Application Registration and Authentication

For your application to access Microsoft backend data (Office 365, SharePoint, …), you have two options:

1. Configure your app package-solution.json file with “Microsoft graph” resource and scope pairs.
2. Register and configure your application with Microsoft Azure AD using your tenant account (Authorization, API, SCOPE).

## Authentication and authorization Using App Package Solution

Example: if your app will use Microsoft graph to make api calls to both the user profile and files stored in Microsoft document libraries of your tenant site,

* 1. Dev Code Base: go to your app code base -> config -> package-solution.json then add this code block under the “solution”:{… } construct and save:

"webApiPermissionRequests": [

{

"resource": "Microsoft Graph",

"scope": "User.Read"

},

{

"resource": "Microsoft Graph",

"scope": "Sites.Read.All"

}

]

* 1. Dev Code Base: Package the App
     1. Cd to your app code base then:
     2. Create a bundle: gulp bundle –-ship
     3. Create a package: gulp package-solution –-ship
  2. Tenant App Catalog: Upload the newly created package to you tenant App catalog : <https://jaxspdev.sharepoint.com/sites/apps/SitePages/Home.aspx>
     1. App catalog -> distribute apps for [sharepoint | office] -> upload
     2. Follow the wizard
  3. Tenant Admin: Grant API Access permissions
     1. <https://jaxspdev-admin.sharepoint.com/_layouts/15/online/AdminHome.aspx#/home>
     2. Login to your tenant admin console
     3. Click on “Advanced” -> “API access” -> “Pending Requests” and approve pending requests as needed

## Authentication and authorization Using App Registration

The basic steps required to configure a service and get a token from the Microsoft identity platform endpoint that your service can use to call Microsoft Graph under its own identity are:

1. Register your app.
2. Configure permissions for Microsoft Graph on your app.
3. Get administrator consent.
4. Get an access token for your app
   1. Sign users in using OpenID connect
   2. Get access token
5. Use the access token to call Microsoft Graph.
6. See: <https://docs.microsoft.com/en-us/graph/auth-v2-service>
7. See: <https://www.youtube.com/watch?time_continue=1&v=IghiKqly-HY&feature=emb_logo>
   1. See: <https://github.com/microsoftgraph/msgraph-training-reactspa/tree/master/demo/graph-tutorial>
8. See: <https://docs.microsoft.com/en-us/graph/permissions-reference>
9. See: <https://docs.microsoft.com/en-us/azure/active-directory/develop/scenario-spa-sign-in?tabs=javascript>
10. Register your application with Microsoft Azure AD
    1. Log onto <https://portal.azure.com/#home>
    2. Click on “View” under the **Manage Azure Active Directory section**
    3. On the left menu, click on “App registration”
    4. On the App registration page, click “New registration”
       1. Enter the name of your app “RQC Access Request App”
       2. Select “Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)”
       3. Set the Web field to <https://localhost:4321> – under the Redirect URI section
       4. Click Register
       5. In this case we got
          1. Client id: 21954173-f0ec-4d81-943a-f9d91d322713
          2. Tenant id: b56bc27e-6da9-4384-9cad-73117b11b0b3
11. Configure permissions for Microsoft Graph on the newly registered application
    1. Logon to your tenant account using <https://portal.azure.com>
    2. Go to App registrations then click on your new app
    3. Click on “Authentication” on the left menu
    4. On the “Authentication” page, under the “Implicit grant”, select the two check boxes (access tokens, and ID tokens)
    5. Click on “Certificates & secrets”
    6. Click “New client secret” then enter a name for the secret key
    7. Click “Add” to generate the ticket
    8. Copy and save the generated key for later use
    9. Set API permissions to application level for both graph and SharePoint access – this requires admin consent
       1. Microsoft graph - <https://docs.microsoft.com/en-us/graph/>
       2. Sharepoint
12. Grant Admin consent - Requesting Admin consent:
    * + - 1. Paste this link on your browser:
          2. https://login.microsoftonline.com**/tenant\_id**/adminconsent?client\_id=**app\_client\_id**&state=12354&redirect\_uri=aut\_redirirect\_id

In our example :

* tenant\_id = b56bc27e-6da9-4384-9cad-73117b11b0b3
* app\_client\_id = 21954173-f0ec-4d81-943a-f9d91d322713
* aut\_redierct\_uri = https%3A%2F%2Flocalhost%3A4321
  + - * 1. Then click access

Redirected link: <http://localhost:4321/?admin_consent=True&tenant=b56bc27e-6da9-4384-9cad-73117b11b0b3&state=12354>

https://login.microsoftonline.com**/**b56bc27e-6da9-4384-9cad-73117b11b0b3/adminconsent?client\_id=1ffb5f51-f2ff-482c-bd76-578ebb88d392

&state=12354&redirect\_uri=https%3A%2F%2Flocalhost%3A3000

1. Implement a service to Get an access token – using axios, or ajax, or fetch, …

**Request Header configuration:**

**Method:** Post

**Host:** login.microsoftonline.com

**Content-Type:** application/x-www-form-urlencoded

**Request data configuration:**

**client\_id** = this\_client\_id

**scope** = https%3A%2F%2Fgraph.microsoft.com%2F.default

**client\_secret** = this\_client\_secret

**grant\_type** = client\_credentials

**Example:** In your app, you can implement a service that uses axios to get access token using the configuration below

POST https://login.microsoftonline.com/{tenant}/oauth2/v2.0/token HTTP/1.1

Host: login.microsoftonline.com

Content-Type: application/x-www-form-urlencoded

client\_id=535fb089-9ff3-47b6-9bfb-4f1264799865

&scope=https%3A%2F%2Fgraph.microsoft.com%2F.default

&client\_secret=qWgdYAmab0YSkuL1qKv5bPX

&grant\_type=client\_credentials

// Line breaks are for legibility only

1. Use the access token to call Microsoft Graph

To use MS graph api calls that reads file content stored in MS SP document library on the cloud, your APP should implement a two-step process:

1. Use graph api to get the downloadWebURL of the target file
2. Use axios to get the content of the file using the link returned in step 1

Example: to fetch the content of test.req\_loc.json from <https://jaxdev.sharepoint.com/Shared%20Documents/models>

* Given SP driveId of “Shared Documents” folder is “b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I”
* Call <https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/root:/models/test.req_loc.json>

This returns a json response with @microsoft.graph.downloadUrl field which you can use to call axios for data extraction

Response from graph:

{

"@odata.context": "https://graph.microsoft.com/v1.0/$metadata#sites('jaxspdev.sharepoint.com')/drives('b%21Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I')/root/$entity",

"@microsoft.graph.downloadUrl": "https://jaxspdev.sharepoint.com/\_layouts/15/download.aspx?UniqueId=8bce2e5c-f6ef-451f-91b9-27e51433d9a2&Translate=false&tempauth=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0..dXhmM2tPVWN0UWR2bXZHY1hPVUlHakpoOWNRODdRMy9mMDFFVjB5S0h4Zz0&ApiVersion=2.0",

"createdDateTime": "2020-04-25T14:55:32Z", ..}

Appendix:

To get access token manually:

https://login.microsoftonline.com/b56bc27e-6da9-4384-9cad-73117b11b0b3/oauth2/authorize?client\_id=21954173-f0ec-4d81-943a-f9d91d322713&response\_type=code&redirect\_uri=http%3A%2F%2Flocalhost%3A4321&response\_mode=query&resource= 21954173-f0ec-4d81-943a-f9d91d322713&state=12345

To get the access token: https://login.microsoftonline.com/b56bc27e-6da9-4384-9cad-73117b11b0b3/oauth2/authorize?client\_id=21954173-f0ec-4d81-943a-f9d91d322713&response\_type=code&redirect\_uri=http%3A%2F%2Flocalhost%3A4321& state=12345

Sample MS Graph queries :

* Using graph explorer - <https://developer.microsoft.com/en-us/graph/graph-explorer>
* List all the document Libraries of tenant jaxdev.sharepoint.com
  + <https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com:/drives>
* List all the document Libraries of the rqcsite sub-site of tenant jaxdev.sharepoint.com
  + <https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com:/sites/rqcsite:/drives>
* List all the children of the /Shared Documents/ drive of tenant jaxdev.sharepoint.com
  + <https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/root/children>
  + In the example we use the drive id of /Shared Document/ folder which was collected from the previous query – tenant’s document libraries listing

<https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/items/01XXZICP5ZB7U62JV3Q5FKD7GZT3YRBSHQ/children> or <https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/root:/models:/children>

List all the children under /Shared Document/models/ -- models id 01XXZICP5ZB7U62JV3Q5FKD7GZT3YRBSHQ

<https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/root:/models/active_au.json>

To get a file content : https://graph.microsoft.com/v1.0/sites/jaxspdev.sharepoint.com/drives/b!Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I/items/01XXZICP7HNAS6LDRRPZGLJLIR6IGY5KDM?select=microsoft.graph.downloadUrl

{

"@odata.context": "https://graph.microsoft.com/v1.0/$metadata#sites('jaxspdev.sharepoint.com')/drives('b%21Rz7Occ-V-keEg8HtjjsX6xBg2qwSArxFmxDMOyfVfjwfxVW1KVqCS4YpTePYBj0I')/items/$entity",

"@odata.etag": "\"{E52568E7-318E-4C7E-B4AD-11F20D8EA86C},1\""

}

Useful Links:

1. <https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-on-behalf-of-flow>
2. <https://docs.microsoft.com/en-us/graph/auth-v2-service>
3. <https://docs.microsoft.com/en-us/sharepoint/dev/spfx/use-msgraph>
4. <https://github.com/microsoftgraph/msgraph-training-reactspa/tree/master/demo/graph-tutorial/src>
5. <https://docs.microsoft.com/en-us/sharepoint/dev/spfx/web-parts/get-started/connect-to-sharepoint>
6. <https://docs.microsoft.com/en-us/azure/active-directory/develop/msal-js-initializing-client-applications>
7. <https://github.com/azureadquickstarts/appmodelv2-webapp-openidconnect-nodejs>
8. <https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-configure-app-access-web-apis#add-permissions-to-access-web-apis>
9. <https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-protocols-oidc>
10. <https://github.com/Azure-Samples/active-directory-javascript-graphapi-v2/blob/quickstart/JavaScriptSPA/authRedirect.js>
11. <https://github.com/AzureAD/microsoft-authentication-library-for-js/issues/1288>
12. <https://medium.com/@franciscopa91/how-to-implement-oidc-authentication-with-react-context-api-and-react-router-205e13f2d49>
13. <https://www.koskila.net/how-to-fix-admin-consent-issues-aadsts65001/>
14. <https://docs.microsoft.com/en-us/graph/api/resources/onedrive?view=graph-rest-1.0>
15. <https://bilot.group/articles/authentication-with-azure-ad-using-msal-react-app/>
16. <https://kaboodlesoftware.com/2019/07/24/creating-web-part-that-support-single-part-app-pages-in-sharepoint-online/>
17. How to deploy : <https://docs.microsoft.com/en-us/sharepoint/dev/spfx/web-parts/single-part-app-pages>

Create users:

1. Lisa Ross