 2: Using Certificates

In this scenario, you will use a certificate to authenticate your app in order to call graph API

### Step:1 Registering and Application

* Navigate to your Azure AD Tenant page
* Click on **App registrations**
* Click on **New registration**
* Fill the requested details
  + Name: az{your\_id}-graphapp1
  + Select supported account types: Accounts in this organizational directory only ({you tenant} only - Single tenant)
  + Redirect URI: <http://localhost:1234>  
    (The redirect URI will not be used in this example)
* Click **Create**

### Step 2: Setup API Permissions

* From the left-hand menu select **API permissions**, and click on **+ Add a permission**
* Select **Microsoft Graph**
* Select **Application permissions**
* Under search type: user.read.all
* From under select permissions, tick the check box beside User.Read.All
* Click on **Add permissions**  
  
* Click on **Grant admin consent** button and confirm the promptMachine generated alternative text:
  API permissions 
  Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. 
  all the permissions the application needs. 
  + Add a permission 
  API / PERMISSIONS NAME 
  Microsoft Graph (2) 
  User. Read 
  User. Read .AII 
  Delegated 
  Application 
  DESCRIPTION 
  Sign in and read user profile 
  Read all users' full profiles 
  ADMIN ct 
  Yes 
  These are the permissions that this application requests statically. You may also request user consent- 
  able permissions dynamically through code. See best practices for requesting permissions 
  Grant consent 
  These permissions have been grant 
  or undefined but aren't in the configured permissions list. If your application require 
  permissions, you should conside 
  ding them to the configured permissions list. 
  Grant admin consent for AdventureWorks AD 

### Step 3: Register a Certificate

*For more details on the PS commands used below follow this link* **(**<https://blogs.msdn.microsoft.com/kaevans/2016/08/12/using-powershell-with-certificates/>**)**

* First, we will need to create a self-signed certificate. Copy and paste the following PowerShell command in a PowerShell windows that is run as Administrator:

$cert=New-SelfSignedCertificate -Subject "CN=GraphApp1Cert" -CertStoreLocation "Cert:\CurrentUser\my" -KeyExportPolicy Exportable -KeySpec Signature

* Use the following PowerShell commands to export the certificate

$bin = $cert.RawData

$base64Value = [System.Convert]::ToBase64String($bin)

$bin = $cert.GetCertHash()

$base64Thumbprint = [System.Convert]::ToBase64String($bin)

$keyid = [System.Guid]::NewGuid().ToString()

$jsonObj = @{customKeyIdentifier=$base64Thumbprint;keyId=$keyid;type="AsymmetricX509Cert";usage="Verify";value=$base64Value}

$keyCredentials=ConvertTo-Json @($jsonObj) | Out-File "creds.txt"

* Open the exported creds.txt file and copy its content

Now that we have a certificate, we can import this into our AAD App registration to be used for authentication.

* Now from the left menu click on **Manifest** and paste the content as value for"KeyCredentials" and then click save

Machine generated alternative text:
p Search resources, services, and docs (G+/) 
Home > AdventureWorks AD - App registrations > graphappl - Manifest 
graphappl - Manifest 
p Search (Ctrl +7) 
Overview 
Quickstart 
Manage 
Branding 
Authentication 
Certificates & secrets 
API permissions 
Expose an API 
Save X Discard Upload Download 
The editor below allows you to update this application by directly modifying its JSON re 
app_ljgggion mepifgs!) 
Owners 
Roles an 
Manifest 
Inistrators (Previ... 
Support + Troubleshooting 
Troubleshooting 
New support request 
1 
2 
3 
4 
5 
6 
7 
8 
9 
10 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
23 
24 
25 
26 
"id". 6eb69d9f-b49b-4f6a-a2ff-47cae36ea233" , 
" acceptMappedC1aims" : 
null, 
" accessTokenAcceptedVersion" . 
null, 
" addlns 
"allowpublicclient": null, 
"appld • 
" appR01es" : 
"oauth2A110wUr1pathMatching": false, 
" createdDateTime • 
" groupMembershipC1aims " : 
null, 
"identifieruris 
" informationalurls" : 
"termsOfService": null, 
"support": null, 
"privacy": null, 
"marketing": null 
"keycredentials" : 
" knownC1ientApp1ications 
" logoUr1 " : 
null, 
" logoutUr1 " : 
null, 
" name" : 
"graphappl" , 
"oauth2A110wIdTokenImp1icitF10w": false, 
"oauth2A110wImp1icitF10w": false, 
"oauth2Permissions" : 

* Copy cmdApp file to your machine and extract it
* Open cmdApp solution
* Update the following key in app.config (replacing your tenant and client IDs)

<add key="ida:AAD" value="https://login.microsoftonline.com/{0}"/>

<add key="ida:Tenant" value="{your\_tenant}"/>

<add key="ida:ClientId" value="{your\_clientid}"/>

<add key="ida:AudienceUri" value="https://graph.microsoft.com"/>

<add key="ida:ReplyUrl" value="http://localhost:1234"/>

<add key="ida:CertificateName" value="CN=GraphApp1Cert"/>

* Set breakpoint at the end of the main function
* Run the app and check the result in the command window

 In this example the user's details are retrieved using the Microsoft Graph API. The certificate is retrieved from the

### Going further:

You can see the self-signed certificate generated by searching for "Manage user certificates" in Control Panel. The certificate is located under **Certificate – Current User** -> **Personal** -> **Certificates**, and is issued to **GraphApp1Cert**.

# Lab 3: Azure AD B2C Authentication

## Part 1: Creating Azure AD B2C Tenant

Creating Azure AD B2C: in this section you will learn how to provision an Azure AD B2C

* Navigate to portal.azure.com
* Click on **+ Create a resource** from the left nav
* Search for **Azure Active Directory B2C** and select it.
* Click on **Create** button
* Select "**Create a new Azure B2C Tenant** "  
  Machine generated alternative text:
  Report a bug 
  Home > New > Azure Active Directory B2C > 
  p Search resources, services, and docs (G +7) 
  ate new 82C Tenant or Link to existing Tenant 
  Create new B2C Tenant or Lin 
  existing Tenant 
  Create a new Azure AD B2C Tenant O 
  Link an existing Azure AD B2C Tenant to my Azure subscription O #
* Provide requested details as following
  + Organization name:  az{your\_id} B2C AD
  + Initial domain name: az{your\_id}b2cad
  + Country or region: United Kingdom
* Then click **Create**
* Click on the **here** link to manage your new directory

Notice the message "No Subscription linked to this B2C tenant or the Subscription needs your attention".

Machine generated alternative text:
Home > Azure AD B2C 
AD B2C 
AdventureWorksADB2C.onmicrosoft.com 
Overview 
Manage 
Applications 
Identity providers 
User attributes 
Users 
Roles and administrators 
Policies 
User flows (policies) 
Identity Experience Framework 
Troubleshoot 
No Subscription linked to this 82C tenant or the Subscription needs your attention. 
Domain name : AdventureWorksADB2C.onmicrosoft.com 
Metered 
: Yes 
Tenar 
Subs( 
Welcome to Azure Active Directory B2C 
1 
Register an application 
The application registration is used to 
secure your directory by allowing only your 
applications to make requests and to make 
sure your users are sent to a trusted place 
after signing in. Get started 
2 
Add identity prc 
Identity providers are 
accounts your users c 
your application. Get 

We will now link this B2C tenant to an existing Azure subscription.

Switch Directories to the location of your target Azure subscription

* Click on **+ Create a resource** from the left nav
* Search for **Azure Active Directory B2C** and select it.
* Click on **Create** button
* Select Link and existing Azure AD B2C Tenant to my Azure Subscription  
  Machine generated alternative text:
  Report a bug 
  p Search resources, services, and docs (G +7) 
  Home > New > Azure Active Directory B2C > 
  Create new 82C Tenant or Link to existing Tenant 
  Create new B2C Tenant or Link to existing Tenant 
  Create a new Azure AD B2 
  nant 
  Link an existing Azure AD B2C Tenant to my Azure subscription O 
* Provide the requested details
  + Azure AD B2C Tenant: pick the B2C tenant you have just created
  + Subscription: Pick you Azure Subscription
  + Resource group: Create new resource group, az{your\_id}-AAD-rg
  + Resource group location: UK west

After the subscription is linked, navigate to the B2C tenant you have just created, you will notice that the subscription is now linked.

## Part 2: Set up sign-up and sign-in with a LinkedIn account using Azure Active Directory B2C

### Step 1: Create a LinkedIn application

To use a LinkedIn account as an identity provider in Azure Active Directory B2C (Azure AD B2C), you need to create an application in your tenant that represents it. If you don't already have a LinkedIn account, you can sign up at <https://www.linkedin.com/>.

* Sign in to the [LinkedIn Developers website](https://www.developer.linkedin.com/) with your LinkedIn account credentials.
* Click **Create App**
  + Enter App Name: az{your\_id}-myLinkedInApp-B2C
  + Company: az{your\_id}-Company-B2C (Create a company using your own login)
  + Application Logo: use the attached image as logo (provide path)
  + Business Email: your business email
* Agree to the LinkedIn API Terms of Use and click **Submit**.
* Copy the values of **Client ID** and **Client Secret**. You can find them under **Auth** tab. You will need both of them to configure LinkedIn as an identity provider in your tenant. Client Secret is an important security credential.  
  Machine generated alternative text:
  B2C 
  Settings 
  AdventureWorks-AADB2C-App 
  Client ID: 86q7yzh3fk2 
  Created: 23 sep, 2019 
  Auth 
  Products 
  Usage & limits 
  Team members 
  Application credentials 
  Client ID: 
  86q7yzh3fk26jq 
  Client Secret: 
* Add a new Redirect URL:  
  https://{your\_tenant\_name}.b2clogin.com/{your\_tenant\_name}.onmicrosoft.com/oauth2/authresp  
  in **Authorized Redirect URLs**. Replace {your\_tenant\_name} with the name of your tenant. You need to use all lowercase letters when entering your tenant name even if the tenant is defined with uppercase letters in Azure AD B2C. Select **Add**, and then click **Update**.  
  Machine generated alternative text:
  Permissions 
  No permissions added 
  OAuth 2.0 settings 
  Redirect URLs: 
  No redirect URLs added 

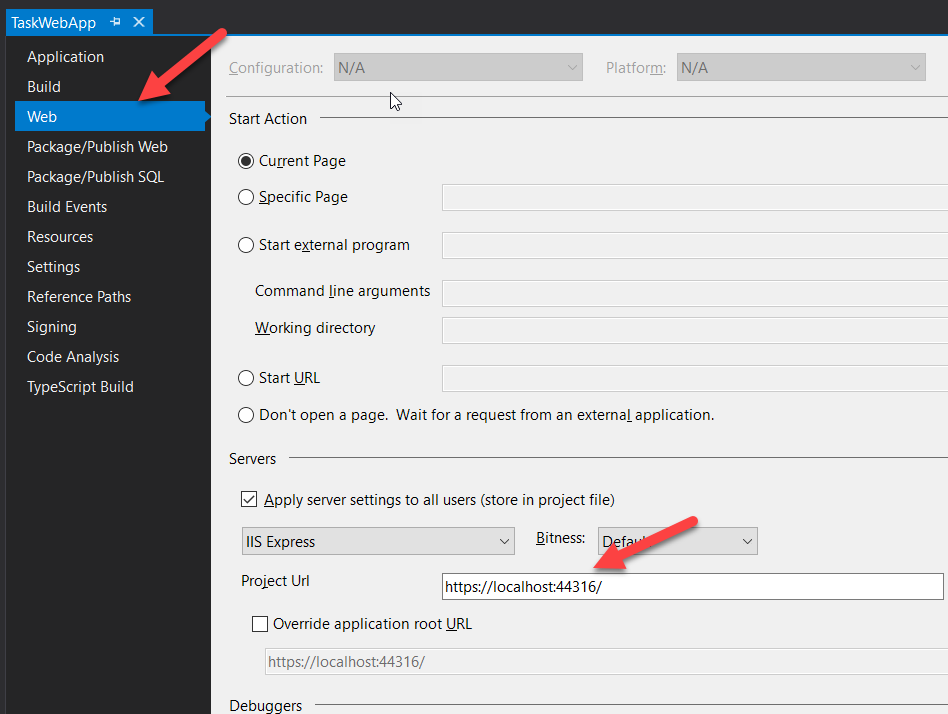
### Step 2: Configure a LinkedIn account as an identity provider

* In the Azure portal:
* Make sure you're using the directory that contains your Azure AD B2C tenant by selecting the **Directory + subscription** filter in the top menu and choosing the directory that contains your tenant.
* Choose **All services** in the top-left corner of the Azure portal, search for and select **Azure AD B2C**.
* Select **Identity providers**, then select **LinkedIn**.  
  Machine generated alternative text:
  Microsoft Azure 
  Create a resource 
  Dashboard 
  — All services 
  FAVORITES 
  All resources 
  t) Resource groups 
  App Services 
  o 
  Function App 
  SQL databases 
  Azure Cosmos DB 
  Virtual machines 
  Load bal 
  ancers 
  Storage accounts 
  Virtual networks 
  Azure Active Directory 
  Monitor 
  visor 
  Security Center 
  st Management + Bill... 
  Help + support 
  Home > Azure AD oc 
  Azure AD B2C 
  p Search (Ctrl +7) 
  Overview 
  Manage 
  Applications 
  Identity providers 
  user attributes 
  Users 
  Roles and administrators 
  Policies 
  User flows (policies) 
  Identity Experience Framework 
  Security 
  O Authentication methods (Previe... 
  Activities 
  Audit logs 
  p Search resources, services, and docs (G+/) 
  Troubleshoot 
  Domain name 
  Metered 
  Subscription ID 
  : AdventureWorksADB2C.onmicrosoft.com 
  : Yes 
  • be5282a4-2cfe-47d1-a1e6-8f3f862d5874 
  Welcome to Azure Active Directory B4C 
  1 
  3 
  Register an application 
  The application registration is used to 
  secure your directory by allowing only your 
  applications to make requests and to make 
  sure your users are sent to a trusted place 
  after signing in. Get started 
  Create a user flow 
  User flows define the experience for your 
  users signing up and signing into your 
  application. Get started 
  2 
* Enter a **Name**. For example, *LinkedIn*.
* For the **Client ID**, enter the Client ID of the LinkedIn application that you created earlier.
* For the **Client secret**, enter the Client Secret that you recorded.
* Select **Save**

## Part 3: Web App using AADB2C

In this section you will learn how to create a WebApp using LinkedIn for sign-up and sign-in.

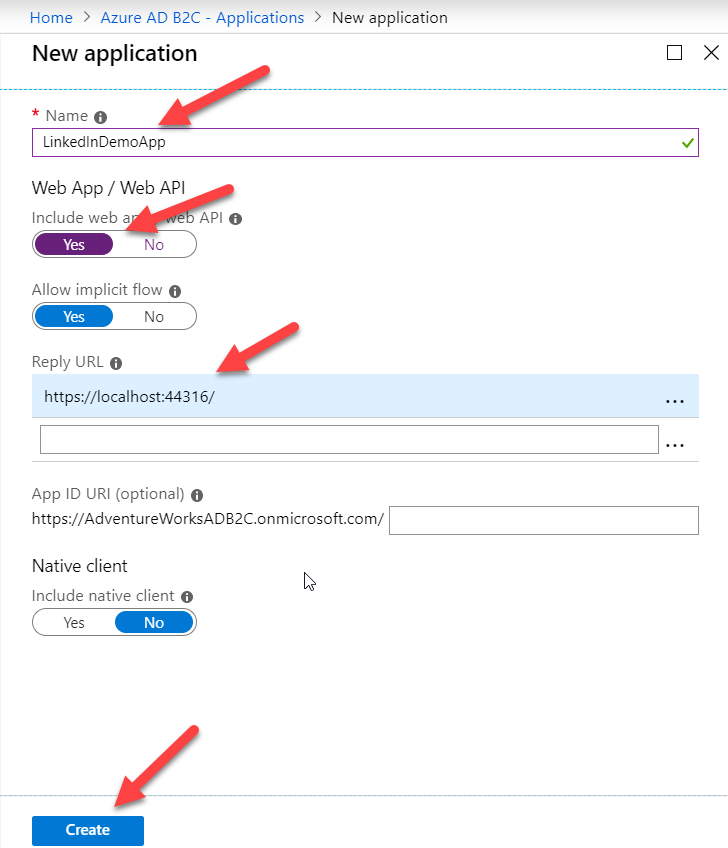
### Download the web application

* Copy provided project or clone app using git clone git clone <https://github.com/azure-samples/active-directory-b2c-dotnet-webapp-and-webapi.git>
* Open the solution using Visual Studio
* We will be working with the **TaskWebApp**
* Right click the **TaskWebApp** project and click properties
* Click on the Web Tab and capture Project Url, we will need this when registering the App  
  

### Create App Registration

In the Azure portal, in your Azure AD B2C tenant:

* Select **Manage** -> **Applications** tab and select **+ Add**
* Provide values for the following fields
  + Name: az{your\_id}-LinkedInB2CDemo
  + Web App/ Web API: Select Yes
  + Reply Url: provide the Project Url value you captured earlier (i.e. https://localhost:44316)
* Leave the rest as-is and click **Create**



### Create Sign up and sign in user flows(policies)

In the Azure portal, in the Azure B2C tenant's blade:

* Click on **Policies** -> **User flows (policies)** and select **+ New user flow**
* On the Recommended tab, select the Sign up and sign in user flow.
* Provide the following values
* Name: SignupSignin
* Select **LinkedIn** and **Email Sign Up** from under Identity providers
* Under "User attributes and claims", select the following (both for **Collect Attribute** and **Return Claim**)
  + Given Name
  + Surname
  + City
* Then click on **Show more**... and take a look at the available attributes. Pick **Job Title**
* Click on **Create** to add the user flow. (A prefix of B2C\_1 is automatically prepended to the name)

**Update Web App configurations**, namely in Web.Config (Client ID, Client Secret, Tenant ID, and User flows, below you will find steps on getting the Client ID and Client Secret Values.. Please refer to earlier labs for the values for the signup and sign in Userflows name and tenant ID

In the Azure portal, in the Azure B2C tenant's blade:

* Open the created App registration and capture the **Application ID**/ Client ID
* Now to create a client Secret, in the left-hand menu select **Keys** and
* Click on **+ Generate Key**.
* Click on **Save**, a secret will be displayed. Make a note of this secret  
  (Note: if the secret contains a double quote (") then generate a new key)

Insert these values in to the web.config file. In Visual Studio:

* Expand **TaskWebApp** and open the Web.Config file.
* Update the following key values

<add key="ida:Tenant" value="{your\_b2c\_tenant}.onmicrosoft.com" />

<add key="ida:TenantId" value="{your\_b2c\_tenant\_id}" />

<add key="ida:ClientId" value="{application id}" />

<add key="ida:ClientSecret" value="{client secret}" />

<add key="ida:AadInstance"  
 value="https://{your\_b2c\_tenant}.b2clogin.com/tfp/{0}/{1}" />

<add key="ida:RedirectUri" value="https://localhost:44316/" />

<add key="ida:SignUpSignInPolicyId" value=" B2C\_1\_SignupSignin" />

* Save your changes and run the application
* Click on **Sign-up / Sign In**

Notice now that you will be presented with the two identity providers which you selected when created the Signup Sign in User flow. Click on **Linked In**

* You can see now that you are presented with LinkedIn sign in page, please sign in using your Linkedin account
* The LinkedIn app we created in previous steps will ask for your consent, click Allow
* You will also be requested the fill/confirm the attributes you have selected when creating the Sign Up and Sign In flow, provide the requested and click continue

Notice now that you are signed in, i.e. you have the option to sign out now.

* Now click on Claims and review the claims
* Sign out and try the email sign up flow