REALDOLMEN
a Gfi Group company

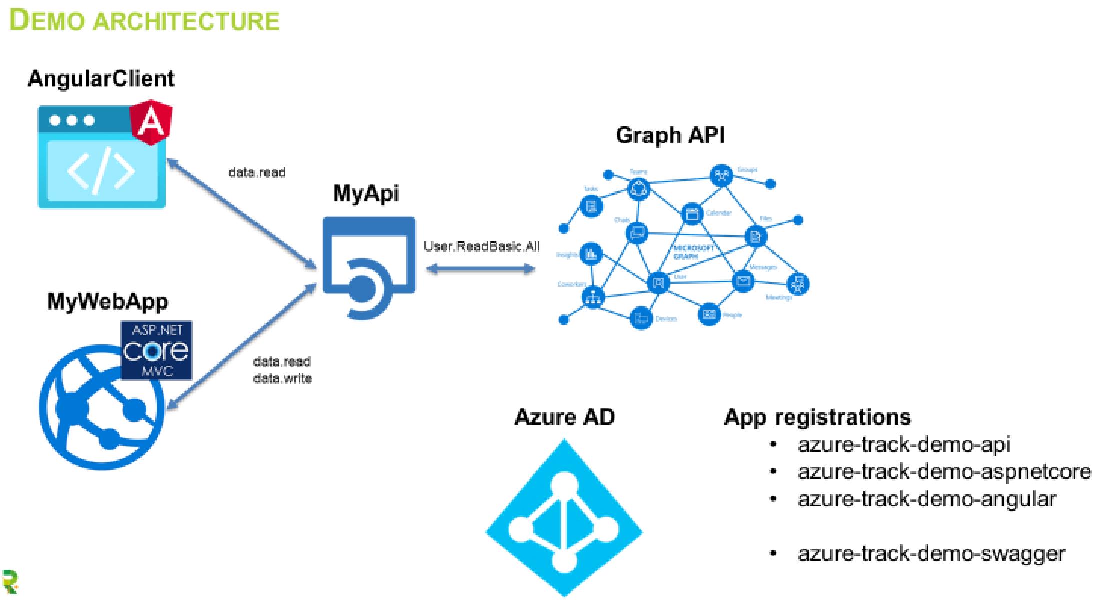
AZURE TRACK

Azure AD Demo

Author: Alexander Reynaert
Date: 25/11/2020
Version: 1.0

Table of contents

1. Architecture



DEMO ARCHITECTURE

App registrations

- azure-track-demo-api
- azure-track-demo-aspnetcore
- azure-track-demo-angular
- azure-track-demo-swagger

- For the API (MyApi) we define 2 scope: data.read & data.write
- The Angular Client have only permission to request the data.read scope
- The Web App (ASP.NET Core) app have permission to request the data.read & data.write scope
- Our API can access the Graph API and have permission to request the User.ReadBasic.All scope from the Graph API.

In our API we check if the user has a scope with the following code:
HttpContext.VerifyUserHasAnyAcceptedScope(ScopesRequiredByApiForWriteData);

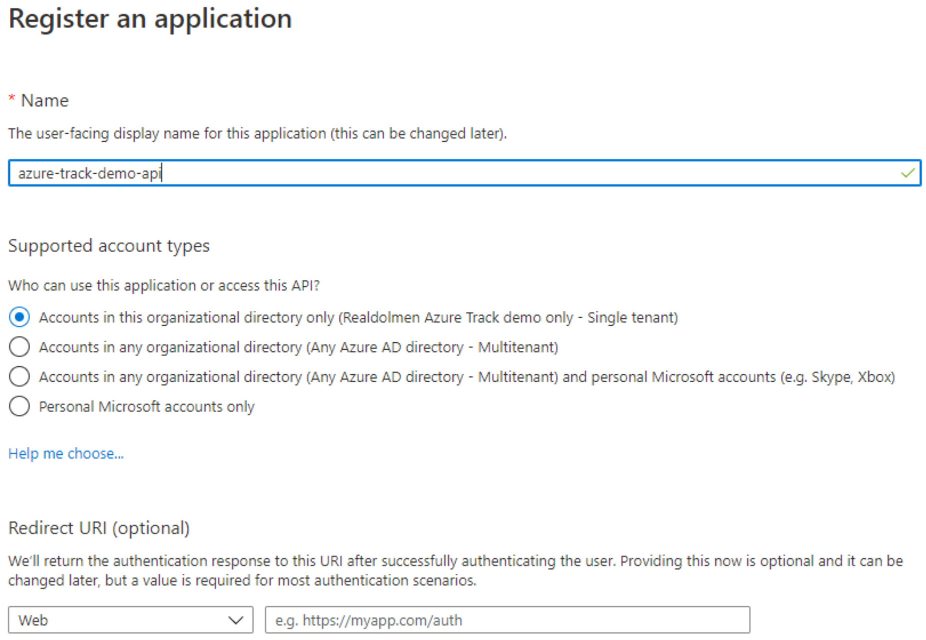
2. Register applications in Azure AD

2.1 azure-track-demo-api

For our API (MyAPI) we register an application with the following data:

- Name: azure-track-demo-api
- Scopes:
  - api://{client}/data.read
  - api://{client}/data.write
- No redirect uri

Step 1: Name & Redirect URI



Register an application

* Name

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

azure-track-demo-api ✓

Supported account types

Who can use this application or access this API?

● Accounts in this organizational directory only (Realdolmen Azure Track demo only - Single tenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

○ Personal Microsoft accounts only

Help me choose...

Redirect URI (optional)

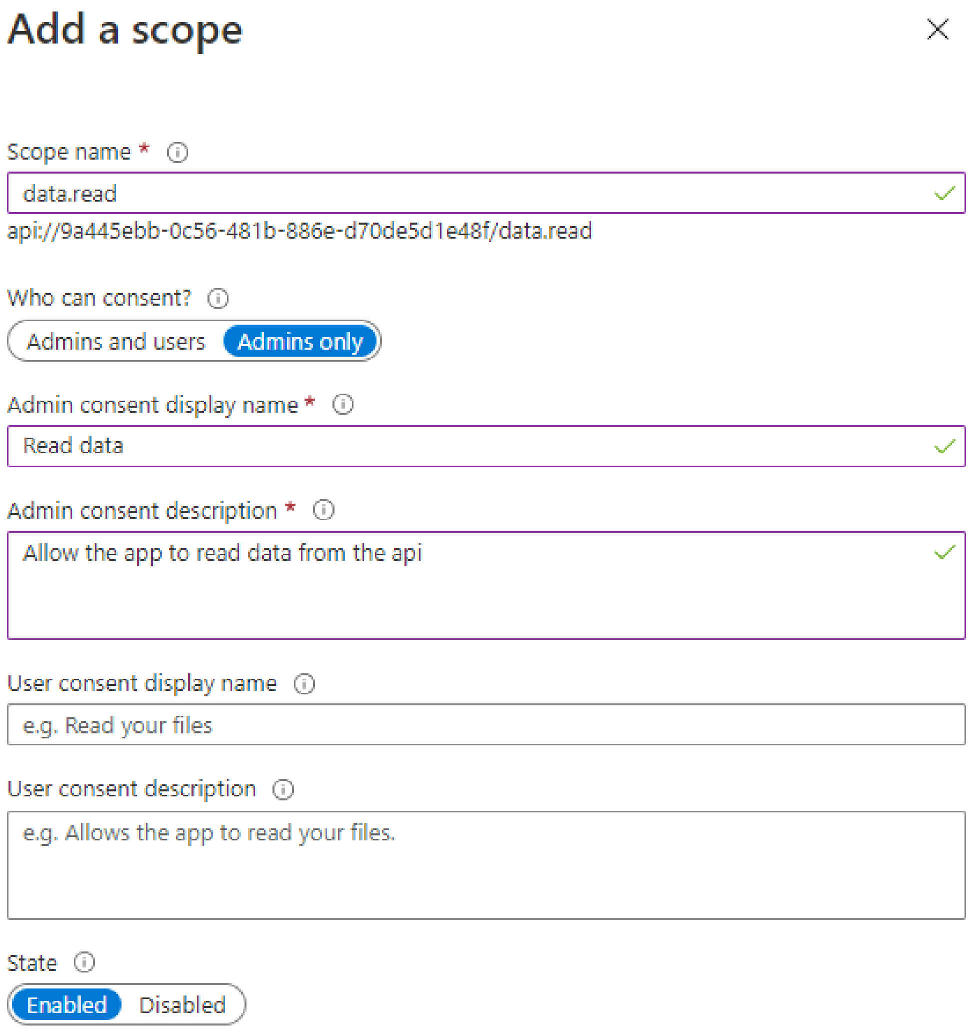
We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web

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

Step 2: goto ‘Expose an API’:

- Save your application ID uri (default is ok)
- Add the 2 scopes (data.read & data.write)
  (choose for ‘Who can consent?’ Admins only, in this case an ‘AD admin’ should give a consent for all users before the scope can be used for an application), you can specify that users give the consent for themselves (useful for personal data).



Add a scope

Scope name *
data.read
api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.read

Who can consent?
○ Admins and users [✓ Admins only]

Admin consent display name *
Read data

Admin consent description *
Allow the app to read data from the api

User consent display name
e.g. Read your files

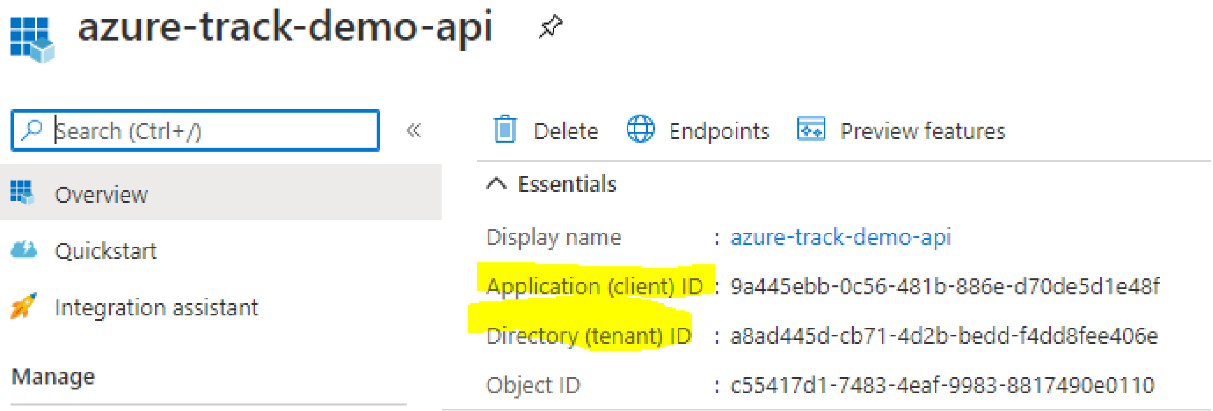
User consent description
e.g. Allows the app to read your files.

State
[✓ Enabled] ○ Disabled

Step 3: the code, open the MyApi project

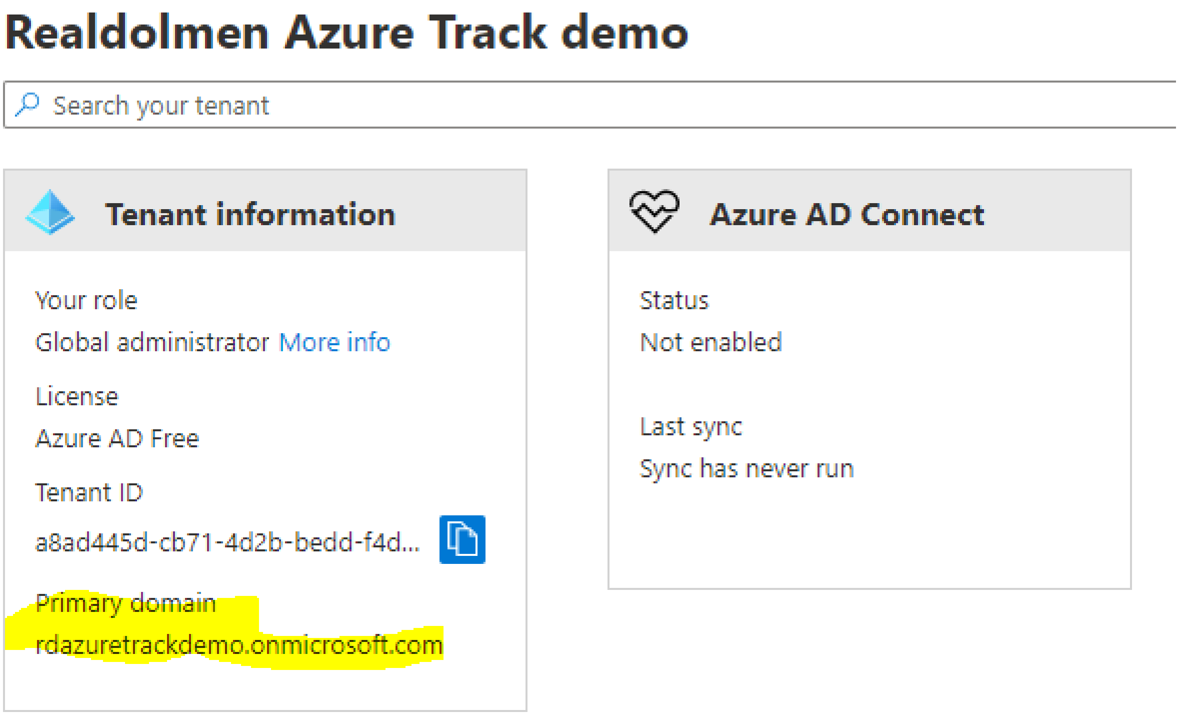
- Add the package Microsoft.Identity.Web via Nuget.
- And add the following code in the Startup class (ConfigureServices method):

  // Adds Microsoft Identity platform (AAD v2.0) support to protect this Api
  services.AddMicrosoftIdentityWebApiAuthentication(Configuration, "AzureAd");
- Open the appsettings file and add the following Json configuration, you need to use you ClientId & Tenant Id from the registered Application (API), the domain is the ‘primary domain’ of the Azure AD:



azure-track-demo-api

Display name : azure-track-demo-api
Application (client) ID : 9a445ebb-0c56-481b-886e-d70de5d1e48f
Directory (tenant) ID : a8ad445d-cb71-4d2b-bedd-f4dd8fee406e
Object ID : c55417d1-7483-4eaf-9983-8817490e0110



Realdolmen Azure Track demo

Your role
Global administrator More info

License
Azure AD Free

Tenant ID
a8ad445d-cb71-4d2b-bedd-f4d...

Primary domain
rdazuretrackdemo.onmicrosoft.com

Status
Not enabled

Last sync
Sync has never run

Appsettings.json code:

```
"AzureAd": {
   "Instance": "https://login.microsoftonline.com/",
   "Domain": "rdazuretrackdemo.onmicrosoft.com",
   "TenantId": "a8ad445d-cb71-4d2b-bedd-f4dd8fee406e",
   "ClientId": "9a445ebb-0c56-481b-886e-d70de5d1e48f"
 }
```

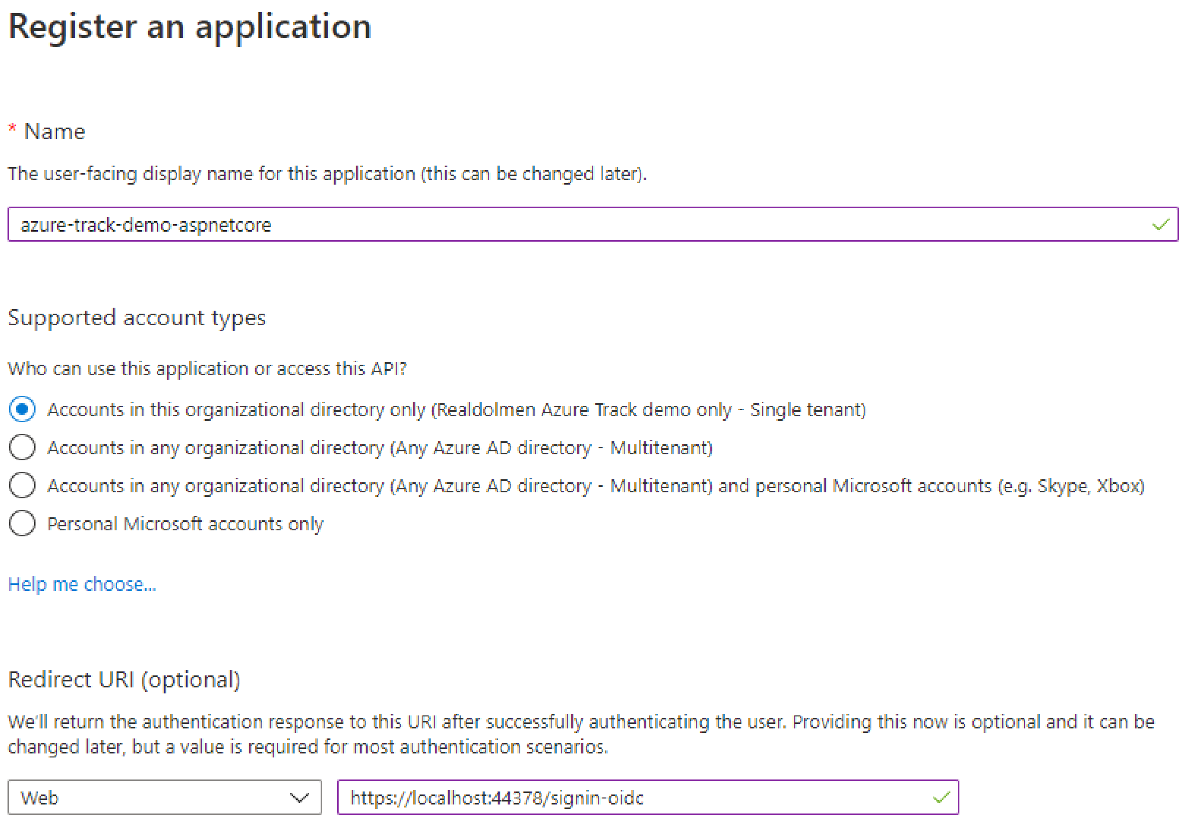
Remark: In the Startup class in the ‘Configure’ method, UseAuthentication() and UseAuthorization() middleware needs to be added between UseRouting() and UseEndpoints()

2.2 azure-track-demo-aspnetcore

We register our WebApp, an ASP.NET Core MVC application. For ASP.NET Core applications we'll use the Implicit flow:

Step 1: name & redirect uri

We choose the 'web' platform and specify a redirect uri (the IDP will redirect to that URI after successful authentication. The Microsoft.Identity.Web functionality will listen on that url where it will parse the information that is provided in the querystring by te IDP.



Register an application

* Name

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

azure-track-demo-aspnetcore ✓

Supported account types

Who can use this application or access this API?

● Accounts in this organizational directory only (Realdolmen Azure Track demo only - Single tenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

○ Personal Microsoft accounts only

Help me choose...

Redirect URI (optional)

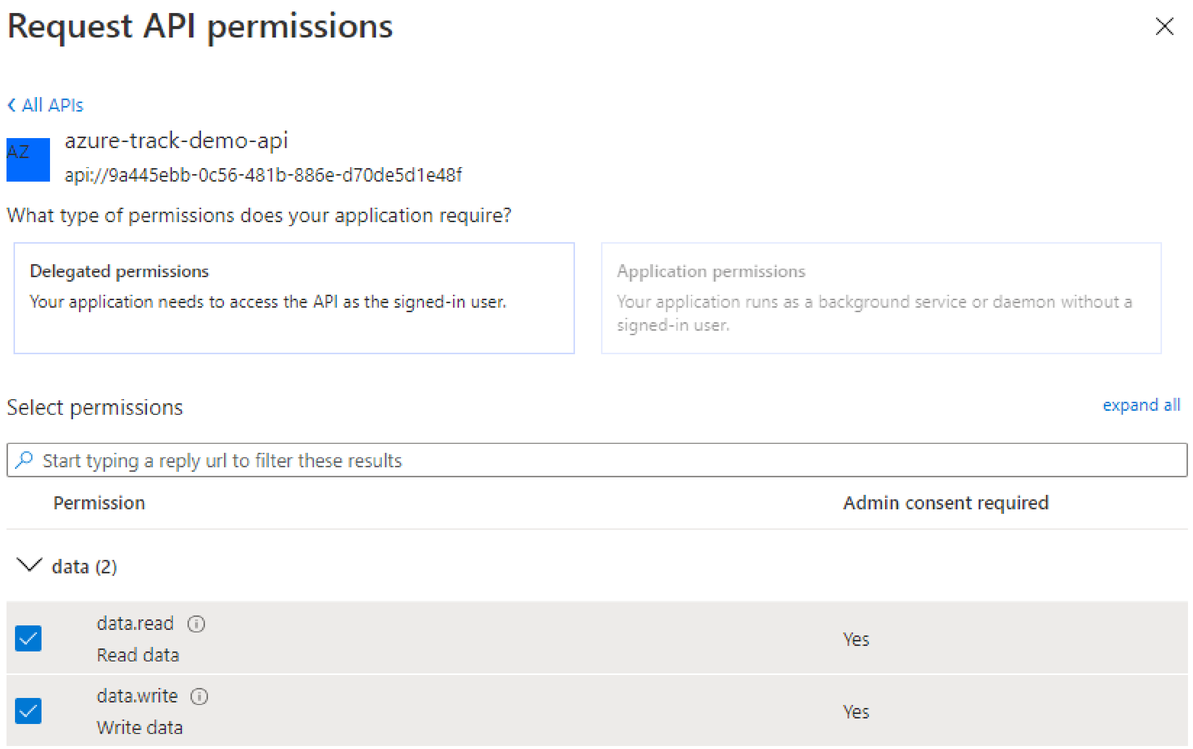
We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web | https://localhost:44378/signin-oidc ✓

Step 2: go to ‘API permissions’

You’ll see that there is already a permission (scope) configured (Users.Read). That scope is required so the client can get the ‘user profile’*: ‘Allows users to sign-in to the app, and allows the app to read the profile of signed-in users. It also allows the app to read basic company information of signed-in users.’*

We add our permissions (scopes) from our own API, click on ‘Add a permission’ -> ‘My APIs’ -> ‘azure-track-api-demo’ -> check both scopes (data.read & data.write):



Request API permissions

‹ All APIs

AZ azure-track-demo-api
api://9a445ebb-0c56-481b-886e-d70de5d1e48f

What type of permissions does your application require?

Delegated permissions
Your application needs to access the API as the signed-in user.

Application permissions
Your application runs as a background service or daemon without a signed-in user.

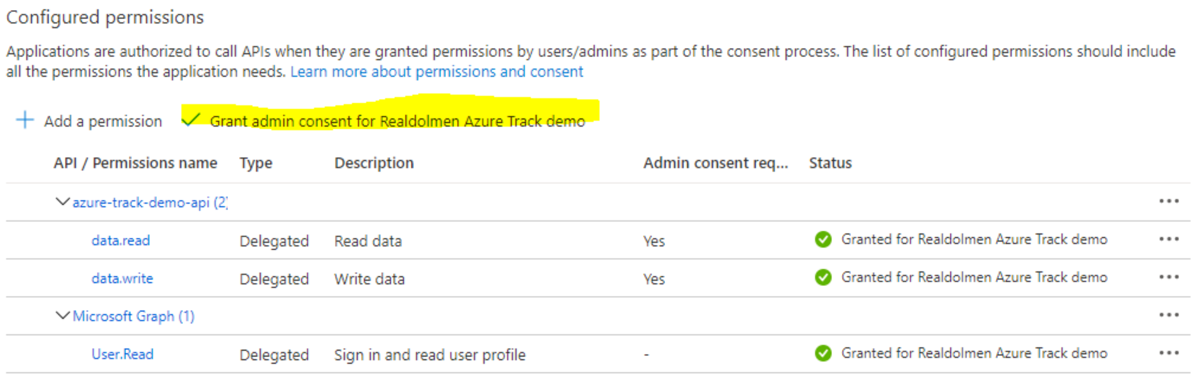
Select permissions

expand all

Start typing a reply url to filter these results

	Permission	Admin consent required
	data (2)	
☑	data.read ⓘ Read data	Yes
☑	data.write ⓘ Write data	Yes

You’ll see that an admin consent is required, you can give an admin consent on these scopes by clicking ‘Grant admin consent for Realdolmen Azure Track demo’ and then the status will be ‘Granted for…’:



Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent

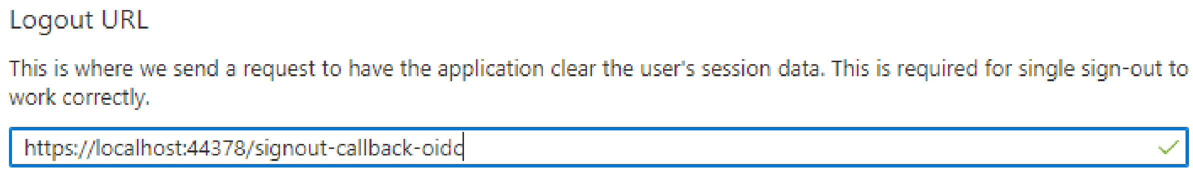
+ Add a permission ✓ Grant admin consent for Realdolmen Azure Track demo

API / Permissions name	Type	Description	Admin consent req...	Status
azure-track-demo-api (2				
data.read	Delegated	Read data	Yes	✓ Granted for Realdolmen Azure Track demo
data.write	Delegated	Write data	Yes	✓ Granted for Realdolmen Azure Track demo
Microsoft Graph (1)				
User.Read	Delegated	Sign in and read user profile	-	✓ Granted for Realdolmen Azure Track demo

Step 3: specify the logout url:

Click on the Authentication menu-item and fill in the logout URL: https://localhost:44378/signout-callback-oidc

After logging out on the IDP it will be redirected to this url the clear the user's session data:



Logout URL

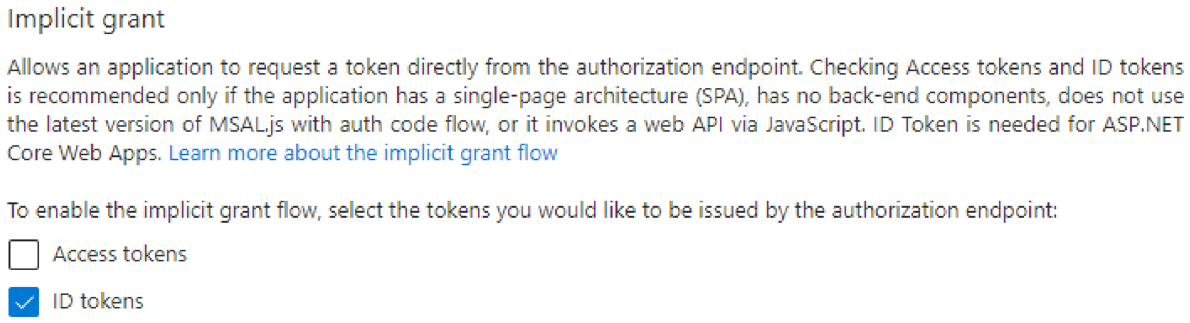
This is where we send a request to have the application clear the user's session data. This is required for single sign-out to work correctly.

https://localhost:44378/signout-callback-oidc

Don't forget to click 'Save'!

Step 5: Enable Implicit flow

Enable the Implicit flow, we need only ID tokens issued by the authorization endpoint:



Implicit grant

Allows an application to request a token directly from the authorization endpoint. Checking Access tokens and ID tokens is recommended only if the application has a single-page architecture (SPA), has no back-end components, does not use the latest version of MSAL.js with auth code flow, or it invokes a web API via JavaScript. ID Token is needed for ASP.NET Core Web Apps. Learn more about the implicit grant flow

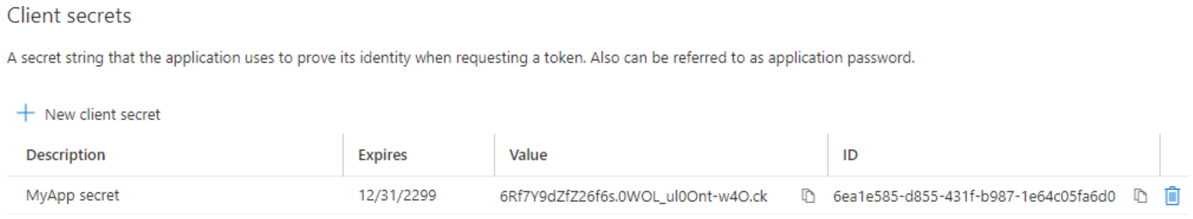
To enable the implicit grant flow, select the tokens you would like to be issued by the authorization endpoint:

☐ Access tokens

☑ ID tokens

Step 4: create a client secret

- Go to Certificates & secrets
- Click on 'New client secret'
- Choose a description, for example 'My App secret' (choose an expiration)
- Click add



Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

+ New client secret

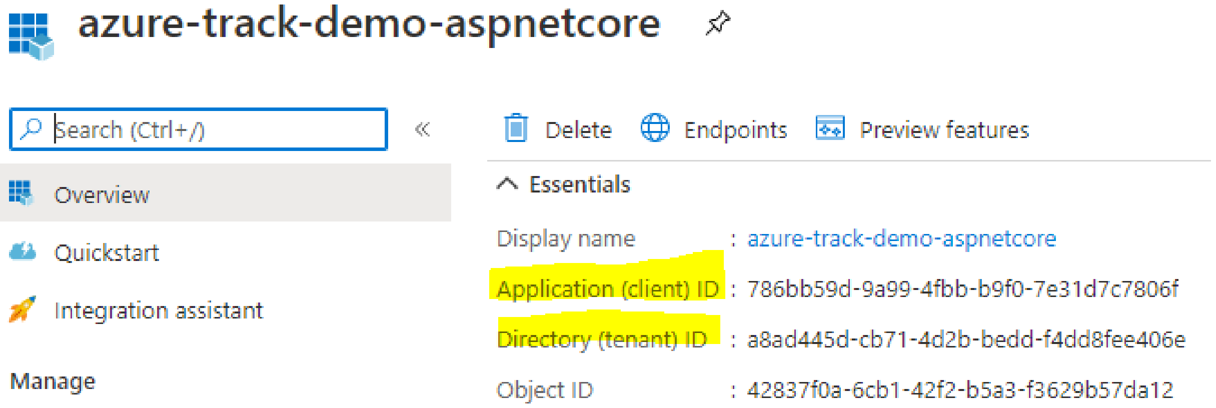
Description	Expires	Value	ID
MyApp secret	12/31/2299	6Rf7Y9dZfZ26f6s.0WOL_ul0Ont-w4O.ck	6ea1e585-d855-431f-b987-1e64c05fa6d0

<u>Note:</u> it's important to 'copy' the secret value because you can only see this secret one time, if you lose it you need to create a new secret.

Step 5: the code, open the MyWebApp project

- Add the package Microsoft.Identity.Web via Nuget.
- And add the following code in the Startup class (ConfigureServices method):

  ```
  // Adds Microsoft Identity platform (AAD v2.0) support to protect this WebApp
    services.AddMicrosoftIdentityWebAppAuthentication(Configuration, "AzureAd")
  ```
- Open the appsettings file and add the following Json configuration, you need to use you ClientId & Tenant Id from the registered Application (WebApp), the domain is the ‘primary domain’ of the Azure AD:



Display name : azure-track-demo-aspnetcore
Application (client) ID : 786bb59d-9a99-4fbb-b9f0-7e31d7c7806f
Directory (tenant) ID : a8ad445d-cb71-4d2b-bedd-f4dd8fee406e
Object ID : 42837f0a-6cb1-42f2-b5a3-f3629b57da12

Appsettings.json code:

```
"AzureAd": {
   "Instance": "https://login.microsoftonline.com/",
   "Domain": "rdazuretrackdemo.onmicrosoft.com",
   "TenantId": "a8ad445d-cb71-4d2b-bedd-f4dd8fee406e",
   "ClientId": " 786bb59d-9a99-4fbb-b9f0-7e31d7c7806f",
   "CallbackPath": "/signin-oidc",
   "SignedOutCallbackPath ": "/signout-callback-oidc"
   //Add in UserSecrets for development and keyvault in production
   "ClientSecret": "your secret created in step 4"
 }
```

Step 6: call the api from our webapp

To call the WebApi from our webapp we created an ‘ApiService’ who uses a HTTPClient to execute Http calls to our WebApi.

First we need to add some more more code in our Startup:

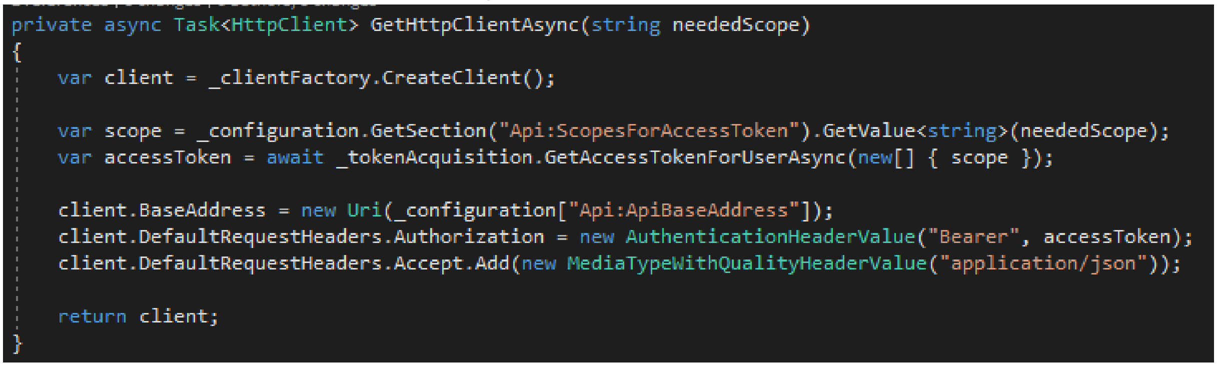
```
services.AddMicrosoftIdentityWebAppAuthentication(Configuration, "AzureAd")
.EnableTokenAcquisitionToCallDownstreamApi(Configuration.GetSection("Api:ScopesForAccessToken").Get<string[]>())
            .AddInMemoryTokenCaches();
```

We specify that we can Inject the ITokenAcquisition service in our code so we can ask for an access_token for one or more specific scopes of our api.

I get the scopes and baseaddress of our API from the appsettings.json file (change scopes with the defined ones in your Api registration):

```
"Api": {
   "ScopesForAccessToken": {
     "Data.Read": "api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.read",
     "Data.Write": "api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.write"
   },
   "ApiBaseAddress": "https://localhost:44390"
 },
```

We create a HttpClient and ask for an access_token on Azure AD with the correct scope and add it to the Authorization header as Bearer for the request to our API:



```
private async Task<HttpClient> GetHttpClientAsync(string neededScope)
{
    var client = _clientFactory.CreateClient();

    var scope = _configuration.GetSection("Api:ScopesForAccessToken").GetValue<string>(neededScope);
    var accessToken = await _tokenAcquisition.GetAccessTokenForUserAsync(new[] { scope });

    client.BaseAddress = new Uri(_configuration["Api:ApiBaseAddress"]);
    client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", accessToken);
    client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

    return client;
}
```

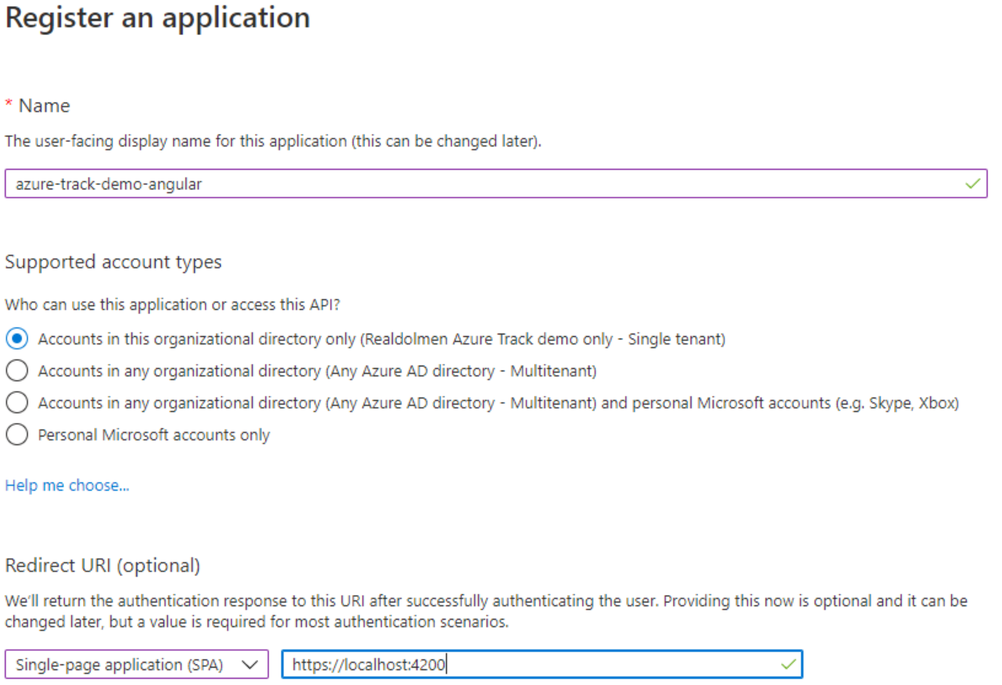
Note: you can ask an access token with more scope to use same the access token to execute more actions.

2.3 azure-track-demo-angular

We register our Angular application. For SPA applications we'll use the Authorization Code + PKCE flow:

Step 1: name & redirect uri

We choose the 'SPÄ' platform and specify a redirect uri, https://localhost:4200 (the IDP will redirect to that URI after successful authentication. Our OpenIdConnect package will parse the querystring data from our url received from Azure AD.



Register an application

* Name

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

azure-track-demo-angular ✓

Supported account types

Who can use this application or access this API?

● Accounts in this organizational directory only (Realdolmen Azure Track demo only - Single tenant)
○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)
○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
○ Personal Microsoft accounts only

Help me choose...

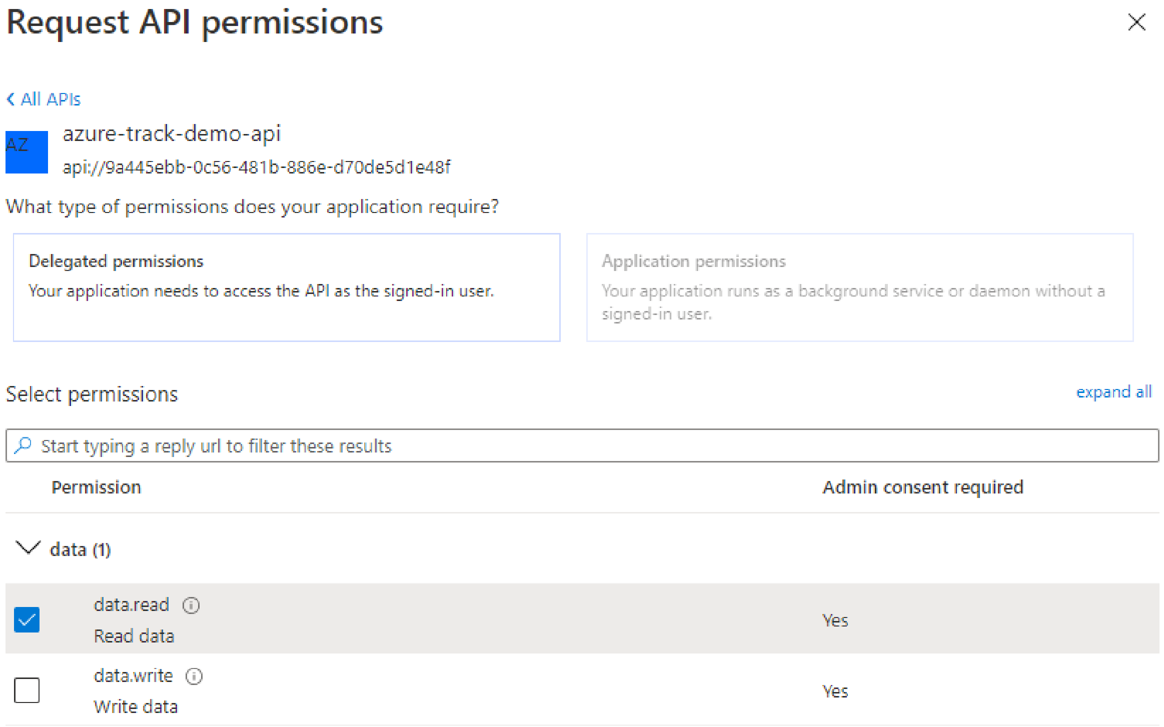
Redirect URI (optional)

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.

Single-page application (SPA) | https://localhost:4200 ✓

Step 2: goto ‘API permissions’

We add our permissions (scopes) from our own API, click on ‘Add a permission’ -> ‘My APIs’ -> ‘azure-track-api-demo’ -> check both scopes only the ‘data.read’ scope because we want only give our Angular client ‘Read access’ on our API for this demo.



Request API permissions

< All APIs

AZ azure-track-demo-api
api://9a445ebb-0c56-481b-886e-d70de5d1e48f

What type of permissions does your application require?

Delegated permissions
Your application needs to access the API as the signed-in user.

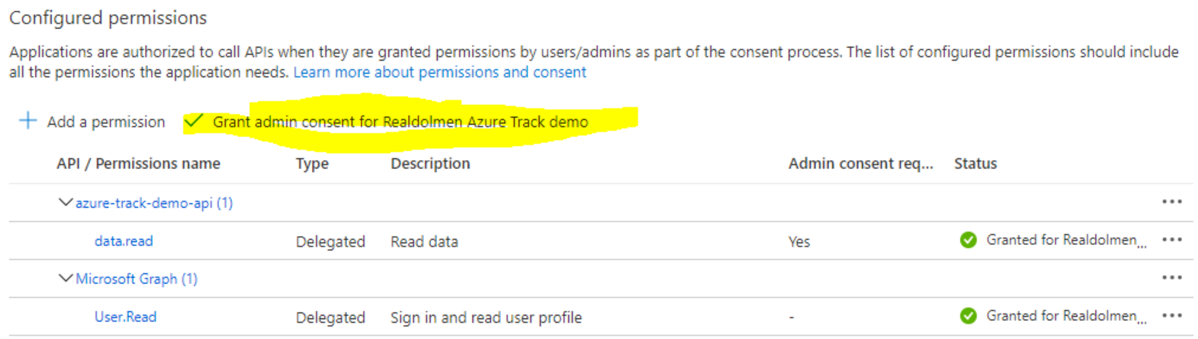
Application permissions
Your application runs as a background service or daemon without a signed-in user.

Select permissions | expand all

Start typing a reply url to filter these results

	Permission	Admin consent required
	data (1)	
☑	data.read Read data	Yes
□	data.write Write data	Yes

You’ll see that an admin consent is required, you can give an admin consent on these scopes by clicking ‘Grant admin consent for Realdolmen Azure Track demo’ and then the status will be ‘Granted for…’:



Configured permissions

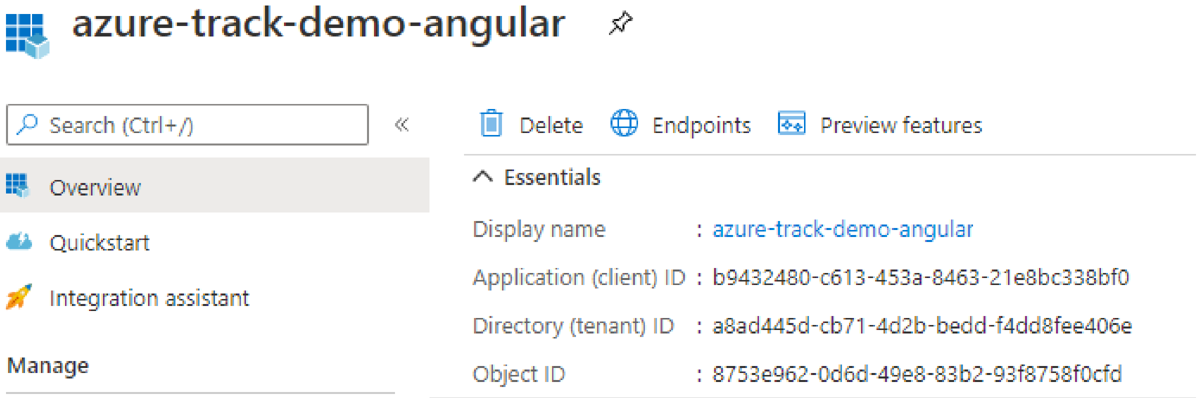
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent

+ Add a permission ✓ Grant admin consent for Realdolmen Azure Track demo

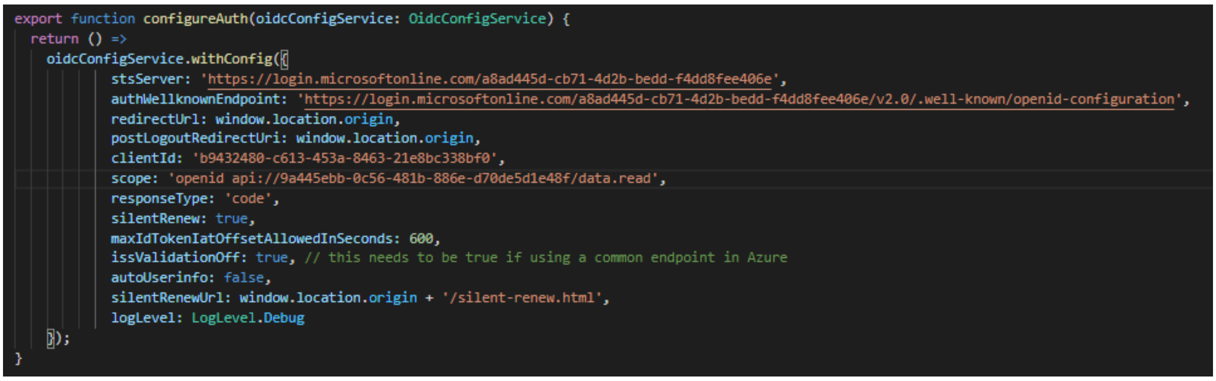
API / Permissions name	Type	Description	Admin consent req...	Status
azure-track-demo-api (1)				
data.read	Delegated	Read data	Yes	✓ Granted for Realdolmen...
Microsoft Graph (1)				
User.Read	Delegated	Sign in and read user profile	-	✓ Granted for Realdolmen...

Step 3: the code, open the AngularOidcClient project with code

- Open the file app.module.ts and change the configuration: (clientid, scope, stsServer, authWellknownEndpoint):
  In the stsServer and authWellknownEndpoint (we only need to change the ‘TenantId’)



Display name : azure-track-demo-angular
Application (client) ID : b9432480-c613-453a-8463-21e8bc338bf0
Directory (tenant) ID : a8ad445d-cb71-4d2b-bedd-f4dd8fee406e
Object ID : 8753e962-0d6d-49e8-83b2-93f8758f0cfd



```
export function configureAuth(oidcConfigService: OidcConfigService) {
  return () =>
    oidcConfigService.withConfig({
            stsServer: 'https://login.microsoftonline.com/a8ad445d-cb71-4d2b-bedd-f4dd8fee406e',
            authWellknownEndpoint: 'https://login.microsoftonline.com/a8ad445d-cb71-4d2b-bedd-f4dd8fee406e/v2.0/.well-known/openid-configuration',
            redirectUrl: window.location.origin,
            postLogoutRedirectUri: window.location.origin,
            clientId: 'b9432480-c613-453a-8463-21e8bc338bf0',
            scope: 'openid api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.read',
            responseType: 'code',
            silentRenew: true,
            maxIdTokenIatOffsetAllowedInSeconds: 600,
            issValidationOff: true, // this needs to be true if using a common endpoint in Azure
            autoUserinfo: false,
            silentRenewUrl: window.location.origin + '/silent-renew.html',
            logLevel: LogLevel.Debug
    });
}
```

After you changed the configuration start your API and Angular app (npm start) and goto https://localhost:4200

NOTE: we use in this angular demo the ‘angular-auth-oidc-client’ package. This package is an standard implementation of OidcClient. We can also use MSAL (js & Angular) but the at the moment of creating this demo the 2.0.0 version for Angular is still in alpha stage. We can use the 1.x version BUT that version does not support the Authorization Code + PKCE flow… It only support the Implicit Flow for SPA’s if we want following the standards concerning secure SPA’s we must use the Authorization Code + PKCE Flow…

We can also conclude here that Azure AD implements the OpenId standards 😉.

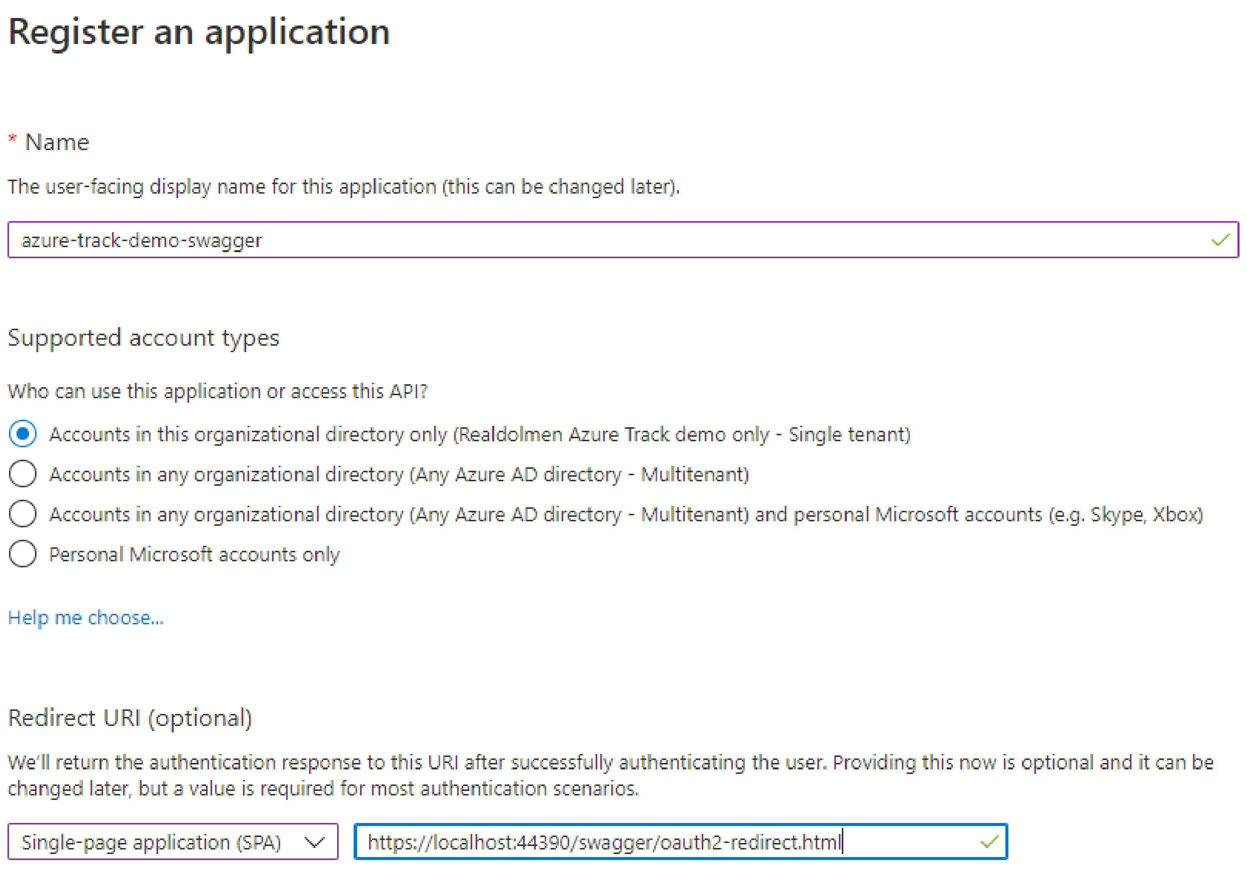
⇨ You we’ll see there is also a silent-renew mechanism included in the demo (you also need to add https://localhost:4200/silent-renew.html in the redirect uri’s of the application).
It’s a mechanism to renew an access token in the background (hidden iframe). An access token is a limited time valid. The silent renew gets a new access token in the background before the old one is expired! If we don’t use this mechanism it would be required to login again after an access token is expired!

2.4 azure-track-demo-swagger

We need also protect our swagger implementation on our API project. Swagger (swashbuckle) sends an api call from javascript therefore we need to use the Authorization Code + PKCE flow.

Step 1: name & redirect uri

We choose the 'SPÄ' platform and specify a redirect uri, https://localhost:44390/swagger/oauth2-redirect.html.



Register an application

* Name

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

azure-track-demo-swagger ✓

Supported account types

Who can use this application or access this API?

● Accounts in this organizational directory only (Realdolmen Azure Track demo only - Single tenant)
○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)
○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
○ Personal Microsoft accounts only

Help me choose...

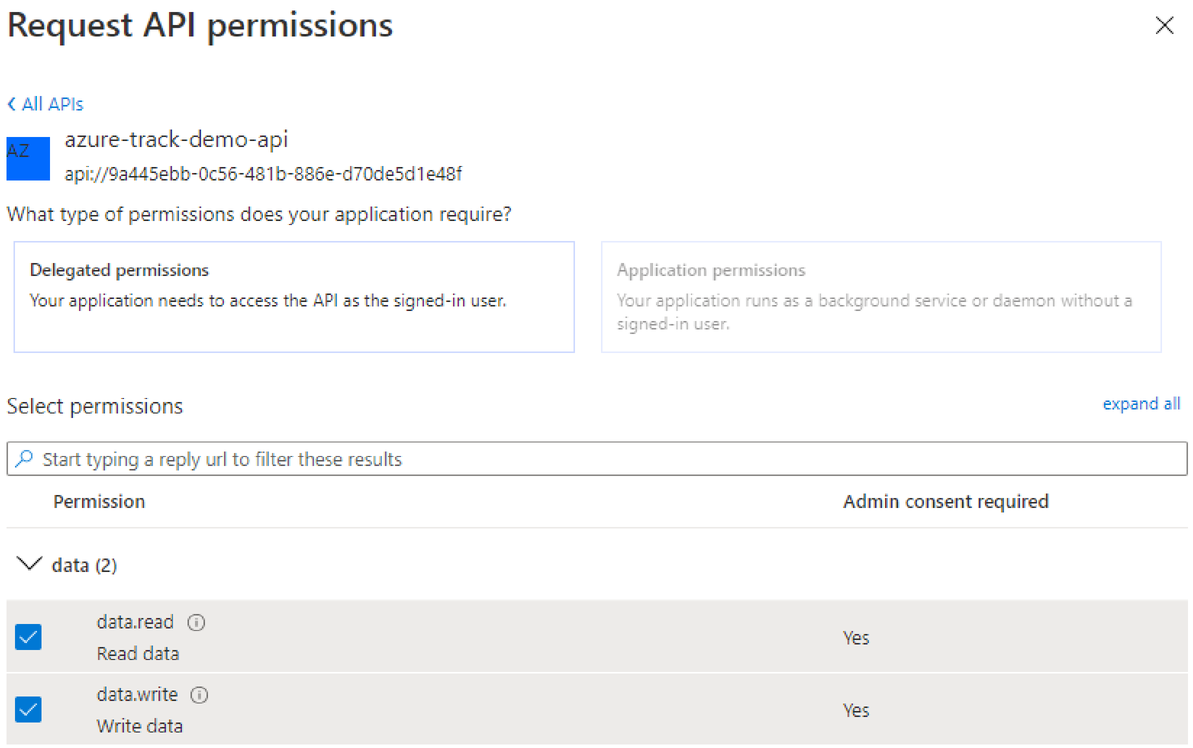
Redirect URI (optional)

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.

Single-page application (SPA) | https://localhost:44390/swagger/oauth2-redirect.html ✓

Step 2: goto ‘API permissions’

We add our permissions (scopes) from our own API, click on ‘Add a permission’ -> ‘My APIs’ -> ‘azure-track-api-demo’ -> check both scopes (data.read & data.write):



Request API permissions

‹ All APIs

AZ azure-track-demo-api
api://9a445ebb-0c56-481b-886e-d70de5d1e48f

What type of permissions does your application require?

Delegated permissions
Your application needs to access the API as the signed-in user.

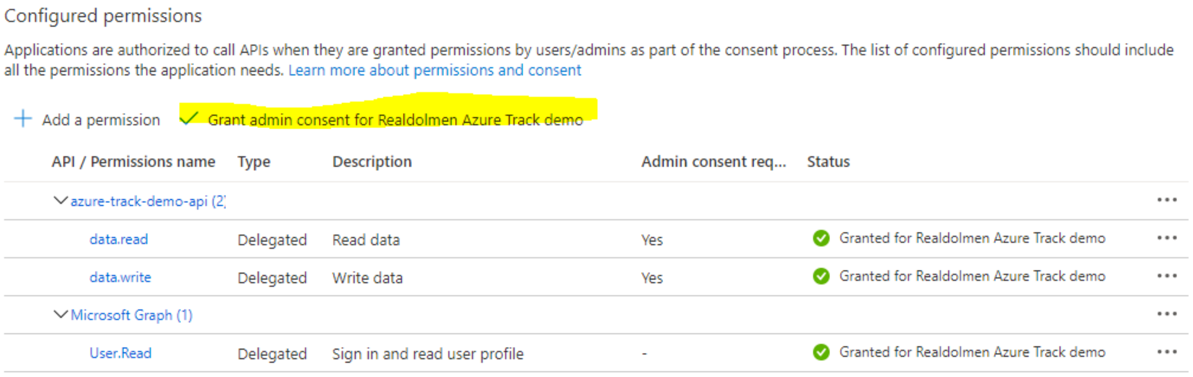
Application permissions
Your application runs as a background service or daemon without a signed-in user.

Select permissions | expand all

Start typing a reply url to filter these results

	Permission	Admin consent required
	data (2)	
☑	data.read Read data	Yes
☑	data.write Write data	Yes

You’ll see that an admin consent is required, you can give an admin consent on these scopes by clicking ‘Grant admin consent for Realdolmen Azure Track demo’ and then the status will be ‘Granted for…’:



Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent

+ Add a permission ✓ Grant admin consent for Realdolmen Azure Track demo

API / Permissions name	Type	Description	Admin consent req...	Status	
azure-track-demo-api (2)					···
data.read	Delegated	Read data	Yes	✓ Granted for Realdolmen Azure Track demo	···
data.write	Delegated	Write data	Yes	✓ Granted for Realdolmen Azure Track demo	···
Microsoft Graph (1)					···
User.Read	Delegated	Sign in and read user profile	-	✓ Granted for Realdolmen Azure Track demo	···

Step 3: the code, open the appsettings.json class from MyApi

- Change the client_id, tenant_id and scopes of the swaggerOAuth section.

```
"SwaggerOAuth": {
    "ClientId": "3741f2f2-82cf-406f-9658-3778d8e048fd",
    "AuthorizationUrl": "https://login.microsoftonline.com/a8ad445d-cb71-4d2b-bedd-f4dd8fee406e/oauth2/v2.0/authorize",
    "TokenUrl": "https://login.microsoftonline.com/a8ad445d-cb71-4d2b-bedd-f4dd8fee406e/oauth2/v2.0/token",
    "Scopes": {
      "MyAPI Read": "api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.read",
      "MyAPI Write": "api://9a445ebb-0c56-481b-886e-d70de5d1e48f/data.write"
    }
  },
```

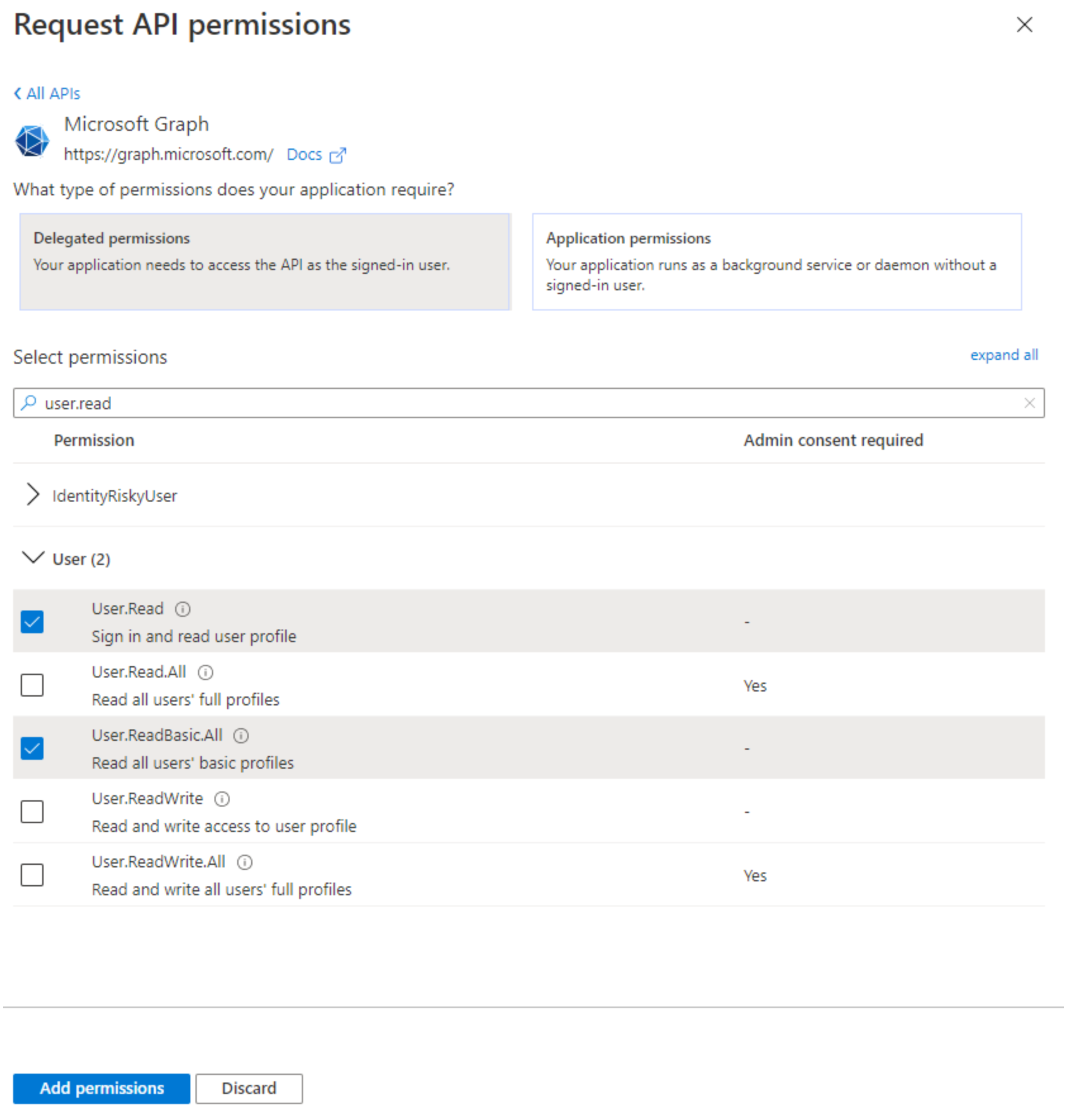
- Check the code in the Startup.cs file of MyAPI, there you can show how you need to configure swagger to add authentication.

2.5 Down stream API

In our architecture we have MyApi that get's a user list (User.ReadBasic.All scope) from the GraphAPI. We need to configure that MyApi can request an access token for an user for the User.ReadBasic.All scope of the Graph API (delegated access).
Our client is not configured to have access to the scope of the Graph API, My API is responsible for this. Sometimes we talk over 'delegated access'.

Step 1: open the 'azure-track-demo-api' application registration
Step 2: goto API permissions:

- Add the User.ReadBasic.All scope



Request API permissions

Microsoft Graph
https://graph.microsoft.com/ Docs

What type of permissions does your application require?

Delegated permissions
Your application needs to access the API as the signed-in user.

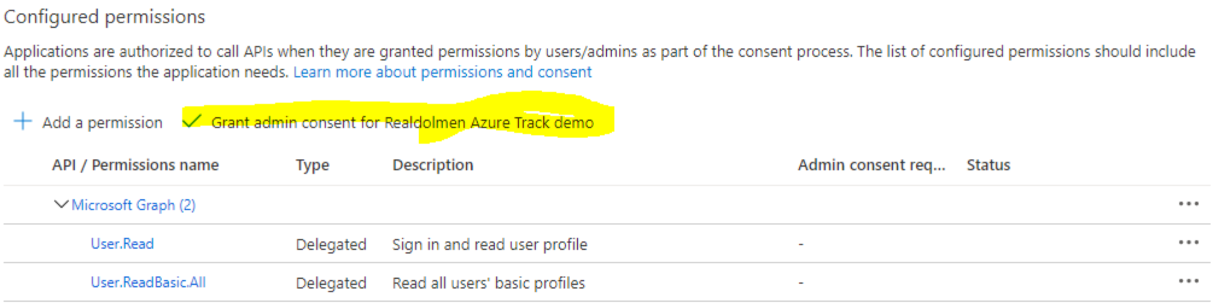
Application permissions
Your application runs as a background service or daemon without a signed-in user.

Select permissions

user.read

	Permission	Admin consent required
	› IdentityRiskyUser	
	∨ User (2)	
☑	User.Read Sign in and read user profile	-
□	User.Read.All Read all users' full profiles	Yes
☑	User.ReadBasic.All Read all users' basic profiles	-
□	User.ReadWrite Read and write access to user profile	-
□	User.ReadWrite.All Read and write all users' full profiles	Yes

You'll see that an admin consent is required, you can give an admin consent on this scope by clicking 'Grant admin consent for Realdolmen Azure Track demo' and then the status will be 'Granted for…':



Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent

API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (2)				
User.Read	Delegated	Sign in and read user profile	-	
User.ReadBasic.All	Delegated	Read all users' basic profiles	-	

Step 3: Add code in Startup class to enable TokeAcquisition (getting an access token for one or more specific scopes):

```
// Adds Microsoft Identity platform (AAD v2.0) support to protect this Api
    services.AddMicrosoftIdentityWebApiAuthentication(Configuration, "AzureAd")
        //only needed to call downstream API
        .EnableTokenAcquisitionToCallDownstreamApi()
        .AddInMemoryTokenCaches();
```

Step 4: get a token: In your code when you need an access token:

```
//we want an access token for the User.ReadBasic.All scope for the MS Graph API
var token = await _tokenAcquisition.GetAccessTokenForUserAsync(new string[] { "User.ReadBasic.All" });
```