



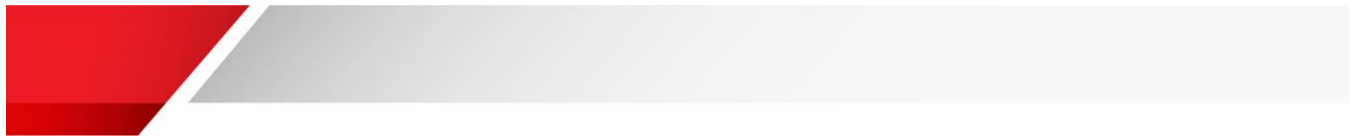
Oracle Digital Assistant TechExchange

Article.

Using MS Teams SSO authentication to access Microsoft Graph APIs from chatbot conversations in Oracle Digital Assistant

Rohit Dhamija, April 2021

This document will guide you through the steps to set up single sign-on (SSO) authentication for an Oracle digital assistant that is exposed through a Microsoft Teams channel.



OBJECTIVE	3
PRE-REQUISITES	4
SOLUTION OVERVIEW.....	5
DEMO	5
CREATE YOUR AAD APPLICATION IN AZURE PORTAL.....	7
1. REGISTER AN APPLICATION	7
2. AUTHENTICATION.....	9
3. CLIENT SECRETS.....	10
4. TOKEN CONFIGURATION	11
5. API PERMISSIONS.....	13
6. EXPOSE AN API.....	15
7. ADD A CLIENT APPLICATION	17
8. MANIFEST	19
9. GRANT TENANT ADMIN PERMISSIONS TO THE AAD APPLICATION	20
UPDATE YOUR MS TEAMS APP WITH SSO DETAIL.....	22
SETUP AUTHENTICATION SERVICE IN ORACLE DIGITAL ASSISTANT	25
ROUTING YOUR SKILL TO MS TEAMS CHANNEL	26
UPDATE SKILL	26
GET SSO ACCESS TOKEN	26
GET ACCESS TOKEN.....	27
CALL GRAPH API.....	30
TEST.....	31
SYSTEM TESTER.....	31
MS TEAMS APP	33
CONCLUSION	34



Objective

The [MS Teams SSO feature](#) is primarily intended to support customers who want to access Oracle Fusion Applications from MS Teams without prompting users for authentication.

The token you receive from the MS Teams SSO endpoint is an “exchange” token and cannot be used directly to invoke graph APIs, therefore cannot be used to access Graph API’s or Oracle Digital Assistant’s [Calendar system components](#).

In order to get a token that works with graph API, the “on-behalf-of” flow must be followed using the SSO token to exchange for an access token with appropriate scopes that will work with graph APIs. More information here: <https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-on-behalf-of-flow>

This document will guide you through the steps to set up single sign-on (SSO) authentication for an Oracle digital assistant that is exposed through a Microsoft Teams channel. Once set up, users just need to log in to Teams with their Azure AD credentials and then seamlessly interact with the digital assistant and access protected resources like profile information, calendar events etc. without having to sign in again.

Pre-requisites

As a pre-requisite, make sure that you have access to the following resources:

- An Microsoft Azure account with an active subscription with admin access
- Oracle Digital Assistant instance version 21.02 or higher
- MS Teams with access to [App Studio](#)
- Your existing skill / new skill should point to your MS Teams Application built using App Studio. In case, it is not setup, please setup one using this [documentation](#). Following this article, you will be updating your MS Teams application for
 - Adding your AAD App Id to configure your app for Single Sign On
 - Adding the resource URL of the app acquiring the auth token for SSO
 - Finally installing your updated application

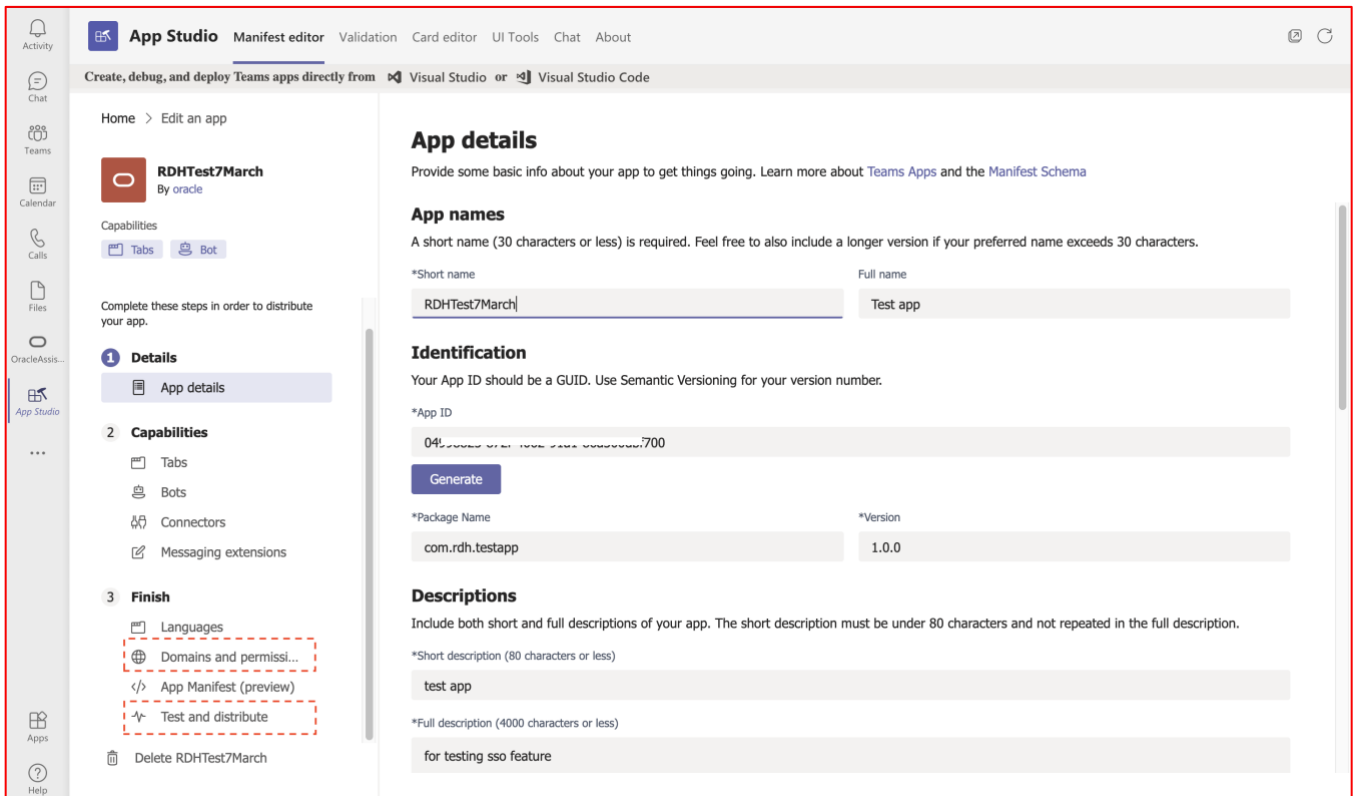


Figure 1 MS Teams app built using App Studio

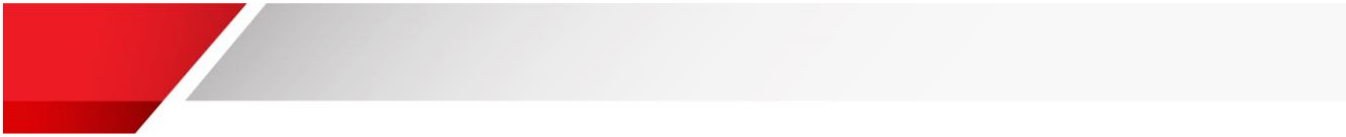


Figure 3 User is able to fetch profile information without need to login again

The skill asks for permission to access the Microsoft account when the user starts interacting. The skill then performs following series of steps:

- Fetches the SSO token
- It uses the SSO token to access the middle tier token
- Uses access token to obtain the user's profile information.



Create your AAD application in Azure Portal

Below are the steps to create your Azure AD application in Azure portal.

1. Register a new application
2. Add authentication settings
3. Create client secrets
4. Add access tokens under Token Configuration
5. Add API permissions
6. Expose the API
7. Add scopes defined by the API and authorize your application
8. Update the manifest file
9. Finally, grant tenant permissions to the application.

1. Register an application

In the Azure portal (https://portal.azure.com/#blade/Microsoft_AAD_RegisteredApps/ApplicationsListBlade), register the application and open the app registration page.

Click **New Registration**.

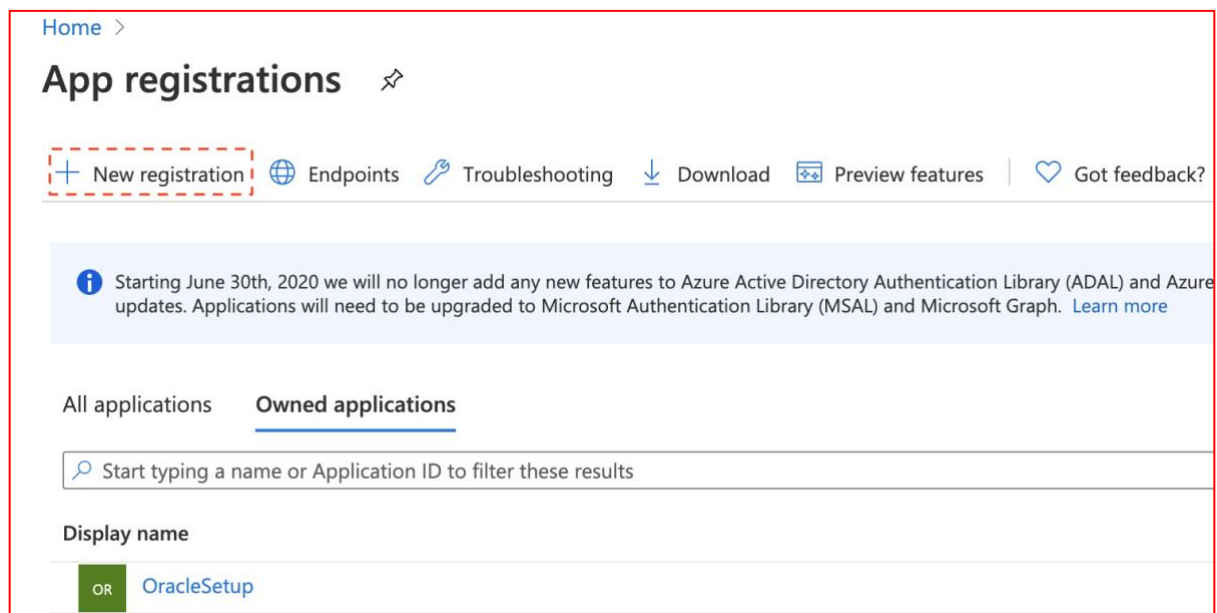


Figure 4 New registration

- Fill in the **Name** field.

- In the **Supported account types** section, select the **Accounts in any organizational directory (Any Azure AD directory - Multitenant)** and **personal Microsoft accounts (e.g. Skype, Xbox)** radio button.

Home > App registrations >

Register an application

* Name

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

Demo SSO ✓

Supported account types

Who can use this application or access this API?

Accounts in this organizational directory only (oracledev 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.

[By proceeding, you agree to the Microsoft Platform Policies](#) ↗

Register

Figure 5 Register an application

- Click **Register**

Once the application is created, you will be navigated to the Overview section. Notice that Application (client) ID and Directory (tenant) ID are created for your app.

NOTE DOWN THE APPLICATION (CLIENT) ID, DIRECTORY (TENANT) ID. YOU WILL BE REQUIRED TO ENTER THIS LATER DURING ODA CONFIGURATION SCREEN. CLICK ON COPY TO CLIPBOARD ICON NEXT TO THE IDS.

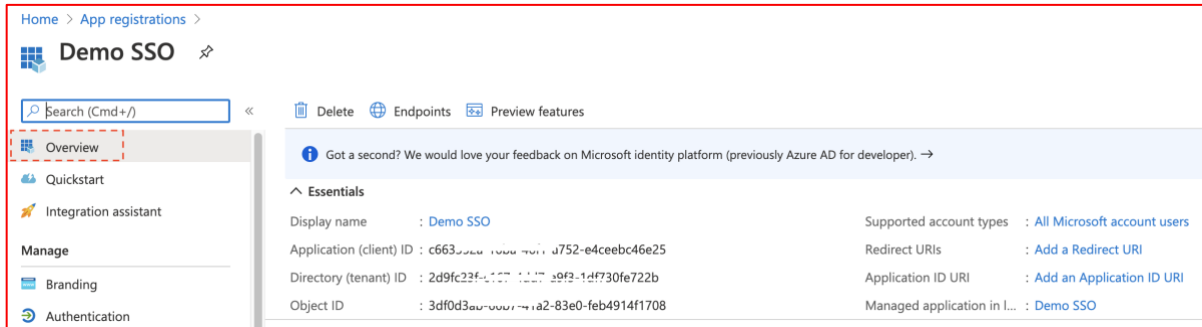


Figure 6 Overview

2. Authentication

Next, you'll add a web platform configuration

- Click on **Authentication** on the left side menu.
- Click **Add a platform**.

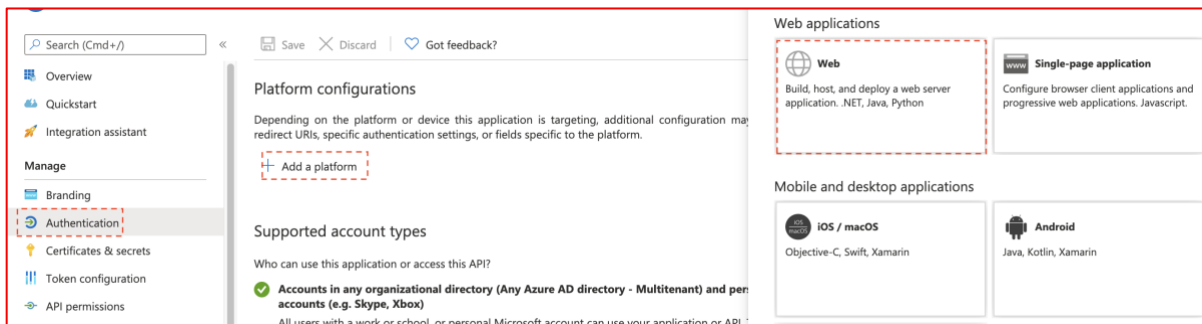


Figure 7 Authentication

- Under **Web application** on the right click on **Web**.
- Under **Redirect URIs** add a redirect URI and click **Configure**.

The format of the redirect URI should be: **<your-oda-url>/connectors/v2/callback**

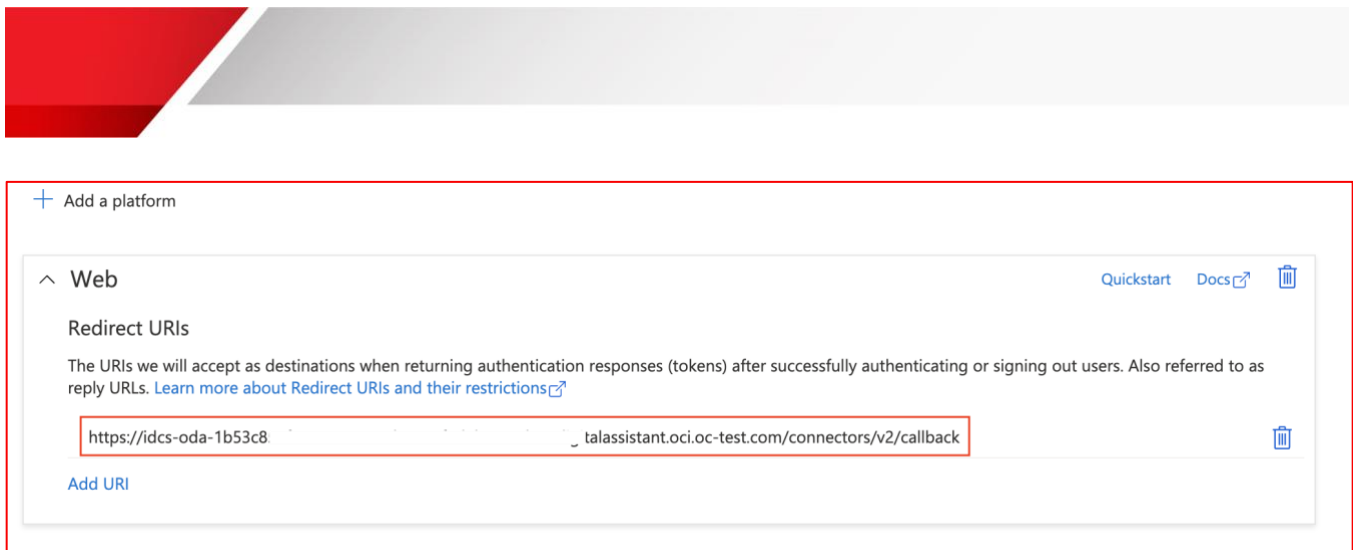


Figure 8 Configure web platform

3. Client Secrets

To Create a client secret

- Select **Certificates and Secret** and select **New Client secret**.
- Give a description, select Never option if you would like this secret to never expire.
- Click **Add**.

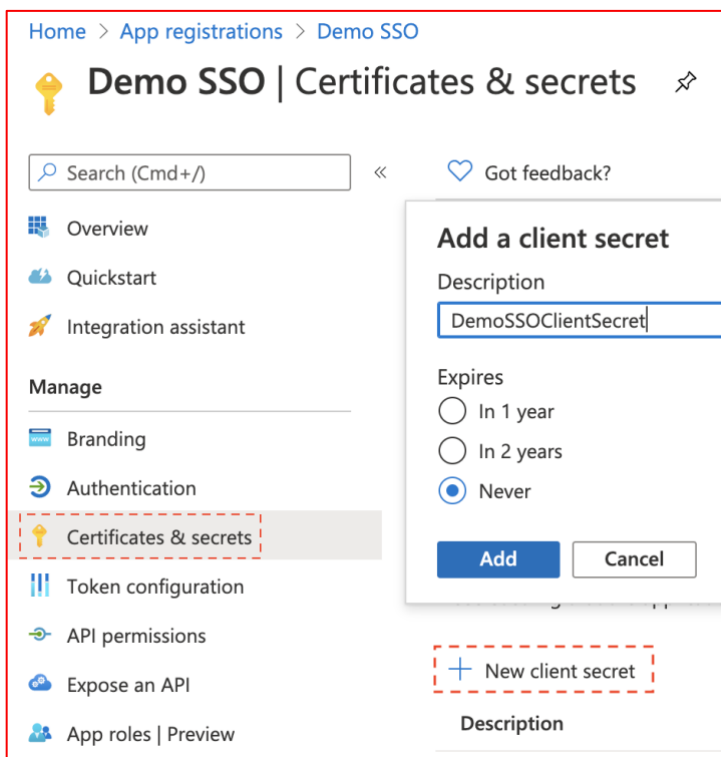


Figure 9 New client secret

Make a note of the client secret value. You will need this later. Click on Copy to clipboard icon next to the client secret value.



Description	Expires	Value	ID
DemoSSOClientSecret	12/31/2299	uA0*****	a06df336-0eed-40e1-b7f0-000a97867fcb  

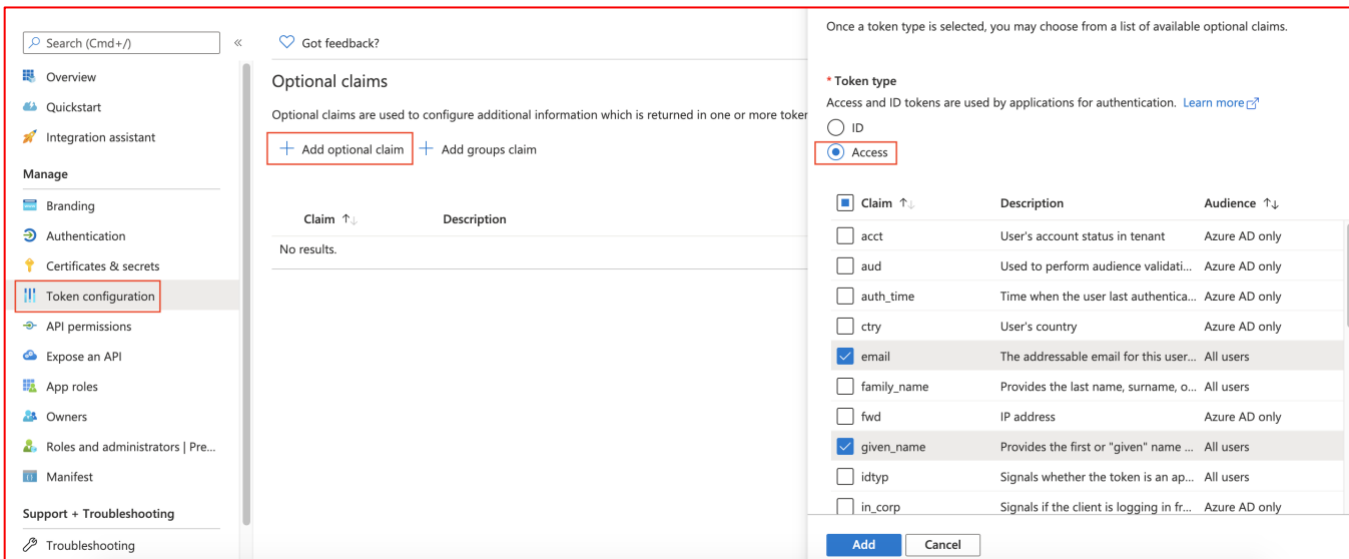
Figure 10 Client secret value

4. Token configuration

Under Token configuration, click **Add optional claim**. An **Add optional claim** popup will appear. Select the Token type as **Access** and select following claims

- **email**
- **given_name**
- **upn**

Finally click **Add**.



Once a token type is selected, you may choose from a list of available optional claims.

* **Token type**
 Access and ID tokens are used by applications for authentication. [Learn more](#)

ID
 Access

<input type="checkbox"/> Claim ↑↓	Description	Audience ↑↓
<input type="checkbox"/> acct	User's account status in tenant	Azure AD only
<input type="checkbox"/> aud	Used to perform audience validati...	Azure AD only
<input type="checkbox"/> auth_time	Time when the user last authentica...	Azure AD only
<input type="checkbox"/> ctry	User's country	Azure AD only
<input checked="" type="checkbox"/> email	The addressable email for this user...	All users
<input type="checkbox"/> family_name	Provides the last name, surname, o...	All users
<input type="checkbox"/> fwd	IP address	Azure AD only
<input checked="" type="checkbox"/> given_name	Provides the first or "given" name ...	All users
<input type="checkbox"/> idtyp	Signals whether the token is an ap...	All users
<input type="checkbox"/> in_corp	Signals if the client is logging in fr...	Azure AD only

Add **Cancel**

Figure 11 Token configuration

Under Add optional claim, Select Turn on the Microsoft Graph email, profile permission (required for claims to appear in token) option and click **Add**

Add optional claim ×

Some of these claims (email, upn) require OpenId Connect scopes to be configured through the API permissions page or by checking the box below. [Learn more](#)

Turn on the Microsoft Graph email, profile permission (required for claims to appear in token).

Add **Cancel**

<input type="checkbox"/> Claim ↑↓	Description	Audience ↑↓
<input type="checkbox"/> acct	User's account status in tenant	Azure AD only
<input type="checkbox"/> auth_time	Time when the user last authentica...	Azure AD only
<input type="checkbox"/> ctry	User's country	Azure AD only
<input checked="" type="checkbox"/> email	The addressable email for this user...	All users
<input type="checkbox"/> family_name	Provides the last name, surname, o...	All users
<input type="checkbox"/> fwd	IP address	Azure AD only
<input type="checkbox"/> given_name	Provides the first or "given" name ...	All users
<input type="checkbox"/> idtyp	Signals whether the token is an ap...	All users
<input type="checkbox"/> in_scp	Signals if the client is logging in fr...	Azure AD only

Add **Cancel**

Figure 12 Add optional claim

5. API permissions

- On the API permission page, you will observe that the required permissions are created automatically.

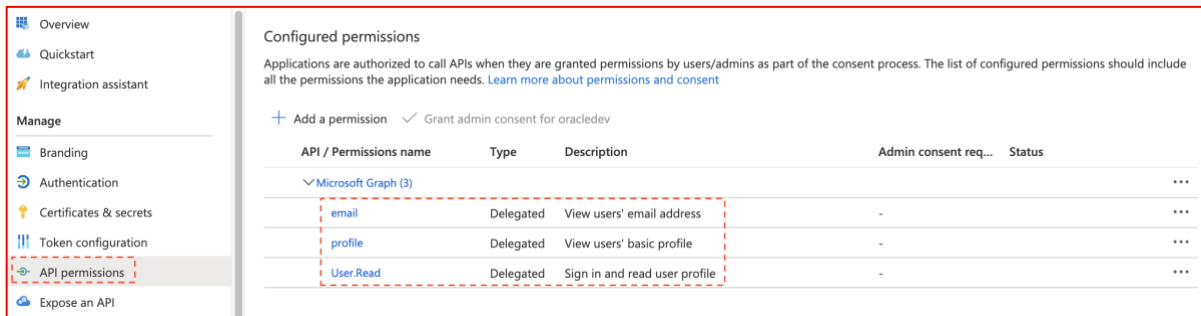


Figure 13 API permissions

- Click on the **Add a Permission** and add **User.ReadBasic.All** permission, since you will need this to access profile information.
- You need to click on Microsoft Graph API, select “Delegated Permissions” and then type User.Read in the Select permissions box.
- Click on Add permission.

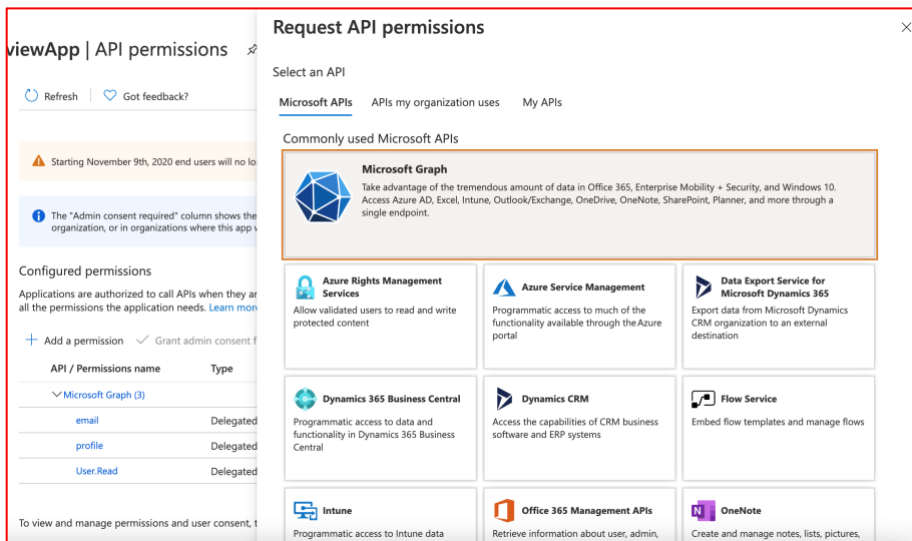


Figure 14 Request API permissions

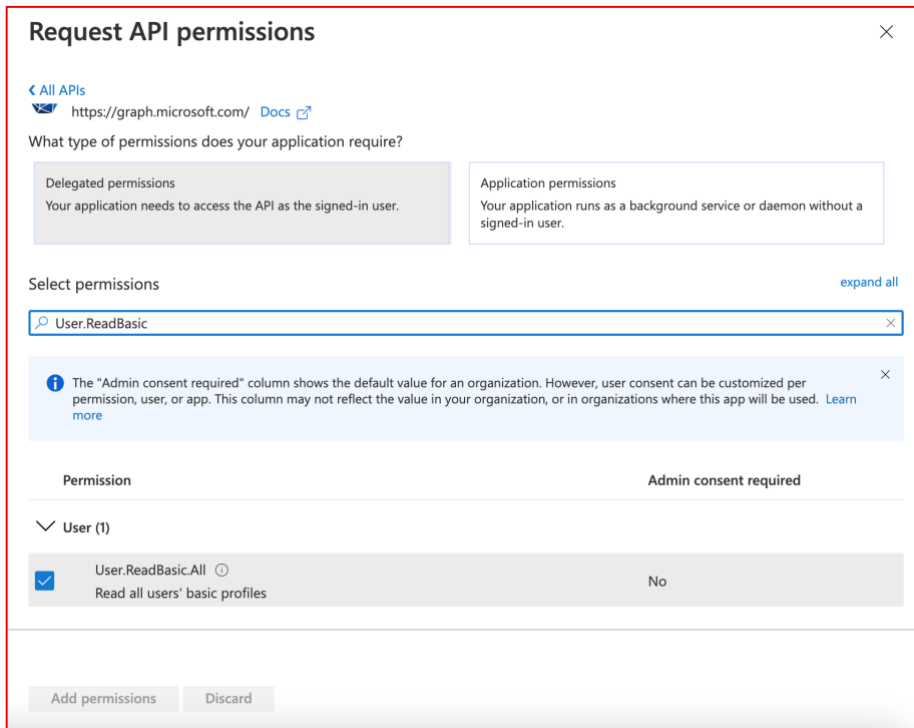


Figure 15 Select permissions

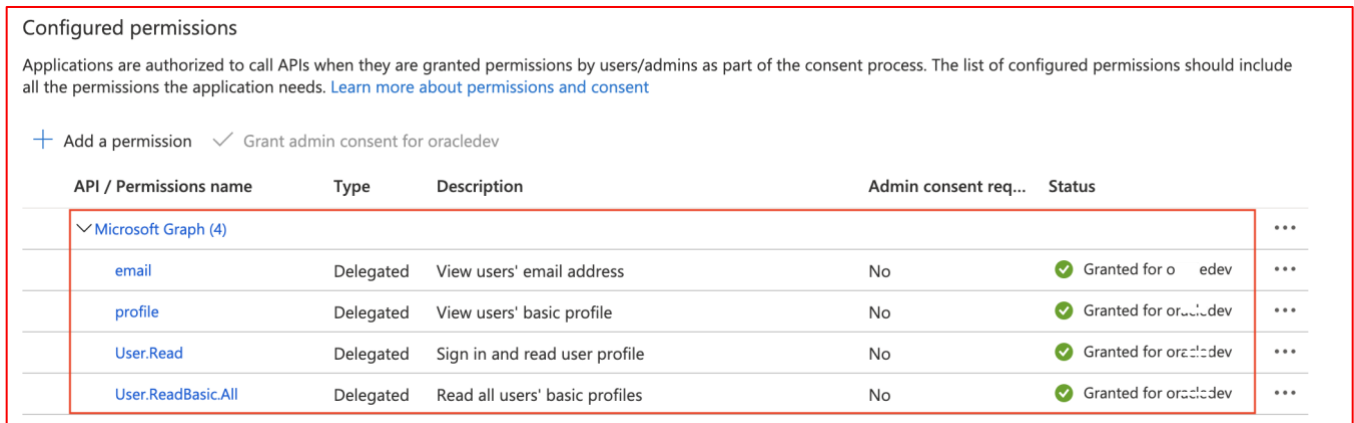


Figure 16 API permission granted to tenancy

NOTE THAT SINCE THE ABOVE PERMISSIONS SHOULD BE GRANTED AT YOUR AZURE TENANCY LEVEL, SO YOU WILL NEED ADMIN RIGHTS FOR THE SAME. PLEASE CONTACT YOUR ADMINISTRATOR TO GRANT YOU THE ACCESS PERMISSIONS.

6. Expose an API

Set Application ID URI

Under Expose an API, set Application ID URI

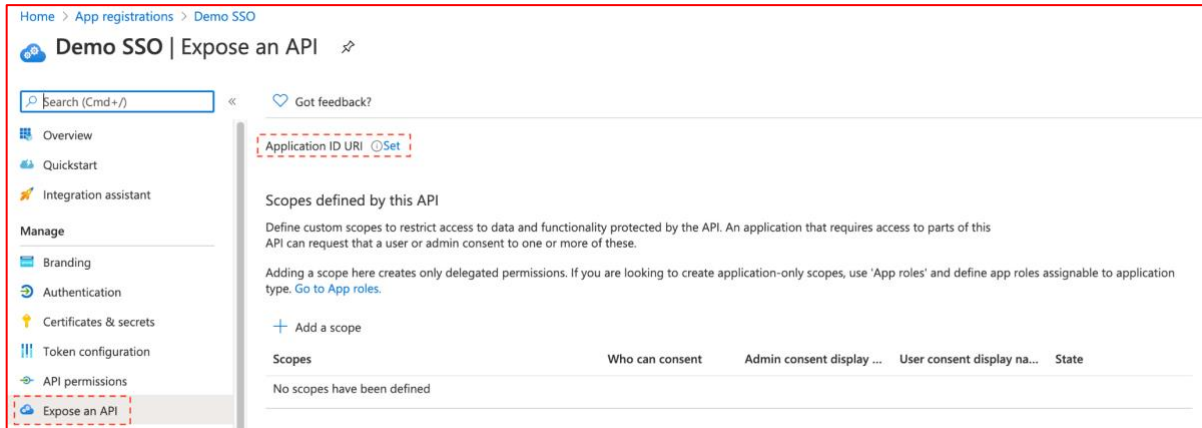


Figure 17 Expose an API

Update the Value in the format:

api://botid-{YourBotId}

YOURBOTID CORRESPONDS TO YOUR TEST APPLICATION'S MICROSOFT APPLICATION ID.

The is the bot id of App Studio Application that you would create later in this article. The following screenshot shows where to look for the bot id. Note this is just for a reference.

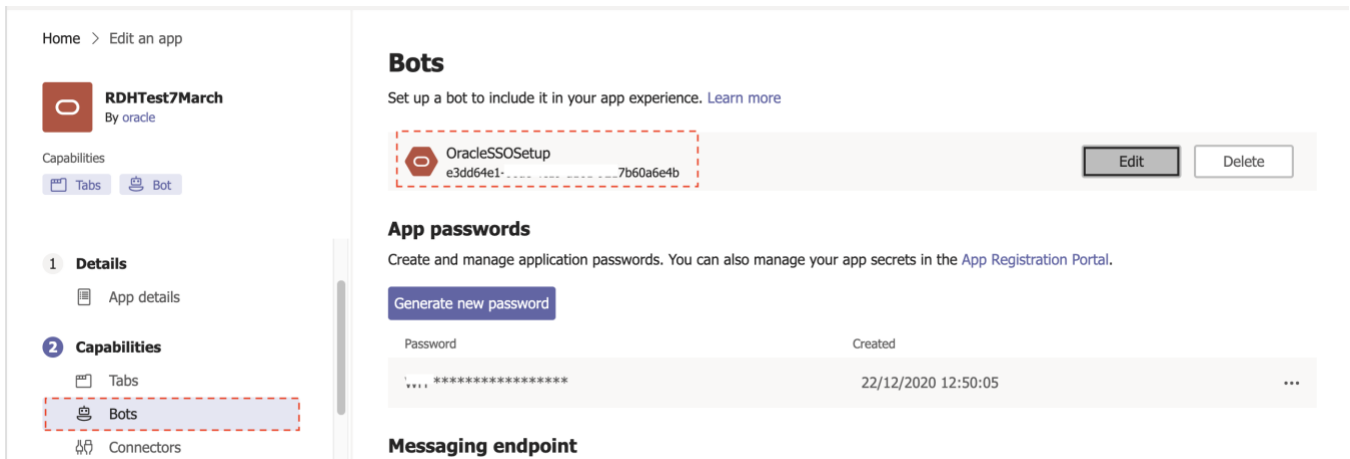
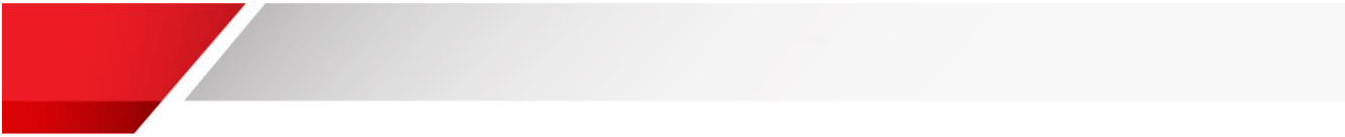


Figure 18 MS Teams – Get the Bot ID



Set the App ID URI

Application ID URI

Figure 19 Set the App ID URI

Add a scope

Click the **Add a scope** button. In the panel that opens, enter:

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires access to parts of this API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create application-only scopes, use 'App roles' and define app roles assignable to application type. [Go to App roles.](#)

- `access_as_user` as the Scope name.
- Set Who can consent? to **Admins and users**.
- Fill in the fields for configuring the admin and user consent prompts with values that are appropriate for the `access_as_user` scope.
- Suggestions:
 - Admin consent title: Admin consent display name
 - Admin consent description: Allows Teams to call the app's web APIs as the current user.
 - User consent title: Teams can access your user profile and make requests on your behalf
 - User consent description: Enable Teams to call this app's APIs with the same rights that you have

Ensure that State is set to **Enabled**

Select **Add scope**

THE DOMAIN PART OF THE SCOPE NAME DISPLAYED JUST BELOW THE TEXT FIELD SHOULD AUTOMATICALLY MATCH THE APPLICATION ID URI SET IN THE PREVIOUS STEP, WITH /ACCESS_AS_USER APPENDED TO THE END



Save Discard Delete

Scope name * ⓘ

 api://botid-e3....._217b60a6e4b/access_as_user

Who can consent? ⓘ
 Admins and users Admins only

Admin consent display name * ⓘ

Admin consent description * ⓘ

User consent display name ⓘ

User consent description ⓘ

State ⓘ
 Enabled Disabled

Figure 20 Scope details

7. Add a client application

In the Authorized client applications section, you identify the applications that you want to authorize to your app's web application.

Each of the following IDs needs to be entered

NOTE THESE ARE EXACT IDS THAT YOU NEED TO USE.

- 1fec8e78-bce4-4aaf-ab1b-5451cc387264 (Teams mobile/desktop application)
- 5e3ce6c0-2b1f-4285-8d4b-75ee78787346 (Teams web application)

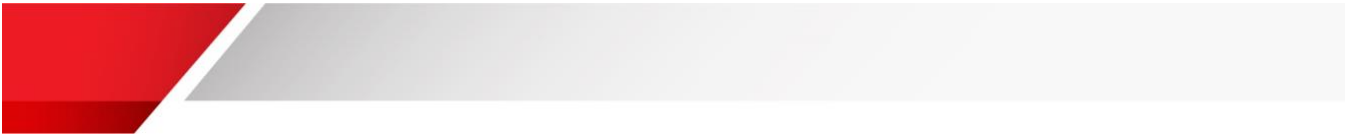
Token configuration
 API permissions
 Expose an API
 App roles | Preview
 Owners

Authorized client applications
 Authorizing a client application indicates that this API trusts the application and users should not be asked to consent when the client calls this API.

+ Add a client application

Client Id Scopes

Figure 21 Add a client application



Add a client application ✕

Client ID ⓘ

1fec8e78-bce4-4aaf-ab1b-5451cc387264 ✓

Authorized scopes ⓘ

api://botid-e3d64... 9217b60a6e4b/access_as_user

Figure 22 Adding client application

Select the Authorized scopes.

Here is how the overall “Expose an API” section should look like

Search (Cmd+) << Got feedback?

Overview
Quickstart
Integration assistant

Manage

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

Support + Troubleshooting

Troubleshooting
New support request

Application ID URI: api://botid-e3dd64...61-9217b60a6e4b

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires access to parts of this API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create application-only scopes, use 'App roles' and define app roles assignable to application type. [Go to App roles.](#)

+ Add a scope

Scopes	Who can consent	Admin consent display ...	User consent display na...	State
api://botid-e3dd64...61-9217b60a6e4b	Admins and users	Teams can access the use...	Team can access your use...	Enabled

Authorized client applications

Authorizing a client application indicates that this API trusts the application and users should not be asked to consent when the client calls this API.

+ Add a client application

Client Id	Scopes
1fec8e78-bce4-4aaf-ab1b-5451cc387264	1
5e3ce6c0-2b1f-4285-8d4b-75ee78787346	1

Figure 23 Preview of Expose an API screen

8. Manifest

Under manifest, set **"acceptMappedClaims"** to **true** and **Save**

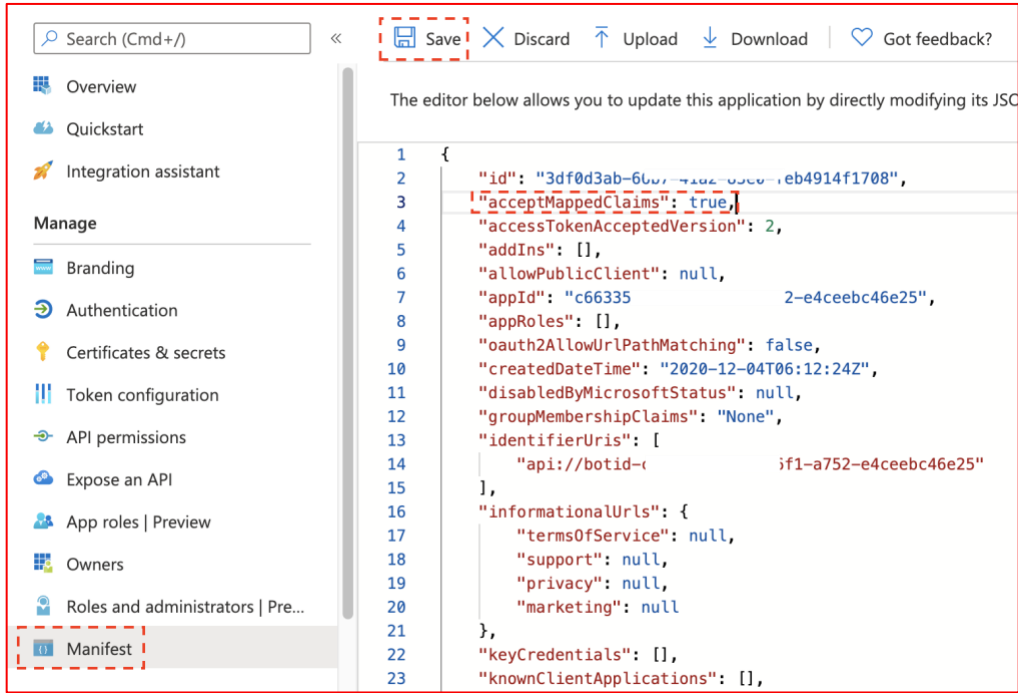


Figure 24 Update Manifest file

The application setup is almost complete now, please go to overview tab and keep **the Application (client) ID**, **Application ID URI** and **Directory (tenant) ID** handy.

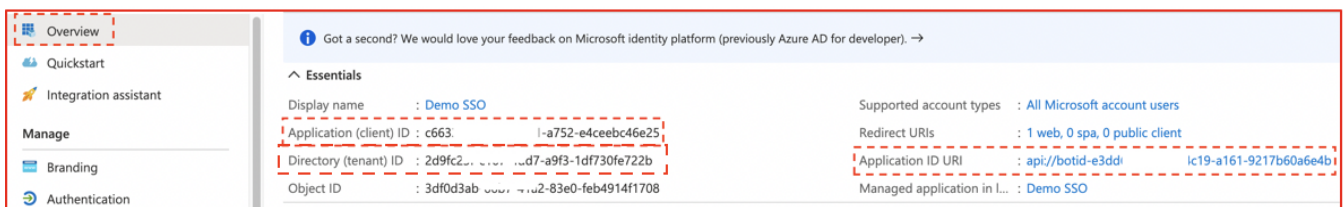


Figure 25 Overview screen

9. Grant tenant admin permissions to the AAD application

Log in with the admin account in a private browser window

- Replace <tenant-id> with the Directory (tenant) ID
- Replace in below URL the client Id value with your application Id

https://login.microsoftonline.com/<tenant-id>/adminconsent?client_id=<client-id>

Load the URL in a browser window where you signed with admin account. Accept the permissions. The prompt should look similar to this:

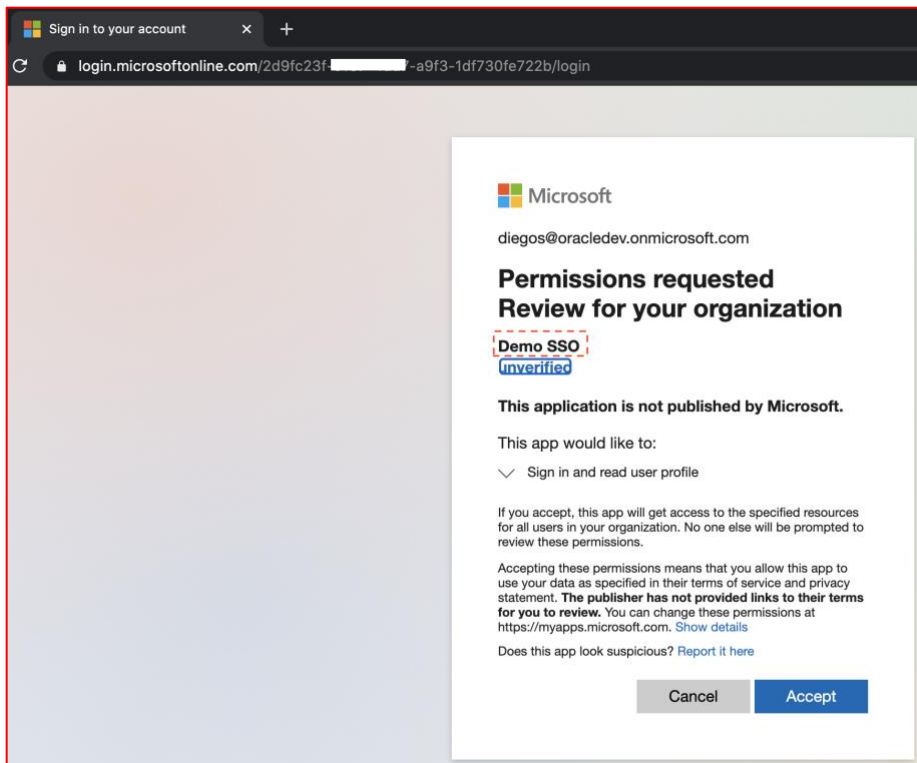


Figure 26 Private browser window

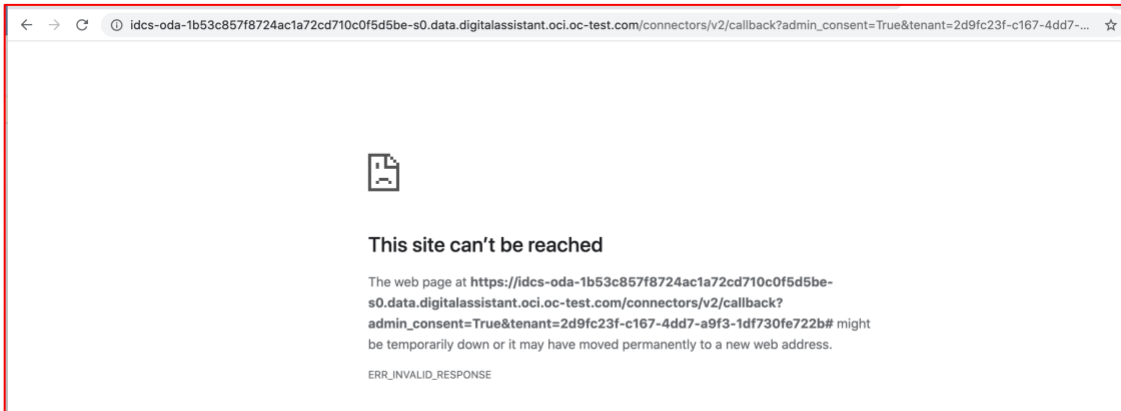


Figure 27 This screen will appear after accepting the permissions

PLEASE IGNORE THE SCREEN AFTER ACCEPTING. THIS MAY SHOW YOU ERROR, HOWEVER THE ERROR DOES NOT MATTER.

Update your MS Teams app with SSO detail

You need to update your app with the above SSO details in MS Teams. Open MS Teams and perform following steps:

Go to <https://teams.microsoft.com> using the same browser window that you used for your AAD Application registration. Do not use the Private window you just used to login using the Admin Account. You can also choose to use the MS Teams Desktop App.

If you do not see the App Studio Icon, click on the 3 dots on the left menu and Find an app by typing App Studio in the text box.

1. Open the App Studio and select **Manifest editor and select your existing app**

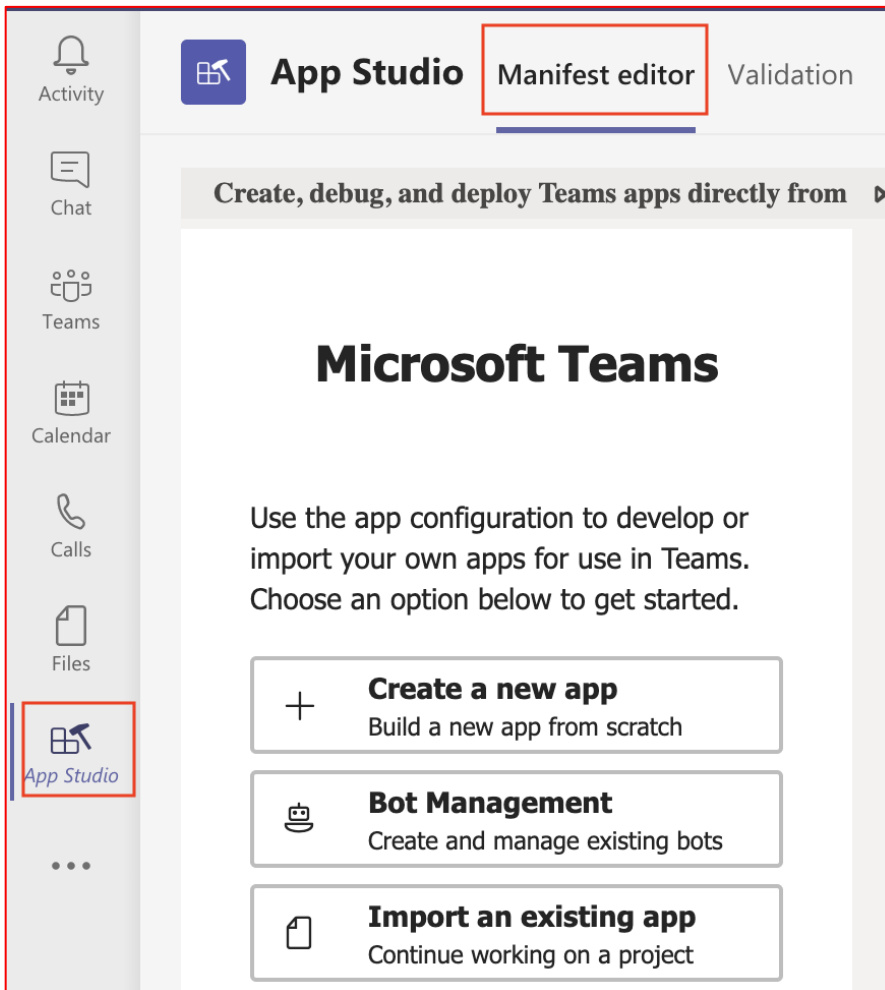


Figure 28 App Studio - Manifest editor

2. Under **Finish**, Select **Domains and Permissions** under **Finish** option
 - o Under **AAD App ID** add your Application (client) ID

- Under **Single-Sign-On** add your Application ID URI

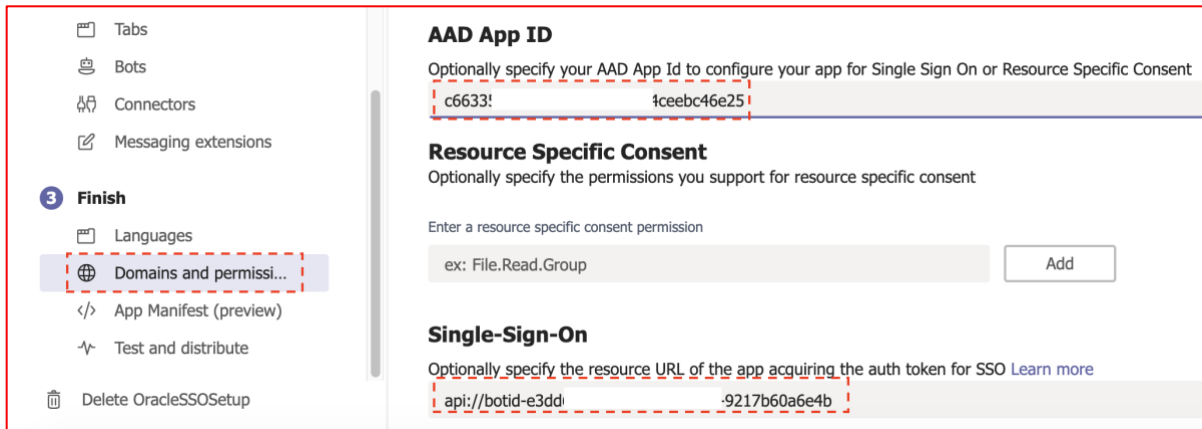


Figure 29 Update SSO details

3. Next, go to **Test and Distribute** section and **install** the application.

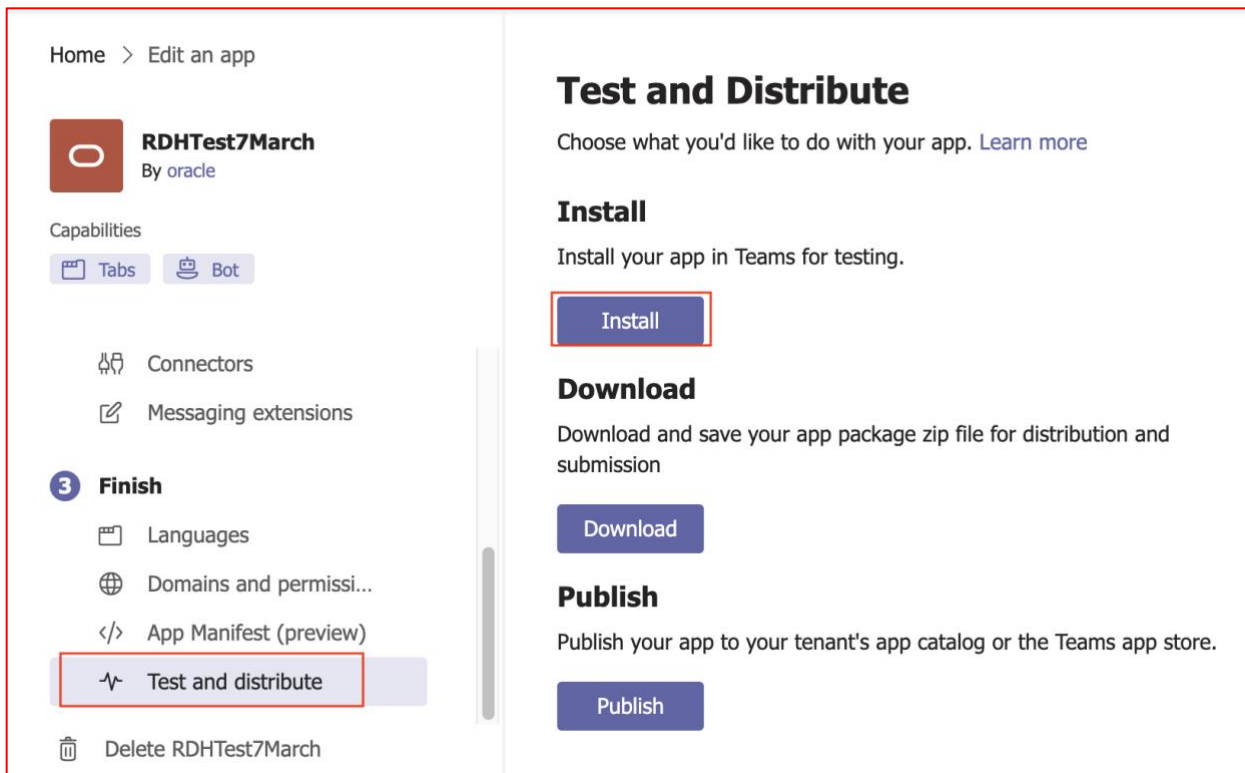



Figure 30 Install the app

- Click on the Add button so the bot gets added to MS Teams



RDHTest7March

✕

Add

test app

for testing sso feature

Bots
Chat with the app to ask questions and find info

Personal app
Keep track of important content and info

Created by: [oracle](#)
Version 1.0.0

Permissions

This app will have permission to:

- Receive messages and data that I provide to it.
- Send me messages and notifications.
- Access my profile information such as my name, email address, company name, and preferred language.
- Receive messages and data that team or chat members provide to it in a channel or chat.
- Send messages and notifications in a channel or chat.
- Access information from this team or chat such as team or chat name, channel list and roster (including team or chat member's names and email addresses) - and use this to contact them.
- user.read
- user.readbasicall

By using RDHTest7March, you agree to the [privacy policy](#) and [terms of](#)

Setup Authentication Service in Oracle Digital Assistant

Add Authentication Services

Enter these values:

Identity Provider: Microsoft Identify Platform

Name: A name to identify the authentication service.

Token Endpoint URL: The IDP's URL for requesting access tokens.

Use `https://login.microsoftonline.com/<Azure-Active-Directory-TenantID>/oauth2/v2.0/token`

Authorization Endpoint: The IDP's URL for the page that users authenticate with by entering their user name and password.

Use `https://login.microsoftonline.com/<Azure-Active-Directory-TenantID>/oauth2/v2.0/authorize`

Client ID and Client Secret: The Application (client) ID and secret of the SSO app

Scope: Scope should be `{Application (client) ID of your SSO app}/access_as_user`

Subject Claim: The access-token profile claim to use to identify the user. Use: email

The screenshot shows the 'Settings - Authentication Services' interface. On the left is a navigation menu with options like Home, Development, Analytics, Settings, Audit Trail, Authentication Services, Data Management, Feature Management, Linked Instance, Translation Services, Additional Services, Downloads..., and Documentation... The main area is titled 'Settings - Authentication Services' and contains a list of services on the left and a configuration form on the right. The service 'DemoAuthenticationService' is selected and highlighted with a red dashed box. The configuration form includes the following fields: Grant Type (Authorization Code), Identity Provider (Microsoft Identity Platform), Name (DemoAuthenticationService), Token Endpoint URL (https://login.microsoftonline.com/2d9fc25f-1037-4307-a9f3-1df730fe722b/oauth2/v2.0/token), Authorization Endpoint URL (https://login.microsoftonline.com/2d9fc25f-1037-4307-a9f3-1df730fe722b/oauth2/v2.0/authorize), Short Authorization Code Request URL (placeholder), Revoke Token Endpoint URL (placeholder), Client ID (c663352a-16e4-4cceb46e25), Client Secret (masked), Scopes (c663352a-16e4-4cceb46e25/access_as_user), Subject Claim (email), and Refresh Token Retention Period (1 days).

Figure 31 Authentication Services

Routing your skill to MS Teams channel

Please ensure that your skill is routed to the MS Teams channel. In case you plan to use the attached skill, then you will need to update the routing.

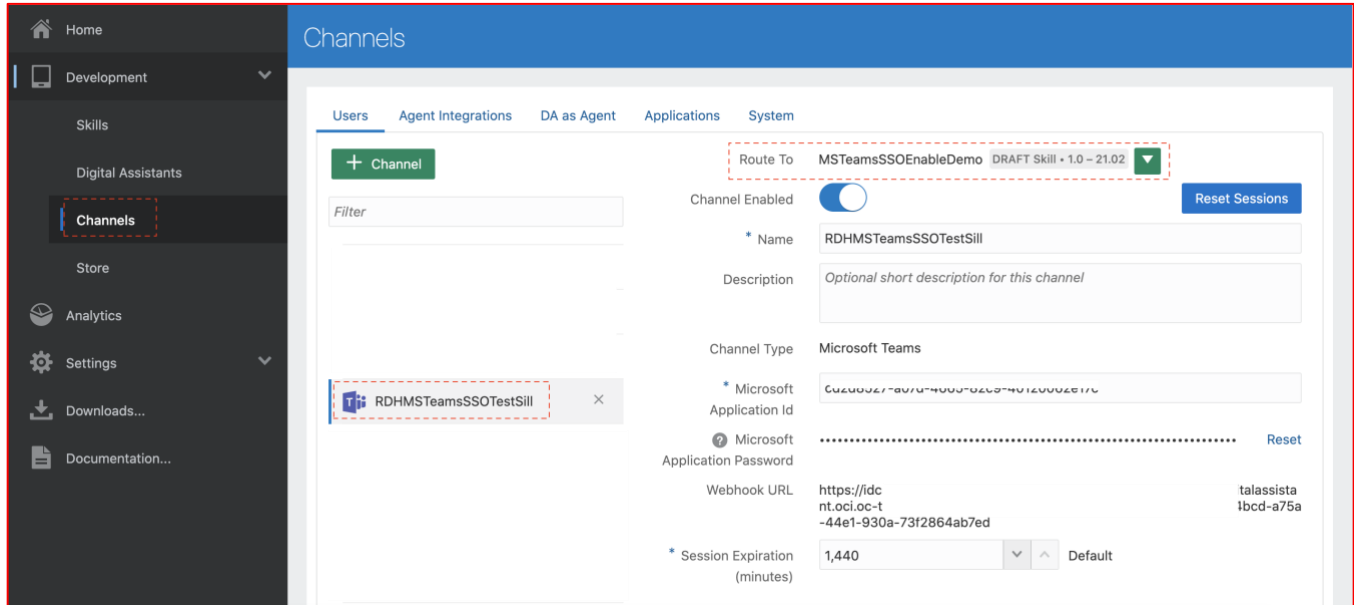


Figure 32 MS Teams channel routing

Update skill

Next, in your skill you will perform following steps

1. Get SSO access token
2. Get Middle Tier access token based on the SSO token
3. Call Graph API using the middle-tier access token

Get SSO access token

You will be using **System.OAuth2AccountLink** component to get the SSO access token.

enableSingleSignOn: set this to **true** so that users who have already signed into MS Teams don't have to sign into the skill. The default is false.

authenticationService: Make sure you point it to your authentication service name.

```

54 MicrosoftOAuth2AccountLink:
55   component: "System.OAuth2AccountLink"
56   properties:
57     prompt: |-
58       Please press the login button for me to re-direct you to Microsoft.
59     authenticationService: "RDHMicrosoftAuthService"
60     authenticatedUserVariableName: "user.userName"
61     accessTokenVariableName: "user.accessToken"
62     showCancelOption: true
63     linkLabel: "Login with Microsoft"
64     cancelLabel: "Cancel action"
65     updateUserProfile: true
66     enableSingleSignOn: true
67   transitions:
68     next: "MicrosoftOAuth2AccountLink"
69   actions:
70     pass: "printSSOToken"
71     fail: "failSSOAccessToken"
72     textReceived: "failSSOAccessToken"

```

Figure 33 System.OAuth2AccountLink component

Get access token

The token you received is an “exchange” token and cannot be used directly to invoke graph APIs, therefore cannot be used to access Graph API’s or Oracle Digital Assistant’s [Calendar system components](#).

In order to get a token that works with graph API, the “on-behalf-of” flow has to be followed using the SSO token to exchange for an access token with appropriate scopes that will work with graph APIs. You may get additional information here: <https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-on-behalf-of-flow>

To request an access token, make an HTTP POST to the tenant-specific Microsoft identity platform token endpoint with the following parameters

<https://login.microsoftonline.com/<tenant>/oauth2/v2.0/token>

You will need to update the custom parameters in skill settings to test it at your end

Tenant: Tenant ID of your Azure portal

grant_type urn:ietf:params:oauth:grant-type:jwt-bearer.

client_id The application (client) ID of SSO app

client_secret The client secret of SSO app

assertion: SSO token

scope: openid email offline_access https://graph.microsoft.com/User.ReadBasic.All

requested_token_use:on_behalf_of



Custom Parameters

+ New Parameter Filter parameters

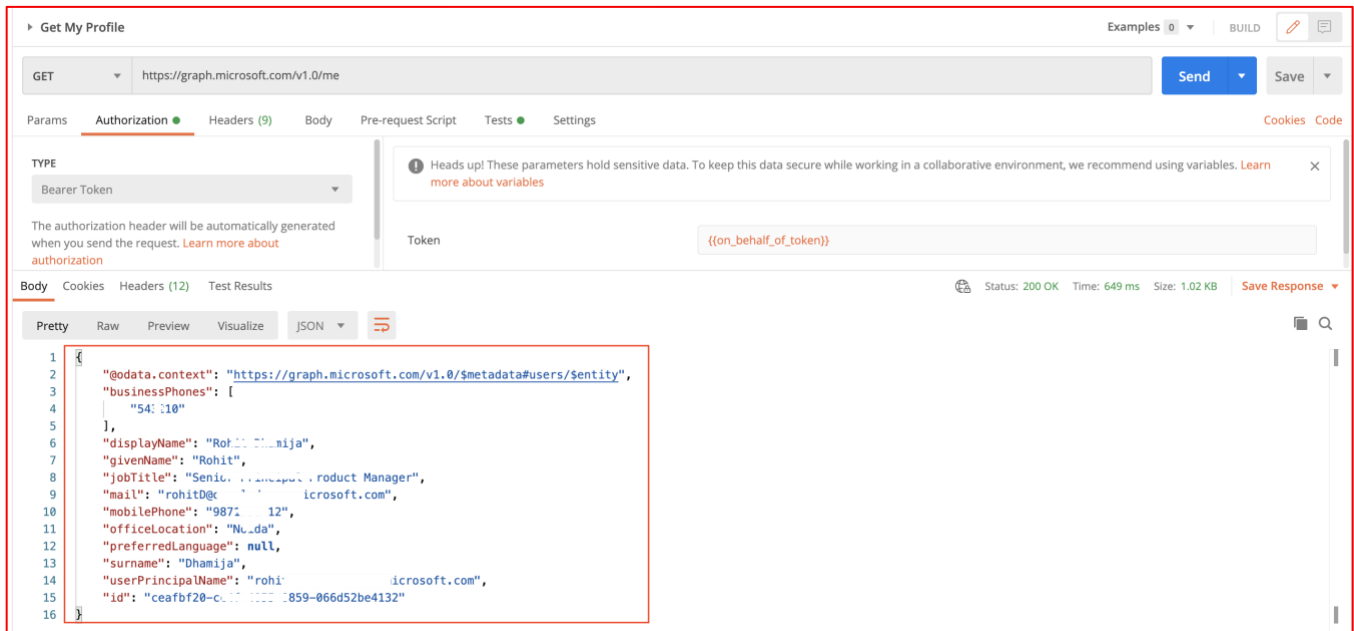
Edit Delete

Name	Display Name	Type	Value	Description
tenant	tenant	String	2d97c8f1-4071-4177-8013-1df730fe722b	
grantType	grantType	String	urn:ietf:params:oauth:grant-type:jwt-be...	
clientId	clientId	String	b1b05b20-3012-4000-b240-0b0a1c4b...	
clientSecret	clientSecret	String	TZNy8dv0c0-0f5T_0sqOrf3~b9kKZ2R...	
scope	scope	String	openid email offline_access https://grap...	
requestedTokenUse	requestedTokenUse	String	on_behalf_of	

Figure 36 Custom parameters

Call Graph API

Finally, you need to use the middle tier access token to call the graph API



The screenshot shows a Postman interface for a GET request to the Graph API endpoint `https://graph.microsoft.com/v1.0/me`. The request is configured with a Bearer Token authorization header and a token value of `{{on_behalf_of_token}}`. The response body is displayed in JSON format, showing user profile information such as `displayName`, `givenName`, `jobTitle`, `mail`, `mobilePhone`, `officeLocation`, `preferredLanguage`, `surname`, `userPrincipalName`, and `id`.

```
1 {
2   "@odata.context": "https://graph.microsoft.com/v1.0/$metadata#users/$entity",
3   "businessPhones": [
4     "54:110"
5   ],
6   "displayName": "Rohit Dhamija",
7   "givenName": "Rohit",
8   "jobTitle": "Senior Product Manager",
9   "mail": "rohitd@icloud.com",
10  "mobilePhone": "9871234567",
11  "officeLocation": "Noida",
12  "preferredLanguage": null,
13  "surname": "Dhamija",
14  "userPrincipalName": "rohitd@icloud.com",
15  "id": "ceafbf20-c000-4000-859-066d52be4132"
16 }
```

Figure 37 Postman Get call to fetch profile information

The skill calls a custom component to get the profile information and save the information in **profileInformation** parameter. Finally, the information is displayed using system's common response component.

```

114   getProfileInformation:
115     component: "getmyprofile"
116     properties:
117       variable: "profileInformation"
118       access_token: "${middleTierToken.value}"
119     transitions:
120     actions:
121       success: "printUserProfile"
122       failure: "globalErrorHandler"
123
124   printUserProfile:
125     component: "System.CommonResponse"
126     properties:
127       processUserMessage: false
128       keepTurn: false
129     metadata:
130       responseItems:
131       - type: "text"
132         text: |-
133
134           Okay. So this is your profile information:
135
136           First Name: "${profileInformation.value.givenName}"
137           Surname: "${profileInformation.value.surname}"
138           Job Title: "${profileInformation.value.jobTitle}"
139           Email: "${profileInformation.value.mail}"
140           Mobile Phone: "${profileInformation.value.mobilePhone}"
141           Office Location: "${profileInformation.value.officeLocation}"
142
143
144     transitions:
145     return: "done"

```

Figure 38 Custom component to fetch profile information and display it using common response component

Test

System Tester

Now you will test your skill in system tester as shown below:

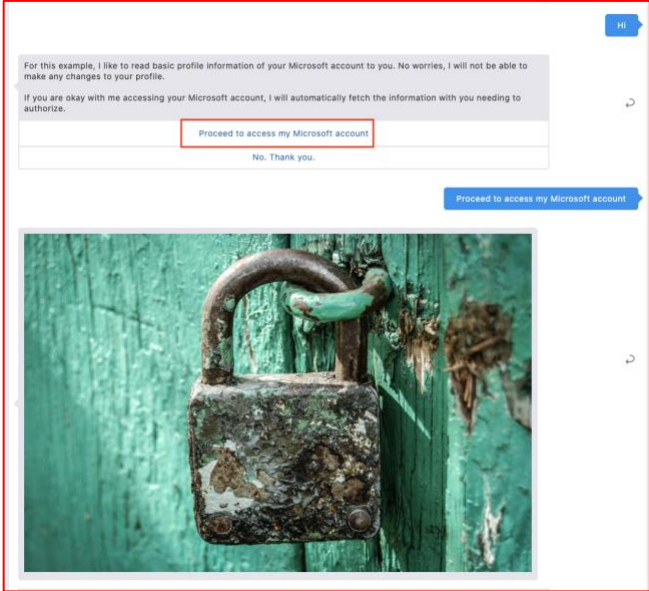
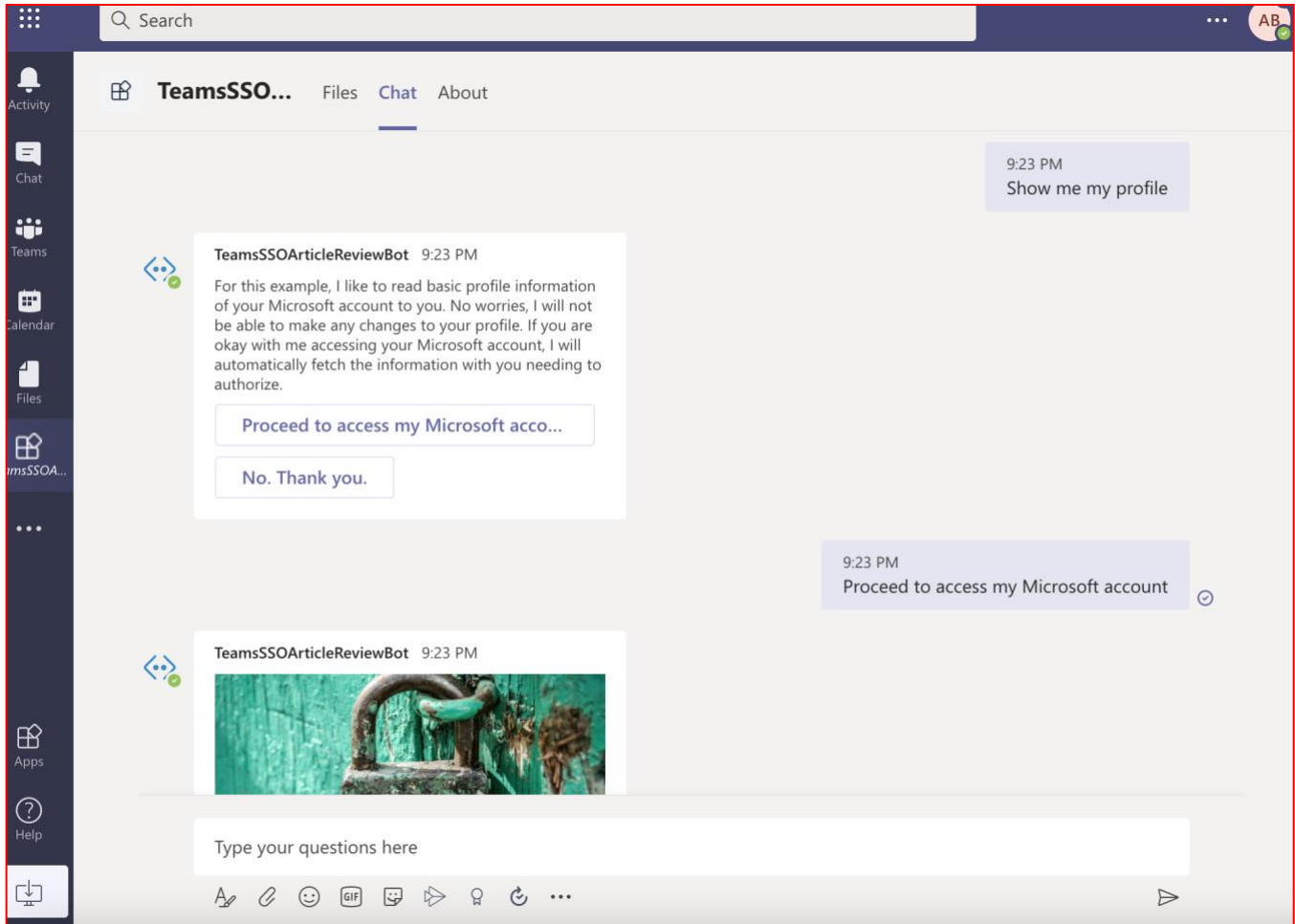


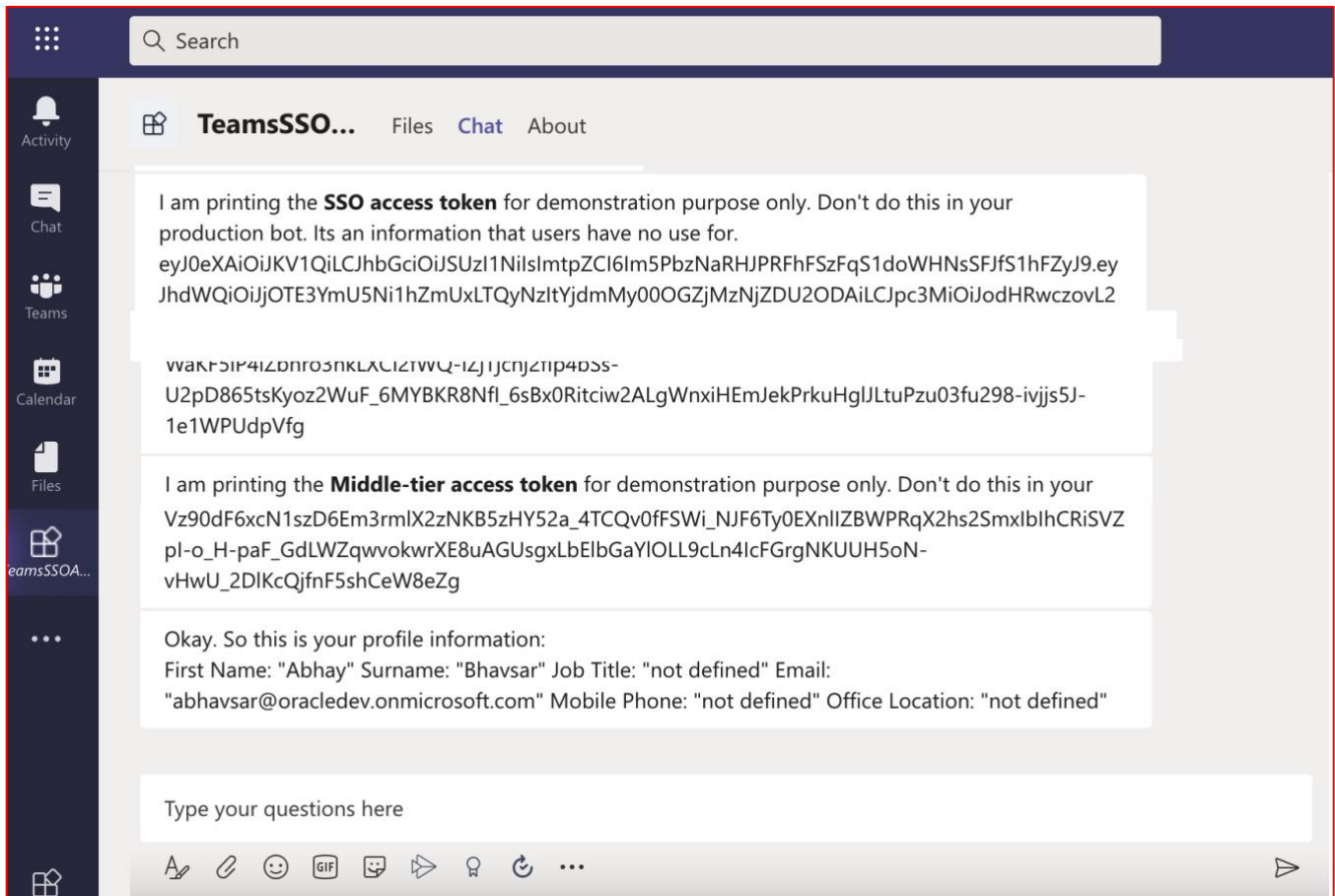
Figure 39 System tester

MS Teams app

Finally, you will test it from MS Teams app



When you click on Process to access my Microsoft account, you are not taken to the Microsoft authentication screen. The System.OAuth2AccountLink component automatically performs the SSO due to the SSO configuration done in MS Teams. You will directly be taken to see the SSO access token, the middle-tier token followed by the profile information.



Conclusion

The article detailed the steps to set up single sign-on (SSO) authentication for an Oracle digital assistant that is exposed through a Microsoft Teams channel. After the set-up, users just need to log in to Teams with their Azure AD credentials and then seamlessly interact with the digital assistant and access the profile information without the need to re-authenticate.