**Deployment Guide**

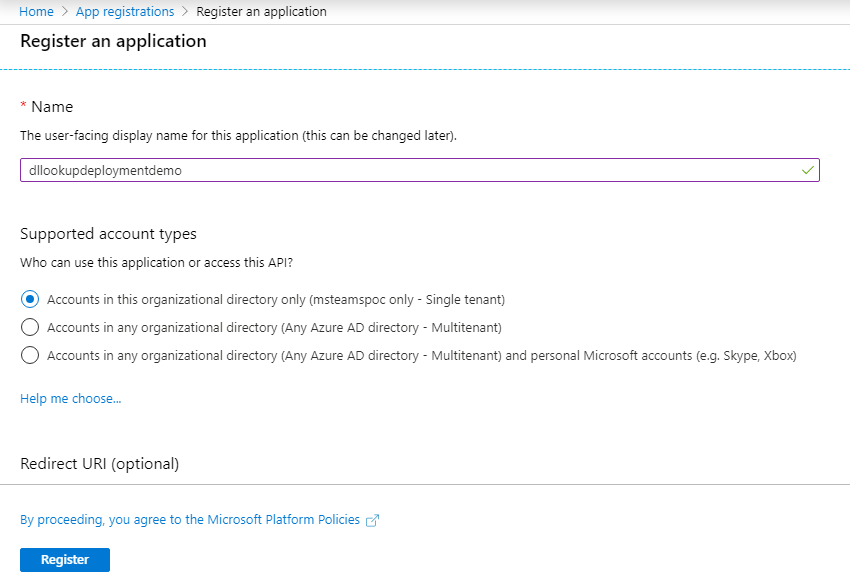
**Prerequisites:**

1. **Git to be installed:** <https://git-scm.com/downloads>
2. **Side loading should be enabled:**

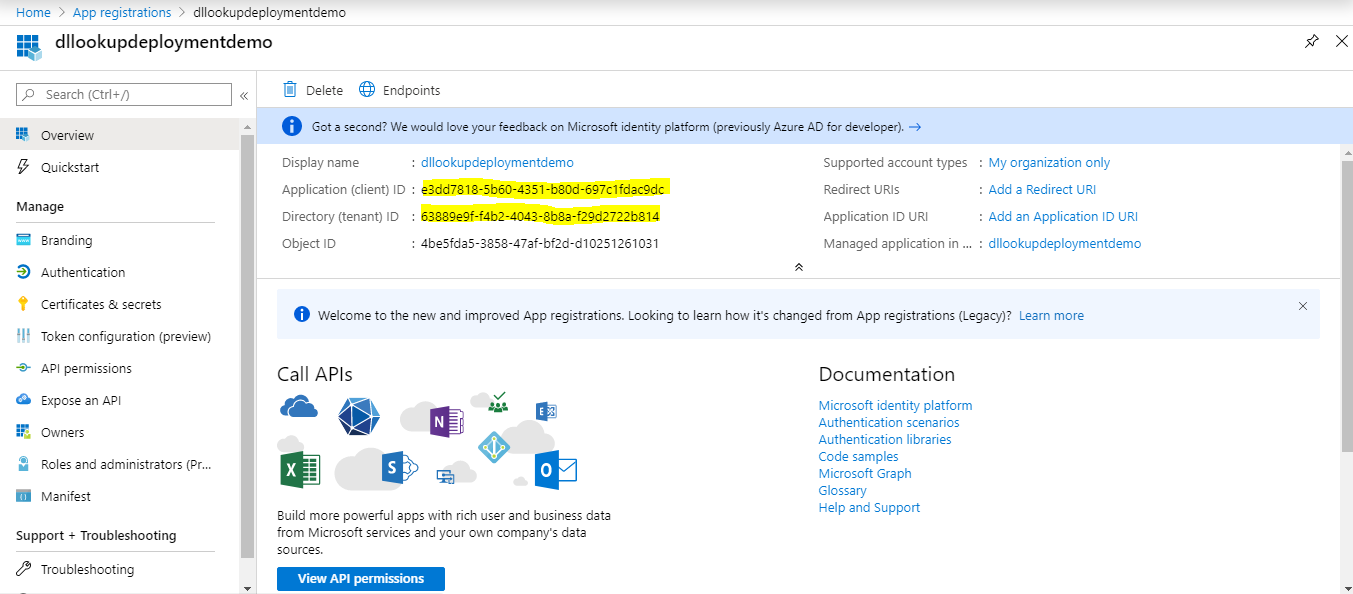
* Open [Teams admin portal](https://admin.teams.microsoft.com/policies/app-setup) and it will show setup policies.
* Click on “Global (Org-wide default)” to the policies. Make sure “**Upload custom apps**” toggle is ON in “Global” else change it and save the change.
* NOTE: It might take 24 hours for this change to take place organization wide.

**Step 1: Register Azure AD application**

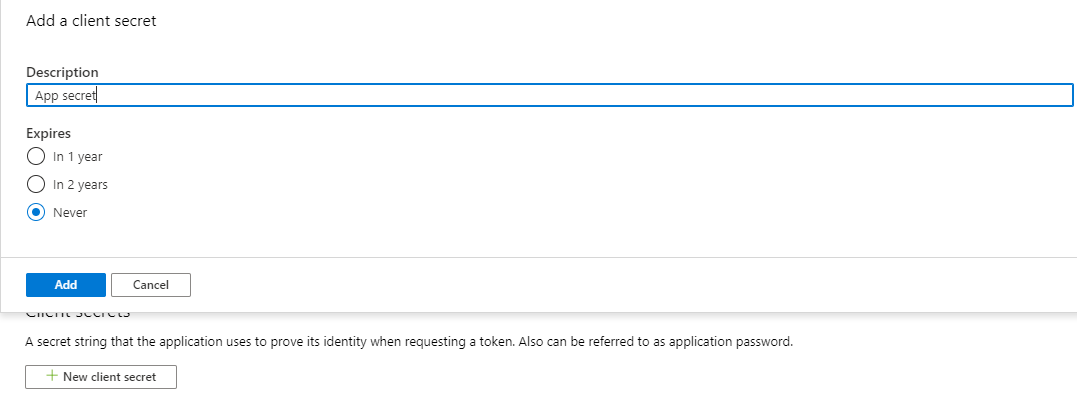
1. Log in to the [Azure Portal](https://portal.azure.com) for your subscription where you have Office 365 tenant, and search with "App registrations" in the search bar. It will redirect to the "App registrations" service.
2. Click on "New registration" to create an Azure AD application.
   1. **Name**: The name of your Teams app(Ex: dllookupdeploymentdemo).
   2. **Supported account types**: Select "**Accounts in this organizational directory only**".
   3. Leave the "Redirect URI" field blank for now.



3. Click on the "Register" button.

4. When the app is registered, you'll be taken to the app's "Overview" page. Copy the **Application (client) ID, Directory (tenant) ID**; we will need it later. Verify that the "Supported account types" is set to “**My organization only”**.

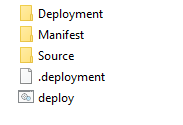
1. On the side rail in the “Manage” section, Click on "**Certificates & secrets**" link. In the “Client secrets” section, click on "+ **New client secret**". Add a description for the secret and choose “Never” in Expires field. Click on "Add".



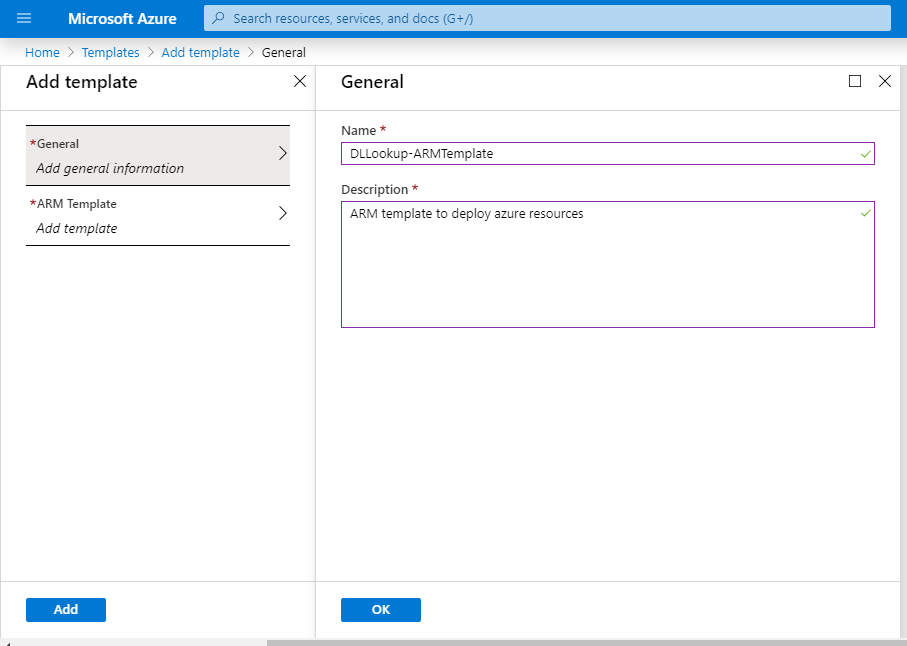
1. Once the client secret is created, copy it’s Value; we will need it later.

**Step 2: Deploy to your Azure subscription**

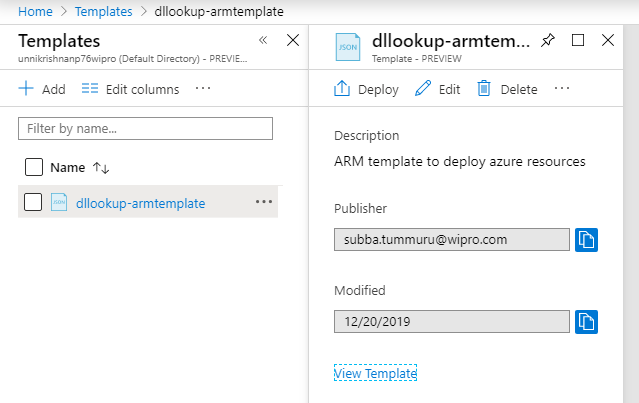
1. Download Source code.



1. Login to Azure portal and search for “Templates”. Templates page will be shown. Click on “+ Add”. Following page will be shown. Enter Name, Description and Click on “Ok”.



1. Click on “ARM Template” in left navigation.
2. Copy the content of “azuredeploy.json” file in Deployment folder and paste it in ARM Tempate text area. Click on “Ok” to proceed.
3. Click on “Add” to save the template details.
4. Click “Refresh” to see the created ARM template.
5. Click on ARM template name and click on “Deploy”.

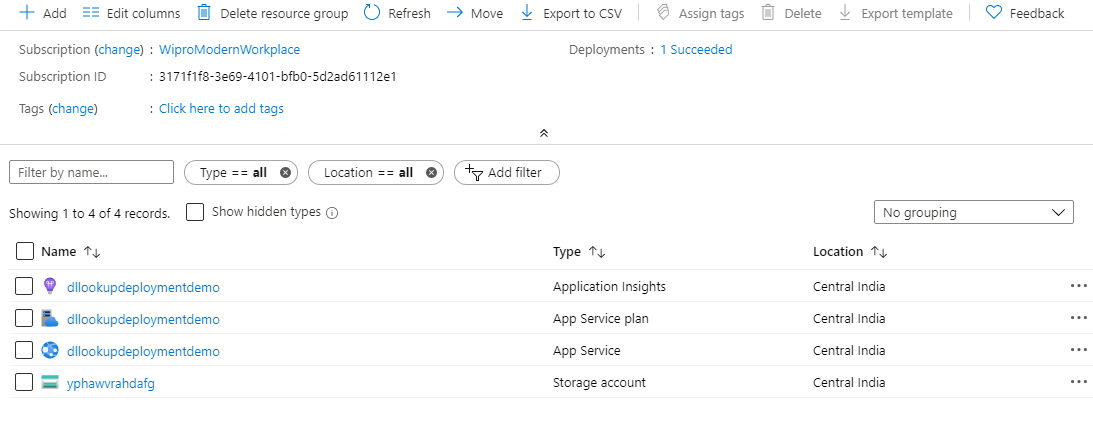


1. Fill the form as shown below:

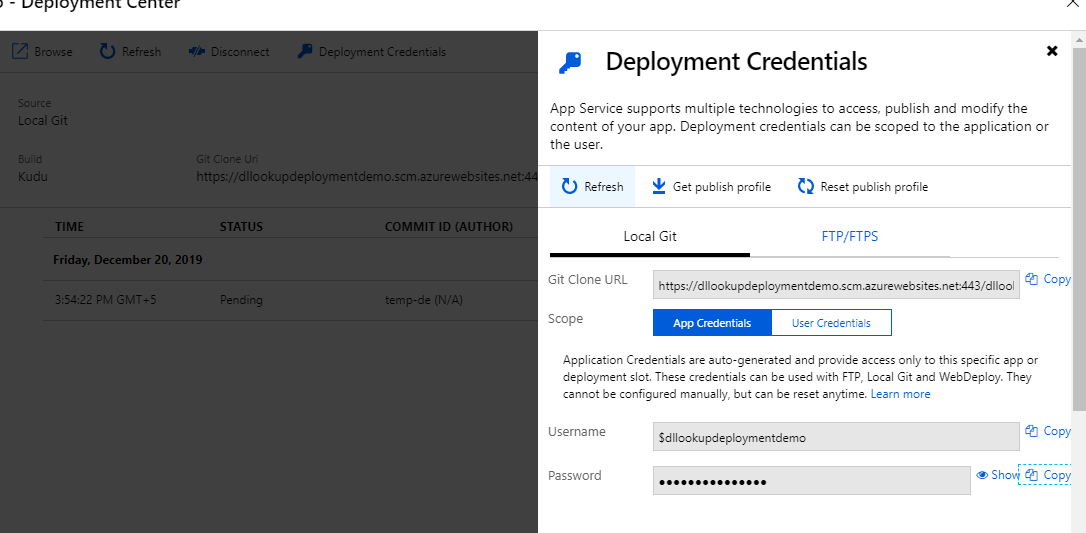
* Subscription: Select the subscription
* Resource Group: We recommend creating a new resource group. Click on “New Resource Group”, Enter resource group name. The resource group location must be in a datacenter that supports: Application Insights; Service Bus; Azure Functions; App Service; Storage Accounts. Select a suitable location.
* Base Resource Name: The app service name [Base Resource Name] must be available. For example, if you select dllookupdeploymentdemo as the base name, the name dllookupdeploymentdemo must be available (not taken); otherwise, the deployment may fail with a Conflict error.
* App Client ID: The application (client) ID which was copied to Notepad
* App Client Secret: The client secret which was copied to Notepad
* App Display Name: Change the name in case required.
* App Description: Change the description in case required.
* Tenant ID: if your Azure subscription is in a different tenant than your Teams app, please change the tenantID field to the tenant in which you are deploying this Teams app.
* Hosting Plan Sku: Change the plan as per preferences. Pricing plan details [here](https://azure.microsoft.com/en-us/pricing/details/app-service/windows/).
* Hosting Plan Size: Change the size as per preferences
* Location: Select the location same as Resource group.
* Graph Scope: Leave as it is.
* Skype Scope: Leave as it is.
* Cache Interval: Leave as it is. Change in case if you prefer to the change presence information caching interval.

Select Checkbox to accept “terms and license” and click on Purchase.

1. It will take a while to create the resources. Check notification area to see the update.
2. Once completed, Click on “Go to Resource group”. Below page will be shown with the list of resources created.



1. Click on app service name to redirect to app service details.
2. Click on “Deployment center”.
3. Select “Local Git” and click on “Continue”.
4. Select “App Service build service” and click on “Continue”.
5. Click on “Finish” button. It will create git url which will be be used for code deployment. Click on “Deployment Credentials” and Copy the username and password to notepad.



1. Open command prompt and navigate to downloaded folder path
2. Enter following commands in command prompt:

git init

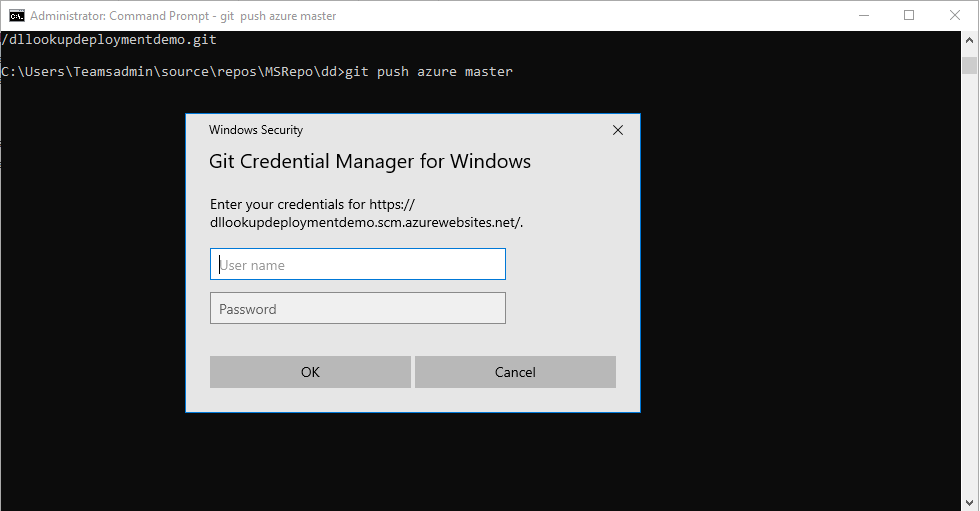
git add .

git commit -m "initial commit"

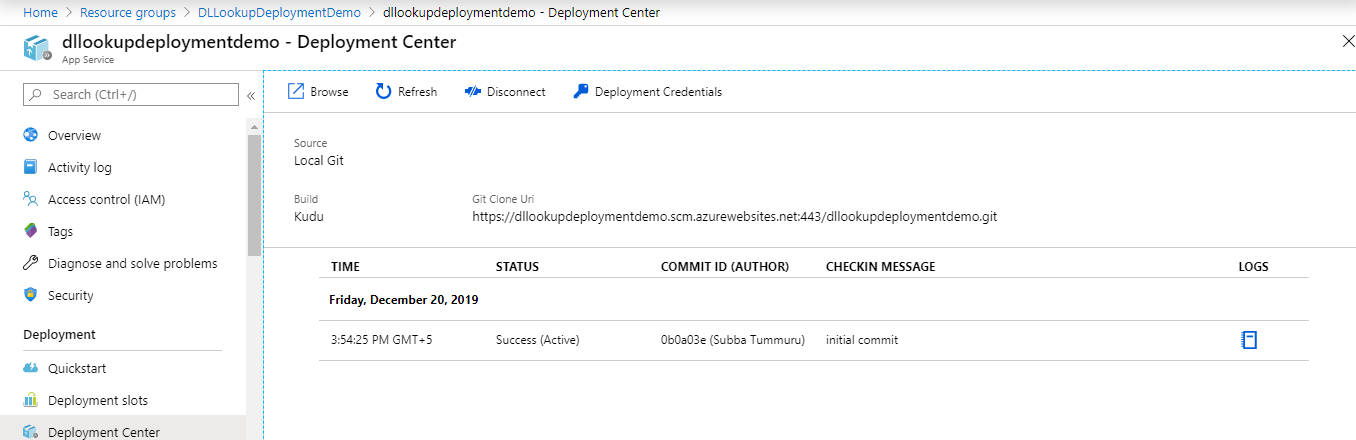
git remote add azure <git url from above>

git push azure master

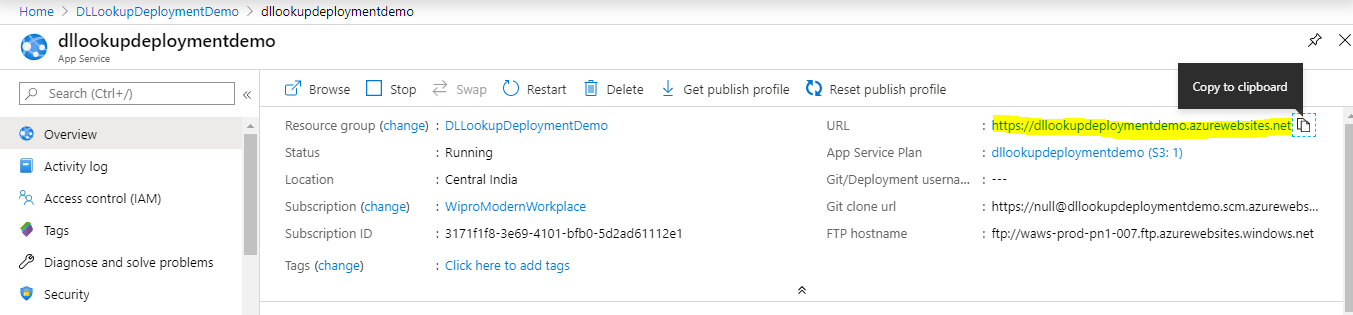
1. Popup opens to enter credentials: Enter deployment credentials which was copied in above steps.



1. It will take some time to deploy the code. Check the progress in deployment center. Once deployment completed, “Success” will be shown in Status.

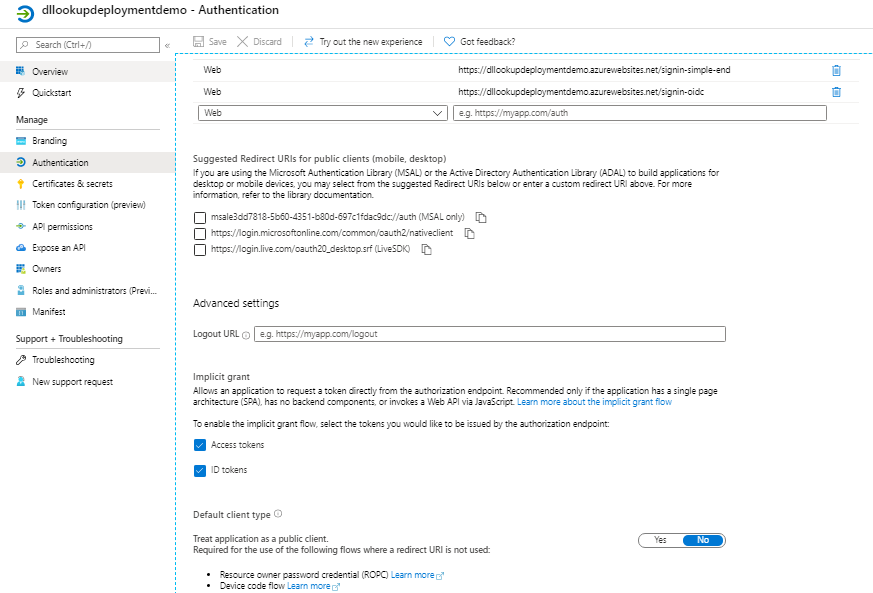


1. Click on Overview and Copy URL

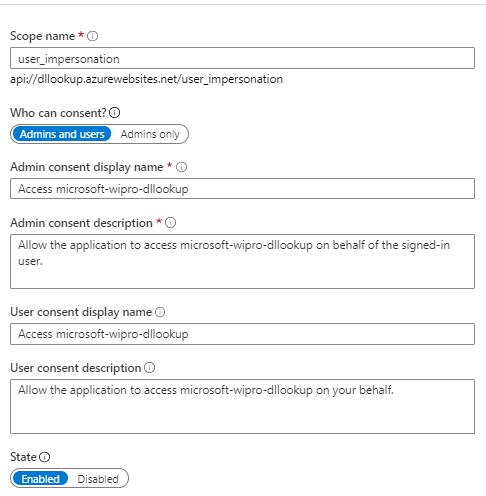


**Step 3: Set up authentication**

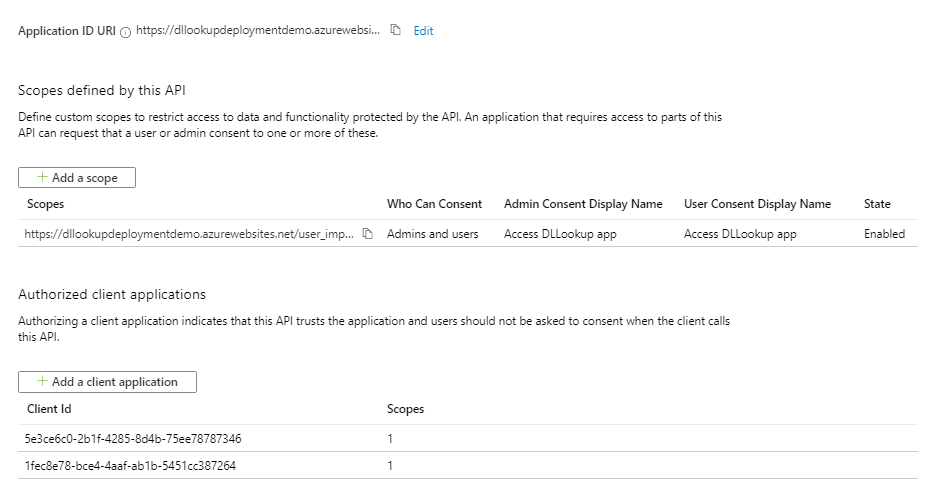
1. In continuation to Step 1. Click on “App Registrations” and open the app registration which was created in Step 1.
2. Under "Manage", click on "**Authentication**" to bring up authentication settings.
3. Add two entries to "Redirect URIs":
   * **Type**: Web
   * **Redirect URI**: Enter "%appDomain%/signin-simple-end" and “%appDomain%/signin-oidc”for in Redirect URL field e.g.  [https://dllookupdeploymentdemo.azurewebsites.net/signin-simple-end](https://dllookupkaiser.azurewebsites.net/signin-simple-end),  [https://dllookupdeploymentdemo.azurewebsites.net/signin-oidc](https://dllookupkaiser.azurewebsites.net/signin-oidc).
4. Under "Implicit grant", check both “**Access tokens**” and "**ID tokens**".
5. Select “No” in Default client type.
6. Under Supported account types, select “Accounts in this organizational directory only”.
7. Click "Save" to commit your changes.



1. Back under "Manage", click on "**Expose an API**".
2. Click on the "Set" link next to "**Application ID URI**" and change the value to "https://<<appDomain>>" e.g. https://dllookupdeploymentdemo.azurewebsites.net.
3. Click "Save" to commit your changes.
4. Click on "**Add a scope**", under "Scopes defined by this API". In the fly out that appears, enter the following values:
   * **Scope name:** user\_impersonation
   * **Who can consent:** Admins and users
   * **Admin and user consent display name:** Provide display name
   * **Admin and user consent description:** Enter “Allow the application to access on behalf of the signed-in user”.



1. Leave remaining fields as it is and Click on "**Add scope**" to commit your changes.
2. Click on "**Add a client application**", under "Authorized client applications". In the fly out that appears, enter the following values:
   * **Client ID**: Provide the following client id : 5e3ce6c0-2b1f-4285-8d4b-75ee78787346
   * **Authorized scopes**: Select the scope that ends with user\_impersonation. (There should only be 1 scope in this list.)
3. Click "Add application" to commit your changes.
4. Click on "**Add a client application**", under "Authorized client applications". In the fly out that appears, enter the following values:
   * **Client ID**: Provide the following client id : 1fec8e78-bce4-4aaf-ab1b-5451cc387264
   * **Authorized scopes**: Select the scope that ends with user\_impersonation. (There should only be 1 scope in this list.)
5. Click "Add application" to commit your changes. After making above changes, screen will be shown as follows:



1. Back under "Manage", click on "**Manifest**".
2. In the editor that appears, find the optionalClaims property in the JSON Azure AD application manifest, and replace it with the following block:

|  |
| --- |
| "optionalClaims": {  "idToken": [],  "accessToken": [  {  "name": "upn",  "source": null,  "essential": false,  "additionalProperties": []  }  ],  "saml2Token": []  }, |

Also verify below properties

|  |
| --- |
| "allowPublicClient": true  "oauth2AllowIdTokenImplicitFlow": true  "oauth2AllowImplicitFlow": true |

16. Click "Save" to commit your changes.

**Step 4: Assign Permissions to your app**

1. Continuation to Step3 (Above steps).
2. Select “**API Permissions**” blade from the left hand side.
3. Click on “**Add a permission**” button to add permission to your app.
4. In Microsoft APIs under Select an API label, select the particular service and give the following permissions,

Under “Commonly used Microsoft APIs”

* Select “**Microsoft Graph**”, then select “**Delegated permissions**” and check the following permissions,
  + Group.Read.All
  + openid
  + profile
  + User.ReadBasic.All

Click on “**Add Permissions**” to commit your changes.

* Click on “Add Permission” and then select “**Skype of Business**”, then follow same steps above to provide the following delegate permissions,
  + Contacts.ReadWrite

Click on “**Add Permissions**” to commit your changes

1. If you are logged in as the Global Administrator, click on the “**Grant admin consent for %tenant-name%**” button to grant admin consent else, inform your Admin to do the same through the portal or follow the steps provided [here](https://docs.microsoft.com/en-us/azure/active-directory/manage-apps/configure-user-consent#grant-admin-consent-through-a-url-request) to create a link and sent it to your Admin for consent.

# Step 5: Create the Teams app packages

1. Open [Teams admin portal](https://admin.teams.microsoft.com/policies/app-setup) and it will show setup policies.
2. Click on “Global (Org-wide default)” to the policies. Make sure “**Upload custom apps**” toggle is ON in “Global” else change it and save the change.

NOTE: It might take 24 hours for this change to take place organization wide.

1. Open Manifest folder from downloaded Source code folder and make following changes in manifest file.
2. Update the placeholders indicated with <<>> in the manifest to values appropriate for your organization.
   * <<App-Client-ID>>: The application (client) ID which was copied to notepad
   * <<Companyname>>: Enter your company name
   * <<WebsiteUrl>>: Company web site URL
   * <<PrivacyUrl>>: Company Privacy URL
   * <<termsOfUseUrl>>: Company Terms of Use URL
   * <<Appdomain>>/dls: Replace App domain value. e.g."<https://dllookupdeploymentdemo.azurewebsites.net/dls>".
   * <<Appdomain>>/help: Replace with App domain value. e.g:https://dllookupdeploymentdemo.azurewebsites.net/help".
   * <<Appdomain>> under Valid Domains: Replace with App domain value. e.g:dllookupdeploymentdemo.azurewebsites.net
   * <<TenantName>> under validDomains: Replace with Tenant name. e.g:msteamspoc.onmicrosoft.com
   * <<App-Client-ID>> under webApplicationInfo: The application (client) ID which was copied to notepad.
   * <<Appdomain>> under webApplicationInfo: Replace with App domain value.
3. Create a ZIP package with the manifest.json, color.png and outline.png. The two image files are the icons for your app in Teams.
   * Name this package “DLLookupManifest.zip”.
   * Make sure that the 3 files are the top level of the ZIP package, with no nested folders.  
     

# Step 6: Run the apps in Microsoft Teams

* 1. Open Microsoft teams desktop app and click on “Apps” in left navigation.
  2. Click on “Upload custom app” which will pop up with two options:
* Upload for me or my teams
* Upload for tenant name (e.g:msteamspoc): Option will be displayed in case if you have permission. Otherwise login with admin account.
  1. Click on “Upload for tenant name (e.g:msteamspoc)” and browse the zipped package.
  2. Uploaded package will be shown under “Built for msteamspoc(tenant name)”. Click on “Distribution Lists Lookup” and click on “Add”. App will be shown.