**Step by Step - How to configure Application to use Azure AD and Oauth2**

**OAUTH2 -** is Token-based security

**OAUTH2 there are 4 roles in the authentication flow**

**Resource owner** - The end-user. The term reflects OAuth's original purpose, giving 3rd party software access on a user's behalf. Other scenarios are also possible.

**Resource server -** The protected asset, typically a web API, which requires a token in order to be accessed. The token validation logic aims to be minimal and can also be stateless.

**Client - The application** - web, mobile, desktop, or device-based, that needs to obtain a token to access the resource server. The client-side OAuth logic is intended to be simple and minimal.

**Authorisation server** - Dedicated server for issuing access tokens to the client, after authenticating the end-user and obtaining authorisation, from the end-user or based on some policy.

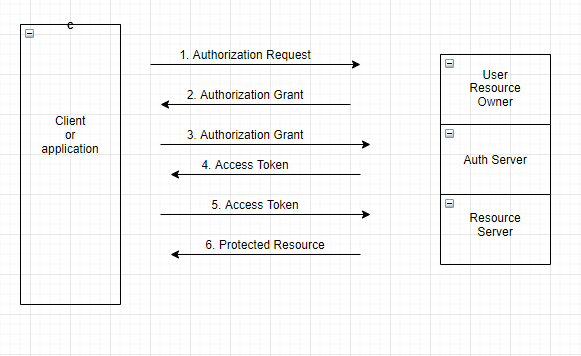
**How can a client obtain a token?**

In order to obtain an access token the client needs to present a valid grant (credential) to the authorisation server.

**Below the common grants and it’s uses:**

1. **Authorisation code -** Intended for traditional web applications with a backend as well as native (mobile or desktop) applications to take advantage of single sign-on via the system browser.
2. **Implicit -** Intended for browser-based (JavaScript) applications without a backend.
3. **Password -** For trusted native clients where the application and the authorisation server belong to the same provider.
4. **Client credentials -** For clients, such as web services, acting on their own behalf.
5. **Refresh token -** A special grant to let clients refresh their access token without having to go through the steps of a code or password grant again.
6. **SAML 2.0 bearer -** Lets a client in possession of a SAML 2.0 assertion (sign-in token) exchange it for an OAuth 2.0 access token.
7. **JWT bearer** - Lets a client in possession of a JSON Web Token (JWT) assertion from one security domain exchange it for an OAuth 2.0 access token in another domain.
8. **Device -** For devices without a browser or with constrained input, such as a smart TV, media console, printer, etc.
9. **Token exchange -** Lets applications and services obtain an access token in delegation and impersonation scenarios.

**Authentication Flow:**



1. The client requests authorization to access service resources from the user

2. If the user authorized the request, the application receives an authorization grant

3. The application requests an access token from the authorization server (API) by presenting authentication of its own identity, and the authorization grant

4. If the application identity is authenticated and the authorization grant is valid, the authorization server (API) issues an access token to the application. Authorization is complete.

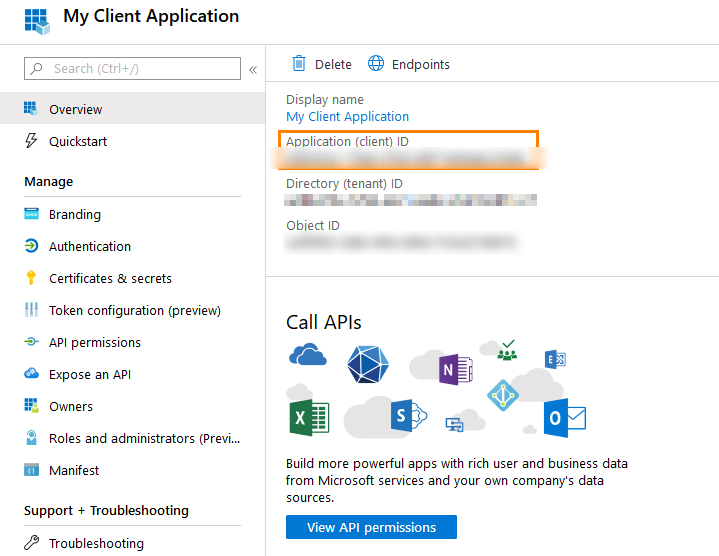
5. The application requests the resource from the resource server (API) and presents the access token for authentication

6. If the access token is valid, the resource server (API) serves the resource to the application

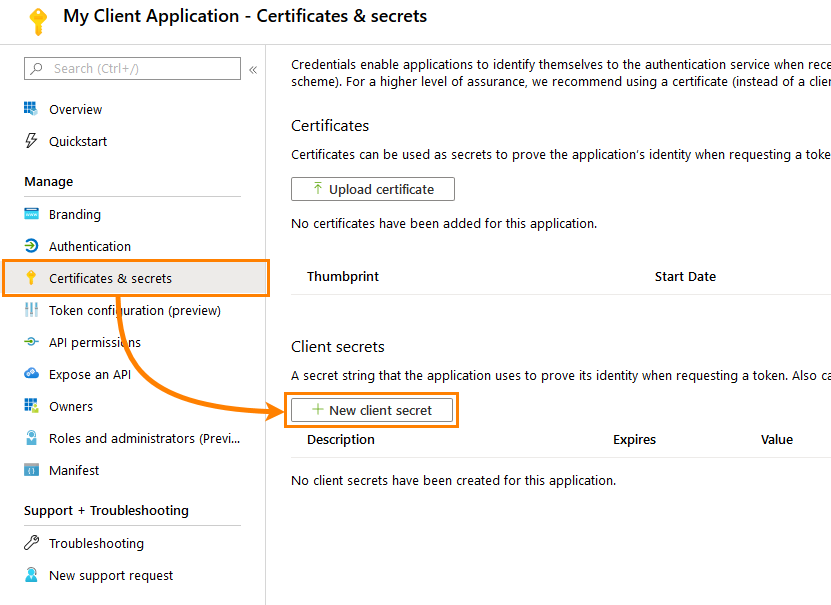
**How to configure application for oauth2 in Azure Active Directory**

**Steps in Azure Portal:**

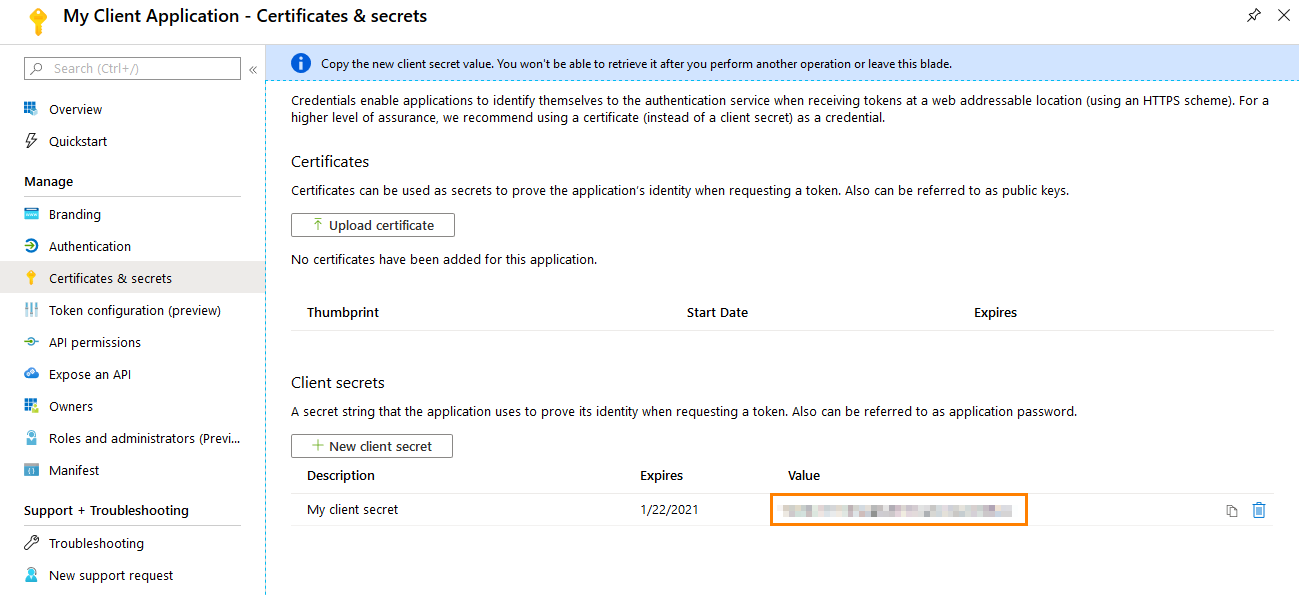
1. Register Application with Azure Active Directory
2. Configure A Client Application - A client application is an application that requests a protected resource. After you register it in Azure Active Directory, you need to perform the following steps to apply the client credentials grant type:
3. Open the Azure Active Directory service. In App registrations, open the registration of your client application.
4. Copy the Application (client) ID to some place. You will need it to link the client to the web service and to configure the request authentication:



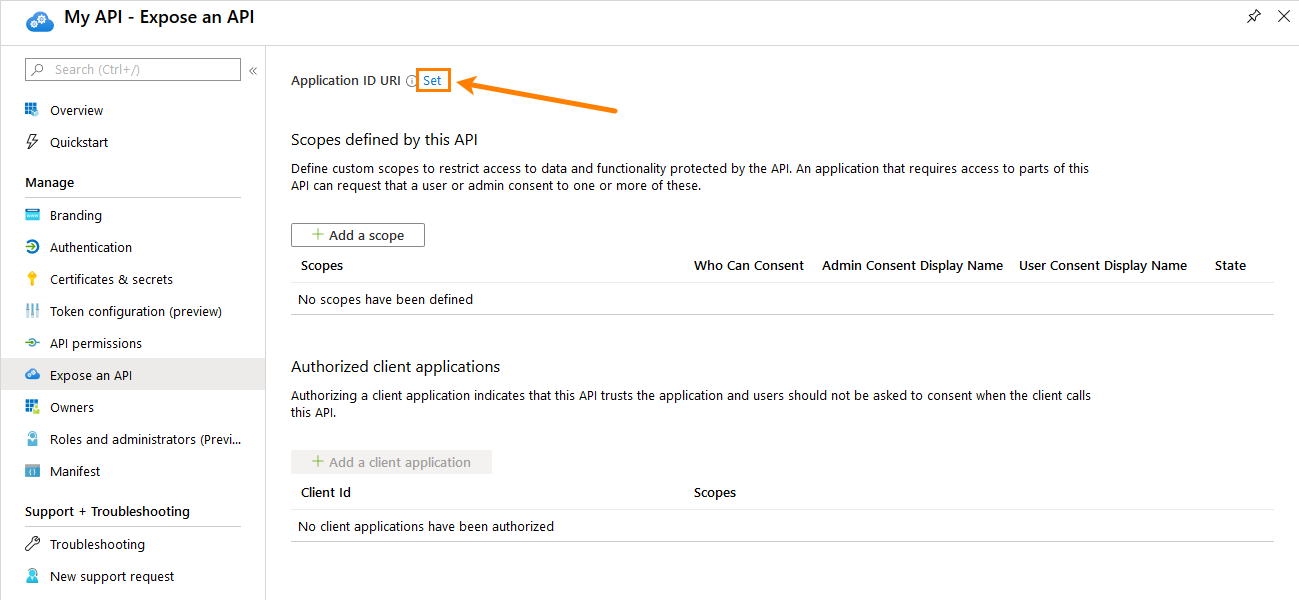
1. In the Client Credentials Grant type, you will need a client secret. To get it, open the Certificates & secrets page and click New client secret: Add a short description and click Add.



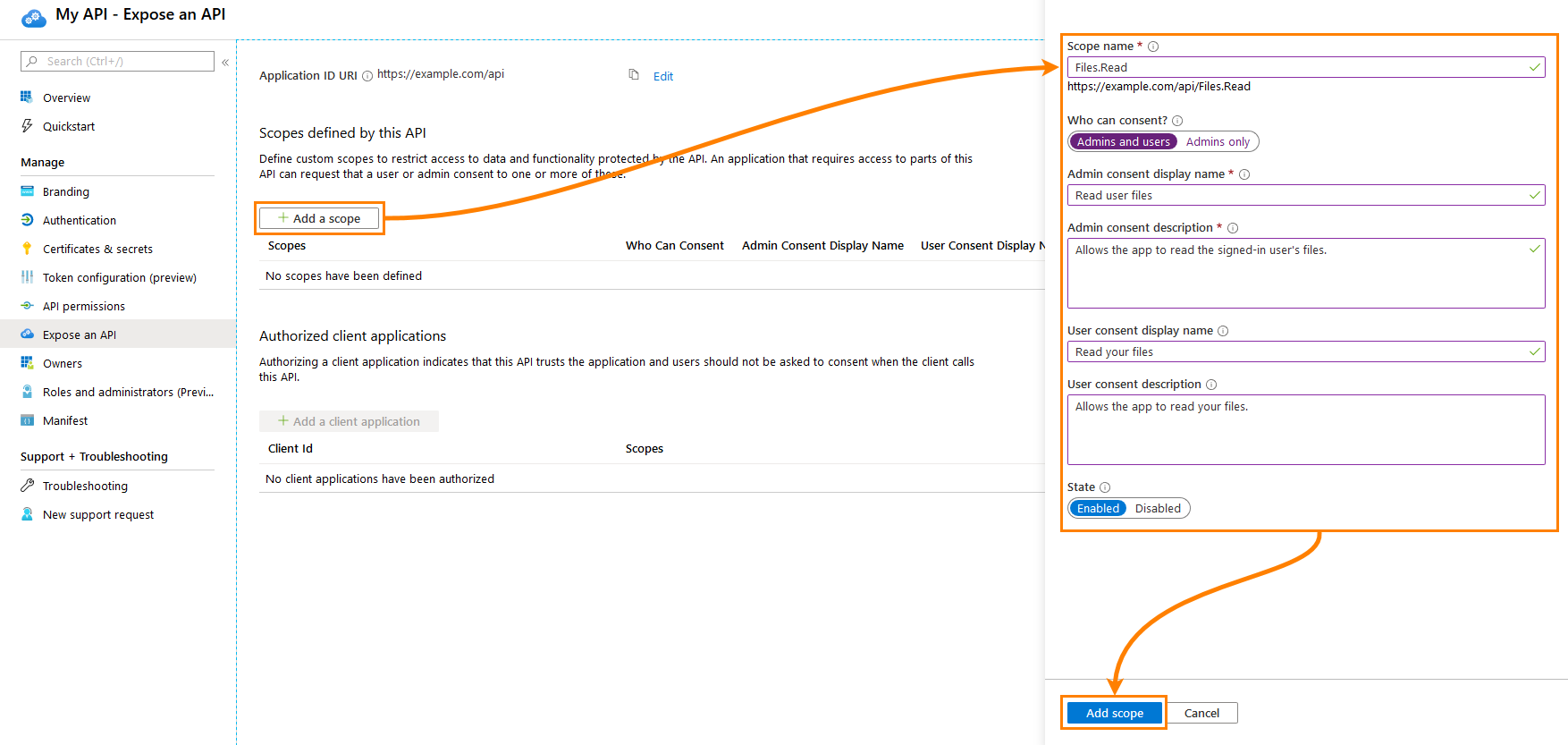
1. Copy the generated value to some place: NOTE: You will not be able to get the client secret after you leave the Certificates & secrets page.



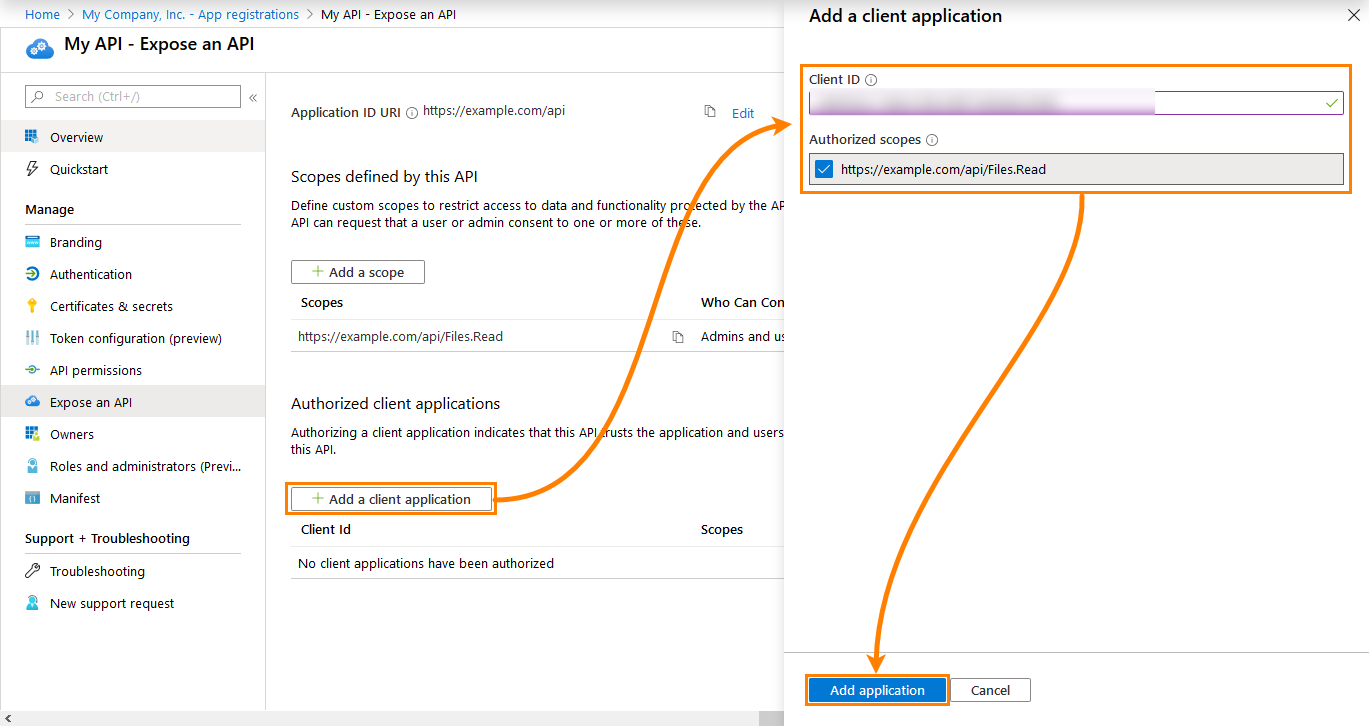
1. Configure A Web Service Application - To configure a web service application, you need to authorize your client application. To do this, perform the following steps:
2. Open the Azure Active Directory service. In App registrations, open the registration of your web service application.
3. Open the Expose an API page.
4. Set the Application ID URI:



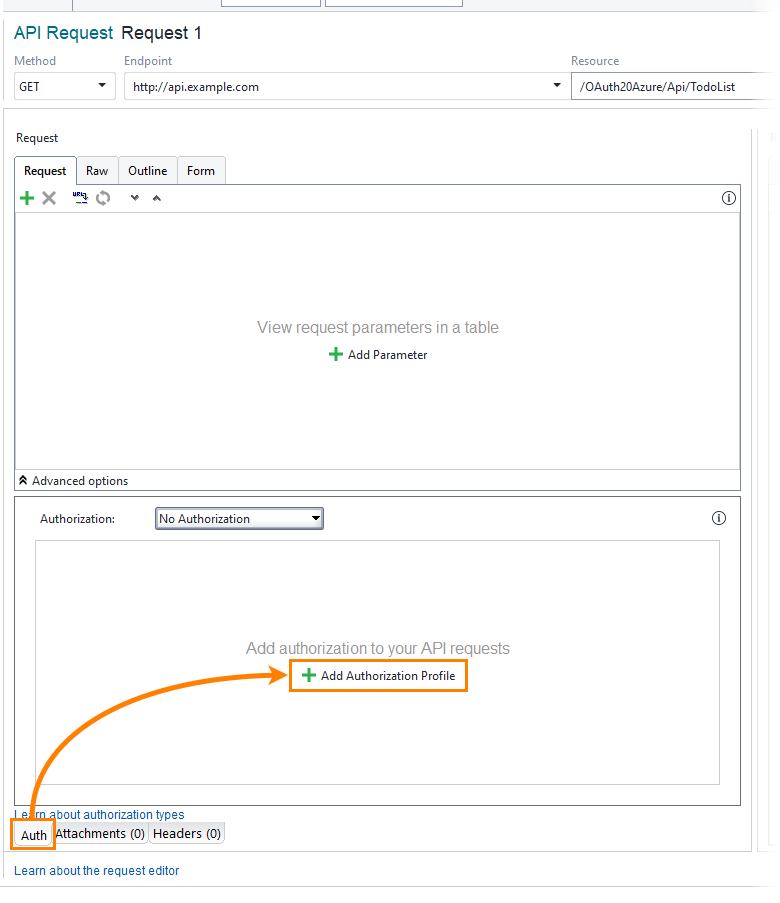
1. When you authorize a client, you specify the scope to restrict client access. To define the scope, click Add a scope and configure it as you need:



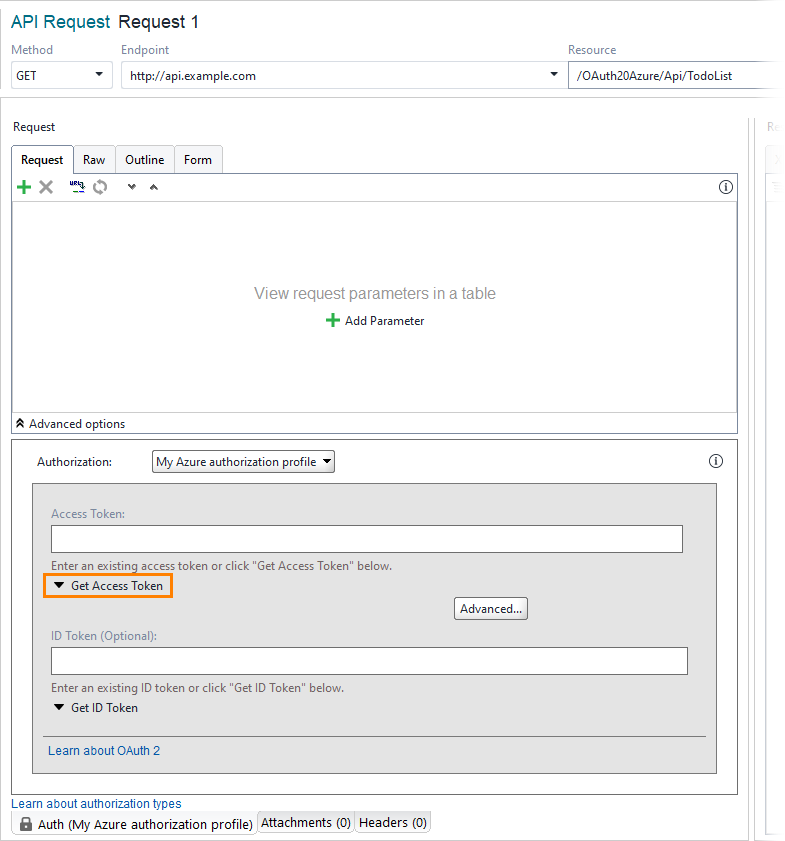
1. To authorize the client application, click Add a client application and specify the Application ID you got earlier:



1. Configure A Request Authentication - Now, you can configure authentication to a protected resource.
2. In ReadyAPI, open a REST request.
3. In the Auth panel, click Add Authorization to add a new authentication profile:



1. Select the OAuth 2.0 (Azure) authentication type.
2. ReadyAPI creates a profile and applies it to the request. Click Get Access Token to configure authentication and get an access token:



1. Select Client Credentials Grant and fill in the required fields. To get the needed values, use data you got from Azure Active Directory earlier:

Client identification: The application ID of your client application. See the Overview page of your application in the Azure Active Directory.

Client Secret: The client secret you created earlier. If you do not have it yet, create it on the Certificates & Secrets page of your application in the Azure Active Directory.

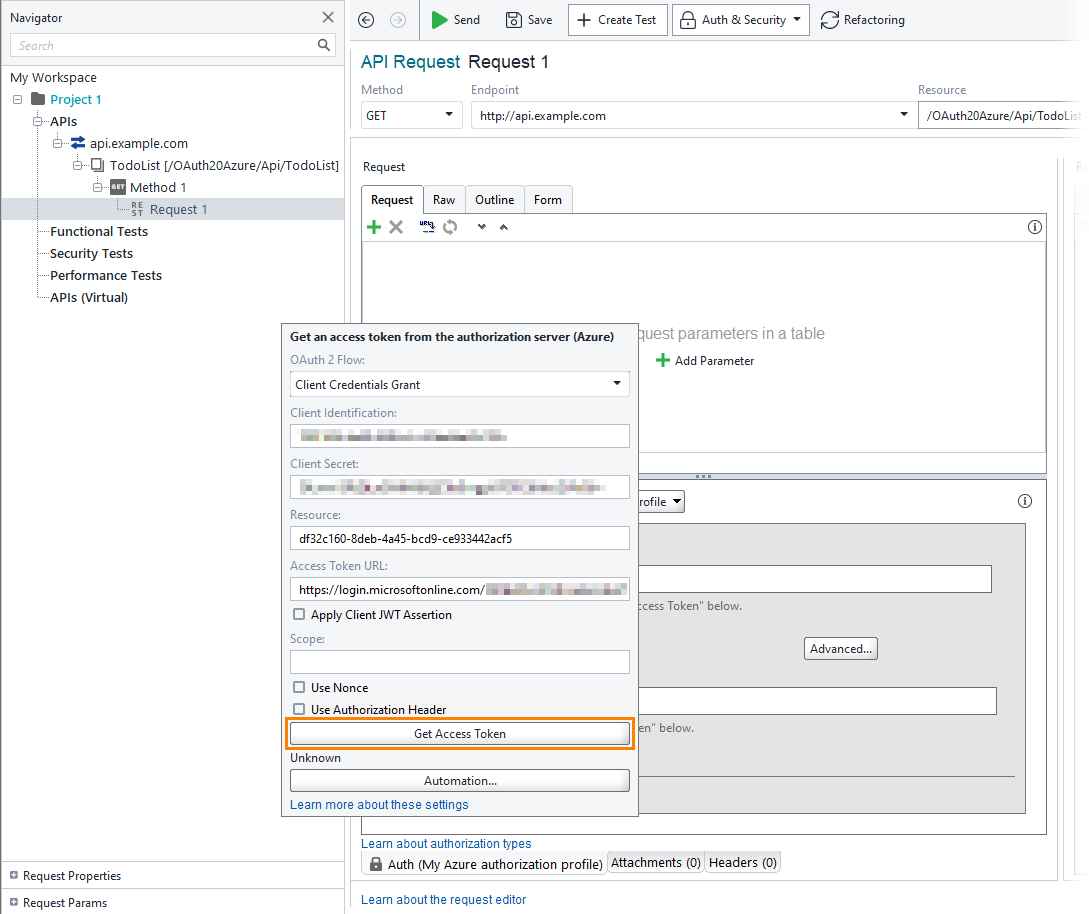
Resource: The Application ID URI of the protected web service. To get it, see the Overview page of your API application in the Azure Active Directory.

Access Token URL: The URL to which ReadyAPI requests an access token. This URL looks as follows:

https://login.microsoftonline.com/<your tenant id>/oauth2/token

To get it, open your Azure Active Directory and click Endpoint.

1. Click Get Access Token to retrieve the token:



1. Now, when you send the request, ReadyAPI sends the access token to authenticate it.

**Powershell:**

**Requirements:**

1. Must have Azure AD Module installed
2. Certificate + Thumbprint

**Steps:**

1. Import-Module AzureAD
2. Connect-AzureAD
3. $azureADAppReg = New-AzureADApplication -DisplayName TestApp -AvailableToOtherTenants $false
4. $azureADAppReg
5. New-AzureADApplicationPasswordCredential -CustomKeyIdentifier PrimarySecret -ObjectId $azureADAppReg.ObjectId -EndDate ((Get-Date).AddMonths(6))
6. $certPath = 'pathtocert'
7. $cert = New-Object -TypeName System.Security.Cryptography.X509Certificates.X509Certificate2($certPath)
8. $keyValue = [System.Convert]::ToBase64String($cert.GetRawCertData())
9. $base64Thumbprint = [System.Convert]::ToBase64String($cert.GetCertHash())
10. New-AzureADApplicationKeyCredential -ObjectId $azureADAppReg.ObjectId -CustomKeyIdentifier $base64Thumbprint -Type AsymmetricX509Cert -Usage Verify -Value $keyValue -EndDate (Get-Date).AddMonths(6)