IDP Login Configuration

The IDP (Identity Provider) login refers to the process of authenticating a user via an external service that manages identities.

The user can log in with an account from an external identity provider, e.g. Azure AD.
This login approach needs to be configured in Keycloak backend depending on the requirements of the customers.

OMN does not manage the authentication and profiles of the IDP users since they are external identities, but the user could map them to work with OMN smoothly.

Currently, the following mappings are supported:

- Mapping Activation for OMN Login (Mapping the activation state)
- Mapping IDP User Profile
- Mapping IDP User Groups

According to the requirements of the customers, the IDP logins could be configured in the keycloak admin console. Keycloak supports both: OpenID Connect and SAML protocols.
This guide will focus on the OpenID Connect protocol.

It needs the administrator access to the keycloak admin console for configuring the IDP logins. The credentials could be found in the `keycloak.env` file from the OMN server.

Furthermore, access to the MS Entra Admincenter (e.g. as Global Administrator) is required.

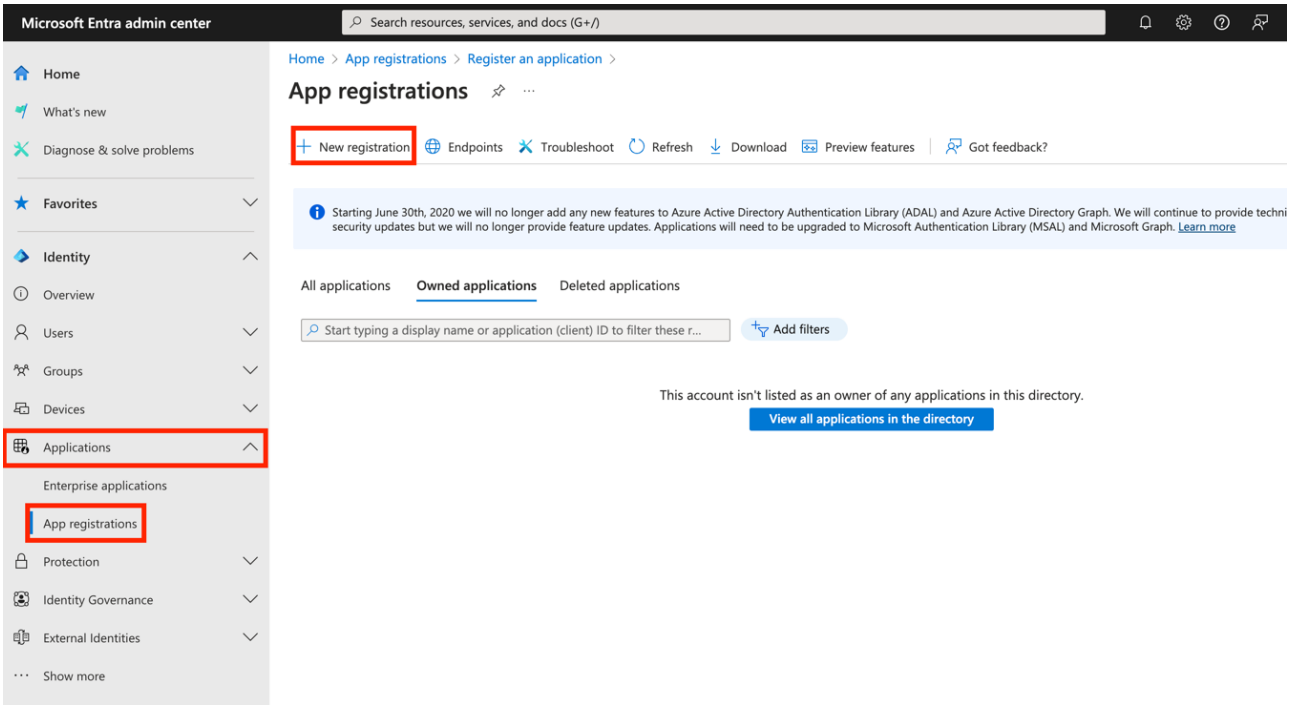
IDP Configuration Example - Azure AD

To configure an IDP login with Azure AD, two setup steps are necessary: with the Microsoft Entra Admincenter (Azure AD) and the Keycloak Admin Console.

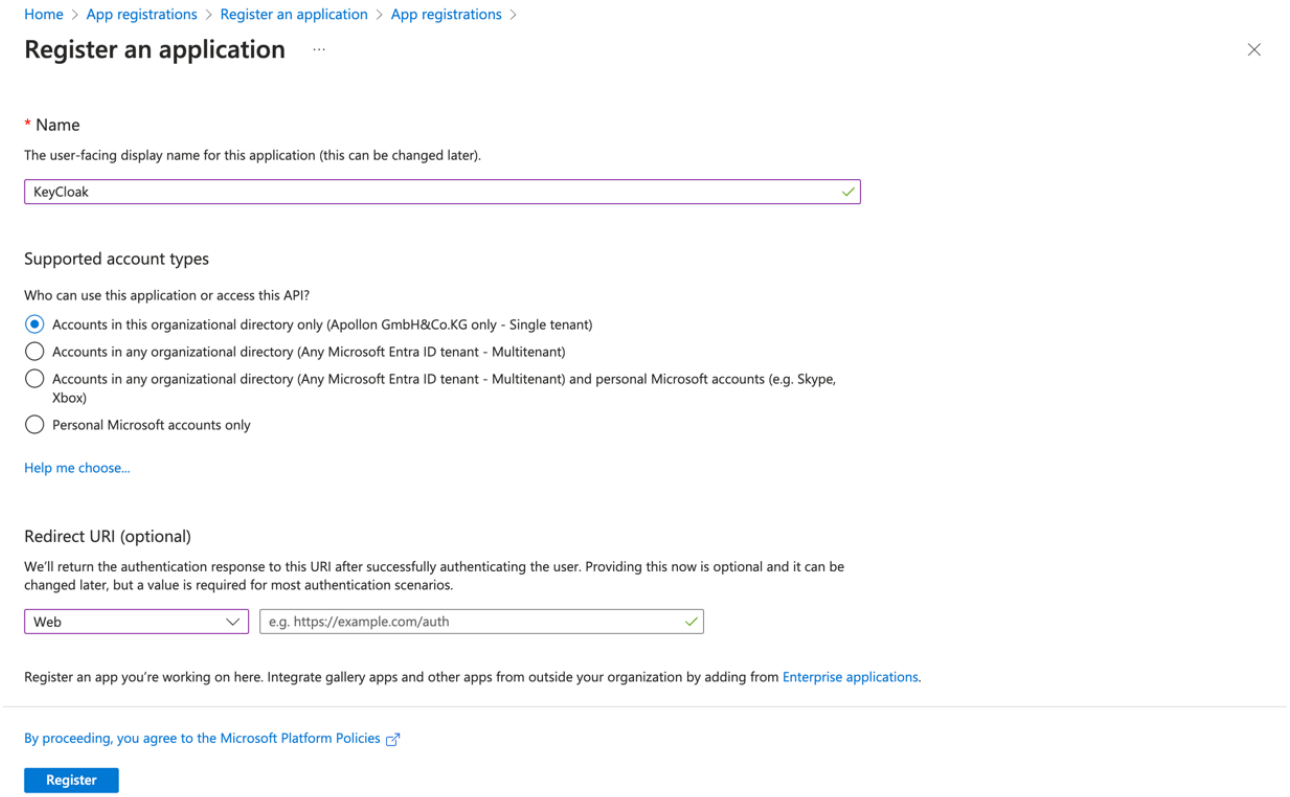
The steps in the following chapters should be followed.

Microsoft Azure AD

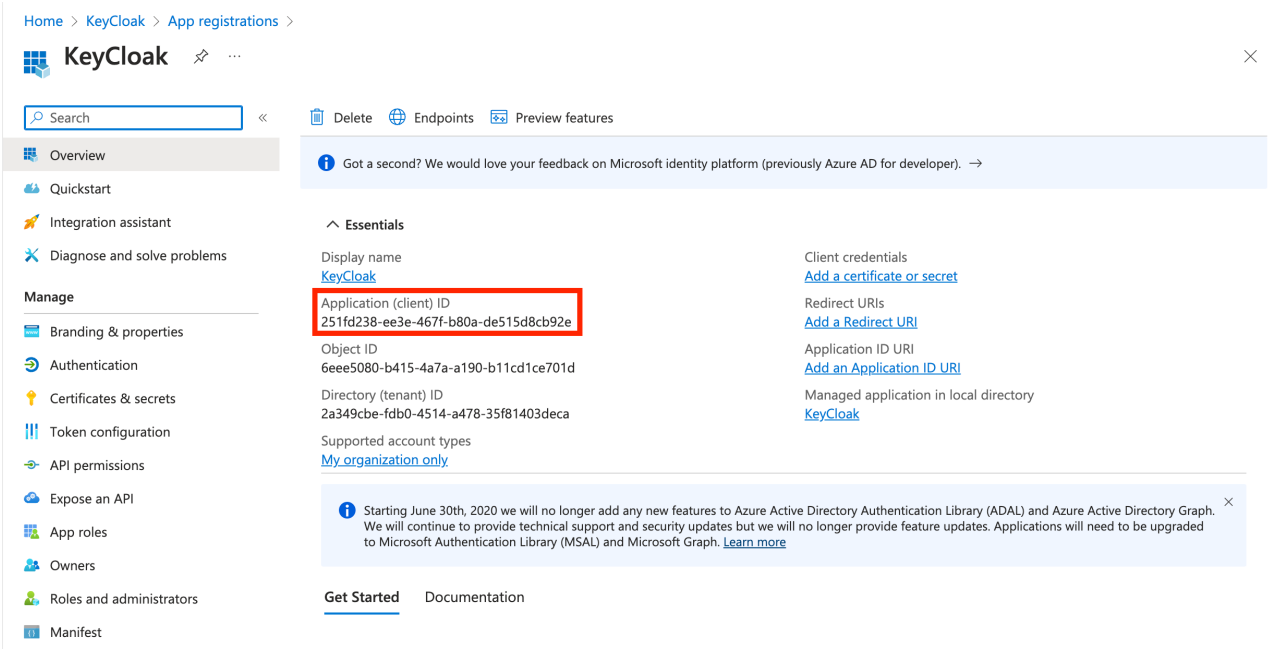
1. Log in to your admin console and navigate to the Applications section:
2. Select `Applications` → `App Registrations` → `New registration`



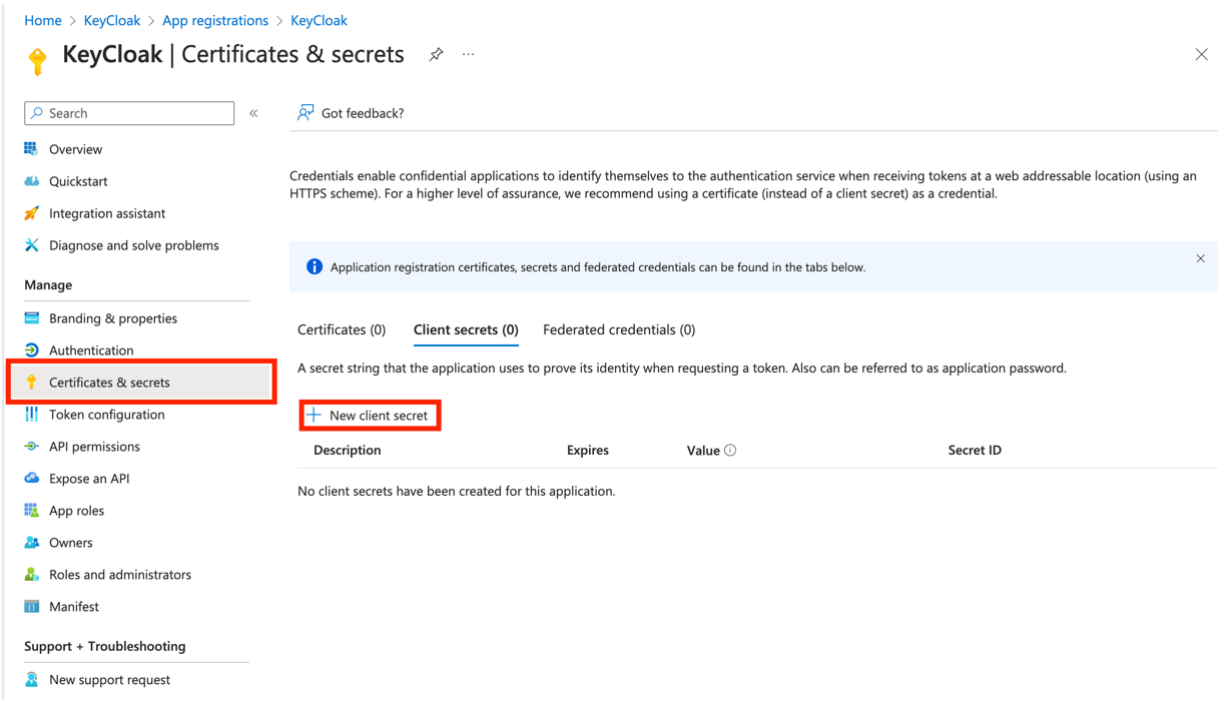
3. Provide a name for the application you are registering and select the account type that you would like to be supported → `single-tenant`.
   Then click the register button.



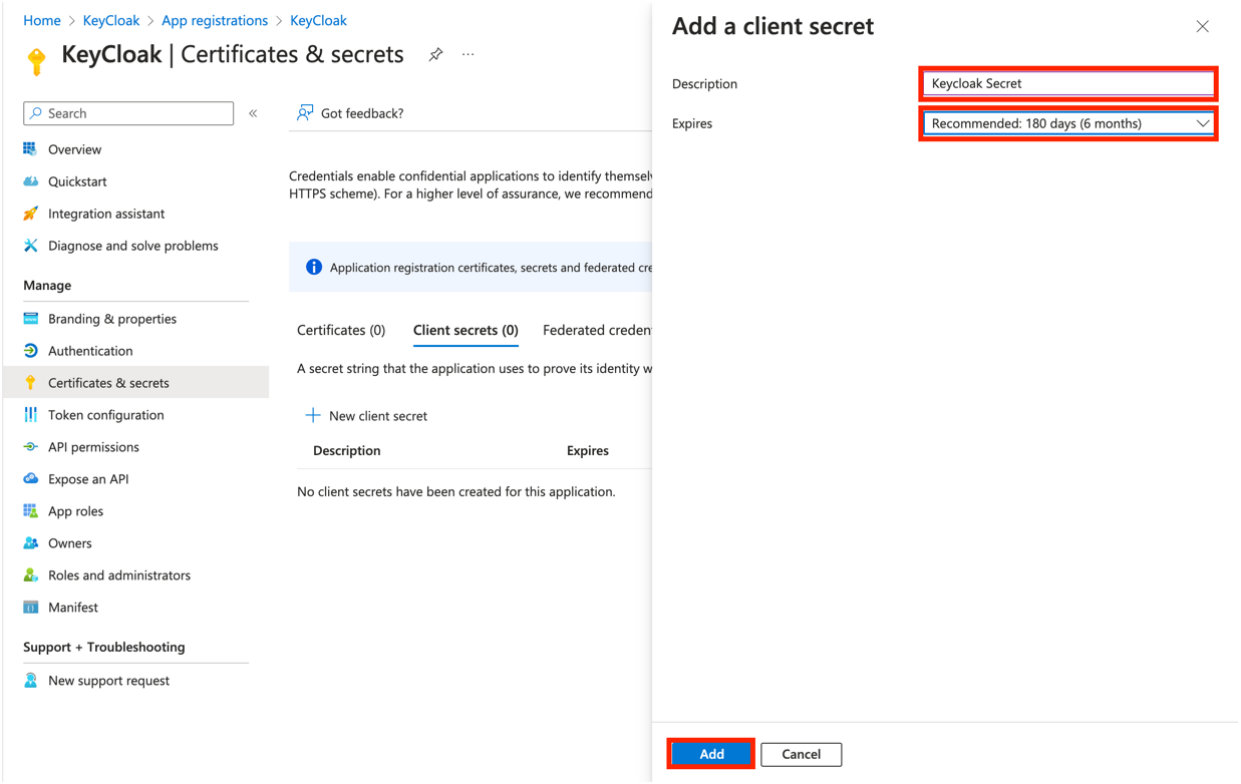
4. After you have registered your application, you will be returned to the Application registration's overview page.
   You need to save your `Application (client) ID`, since you will need it later.



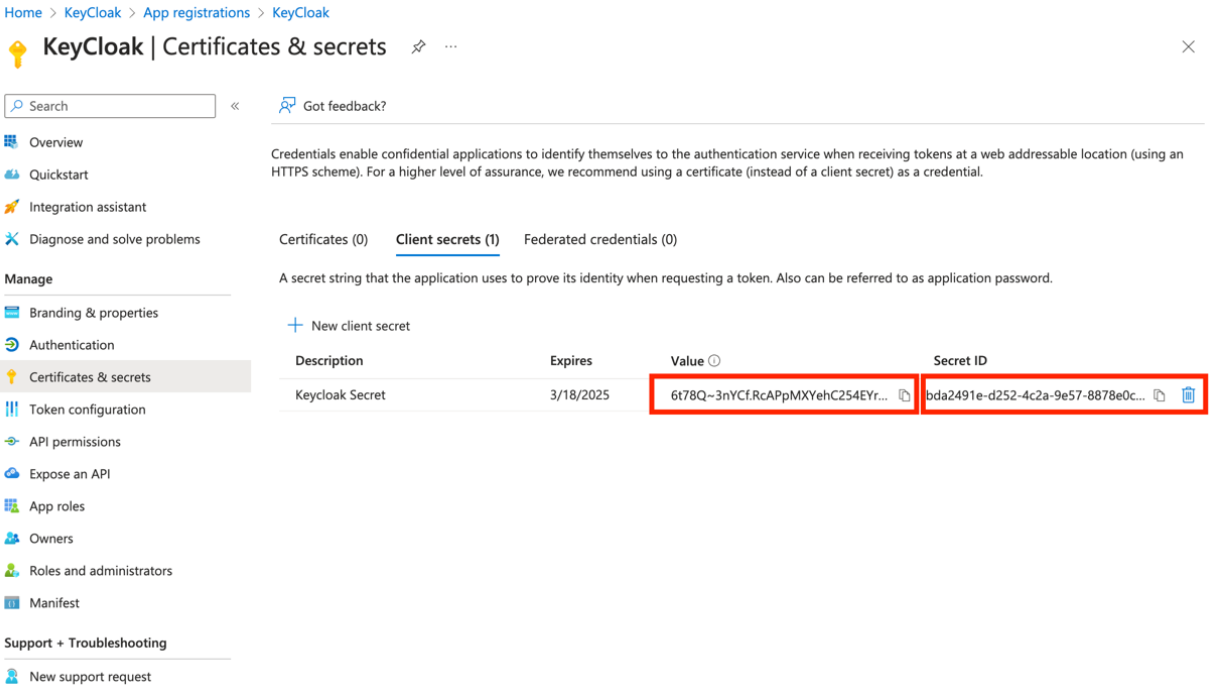
5. Create a Client Secret for your Application:
   - On the left side, navigate to `Certificates and secrets`. Create a `New client secret`.



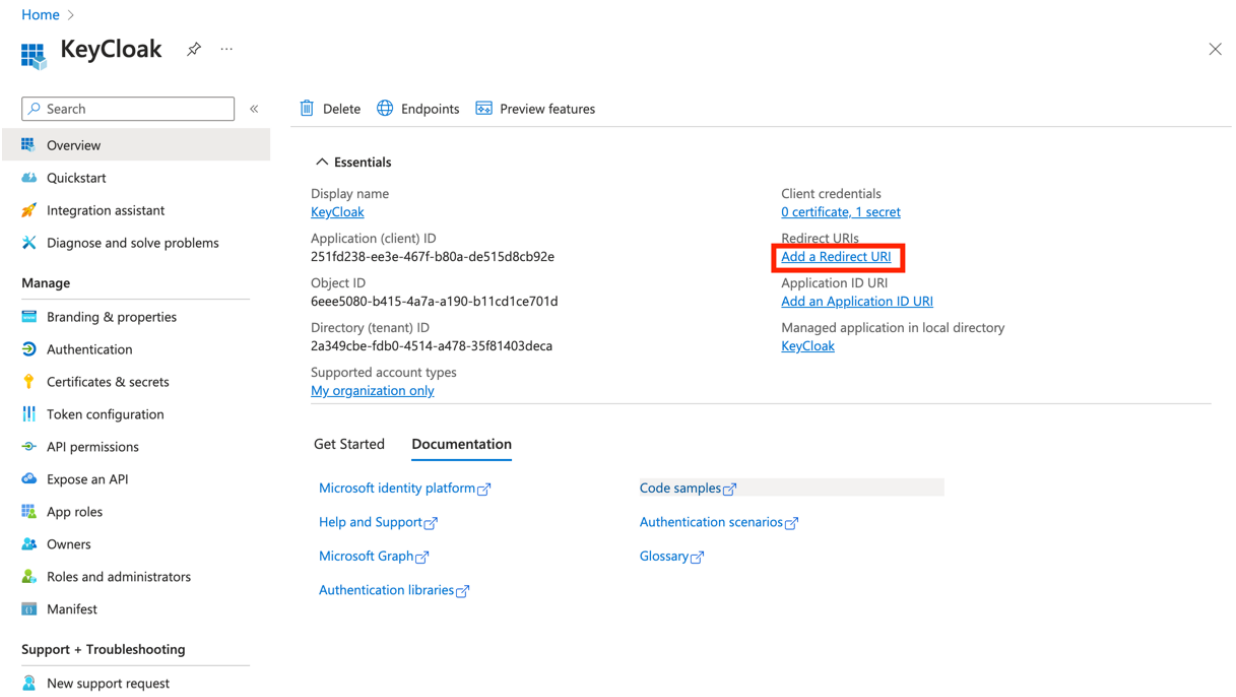
   - Add a description for your new secret and set an expiration for it.



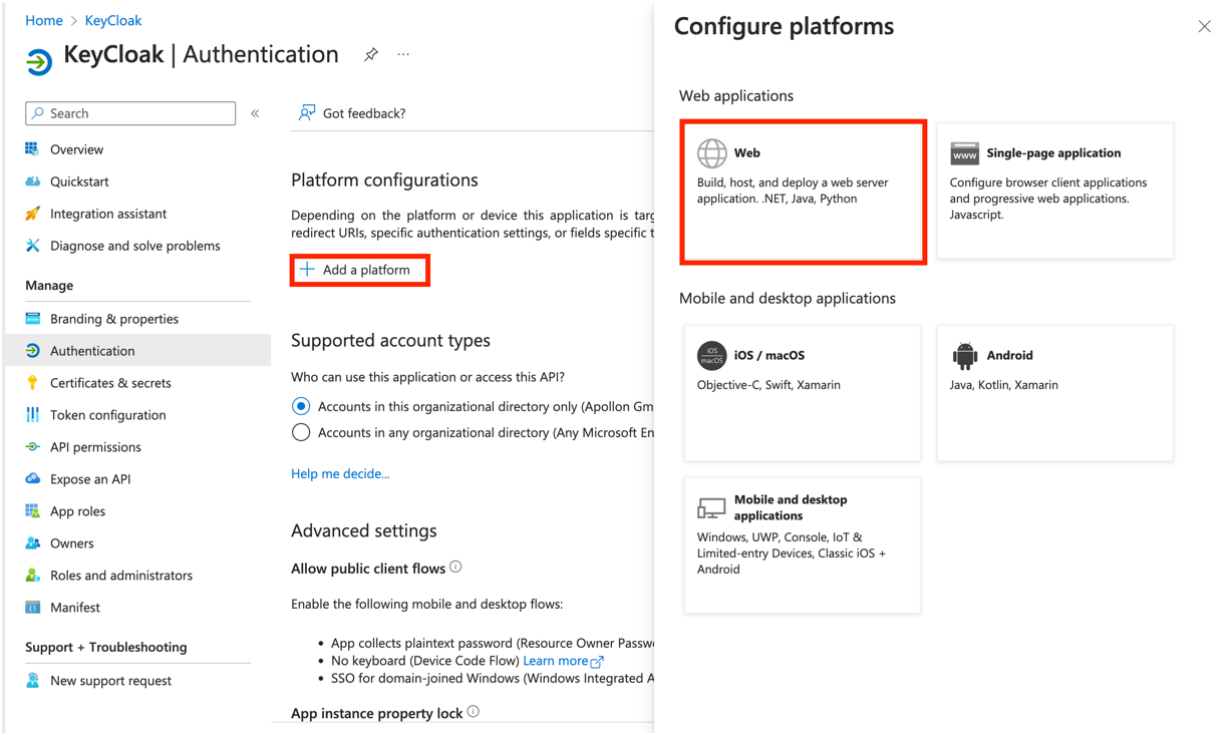
- Please save the `Value` and the `Secret ID` for later.



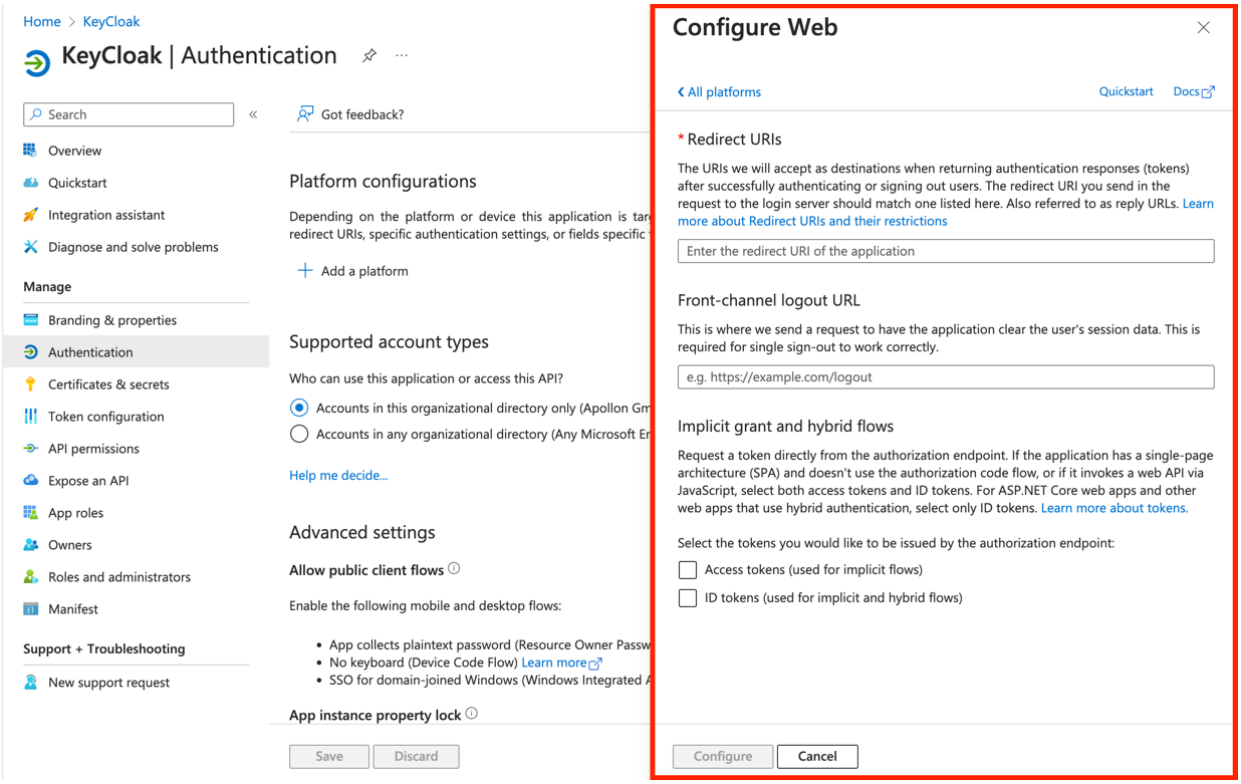
6. Add the Redirect URI generated from Keycloak to the Azure Application Registration.
   - Select `Add a Redirect URI`



- Select `Add a plattform` → `Web`



- Add your `Redirect URI` you saved when configuring your identity provider in Keycloak and paste it into the Redirect URI option for the platform you're configuring.



Keycloak

1. Create OpenID Connect v1.0 as an Identity Provider in your Realm
    - Log in to the Keycloak admin console and make sure the `OMN` realm is selected.
    - Create Realm and Client in Keycloak Administration Console

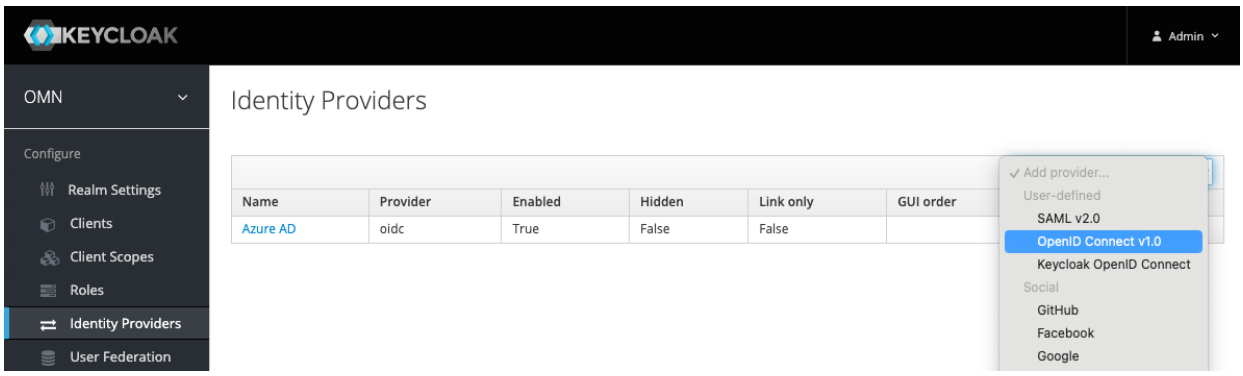
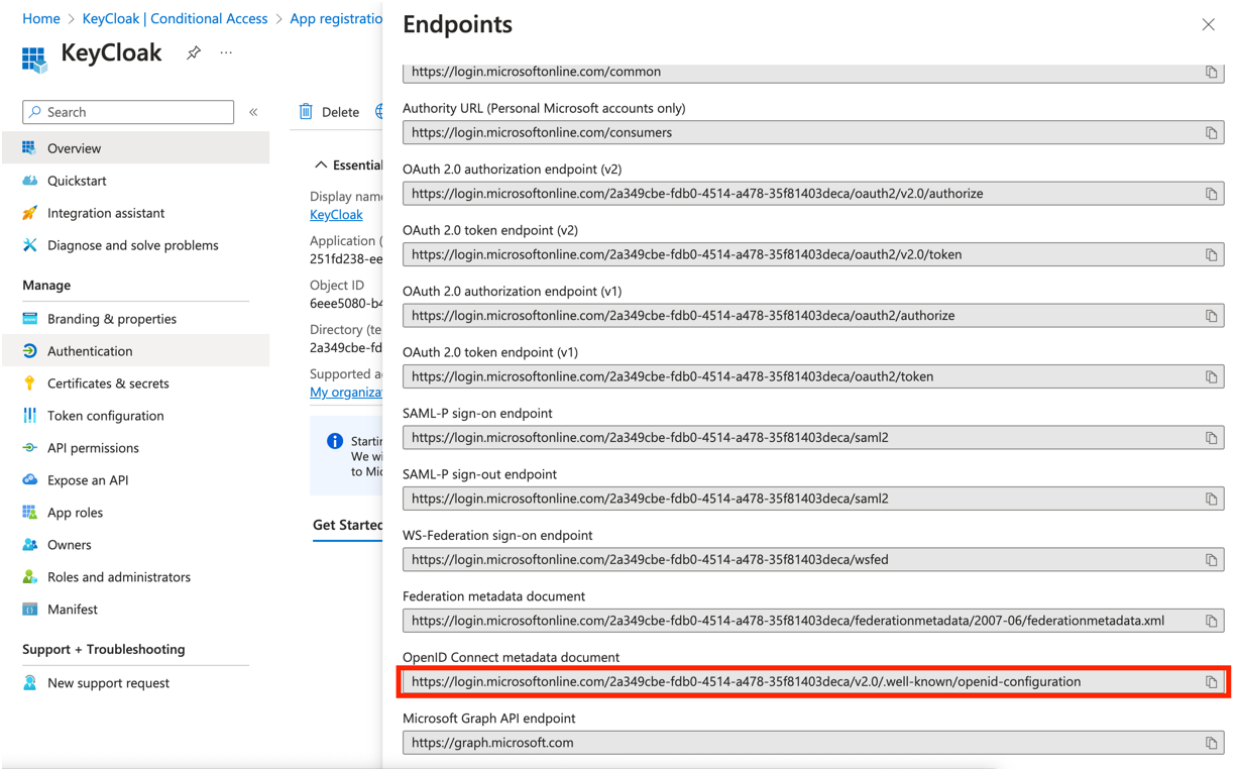


Figure 1. Create IDP with OpenID Connect Protocol

    - Select `Identity Providers` → `Add provider` → `OpenID Connect v1.0`



- Under `Import External IDP Config` you need to add the External link from Entra AAD. For this go to `Endpoints` and copy the `OpenID Connect metadata document` link.

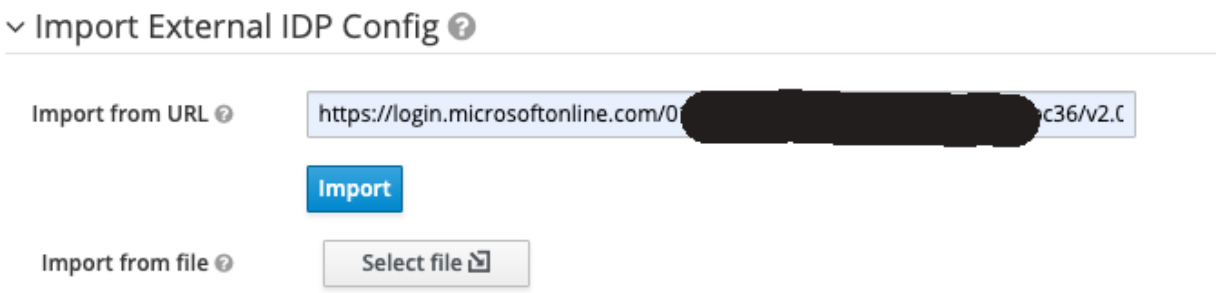


Figure 2. Import the metadata from IDP URL

2. Import the IDP configuration from the Azure AD metadata URL (Ask the customer for the metadata URL):
   - Fill the field **Import from URL** with the information you can find here, for example.
   - Click on the `Import` button.
3. Fill the **Alias** field with the expected IDP name, e.g. `azuread`, the `Redirect URI` field will be filled automatically.
4. Use the generated **Redirect URI** to register an application in the Azure AD side for getting the `Client ID` and `Client Secret` from Azure AD (Ask the customer for help).
5. Fill the **Client ID** and **Client Secret** fields with the values from the Azure AD application configuration.
6. Set **First Login Flow** to `OMN First Broker Login`
7. Set **Post Login Flow** to `IDP-PostLogin`
8. Set **Client Authentication** to `Client Secret sent as post`

9. Set **Prompt** to `login`
10. Set **Default Scopes** to `openid profile`
11. Click on the `Save` button

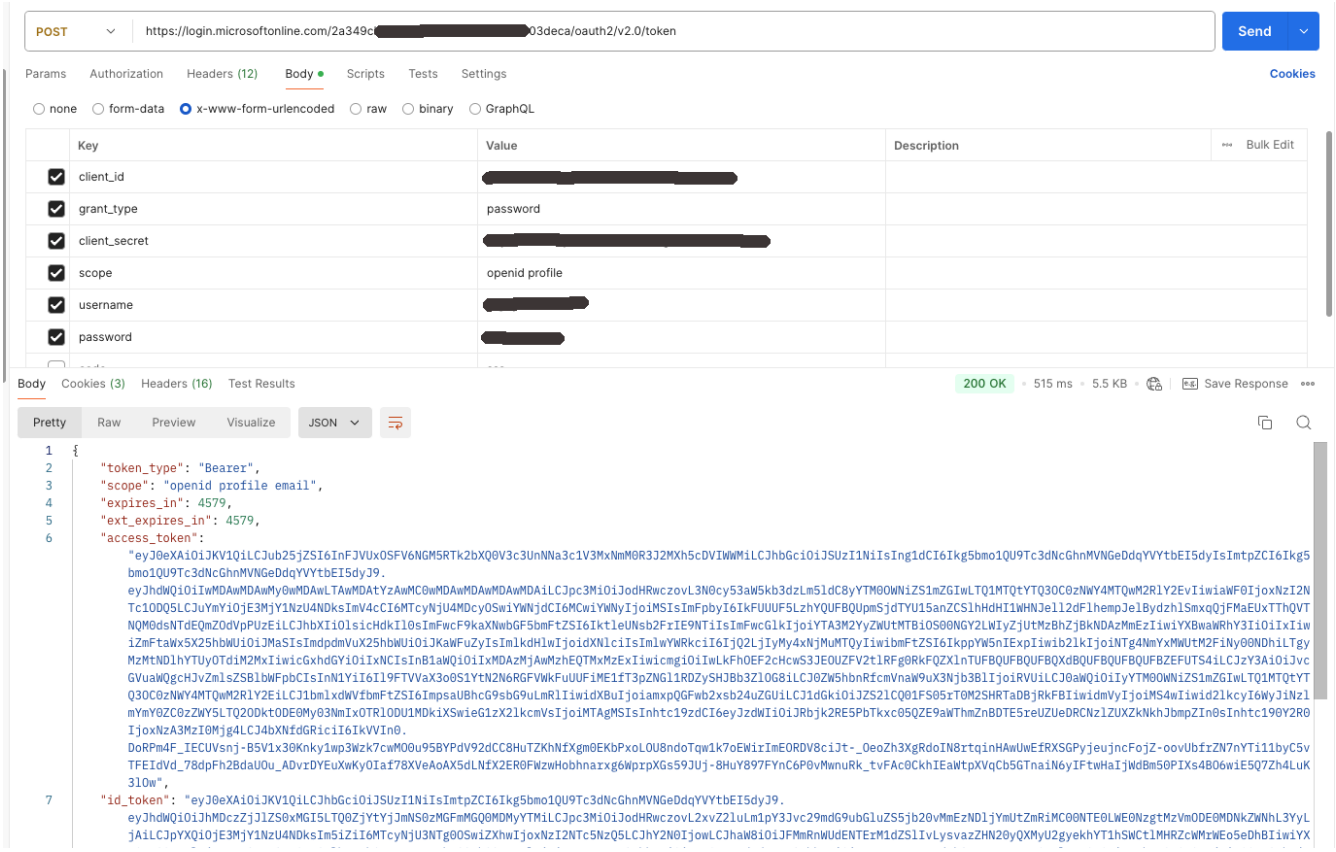
Validation of IDP configuration

To validate the IDP side configuration, you could use the tools like `Postman` or `Curl` to get the token from the token endpoint which could be found from the field `Token URL` in the IDP configuration above. Also replace the `client_id`, `client_secret`, `username`, and `password` with the values from the IDP.

Query token by Curl

```
curl --location 'https://login.microsoftonline.com/YOUR_TENANT_ID/oauth2/v2.0/token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=YOUR_CLIENT_ID' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'client_secret=YOUR_CLIENT_SECRET' \
--data-urlencode 'scope=openid profile' \
--data-urlencode 'username=YOUR_USERNAME' \
--data-urlencode 'password=YOUR_PASSWORD'
```

Query token by Postman


POST
https://login.microsoftonline.com/2a349c[redacted]03deca/oauth2/v2.0/token
Send
Params
Authorization
Headers (12)
Body
Scripts
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Key
Value
Description
Bulk Edit
client_id
grant_type
password
client_secret
scope
openid profile
username
password
Body
Cookies (3)
Headers (16)
Test Results
200 OK
515 ms
5.5 KB
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"token_type": "Bearer",
"scope": "openid profile email",
"expires_in": 4579,
"ext_expires_in": 4579,
"access_token":
"eyJ0eXAiOiJKV1QiLCJub25jZSI6InFJVUxOSFV6NGM5RTk2bXQ0V3c3UnNNa3c1V3MxNmM0R3J2MXh5cDVIWWMiLCJhbGciOiJSUzI1NiIsIng1dCI6Ikg5bmo1QU9Tc3dNcGhnMVNGeDdqYVYtbEI5dyIsImtpZCI6Ikg5bmo1QU9Tc3dNcGhnMVNGeDdqYVYtbEI5dyJ9.
eyJhdWQiOiIwMDAwMDAwMy0wMDAwLTAwMDAtYzAwMC0wMDAwMDAwMDAwMDAiLCJpc3MiOiJodHRwczovL3N0cy53aW5kb3dzLm5ldC8yYTM0OWNiZS1mZGIwLTQ1MTQtYTQ3OC0zNWY4MTQwM2RlY2EvIiwiaWF0IjoxNzI2NTc1ODQ5LCJuYmYiOjE3MjY1NzU4NDksImV4cCI6MTcyNjU4MDcyOSwiYWNjdCI6MCwiYWNyIjoiMSIsImFpbyI6IkFUUUF5LzhYQUFBQUpmSjdTYU15anZCSlhHdHI1WHNJell2dFlhempJelBydzhlSmxqQjFMaEUxTThQVTNQM0dsNTdEQmZOdVpPUzEiLCJhbXIiOlsicHdkIl0sImFwcF9kaXNwbGF5bmFtZSI6IktleUNsb2FrIE9NTiIsImFwcGlkIjoiYTA3M2YyZWUtMTBiOS00NGY2LWIyZjUtMzBhZjBkNDAzMmEzIiwiYXBwaWRhY3IiOiIxIiwiZmFtaWx5X25hbWUiOiJMaSIsImdpdmVuX25hbWUiOiJKaWFuZyIsImlkdHlwIjoidXNlciIsImlwYWRkciI6IjQ2LjIyMy4xNjMuMTQyIiwibmFtZSI6IkppYW5nIExpIiwib2lkIjoiNTg4NmYxMWUtM2FiNy00NDhiLTgyMzMtNDlhYTUyOTdiM2MxIiwicGxhdGYiOiIxNCIsInB1aWQiOiIxMDAzMjAwMzhEQTMxMzExIiwicmgiOiIwLkFhOEF2cHcwS3JEOUZFV2tlRFg0RkFQZXlnTUFBQUFBQUFBQXdBQUFBQUFBQUFBZEFUTS4iLCJzY3AiOiJvcGVuaWQgcHJvZmlsZSBlbWFpbCIsInN1YiI6Il9FTVVaX3o0S1YtN2N6RGFVWkFuUUFiME1fT3pZNGl1RDZySHJBb3ZlOG8iLCJ0ZW5hbnRfcmVnaW9uX3Njb3BlIjoiRVUiLCJ0aWQiOiIyYTM0OWNiZS1mZGIwLTQ1MTQtYTQ3OC0zNWY4MTQwM2RlY2EiLCJ1bmlxdWVfbmFtZSI6ImpsaUBhcG9sbG9uLmRlIiwidXBuIjoiamxpQGFwb2xsb24uZGUiLCJ1dGkiOiJZS2lCQ01FS05rT0M2SHRTaDBjRkFBIiwidmVyIjoiMS4wIiwid2lkcyI6WyJiNzlmYmY0ZC0zZWY5LTQ2ODktODE0My03NmIxOTRlODU1MDkiXSwieG1zX2lkcmVsIjoiMTAgMSIsInhtc19tdCI6eyJzdWIiOiJRbjk2RE5PbTkxc05QZE9aWThmZnBDTE5reUZUeDRCNzlZUXZkNkhJbmpZIn0sInhtc190Y2R0IjoxNzA3MzI0Mjg4LCJ4bXNfdGRiciI6IkVVIn0.
DoRPm4F_IECUVsnj-B5V1x30Knky1wp3Wzk7cwMO0u95BYPdV92dCC8HuTZKhNfXgm0EKbPxoLOU8ndoTqw1k7oEWirImEORDV8ciJt-_OeoZh3XgRdoIN8rtqinHAwUwEfRXSGPyjeujncFojZ-oovUbfrZN7nYTi11byC5v
TFEIdVd_78dpFh2BdaUOu_ADvrDYEuXwKyOIaf78XVeAoAX5dLNfX2ER0FWzwHobhnarxg6WprpXGs59JUj-8HuY897FYnC6P0vMwnuRk_tvFAc0CkhIEaWtpXVqCb5GTnaiN6yIFtwHaIjWdBm50PIXs4BO6wiE5Q7Zh4LuK
3lOw",
"id_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6Ikg5bmo1QU9Tc3dNcGhnMVNGeDdqYVYtbEI5dyJ9.
eyJhdWQiOiJhMDczZjJlZS0xMGI5LTQ0ZjYtYjJmNS0zMGFmMGQ0MDMyYTMiLCJpc3MiOiJodHRwczovL2xvZ2luLm1pY3Jvc29mdG9ubGluZS5jb20vMmEzNDljYmUtZmRiMC00NTE0LWE0NzgtMzVmODE0MDNkZWNhL3YyL
jAiLCJpYXQiOjE3MjY1NzU4NDksIm5iZiI6MTcyNjU3NTg0OSwiZXhwIjoxNzI2NTc5NzQ5LCJhY2N0IjowLCJhaW8iOiJFMmRnWUdENTErM1dZSlIvLysvazZHN20yQXMyU2gyekhYT1hSWCtlMHRZcWMrWEo5eDhBIiwiYX

Then check the following points:

- The access token and ID token should be returned successfully.
- The user profile data should be included in the ID token.
- The user groups should be included in the ID token.
- The `login_hint` claim should be included in the ID token (optional).

You can use https://jwt.io to decode the ID token and check above mentioned points.

Mapping Activation for OMN Login

Normally in OMN a new created user is not to be activated by default.
In the case of IDP login, the users could be activated automatically by mapping to the built-in role `omn-login-role`.

Go to the `Mappers` tab and create a new mapper with the following settings:

- **Name**: `omn-login-role-mapper`
- **Sync Mode**: `Force`
- **Mapper Type**: `Claim to Role`
- **Claim**: `groups`
  ⇒ The claim name which provides the user groups from Azure AD
- **Claim Value**: `your-target-group-from-azure-ad`
  ⇒ The group name whose members should be activated in OMN automatically.
- **Role**: `omn-login-role`

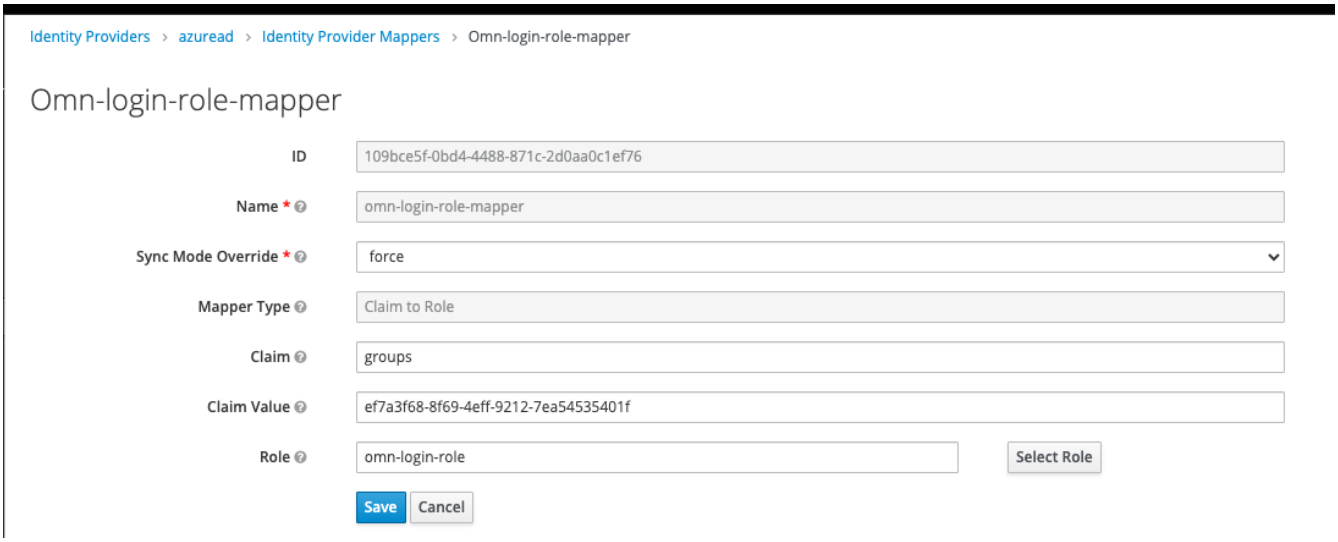


Figure 4. Mapping Activation State

If there are multiple groups in Azure AD which should be activated in OMN, you can create multiple mappers with different `Claim Value` settings for each group.

After that, all users from the configured groups will be activated automatically and others have to be activated manually by the administrator.

This automatic activation will be only executed at the first IDP login.
After that, the user could be activated or deactivated only by administrator manually.

Mapping IDP User Profile

An OMN user has own profile properties which could be also mapped from the IDP user profile data.
OMN has the following standard profile properties:

- userName
- firstName
- lastName
- email
- phone
- fax
- zip
- address
- city
- country
- department
- company

Keycloak normally does the mapping automatically for the standard claims like `username`, `email`, `given_name`, `family_name`, etc.
Others, e.g. company, should be mapped manually following the next steps:

Go to the `Mappers` tab and create a new mapper with the following settings:

- **Name**: `omn-userinfo-company`
- **Sync Mode**: `Force`
- **Mapper Type**: `Attribute Importer`
- **Claim**: `companyName`
  ⇒ The claim name matching the `company name` from Azure AD
- **User Attribute Name**: `company`
  ⇒ The OMN user profile property name from the above properties list

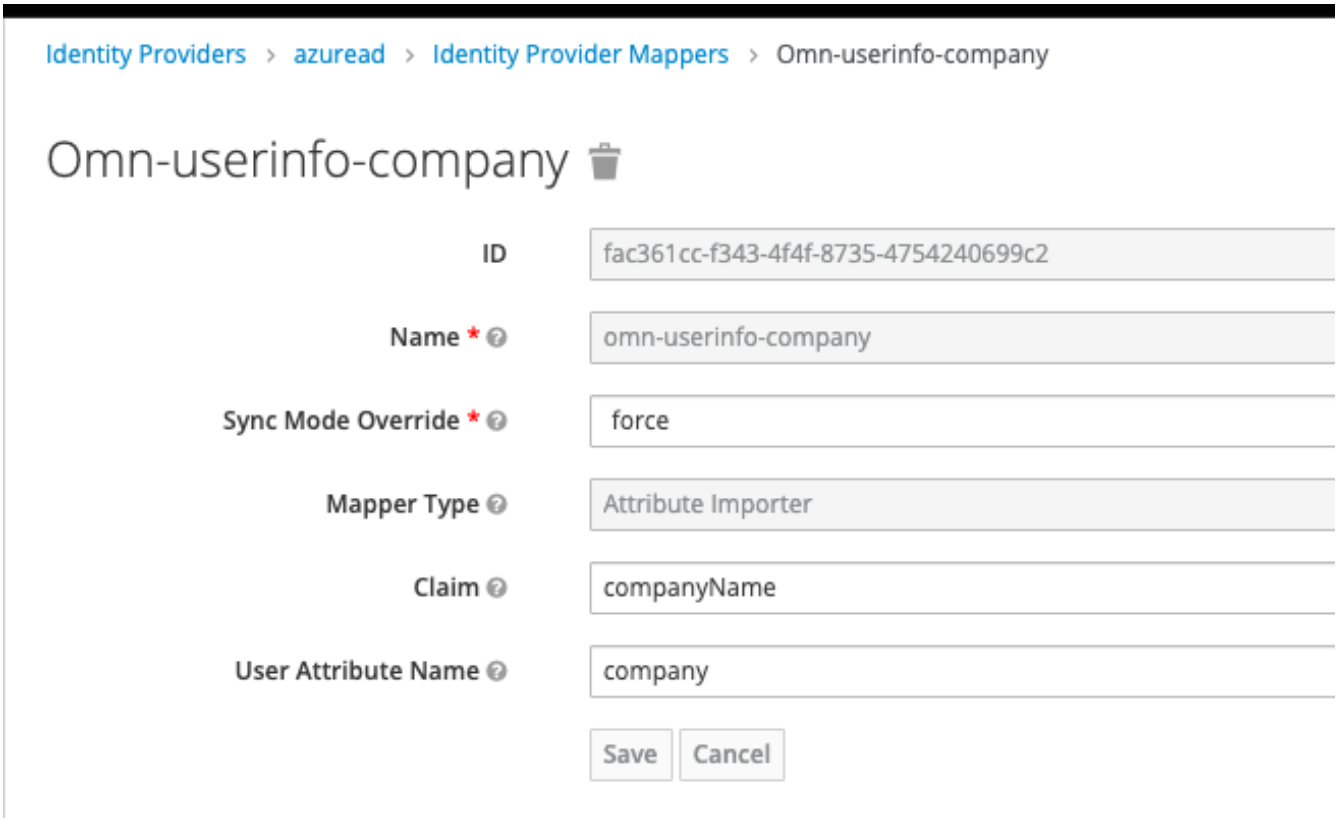


Figure 5. Mapping User Profile

After that, the `company` property will be filled with the value from the Azure AD user profile. The same could be done for other user profile properties.

The user profile mapping will be executed at each IDP login so that the user profile data will be always up-to-date.

Mapping IDP User Groups

OMN has user groups with different access rights, that could be also mapped from the IDP user groups. For example, the user from the group `admin` in Azure AD could be mapped to the `admin` group in OMN.

Before the mapping, the groups should be synchronized from the OMN LDAP into Keycloak.

Go to User `Federation` → `omn-ldap` → `Mappers` → `omn-ldap-group`, and click on `Sync LDAP Groups to Keycloak` button.

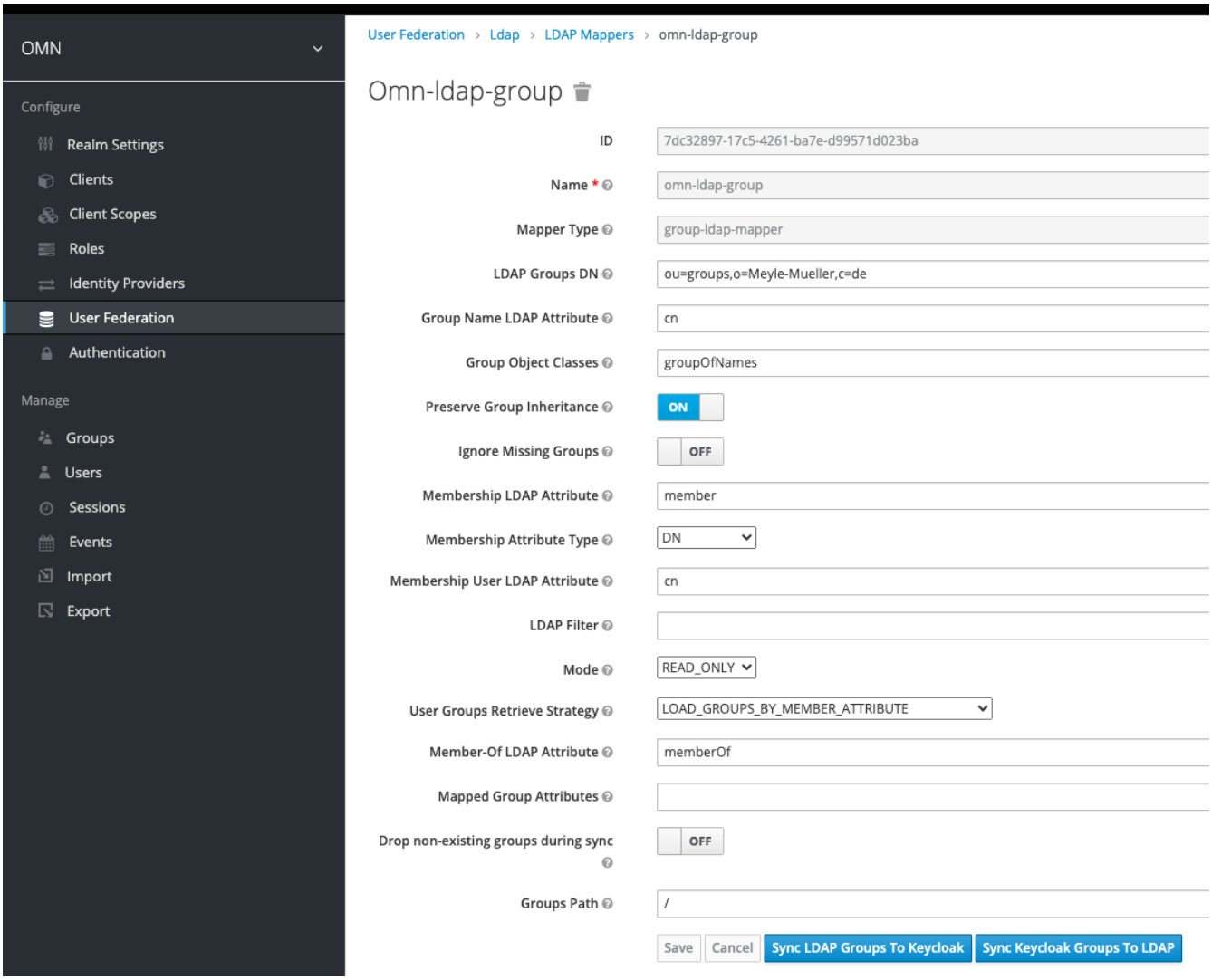


Figure 6. Sync OMN Groups

Go to `Identity Providers` → `Azure AD`, then open the `Mappers` tab and create a new mapper with the following settings:

- **Name**: `omn-group-admin-mapper`
- **Sync Mode**: `Force`
- **Mapper Type**: `Advanced Claim to Group`
- **Claims**: ⇒ add a claim key-value pair key: `groups` ⇒ The claim name which provides the user groups from Azure AD value: `admin-IDP` ⇒ The IDP group name which needs to be mapped to the OMN group
- **Group**: `manager-gui`
  ⇒ Choose the target OMN group by clicking on the `Select Group` button

The same could be done for other groups.

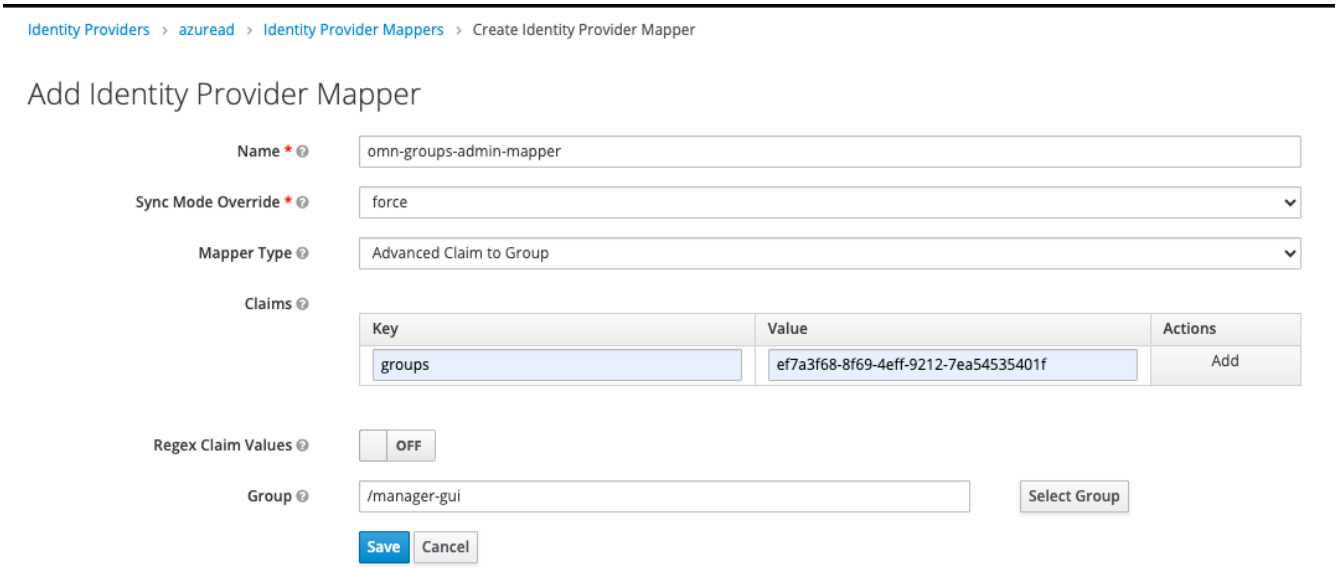
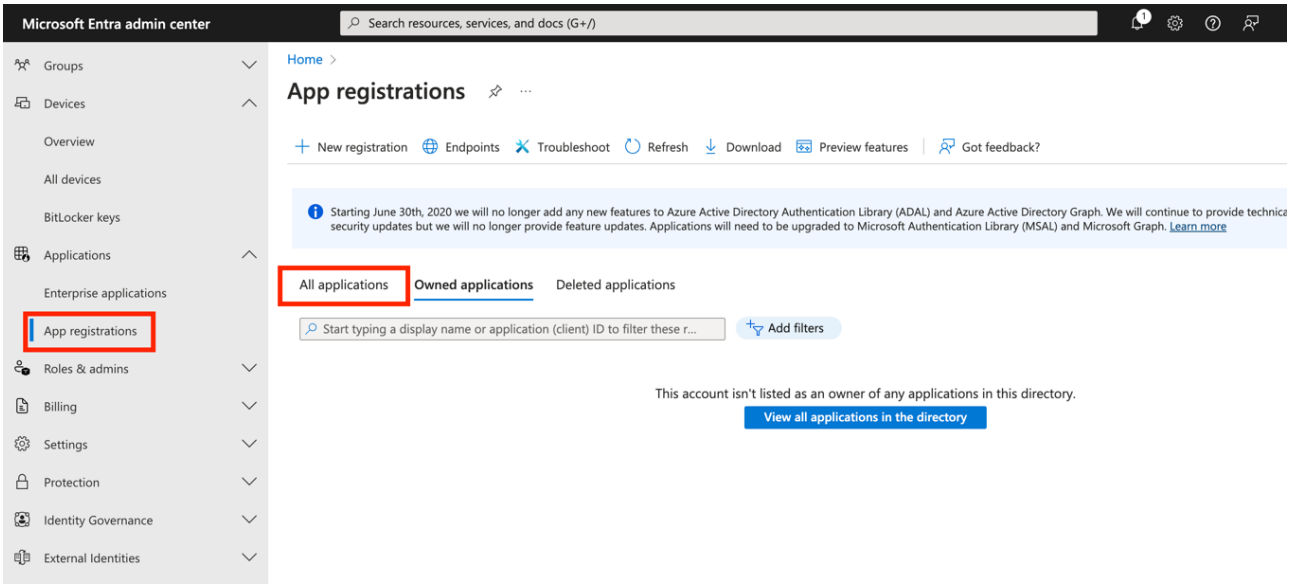


Figure 7. Mapping Group

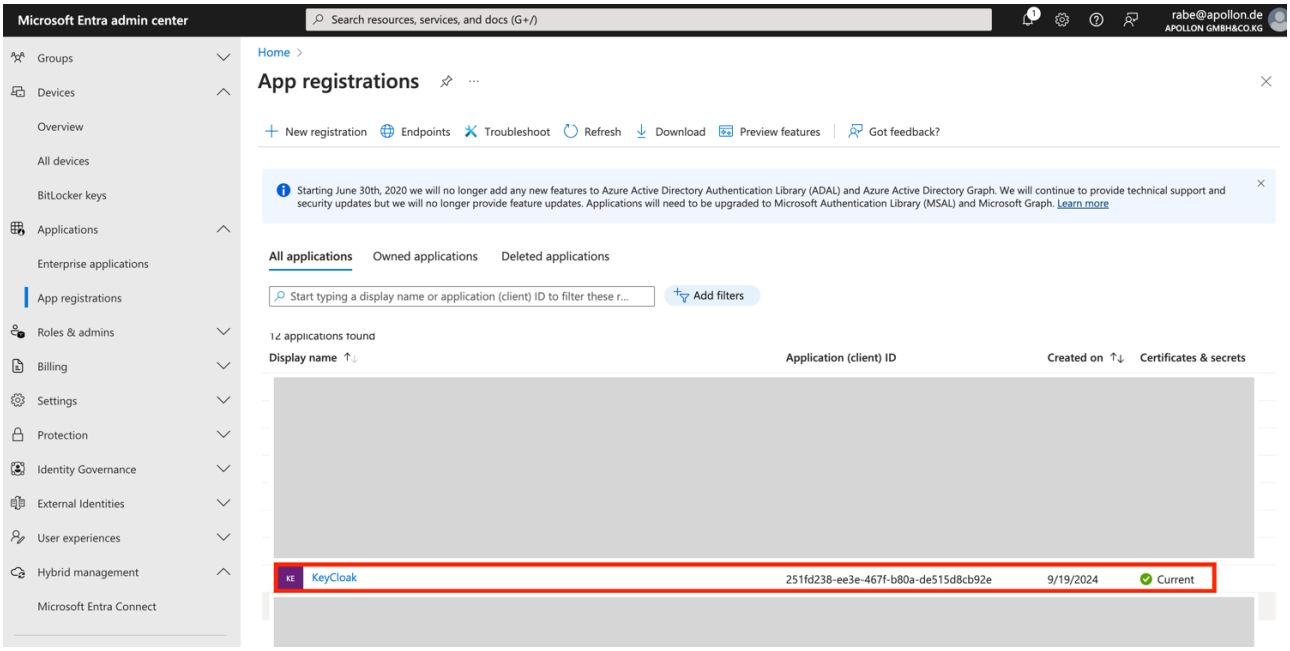
Configure ID Token with additional user information (Azure AD)

The following example shows the configuration or additional claims. In this case, we configure the IDP Token endpoint to print the user groups, additionally.

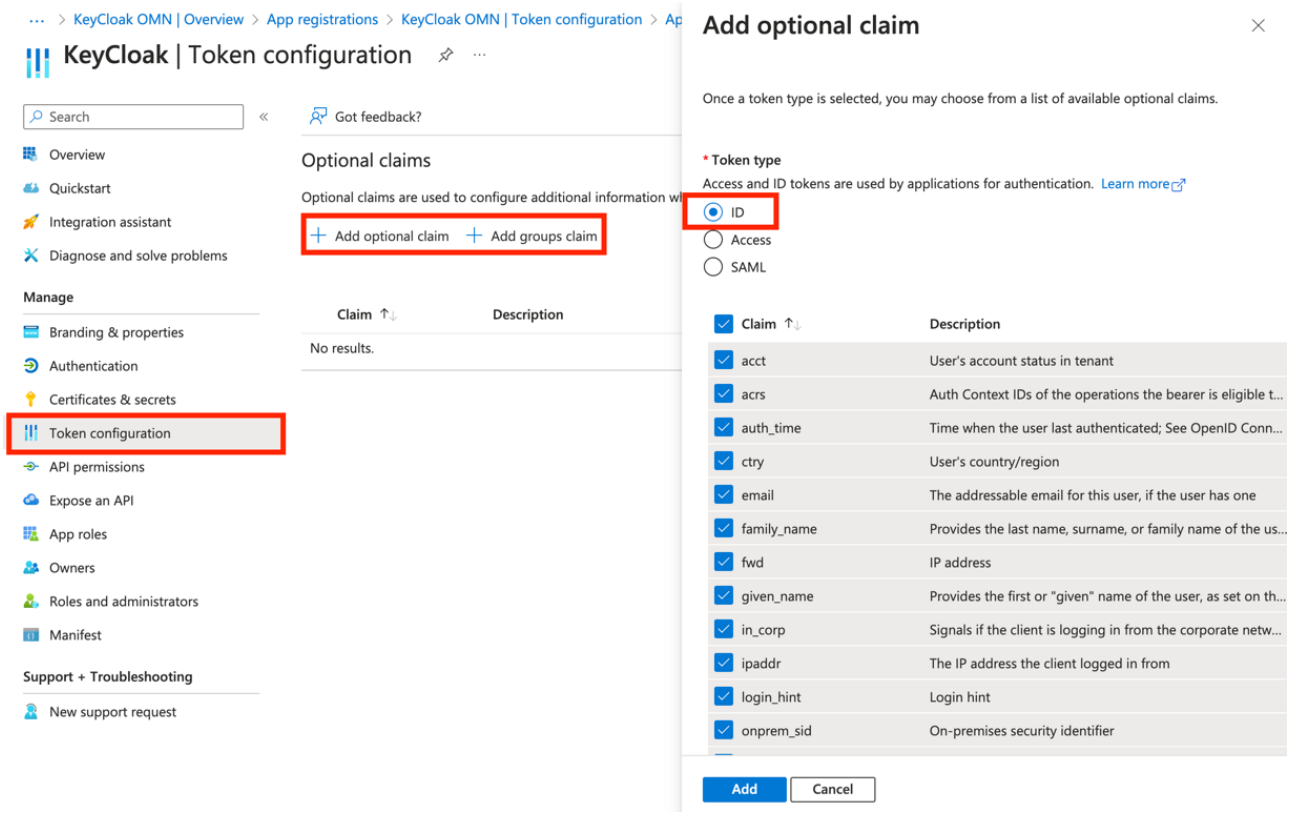
1. Log in to your admin console and navigate to the Applications section:
2. Select `Applications` → `App Registrations` → `All applications`



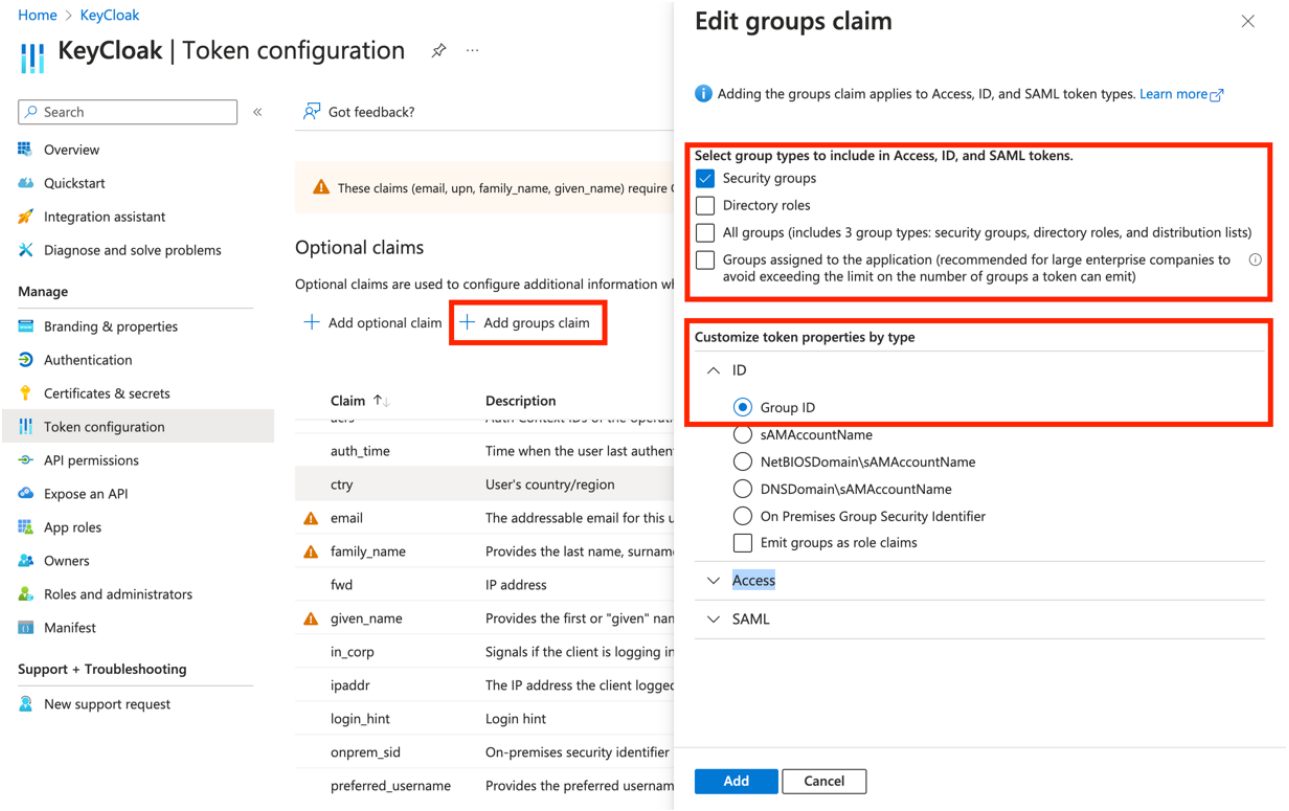
3. Select the `KeyCloak` app instance created previously.



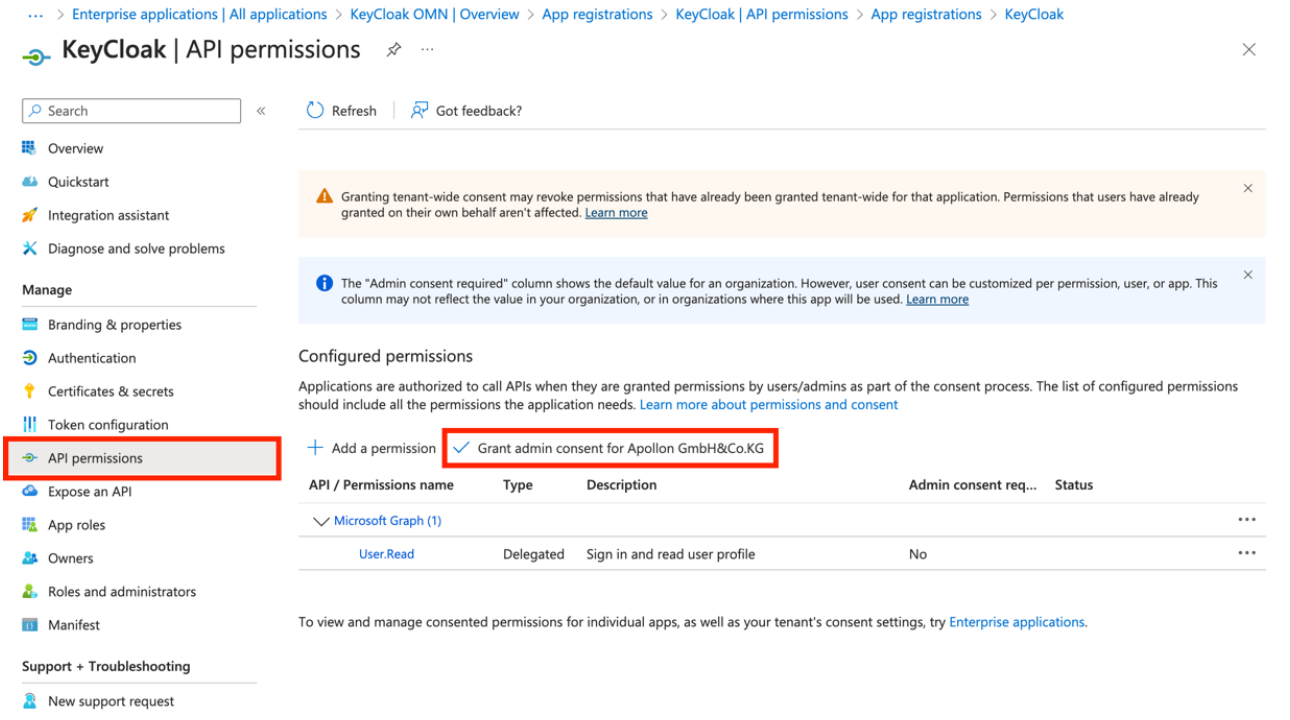
4. Go to `Token configuration` → `Add optional Claim` → `ID` → Select needed claims, e.g. Claim all and Click `Add` Button.

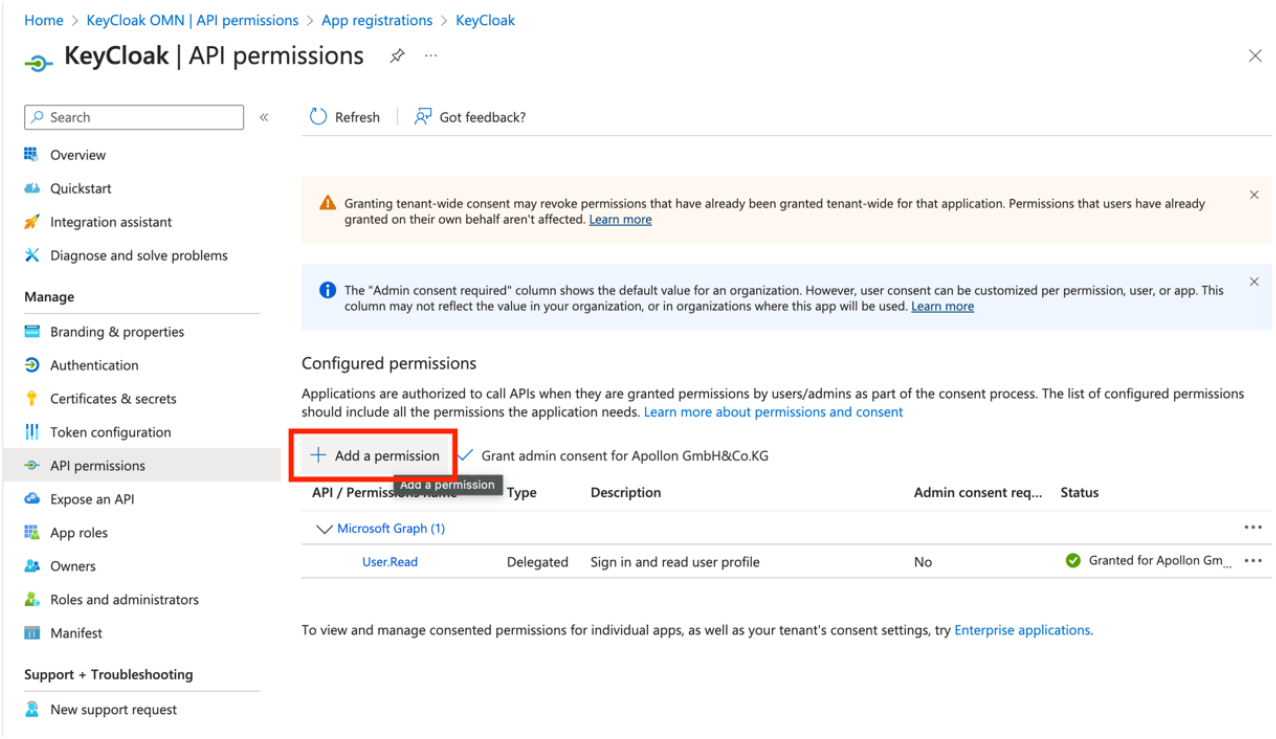


5. Go to `Add groups claim` → select e.g. `Security groups` and the token ID with e.g. `Group ID`. Then Click the `Add` button.

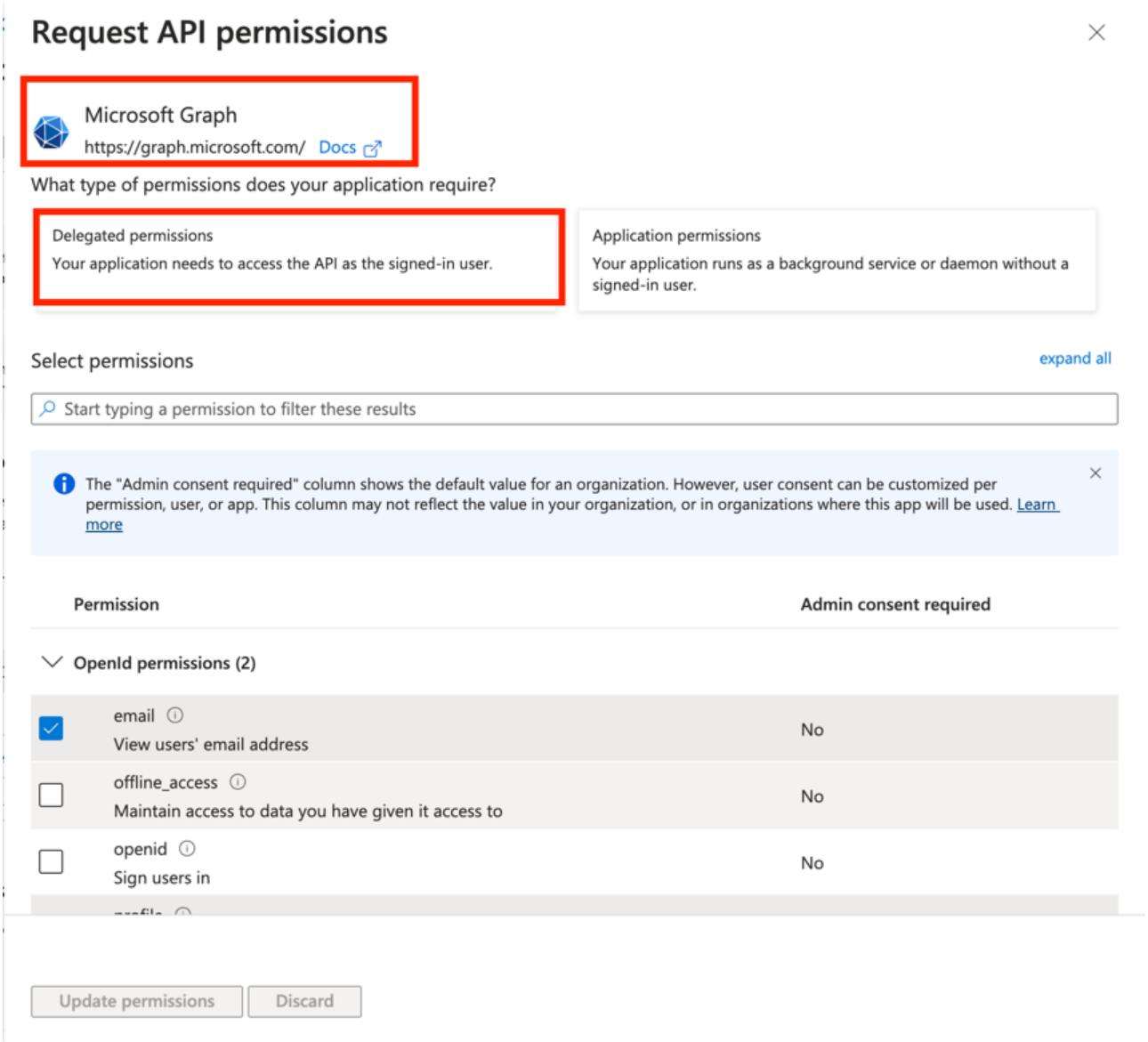


6. Go to `API permissions` → `Add a permission`

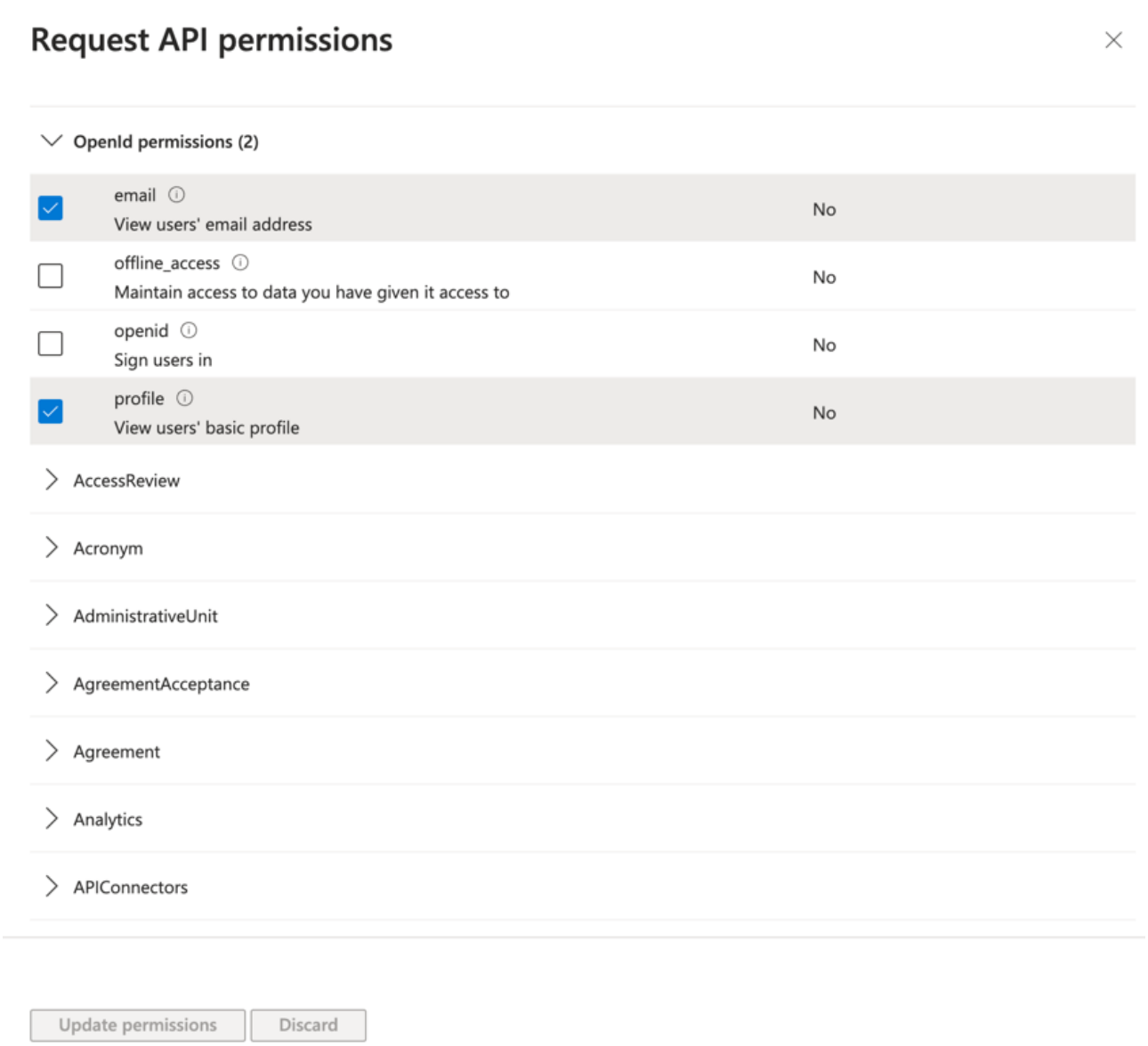




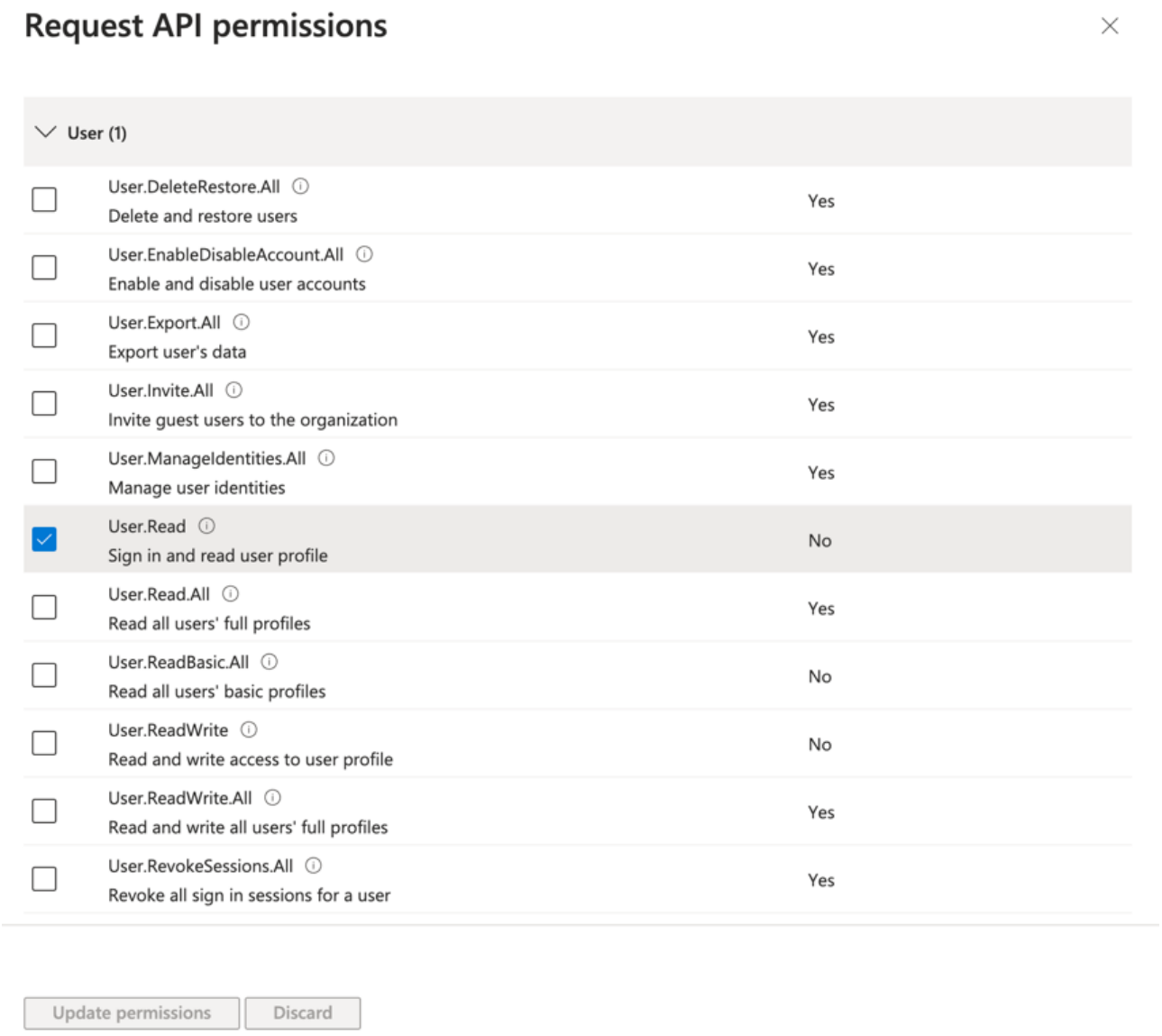
7. Go to `Microsoft Graph` → `add a permission`
8. Go to `Microsoft Graph` → `Delegated permission` is selected.



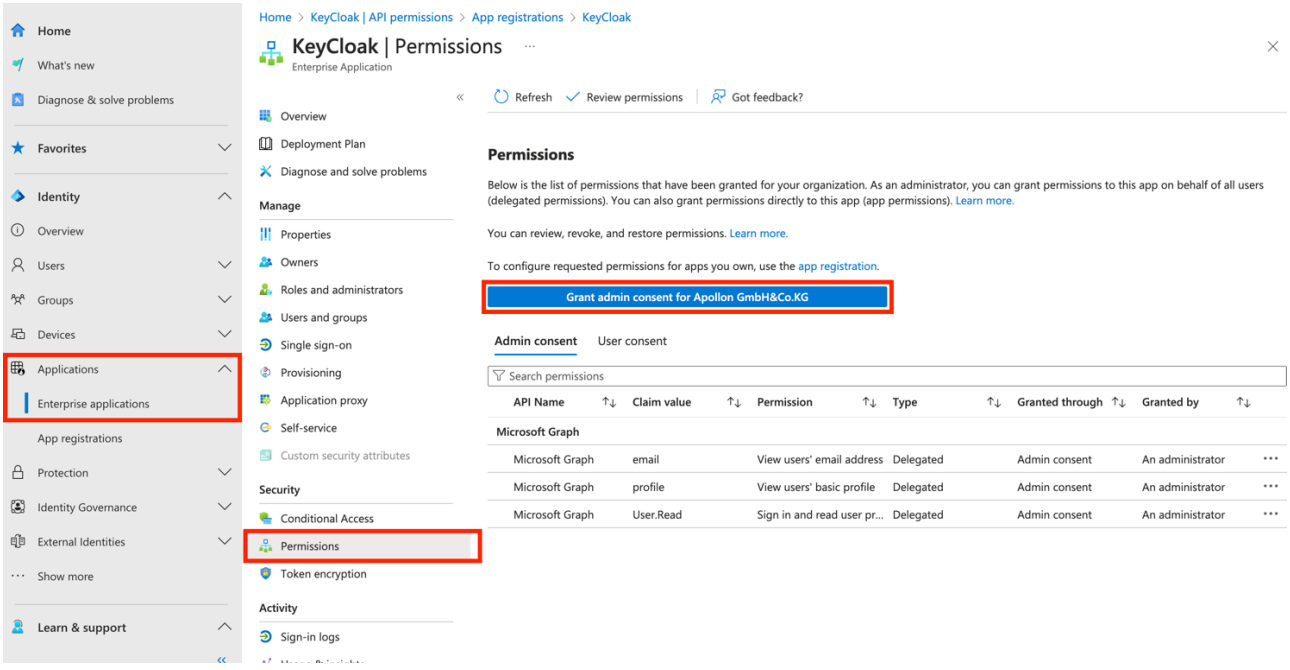
9. There you need the following permission set:
   `OpenId permissions`: check that `email` and `profile` is selected.



10. In `User` check that `User.Read` is selected.



11. Navigate to Applications→ Enterprise applications → Permissions and Grant the Admin Consent⋯.



12. Accept the registration.

This is just an example of a test configuration. More information for Configure optional claims from Microsoft.