Monitoring Expiring Client Secrets in Azure Logic Apps

I want to share how Logic Apps can help address the challenge of expiring client secrets in Azure. As part of my Logic App flows, I leverage App Registrations in Azure for authentication, relying on client secrets. However, these secrets have a tendency to expire, which can lead to disruptions in my flows if not addressed in a timely manner. To proactively manage this issue, I needed a solution to receive notifications before client secrets expire. Unfortunately, there wasn't a built-in solution available, so I turned to Logic Apps once again to tackle this problem by leveraging the Graph API.

With the Graph API, I can retrieve crucial information about App Registrations in Azure, including the end dates of client secrets. Armed with this information, I decided to create a Logic Apps flow that would send me a notification a few weeks before a secret's expiration date. By using the available data via the Graph API, I could filter and identify expiring client secrets. These notifications could then be sent via email or routed to a Teams channel.

By building this Logic Apps flow, I gained visibility into the status of client secrets and ensured that I could take timely action to prevent any disruptions in my flows. This approach provided a custom solution to an otherwise unaddressed challenge, and the information obtained from the Graph API proved instrumental in implementing this functionality within Logic Apps.

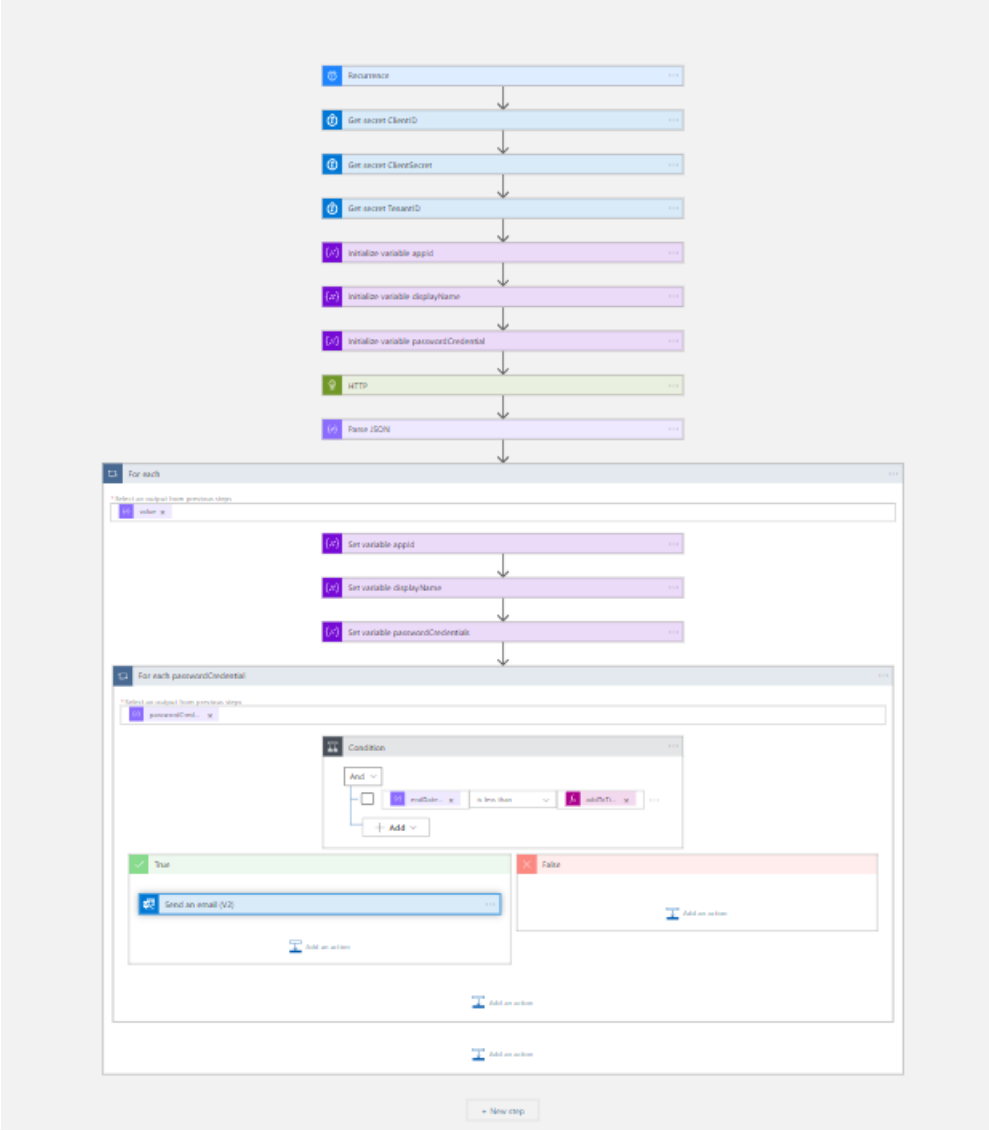
To learn more about the specifics of the implementation and see a step-by-step guide, please follow this documentation. I have also included the complete Logic Apps flow in an ARM Template, which can be downloaded from my GitHub repository.

Key Highlights:

- App Registrations in Azure are crucial for authenticating Logic App flows.
- Client secrets associated with these App Registrations can expire, causing flow disruptions.
- An out-of-the-box solution for managing expiring client secrets is not available.
- The Graph API provides access to information about client secrets, including their end dates.
- A Logic Apps flow can be designed to monitor and notify about expiring client secrets.
- Notifications can be sent via email or Teams channel.
- The implementation requires appropriate permissions and access to necessary Azure resources.

For a detailed walkthrough of setting up the Logic Apps flow and addressing this challenge, please refer to document. The document provides step-by-step instructions, screenshots, and additional insights into the solution's architecture and implementation.

This is what the complete flow looks like.



This flow monitors all App Registration client secrets.

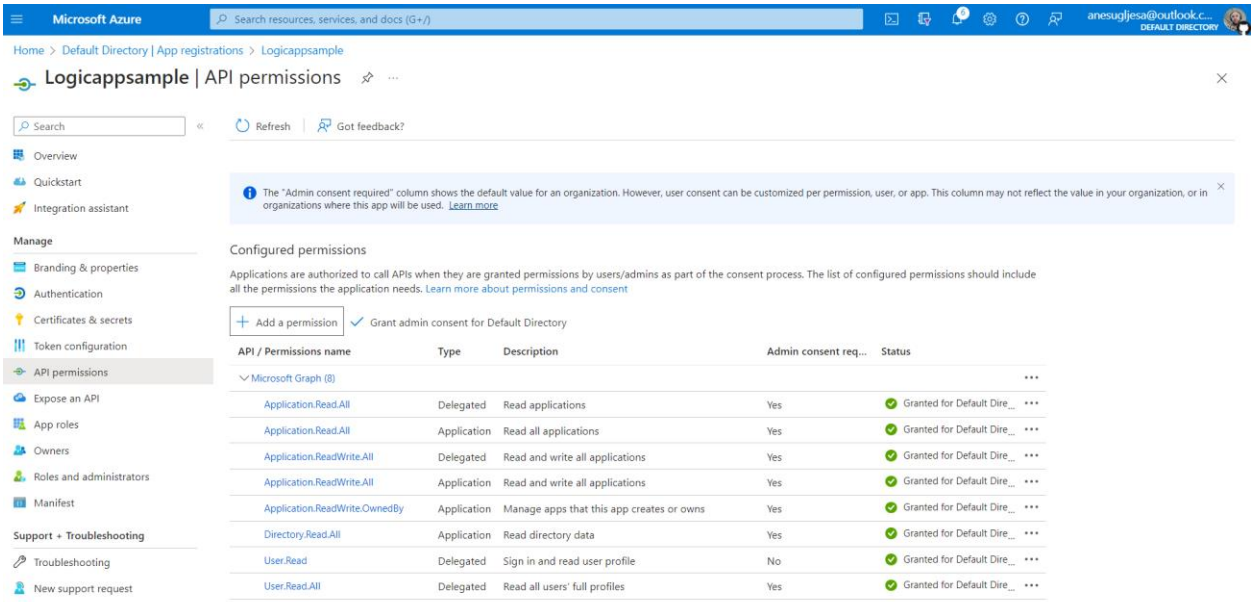
The requirements

To get this logic Apps flow up and running we have a few **requirements**.

Via an **HTTP** action, we're going to query Graph API for the information we need. This HTTP uses an **Azure App Registration**, which is our first requirement.

The App Registration is used for authentication in the HTTP actions and also for granting the required permissions. The minimum required **permission** for this Logic App is shown: *Application.Read.All, Application.ReadWrite.OwnedBy, Application.ReadWrite.All, Directory.Read.All*

Learn more on link: https://learn.microsoft.com/en-us/graph/api/application-list?view=graph-rest-1.0&tabs=http



To secure the HTTP action I use **Azure Key Vault**. The Key Vault holds the **client secret** and keeps this secure in the flow. I also added the **tenant id** and **client id** to the vault, so I only have to query the Key vault for this information.

This flow only uses one HTTP action, if you have several HTTP actions in a flow, in every HTTP action the tenant and client id needs to be added. By storing the information in a variable or in a Key Vault, we don't have to copy/ paste the ids in every HTTP action.

Microsoft Azure | Search resources, services, and docs (G+/) | anesugljesa@outlook.c... DEFAULT DIRECTORY

Home > logic-app-prod

logic-app-prod | Secrets

Key vault

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Access policies
- Events

Objects

- Keys
- Secrets

Generate/Import | Refresh | Restore Backup | View sample code | Manage deleted secrets

Name	Type	Status	Expiration date
ClientID		✓ Enabled	
ClientSecret		✓ Enabled	
TenantID		✓ Enabled	

Microsoft Azure | Search resources, services, and docs (G+/) | anesugljesa@outlook.c... DEFAULT DIRECTORY

Home > logic-app-prod

logic-app-prod | Access configuration

Key vault

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Access policies
- Events

Objects

- Keys
- Secrets
- Certificates

Settings

- Access configuration
- Networking
- Microsoft Defender for Cloud

Refresh

Configure your options on access policy for this key vault

To access a key vault in data plane, all callers (users or applications) must have proper authentication and authorization. Authentication establishes the identity of the caller. Authorization determines which operations the caller can execute. Learn more

Permission model

Grant data plane access by using a Azure RBAC or Key Vault access policy

- ○ Azure role-based access control (recommended)
- ◉ Vault access policy

Go to access policies

Resource access

Choose among the following options to grant access to specific resource types

- ☐ Azure Virtual Machines for deployment
- ☐ Azure Resource Manager for template deployment
- ☐ Azure Disk Encryption for volume encryption

Microsoft Azure | Search resources, services, and docs (G+/) | anesugljesa@outlook.c... DEFAULT DIRECTORY

Home > logic-app-prod

logic-app-prod | Access policies

Key vault

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Access policies
- Events

Create | Refresh | Delete | Edit

Access policies enable you to have fine grained control over access to vault items. Learn more

Search | Permissions : All | Type : All

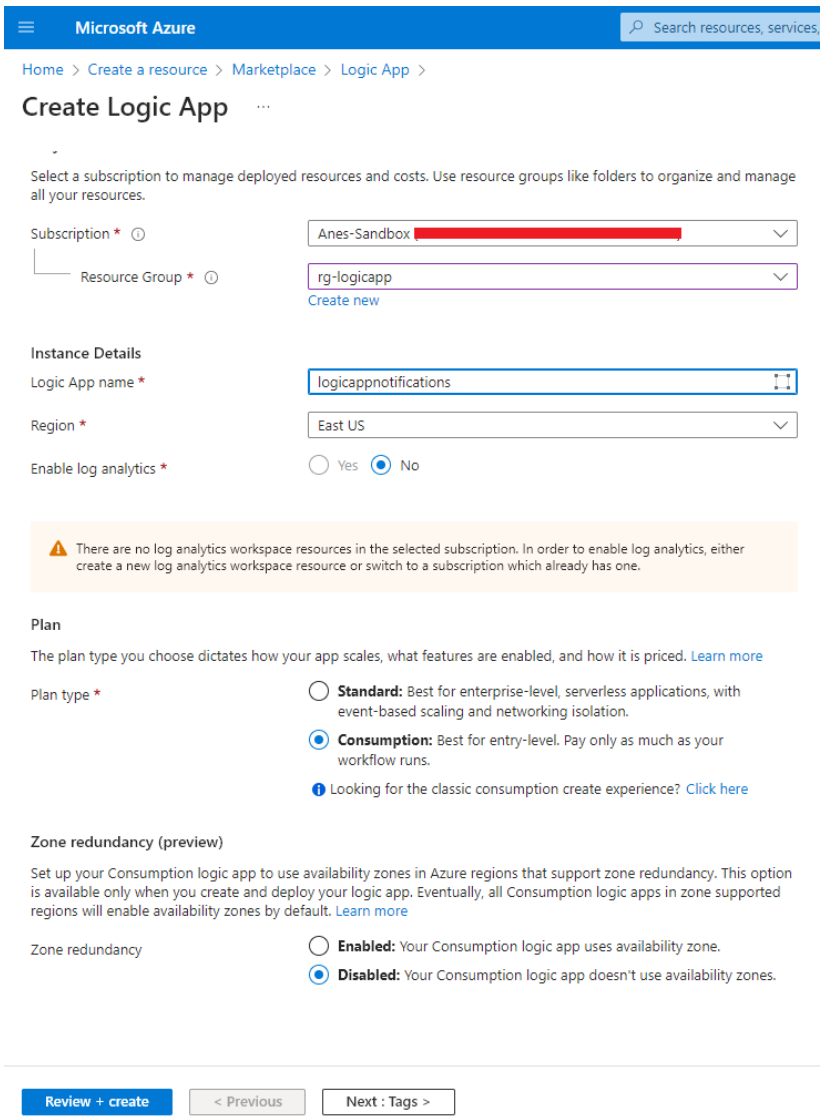
Showing 1 to 1 of 1 records.

Name	Email	Key Permissions	Secret Permissions	Certificate Permissions
USER				
Anes Ugljesa	[redacted]		All	

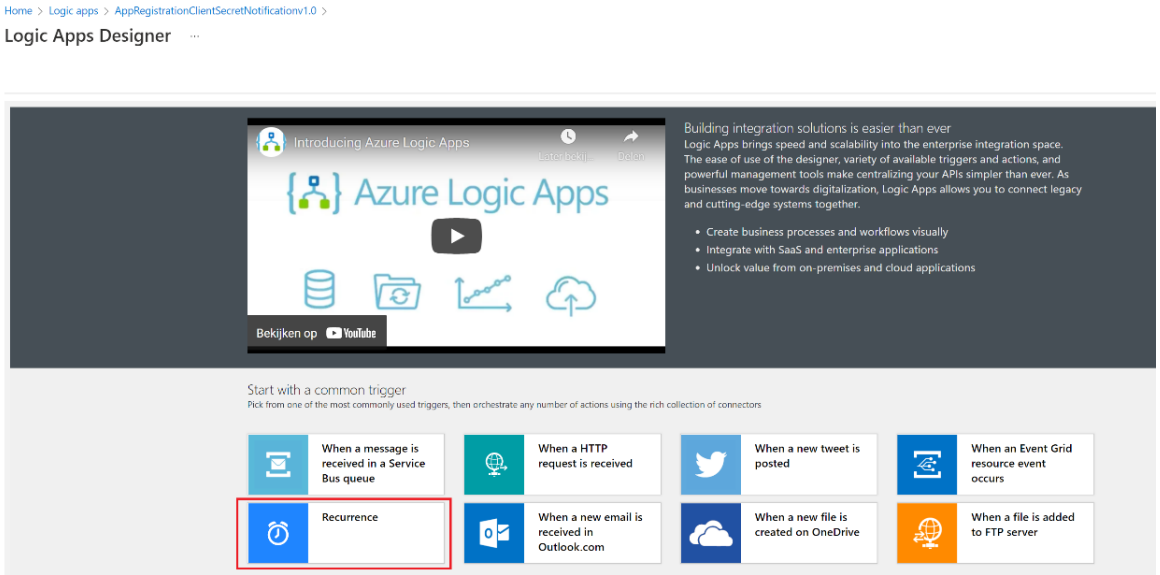
Setup the Logic Apps flow

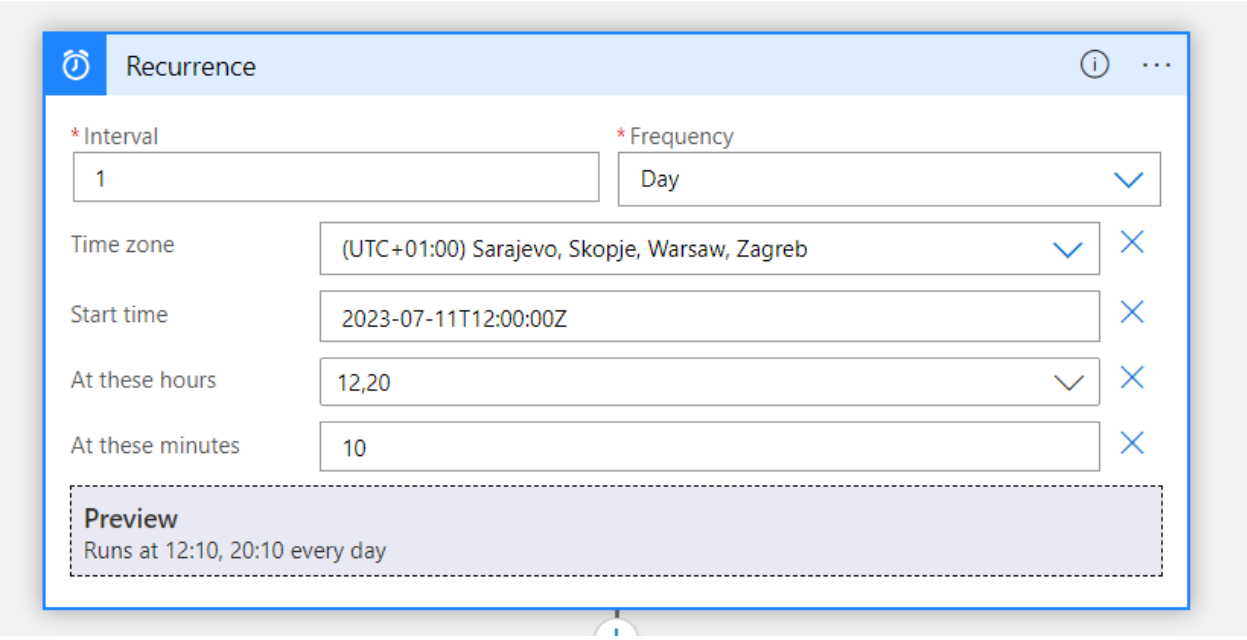
Let's configure the Logic Apps flow.

Sign in to the [Azure portal](#) and open the **Logic App** service. I created a blank Logic App of **type Consumption**.



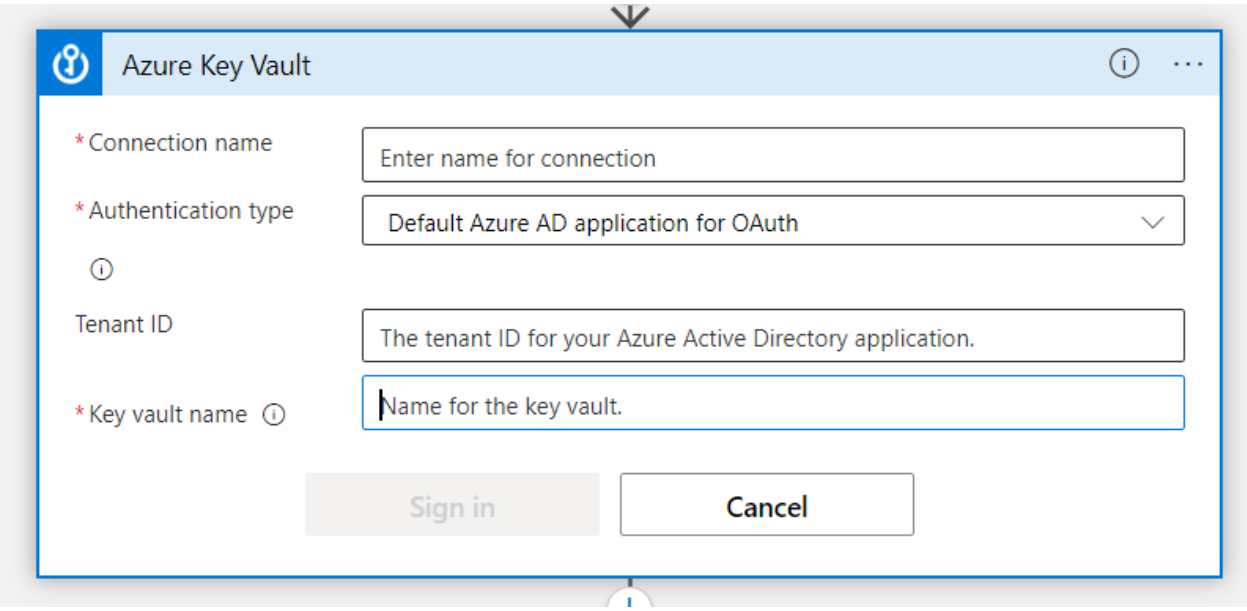
When the creation of the Logic App is finished, open the flow. A few templates are shown, choose **Recurrence**.



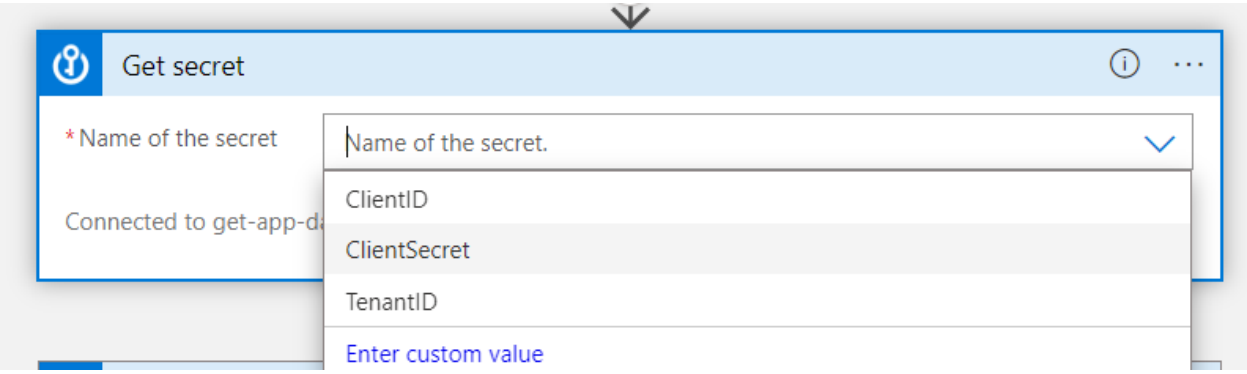


Recurrence in Azure App Designer allows you to schedule the execution of a workflow or action at regular intervals. It involves defining the start time, days, and minutes for the recurrence pattern. This enables you to automate repetitive tasks and ensure they are executed at specific intervals according to your requirements.

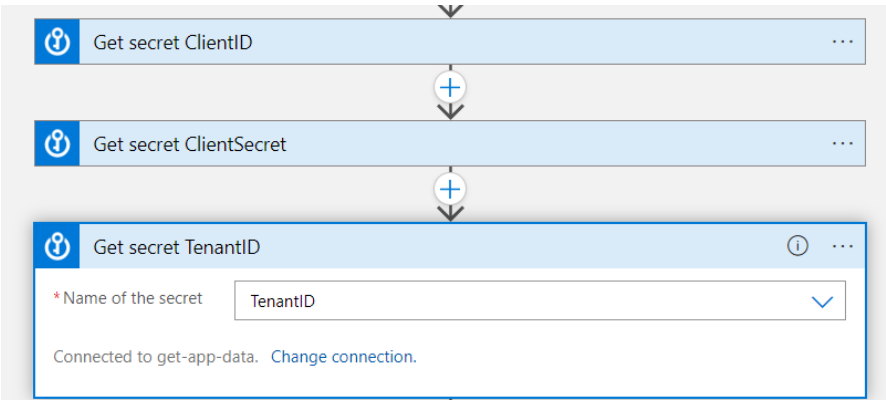
The first action we're going to add is the **Get secret** action, which is an **Azure Key Vault** action. With this action, we retrieve the tenant id, which we use in the HTTP action. Search on Azure Key and select Get secret.



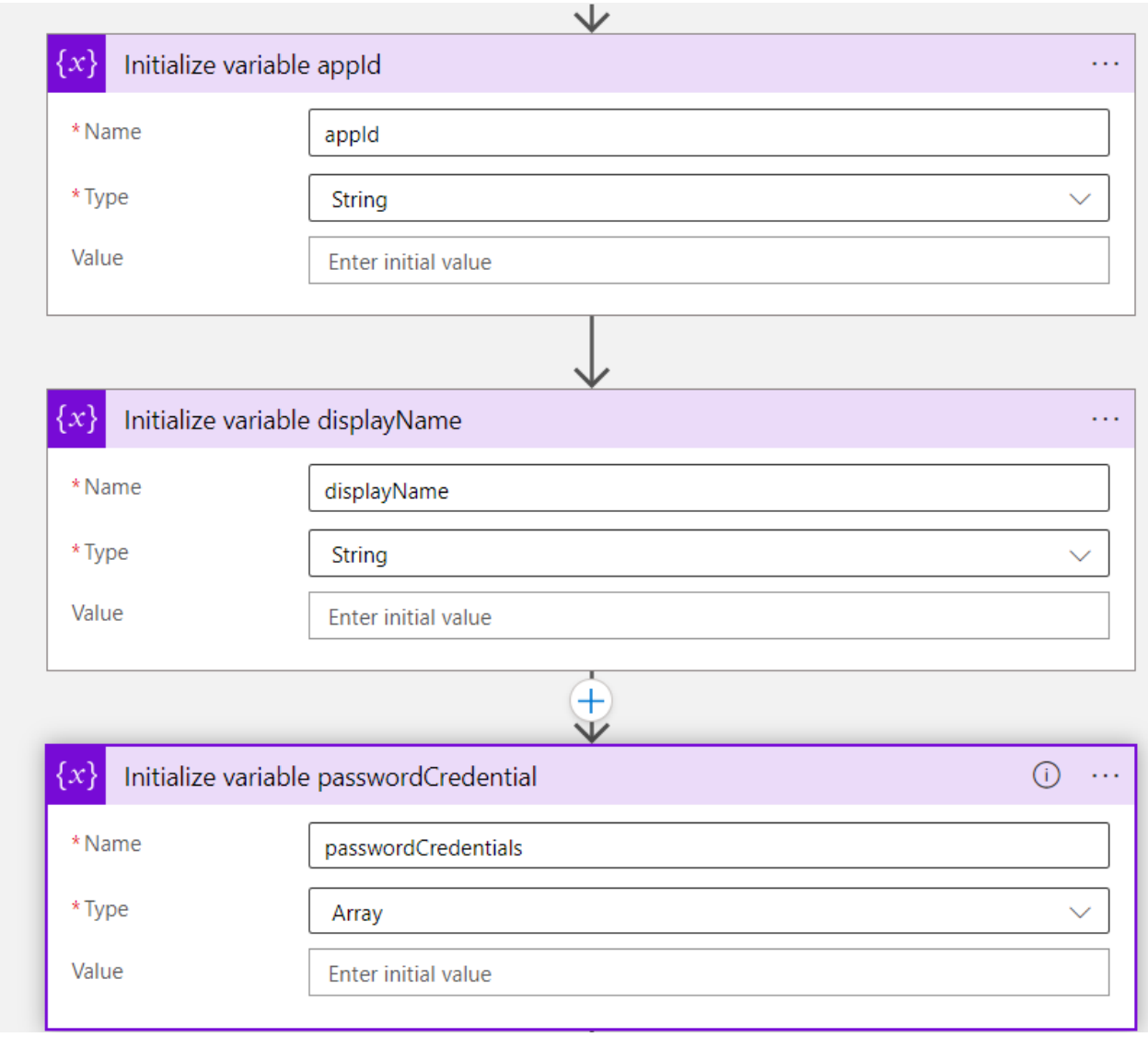
Enter the **name** of the connection, Vault name and select your **tenant**. Next, click **Sign in** to authenticate to the Key Vault.



Choose **client id** from the drop-down list. Repeat the Key Vault steps to also retrieve the **tenant id** and **client secret**.



In the next action, we are going to **initialize variables** for appId, displayName, and passwordCredentials. Search for variables and select **Initialize variable** (three times).



The next step we're going to add is an **HTTP** Action, to query Graph for the App Registrations.

Choose **GET** as **Method**.
As **URI** enter:

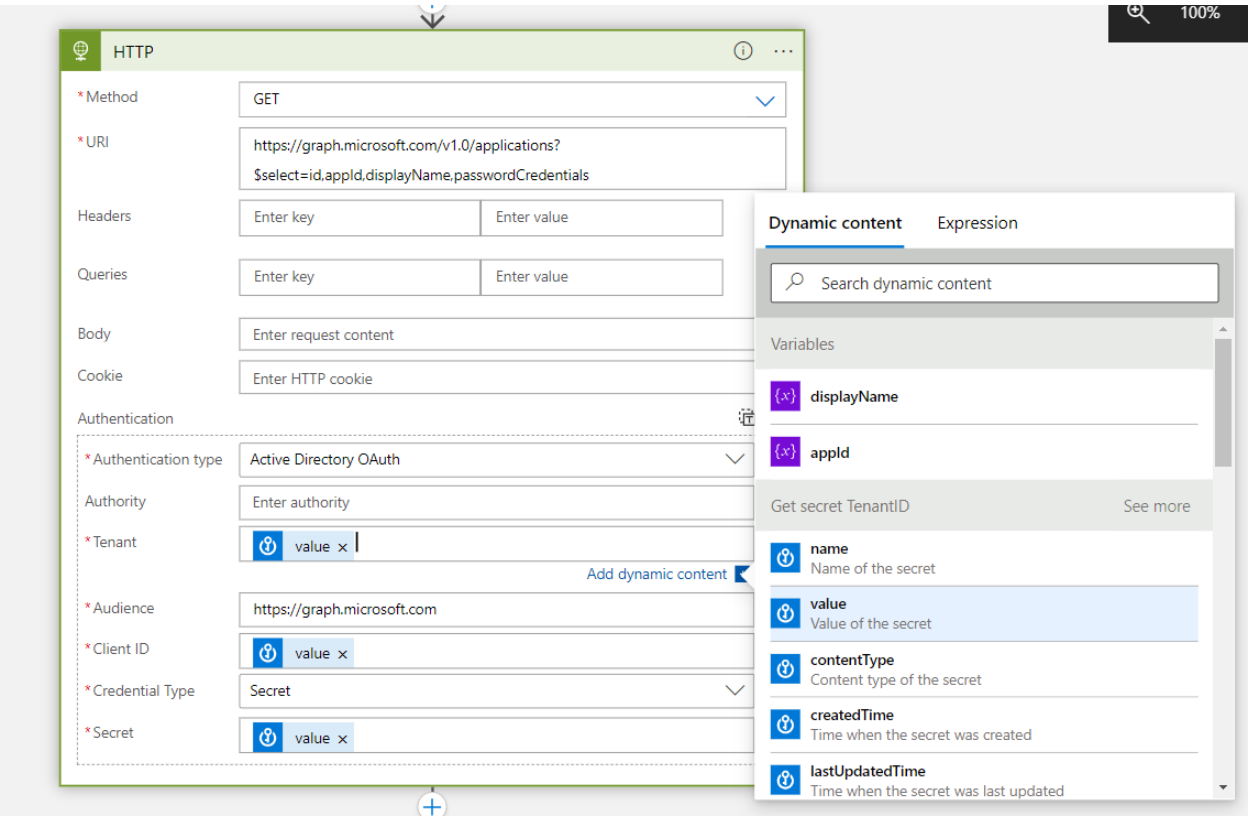
https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials

With this select, we only select the items which are relevant to the information I want to use later in this flow. If you need more information, the URL needs to be expanded with that information.

Choose **Add new parameter** and check **Authentication**. Select **Active Directory OAuth** as authentication type.

As **tenant, client id**, and **secret**, we add the Dynamic content **Value**. Make sure you pick the value from the correct Key Vault action.

Enter https://graph.microsoft.com as **Audience**.



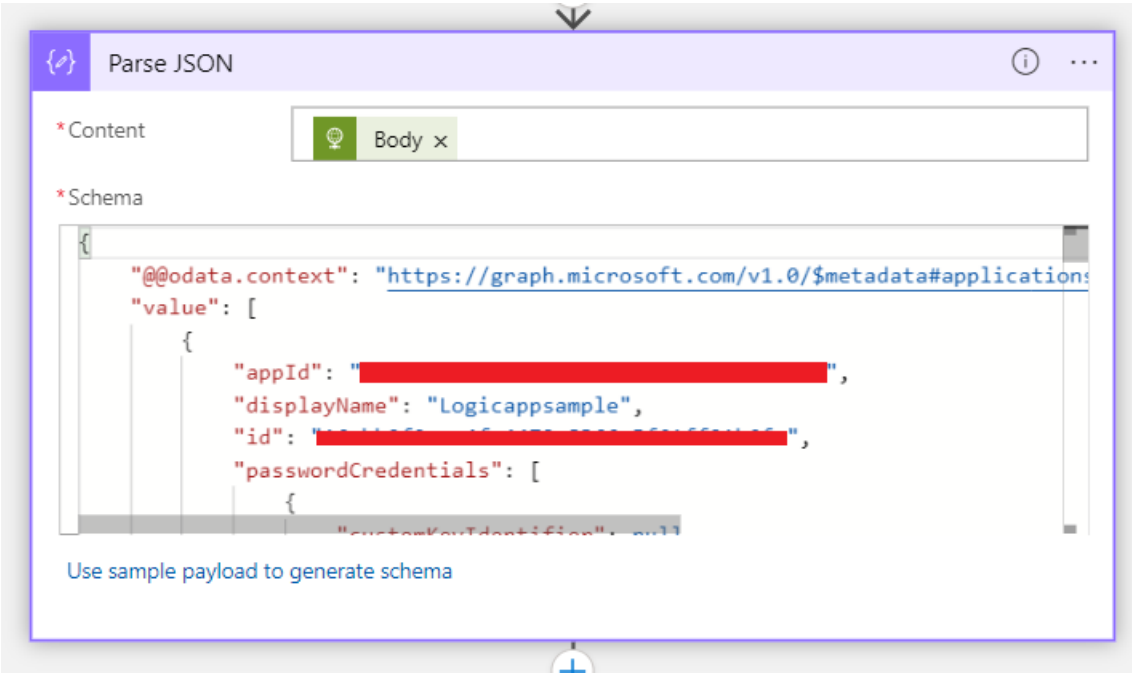
To use the information, we received (by executing this HTTP action) in the upcoming actions, we need to parse the received information with a **Parse JSON** action.

Add a Parse JSON action to the flow. We're going to parse the response **body** of the previous HTTP action. In the **Content** box, we add Body, from the previous HTTP action.

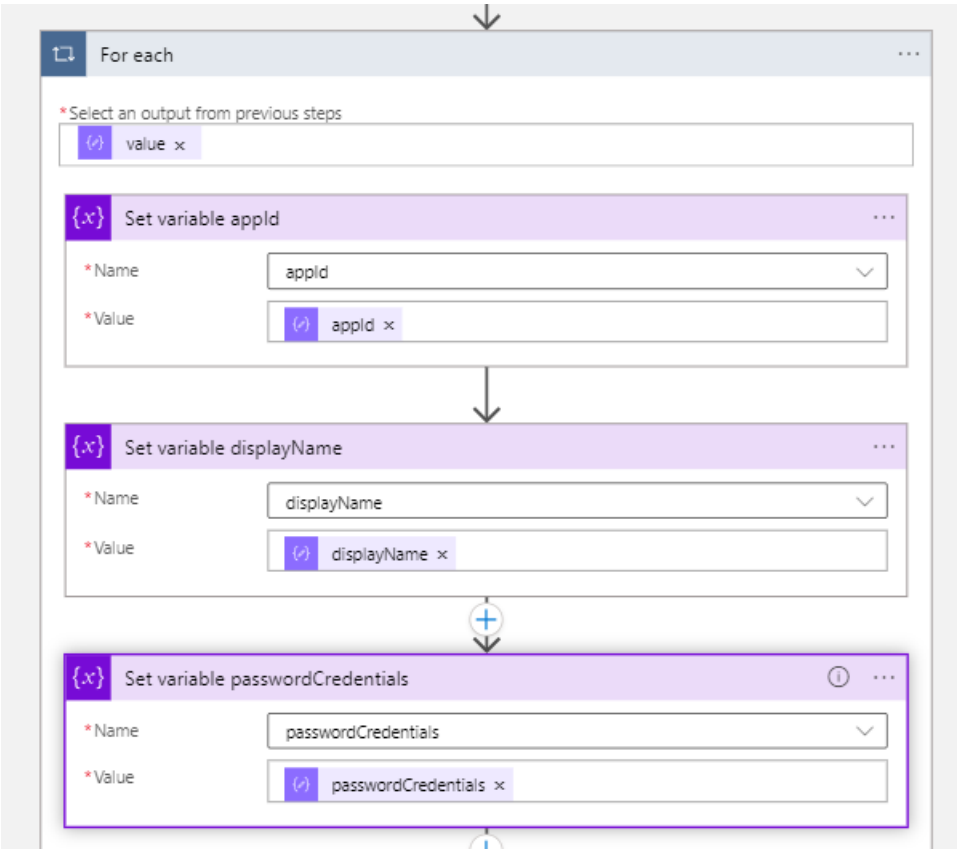
To fill the schema with the correct information, we can add an **example payload**. The information to use as an example payload can be 'generated' by running the flow. **Save** the flow and hit **Run trigger** to start the flow.

Open the Runs history from the flow and open the HTTP action. Copy the body.

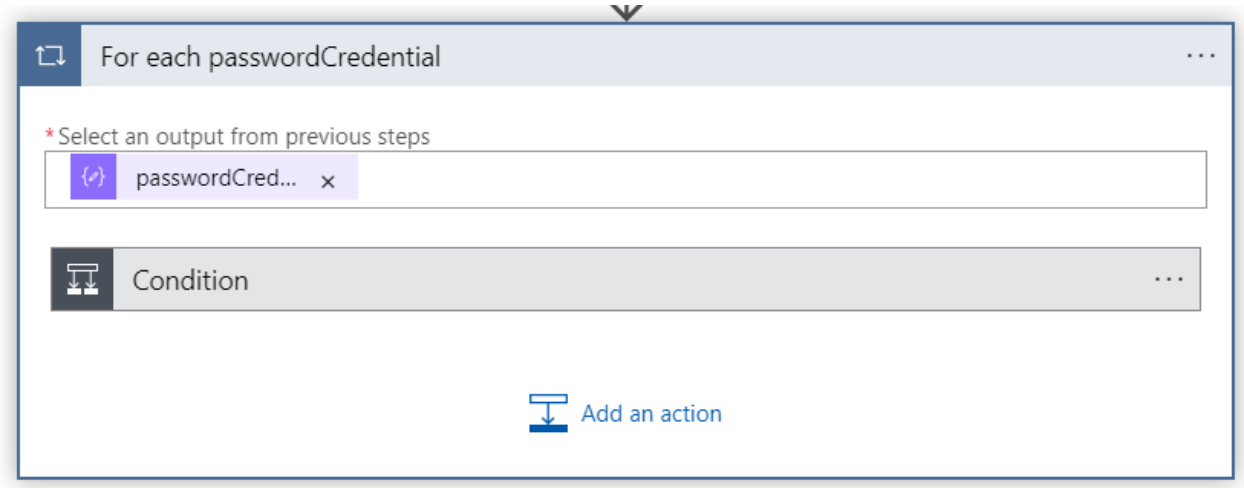
Click **Use sample payload** in the Parse JSON action, past the information which we copied from the runs history in the new pop-up, and click Done.



Add **For each** action. Add Set variable actions and repeat the steps. As value we add dynamic content from the Parse JSON action.



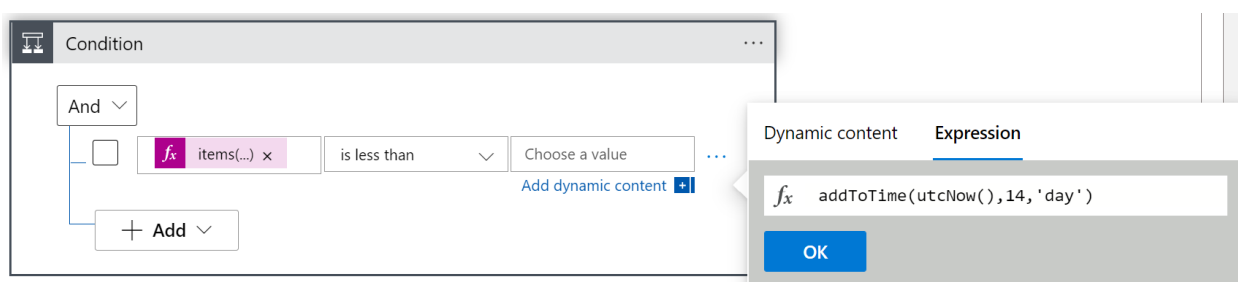
Add another **For each** action, which is a Control action. As the output select passwordCredentials from the Parse JSON action.



Add a **Condition** action, which is also a Control action. With this action, we're going to determine if the end date of the secret (passwordCredential) falls between today and 14 days ago. If that's true it is expiring and a follow up action is performed.

To grab the end date from the passwordCredentials we need to use the below **expression**: items('For_each_passwordCredential')?['endDateTime']

Select **is less than** from the drop-down list. In the right text box enter the below expression to add the current date – 14 days: addToTime(utcNow(),14,'day')



Under **True**, we need to add the action which sends us the notification. I choose to send an **email from a mailbox**.

We can enter text, but also use data from previous actions in the body of the email. I used **displayName from the Parse JSON** action to display the App Registration name.

And I used **expressions** to get information from the passwordCredential variable. These expressions are comparable to the expression used to get the end date of the secret.

True

Send an email (V2)

* Body

Font 12

One of the client secrets is going to expire from Azure App Registration;
displayName ×

Details:

Secret ID: keyId ×
Display Name: displayName ×
Expiration date: endDateTime ×

App Registration location;
https://portal.azure.com/#blade/Microsoft_AAD_RegisteredApps/ApplicationMenuBlade/Overview/appId/ appId ×

Please take action as soon as possible.

* Subject: Azure App Registration secret is going to expire!

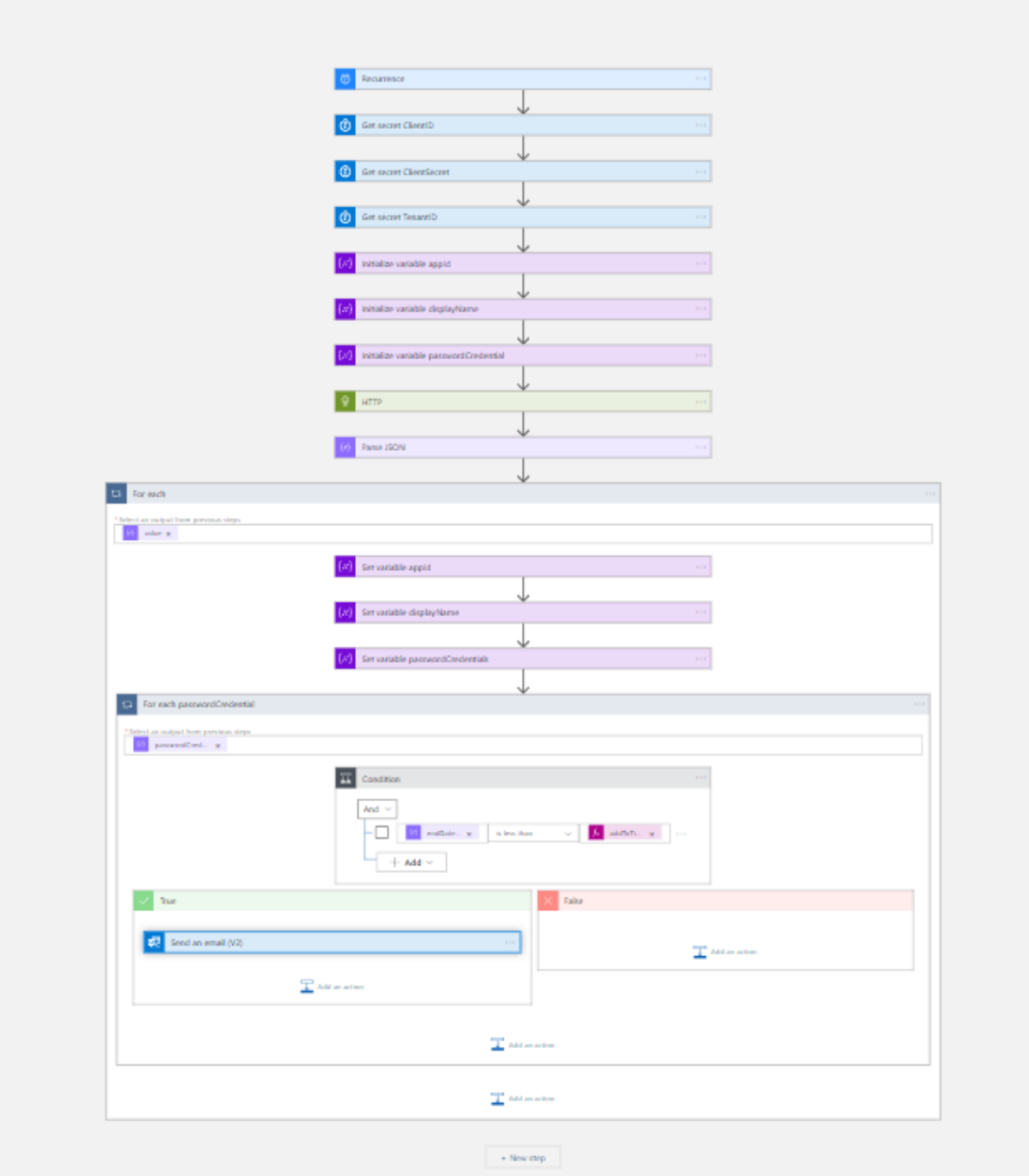
* To: anesugljesa@outlook.com

Importance: High

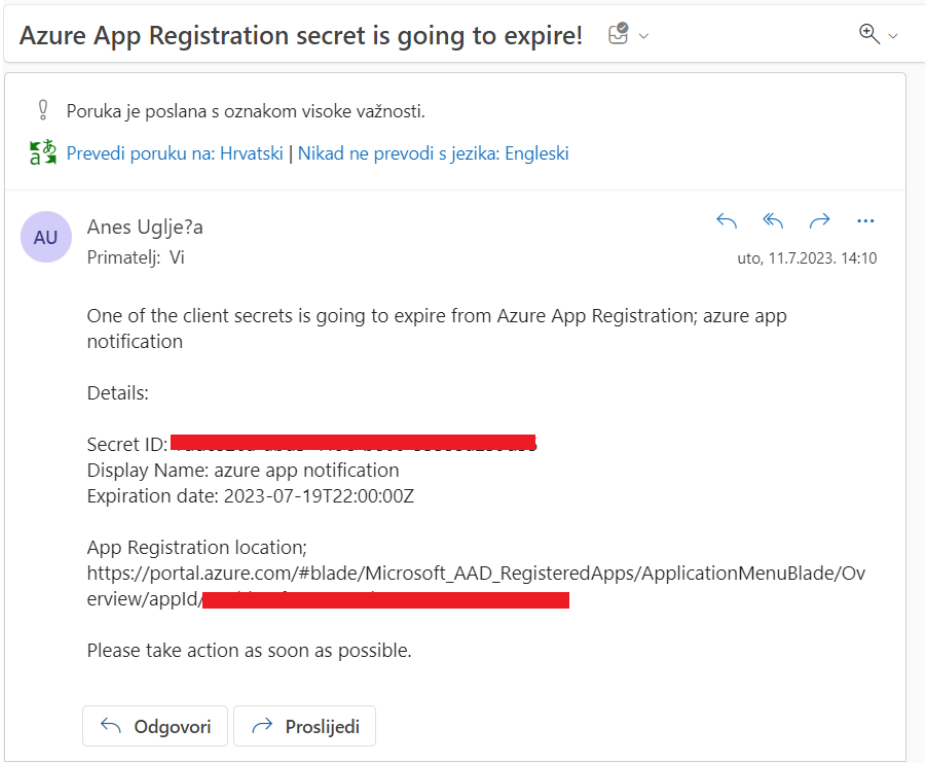
Add new parameter

Connected to Outlook.com. Change connection.

This is our complete flow.



This is how email looks like. I also added the URL to the App Registration for which I use the appId from the Parse JSON action.



Azure App Registration secret is going to expire!

Poruka je poslana s oznakom visoke važnosti.

Prevedi poruku na: Hrvatski | Nikad ne prevodi s jezika: Engleski

AU

Anes Uglje?a

Primatelj: Vi

uto, 11.7.2023. 14:10

One of the client secrets is going to expire from Azure App Registration; azure app notification

Details:

Secret ID: [redacted]
Display Name: azure app notification
Expiration date: 2023-07-19T22:00:00Z

App Registration location;
https://portal.azure.com/#blade/Microsoft_AAD_RegisteredApps/ApplicationMenuBlade/Overview/appId/[redacted]

Please take action as soon as possible.

Odgovori | Proslijedi

Postman configuration for azure.

1. Install Postman: Download and install Postman from the official website https://www.postman.com/downloads/
2. Obtain Azure AD Credentials: Depending on the Azure API you want to interact with, you may need to create an Azure AD application and obtain the necessary credentials (such as Client ID, Client Secret, and Tenant ID).
3. Create a New Postman Environment: Open Postman and click on the "New" button to create a new request.
4. Define Environment Variables: In the Postman interface, click on the "Manage Environments" button (the eye icon in the top-right corner), then click on "Add" to create a new environment.
5. Configure Environment Variables: In the environment configuration, add the following variables based on your Azure AD credentials:
    - ClientID: The Client ID of your Azure AD application.
    - ClientSecret: The Client Secret of your Azure AD application.
    - TenantID: The Tenant ID associated with your Azure AD.
6. Set Environment as Active: Select the newly created environment from the drop-down menu in the top-right corner of the Postman interface.
7. Configure Authorization: In your API requests, add the following authorization configuration:
    - Set the request method, URL, and body (if required).
    - In the "Authorization" tab, select the "OAuth 2.0" type.
    - Click on the "Get New Access Token" button.
    - Fill in the following fields:
        a. Token Name: Choose a name for your token.
        b. Grant Type: Select "Authorization Code".
        c. Click on Authorize using browser.
        d. Callback URL: https://graph.microsoft.com/.default
        e. Auth URL: https://login.microsoftonline.com/{azureTenantId}/oauth2/v2.0/authorize
        f. Token URL: https://login.microsoftonline.com/{azureTenantId}/oauth2/v2.0/token
        g. Client ID: {{ ClientID }} (use the environment variable).
        h. Client Secret: {{ ClientSecret }} (use the environment variable).
        i. Scope: Specify the required scopes for your API.
    - Click on the "Request Token" button to generate the access token.
8. Send the Request: After configuring the authorization, you can click the "Send" button to execute the request with the provided access token.

By following these steps, you can configure Postman to interact with Azure APIs using the appropriate credentials and authentication mechanism.

The minimum required **permission** for this is shown below. We need to grant admin consent.

Microsoft Azure | Search resources, services, and docs (G+/) | anesugljesa@outlook.c... DEFAULT DIRECTORY

Home > Default Directory | App registrations > Logicappsample

Logicappsample | API permissions

Search

- Overview
- Quickstart
- Integration assistant

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API
- App roles
- Owners
- Roles and administrators
- Manifest

Support + Troubleshooting

- Troubleshooting
- New support request

Refresh | Got feedback?

The "Admin consent required" column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization, or in organizations where this app will be used. Learn more

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 Default Directory

API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (8)				
Application.Read.All	Delegated	Read applications	Yes	Granted for Default Dire...
Application.Read.All	Application	Read all applications	Yes	Granted for Default Dire...
Application.ReadWrite.All	Delegated	Read and write all applications	Yes	Granted for Default Dire...
Application.ReadWrite.All	Application	Read and write all applications	Yes	Granted for Default Dire...
Application.ReadWrite.OwnedBy	Application	Manage apps that this app creates or owns	Yes	Granted for Default Dire...
Directory.Read.All	Application	Read directory data	Yes	Granted for Default Dire...
User.Read	Delegated	Sign in and read user profile	No	Granted for Default Dire...
User.Read.All	Delegated	Read all users' full profiles	Yes	Granted for Default Dire...

Home | Workspaces | API Network | Explore | Search Postman | Invite | Upgrade

My Workspace | New | Import

Collections | Environments | History

logic-app-on-azure
- GET https://graph.microsoft.com/v1...
- GET https://graph.microsoft.com/v1...

GET https://graph.microsof | Microsoft Graph environm | Microsoft Graph environment

logic-app-on-azure / https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials | Save

GET | https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials | Send

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

Type: OAuth 2.0

The authorization data will be automatically generated when you send the request. Learn more about authorization

Add authorization data to: Request Hea...

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables

Current Token

This token is only available to you. Sync the token to let collaborators on this request use it.

Token: Available Tokens

[redacted]

Header Prefix: Bearer

Home | Workspaces | API Network | Explore | Search Postman | Invite | Upgrade

My Workspace | New | Import

Collections | Environments | History

logic-app-on-azure
- GET https://graph.microsoft.com/v1...
- GET https://graph.microsoft.com/v1...

GET https://graph.microsof | Microsoft Graph environm | Microsoft Graph environment

logic-app-on-azure / https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials | Save

GET | https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials | Send

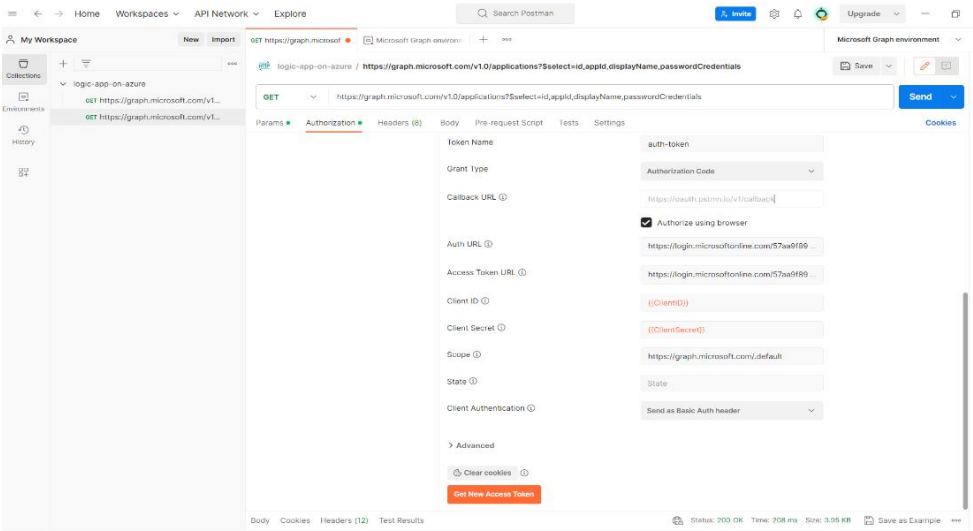
Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

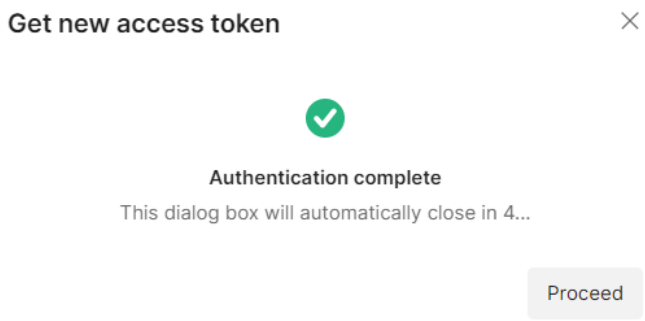
Token Name	auth111
Grant Type	Authorization Code
Callback URL	https://oauth.pstmn.io/v1/callback
	☑ Authorize using browser
Auth URL	https://login.microsoftonline.com/57aa9f89 ...
Access Token URL	https://login.microsoftonline.com/57aa9f89 ...
Client ID	{{ClientID}}
Client Secret	{{ClientSecret}}
Scope	https://graph.microsoft.com/.default
State	State
Client Authentication	Send as Basic Auth header

> Advanced

Clear cookies

Get New Access Token


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
My Workspace
New
Import
Microsoft Graph environment
Collections
Environments
History
logic-app-on-azure
GET https://graph.microsoft.com/v1...
GET https://graph.microsoft.com/v1...
logic-app-on-azure / https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials
Save
GET
https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
Token Name
auth-token
Grant Type
Authorization Code
Callback URL
https://oauth.pstmn.io/v1/callback
Authorize using browser
Auth URL
https://login.microsoftonline.com/57aa9f89 ...
Access Token URL
https://login.microsoftonline.com/57aa9f89 ...
Client ID
{{ClientID}}
Client Secret
{{ClientSecret}}
Scope
https://graph.microsoft.com/.default
State
State
Client Authentication
Send as Basic Auth header
Advanced
Clear cookies
Get New Access Token
Body
Cookies
Headers (12)
Test Results
Status: 200 OK
Time: 208 ms
Size: 3.95 KB
Save as Example


Get new access token
Authentication complete
This dialog box will automatically close in 4...
Proceed

MANAGE ACCESS TOKENS

All Tokens Delete

auth-token

auth-token

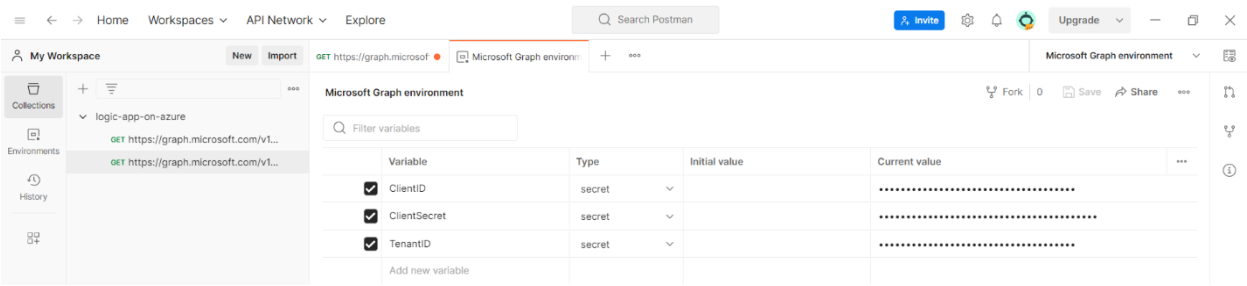
Token Details

Use Token

Token Name auth-token

Access Token

[redacted]
akIBRjRYcjhTUEFlem1EeFZIYXB0c2MiLCJhbGciOiJSUzI1NiIsIng1dCI
6Ii1LSTNROW5OUjdiUm9meG1lWm9YcWJlWkdldyIsImtpZCI6Ii1LST
NROW5OUjdiUm9meG1lWm9YcWJlWkdldyJ9.eyJhdWQiOiJodHRw
czovL2dyYXBoLm1pY3Jvc29mdC5jb20iLCJpc3MiOiJodHRwczovL
3N0cy53aW5kb3dzLm5ldC81N2FhOWY4OS00YzQ2LTRhYTUtYTE
zNS0wOGFiZTM1YTZkOWUvIiwiaWF0IjoxNjg5MDcxNzU2LCJuYmY
iOjE2ODkwNzE3NTYsImV4cCI6MTY4OTA3NzIxMiwiYWNjdCI6MC
wiYWNyIjoiMSIsImFpbyI6IkFWUUFxLzhUQUFBQWdlRjNZL0MxaUR
2K1ZENjIzOW80dHh2VzB1Zm53c1d5ZzVHNDlLUnV0M1AweHZVOF
dBUVNFc29ZMzBKVTA5Yi9TVXVXTytUUk83bnhHc0Q0U1Znc2JIc
3ZNSkRyeW9WejVNMkZ3alVqMWNNPSIsImFsdHNlY2lkIjoiMTpsaX
ZlLmNvbTowMDAzN0ZGRTM1NTJGMkMyIiwiYW1yIjpbInB3ZCJdLC
JhcHBfZGlzcGxheW5hbWUiOiJMb2dpY2FwcHNhbXBsZSIsImFwcG



Home Workspaces API Network Explore Search Postman Invite Upgrade

My Workspace New Import

Collections Environments History

logic-app-on-azure
GET https://graph.microsoft.com/v1...
GET https://graph.microsoft.com/v1...

GET https://graph.microsof Microsoft Graph environm

Microsoft Graph environment

logic-app-on-azure / https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials

Save

GET https://graph.microsoft.com/v1.0/applications?$select=id,appId,displayName,passwordCredentials Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Client ID {{ClientID}}

Client Secret {{ClientSecret}}

Scope https://graph.microsoft.com/.default

State State

Client Authentication Send as Basic Auth header

> Advanced

Clear cookies

Get New Access Token

Body Cookies Headers (12) Test Results Status: 200 OK Time: 323 ms Size: 3.95 KB Save as Example

Pretty Raw Preview Visualize JSON


```
{
    "@odata.context": "https://graph.microsoft.com/v1.0/$metadata#applications(id,appId,displayName,passwordCredentials)",
    "value": [
        {
            "id": [redacted],
            "appId": [redacted],
            "displayName": "azure app notification",
            "passwordCredentials": [
                {
                    "customKeyIdentifier": null,
                    "displayName": "client secret",
```


Microsoft Azure Search resources, services, and docs (G+/) anesugljesa@outlook.c... DEFAULT DIRECTORY

Home > Default Directory | App registrations >

Logicappsample

Search

Overview
Quickstart
Integration assistant

Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles
Owners
Roles and administrators
Manifest

Support + Troubleshooting
Troubleshooting
New support request

Delete Endpoints Preview features

Got a second? We would love your feedback on Microsoft identity p

Essentials

Display name : Logicappsample

Application (client) ID : [redacted]

Object ID [redacted]

Directory (tenant) ID : [redacted]

Supported account types : My organization only

Starting June 30th, 2020 we will no longer add any new feature
but we will no longer provide feature updates. Applications will

Get Started Documentation

Build your

The Microsoft identity platform is
standards-based authentic

Endpoints

Copy to clipboard

OAuth 2.0 authorization endpoint (v2)
https://login.microsoftonline.com/[redacted]/oauth2/v2.0/authorize

OAuth 2.0 token endpoint (v2)
https://login.microsoftonline.com/[redacted]/oauth2/v2.0/token

OAuth 2.0 authorization endpoint (v1)
https://login.microsoftonline.com/[redacted]oauth2/authorize

OAuth 2.0 token endpoint (v1)
https://login.microsoftonline.com/[redacted]/oauth2/token

OpenID Connect metadata document
https://login.microsoftonline.com/[redacted]/v2.0/.well-known/openid-configuration

Microsoft Graph API endpoint
https://graph.microsoft.com

Federation metadata document
https://login.microsoftonline.com/[redacted]/federationmetadata/2007-06/federationmetadata.xml

WS-Federation sign-on endpoint
https://login.microsoftonline.com/[redacted]wsfed

SAML-P sign-on endpoint
https://login.microsoftonline.com/[redacted]/saml2

SAML-P sign-out endpoint
https://login.microsoftonline.com/[redacted]saml2