

FAQ Multi-language

Deployment Guide for FAQPlus application

REVISION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Update** | **Updated By** |
| 0.1 | 05/09/2020 | Draft Version | Soumyajeet Swain |

Contents

[Overview 3](#_Toc47954568)

[Deployment Guide 3](#_Toc47954569)

[**Prerequisites** 3](#_Toc47954570)

[**Step 1: Register Azure AD applications** 3](#_Toc47954571)

[**Step 2: Deploy to your Azure subscription** 3](#_Toc47954572)

[**Step 3: Set up authentication for the configuration app** 3](#_Toc47954573)

[**Step 4: Create the QnA Maker knowledge base** 3](#_Toc47954574)

[**Multi-Turn Enablement** 3](#_Toc47954575)

[**Step 5: Finish configuring the FAQ Plus app** 3](#_Toc47954576)

[**Notes** 3](#_Toc47954577)

[**Step 6: Create the Teams app packages** 3](#_Toc47954578)

[**Step 7: Deployment steps for updating an existing application** 3](#_Toc47954579)

[**Step 8: Run the apps in Microsoft Teams** 3](#_Toc47954580)

Overview

We are looking to allow a team for a centralized way to collate all the KBs with multi-language support (ENG, SPA, POR). The intention is to provide the front-end layer to this as an app for all the users. The App should have an ability to

* Provide response to user query in the selected language
* Record feedback
* Ask an expert
* Show multi-language support

Deployment Guide

**Prerequisites**

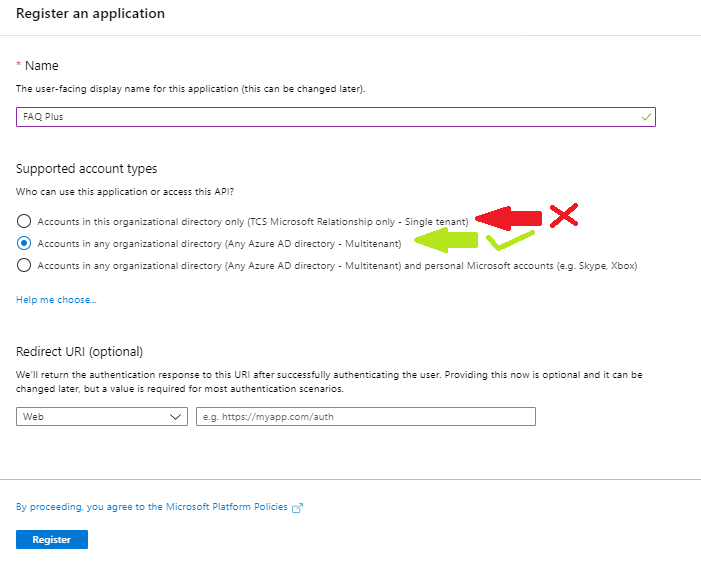
To begin, you will need:

* An Azure subscription where you can create the following kind of resources:
  + App service
  + App service plan
  + Bot channels registration
  + Azure storage account
  + Azure search
  + Azure function
  + QnAMaker cognitive service
  + Application Insights
* A team in Microsoft Teams with your group of experts. (You can add and remove team members later!)
* A copy of the FAQ Plus app GitHub repo (<https://github.com/OfficeDev/microsoft-teams-apps-faqplus>) for initial deployment of the bot
* Latest Code for FAQ+ bot, config app, Azure function app
* A reasonable set of Question and Answer pairs to set up the knowledge base for the bot.
* VisualStudio, .net core 2.1 (which is required by the app) and nodeJS > 10.x.

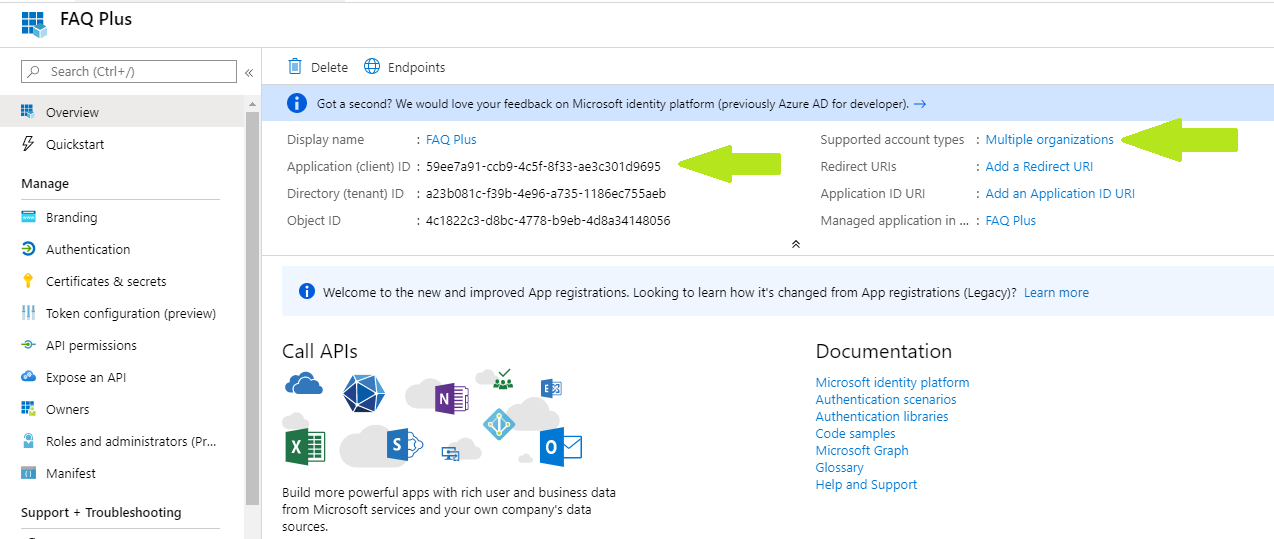
**Step 1: Register Azure AD applications**

Register two Azure AD applications in your tenant's directory: one for the bot, and another for the configuration app.

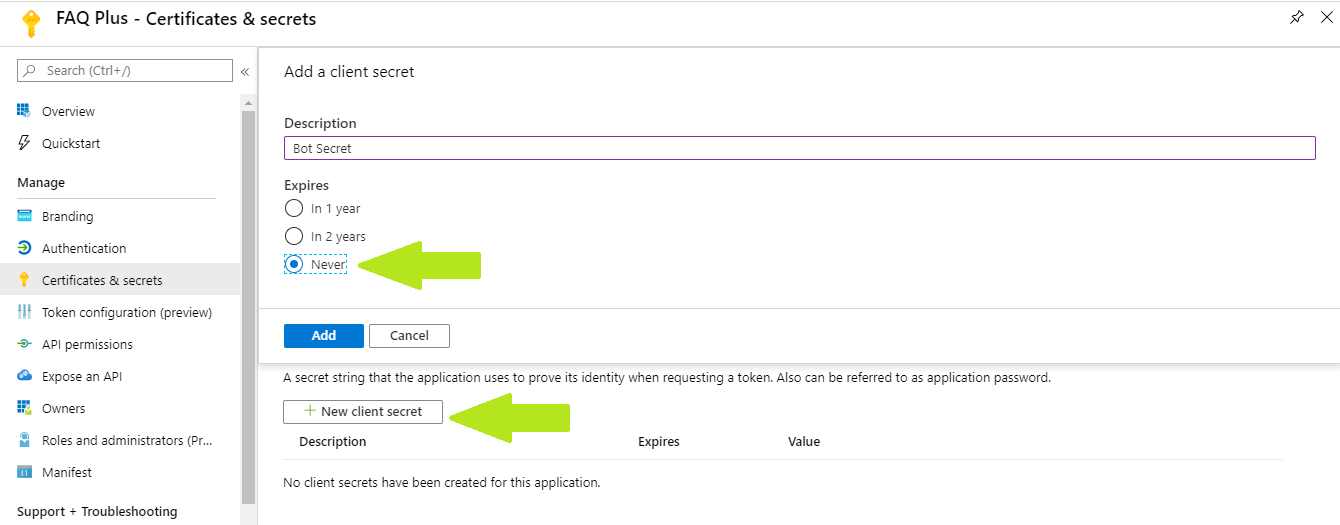
1. Log in to the Azure Portal for your subscription, and go to the "App registrations" blade [here](https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredApps).
2. Click on "New registration", and create an Azure AD application.
   1. **Name**: The name of your Teams app - if you are following the template for a default deployment, we recommend "FAQ Plus".
   2. **Supported account types**: Select "Accounts in any organizational directory"
   3. Leave the "Redirect URL" field blank.



1. Click on the "Register" button.
2. When the app is registered, you'll be taken to the app's "Overview" page. Copy the **Application (client) ID**; we will need it later. Verify that the "Supported account types" is set to **Multiple organizations**.



1. On the side rail in the Manage section, navigate to the "Certificates & secrets" section. In the Client secrets section, click on "+ New client secret". Add a description for the secret and select an expiry time. Click "Add".

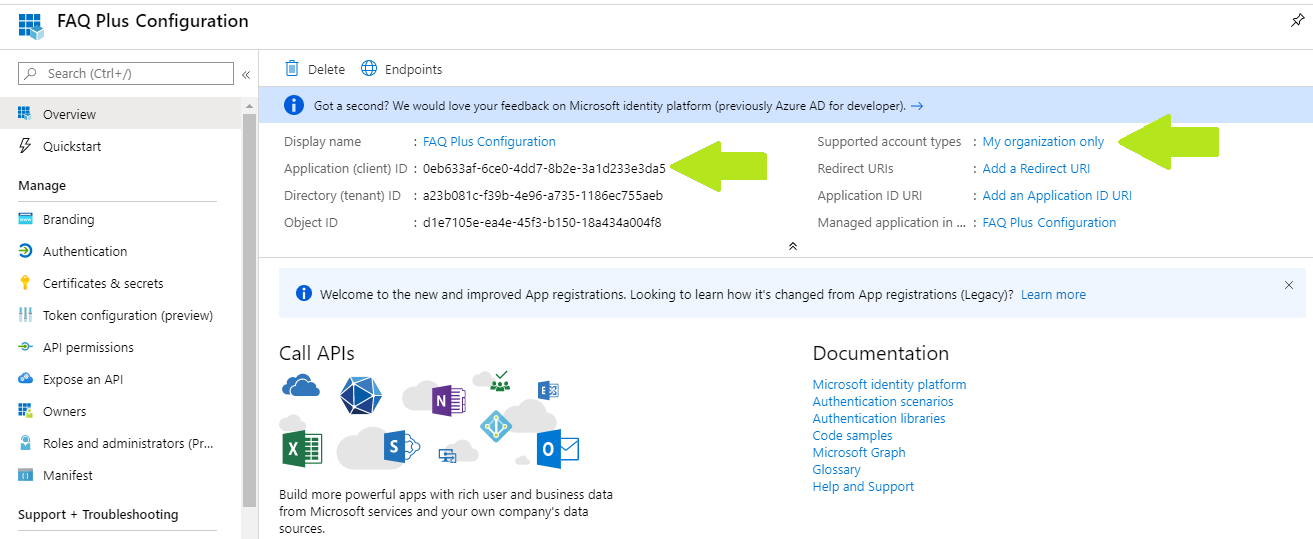


1. Once the client secret is created, copy its **Value**; we will need it later.
2. Go back to “App registrations”, then repeat steps 2-3 to create another Azure AD application for the configuration app.
   1. **Name**: The name of your configuration app. We advise appending “Configuration” to the name of this app; for example, “FAQ Plus Configuration”.
   2. **Supported account types**: Select "Account in this organizational directory only"
   3. Leave the "Redirect URL" field blank for now.

At this point you have 4 unique values:

* Application (client) ID for the bot
* Client secret for the bot
* Application (client) ID for the configuration app
* Directory (tenant) ID, which is the same for both apps

We recommend that you copy these values into a text file, using an application like Notepad. We will need these values later.



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

1. Click on the "Deploy to Azure" button below.

[Deploy to Azure](https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fraw.githubusercontent.com%2FOfficeDev%2Fmicrosoft-teams-apps-faqplus%2Fmaster%2FDeployment%2Fazuredeploy.json)

1. When prompted, log in to your Azure subscription.
2. Azure will create a "Custom deployment" based on the ARM template and ask you to fill in the template parameters.
3. Select a subscription and resource group.

* We recommend creating a new resource group.
* The resource group location MUST be in a datacenter that supports: Application Insights; Azure Search; and QnA Maker. For an up-to-date list, click [here](https://azure.microsoft.com/en-us/global-infrastructure/services/?products=logic-apps,cognitive-services,search,monitor), and select a region where the following services are available:
* Application Insights
* QnA Maker
* Azure Search

1. Enter a "Base Resource Name", which the template uses to generate names for the other resources.

* The app service names [Base Resource Name], and [Base Resource Name]-qnamaker must be available. For example, if you select contosofaqplus as the base name, the names contosofaqplus, and contosofaqplus-qnamaker must be available (not taken); otherwise, the deployment will fail with a Conflict error.
* Remember the base resource name that you selected. We will need it later.

1. Fill in the various IDs in the template:
   1. **Bot Client ID**: The application (client) ID of the Microsoft Teams Bot app
   2. **Bot Client Secret**: The client secret of the Microsoft Teams Bot app
   3. **Config App Client Id**: The application (client) ID of the configuration app
   4. **Tenant Id**: The tenant ID above

Make sure that the values are copied as-is, with no extra spaces. The template checks that GUIDs are exactly 36 characters.

1. Fill in the "Config Admin UPN List", which is a semicolon-delimited list of users who will be allowed to access the configuration app.

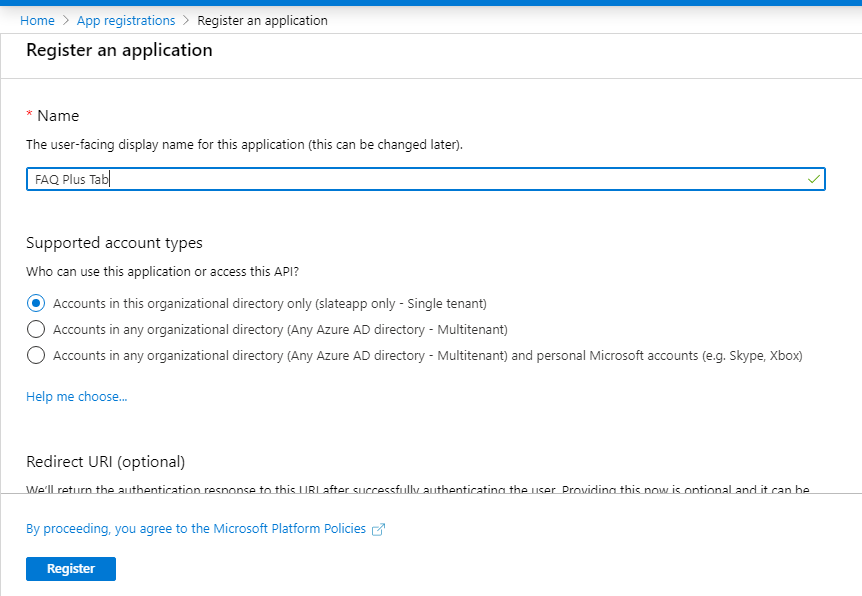
* For example, to allow Megan Bowen ([meganb@contoso.com](mailto:meganb@contoso.com)) and Adele Vance ([adelev@contoso.com](mailto:adelev@contoso.com)) to access the configuration app, set this parameter to meganb@contoso.com;adelv@contoso.com.
* You can change this list later by going to the configuration app service's "Configuration" blade.

1. If you wish to change the app name, description, and icon from the defaults, modify the corresponding template parameters.
2. Agree to the Azure terms and conditions by clicking on the check box "I agree to the terms and conditions stated above" located at the bottom of the page.
3. Click on "Purchase" to start the deployment.
4. Wait for the deployment to finish. You can check the progress of the deployment from the "Notifications" pane of the Azure Portal. It can take more than 10 minutes for the deployment to finish.
5. Once the deployment has finished, you would be directed to a page that has the following fields:

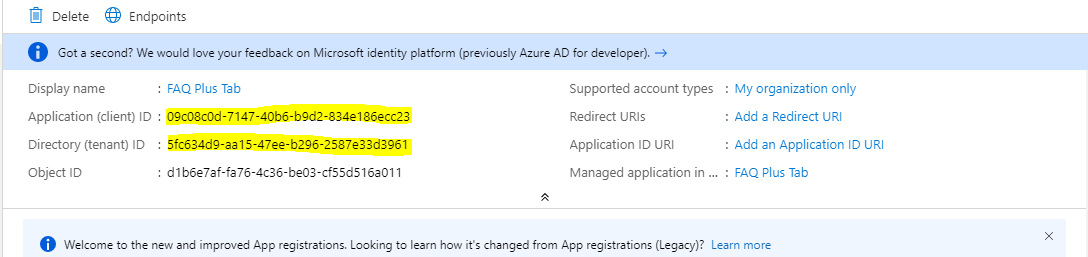
* botId - This is the Microsoft Application ID for the FAQ Plus bot.
* appDomain - This is the base domain for the FAQ Plus Bot.

**Step 3: Add App registration for Tab authentication.**

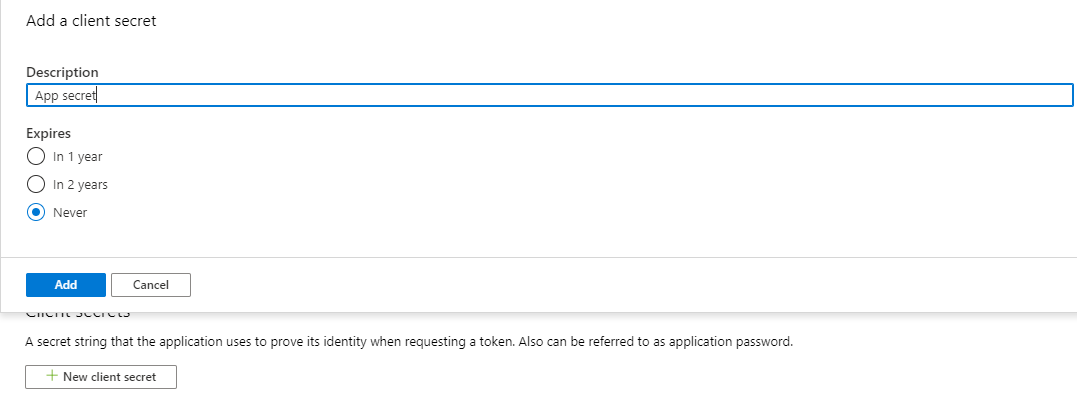
1. Log in to the Azure Portal for your subscription, and go to the "App registrations" blade [here](https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredApps).
2. Click on "New registration", and create an Azure AD application.
   1. **Name**: The name of your Teams app, we recommend "FAQ Plus Tab".
   2. **Supported account types**: Select " Accounts in this organizational directory only"
   3. Leave the "Redirect URL" field blank.



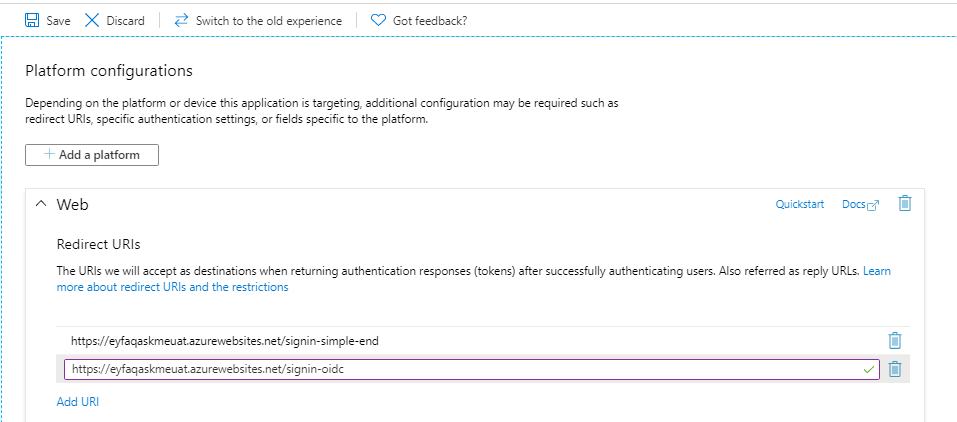
1. Click on the "Register" button.
2. When the app is registered, you'll be taken to the app's "Overview" page. Copy the Application (client) ID, Directory (tenant) ID; we will need it later. Verify that the "Supported account types" is set to “My organization only”.



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".



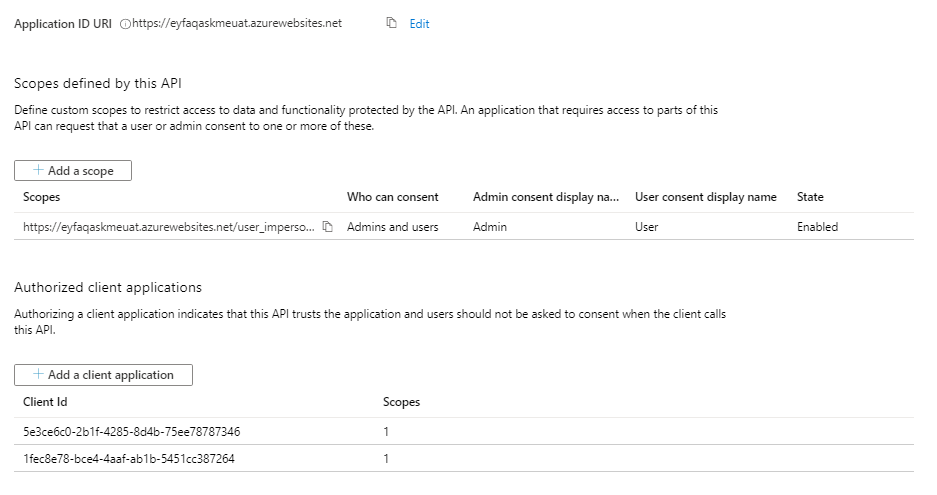
1. Once the client secret is created, copy it’s Value; we will need it later.
2. Under "Manage", click on "Authentication" to bring up authentication settings.
3. Click on “+Add Platform”, and then click on “Web”. Add two entries to "Redirect URIs":
4. Type: Web
5. Redirect URI: Enter "https://<<appDomain>>/signin-simple-end" and “https://<<appDomain>>/signin-oidc”for in Redirect URL field
6. Under "Implicit grant", check both “Access tokens” and "ID tokens".
7. Select “No” in Default client type.
8. 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>>
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. To allow application to work in Microsoft Teams Web client, follow below steps:
   * Click on "Add a client application", under "Authorized client applications". In the fly out that appears, enter the following values:
     1. Client ID: Provide the following client id : 5e3ce6c0-2b1f-4285-8d4b-75ee78787346
     2. Authorized scopes: Select the scope that ends with user\_impersonation. (There should only be 1 scope in this list.)
   * Click "Add application" to commit your changes.
3. To allow application to work in Microsoft Teams Desktop client, follow below steps:
   * Click on "Add a client application", under "Authorized client applications". In the fly out that appears, enter the following values:
     1. Client ID: Provide the following client id : 1fec8e78-bce4-4aaf-ab1b-5451cc387264
     2. Authorized scopes: Select the scope that ends with user\_impersonation. (There should only be 1 scope in this list.)
   * Click "Add application" to commit your changes.
4. 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 |

1. Click "Save" to commit your changes.

**Step 4: Set up authentication for the configuration app**

1. Note the location of the configuration app that you deployed, which is https://[BaseResourceName]-config.azurewebsites.net. For example, if you chose "contosofaqplus" as the base name, the configuration app will be at https://contosofaqplus.azurewebsites.net
2. Go back to the "App Registrations" page [here](https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredAppsPreview).
3. Click on the configuration app in the application list. Under "Manage", click on "Authentication" to bring up authentication settings.
4. Add new entries to "Redirect URLs":

* **Type**: Web
* **Redirect URLs**: If your configuration app's URL is

[https://contosofaqplus-config.azurewebsites.net](https://contosofaqplus.azurewebsites.net/), then add the following three entries as the Redirect URLs:

* + [https://contosofaqplus-config.azurewebsites.net](https://contosofaqplus.azurewebsites.net/)
  + [https://contosofaqplus-config.azurewebsites.net/signin](https://contosofaqplus.azurewebsites.net/signin)
  + [https://contosofaqplus-config.azurewebsites.net/configuration](https://contosofaqplus.azurewebsites.net/configuration)

Note: Please refer to Step 3.1 for more details about the URL.

1. Under "Implicit grant", check "Access tokens" and "ID tokens".
2. Click "Save" to commit your changes.

**Step 5: Create QnA Maker knowledge base one per language (English, Español, Portugués)**

Create knowledge base for each language, following the instructions in the [QnA Maker documentation](https://docs.microsoft.com/en-us/azure/cognitive-services/qnamaker/tutorials/create-publish-query-in-portal" \l "create-a-knowledge-base).

For English Knowledgebase, skip the step, "Create a QnA service in Microsoft Azure", because the ARM template that you deployed in Step 2 "Deploy to your Azure subscription" already created the QnA service. Proceed directly to the next step, "Connect your QnA service to your KB".

For Spanish and Portuguese knowledgebases, name the QnA service with “[-qnamaker-spanish](https://portal.azure.com/#@unnikrishnannairpoutlook.onmicrosoft.com/resource/subscriptions/ec605fd6-b601-4513-a8a4-50640902edfa/resourceGroups/FAQv2plusApp/providers/Microsoft.CognitiveServices/accounts/dev-faqplusv2-qnamaker-spanish)” and “[-qnamaker-portuguese](https://portal.azure.com/#@unnikrishnannairpoutlook.onmicrosoft.com/resource/subscriptions/ec605fd6-b601-4513-a8a4-50640902edfa/resourceGroups/FAQv2plusApp/providers/Microsoft.CognitiveServices/accounts/dev-faqplusv2-qnamaker-spanish)” suffix respectively to the base resource name of the app.

Use the following values when connecting to the QnA service:

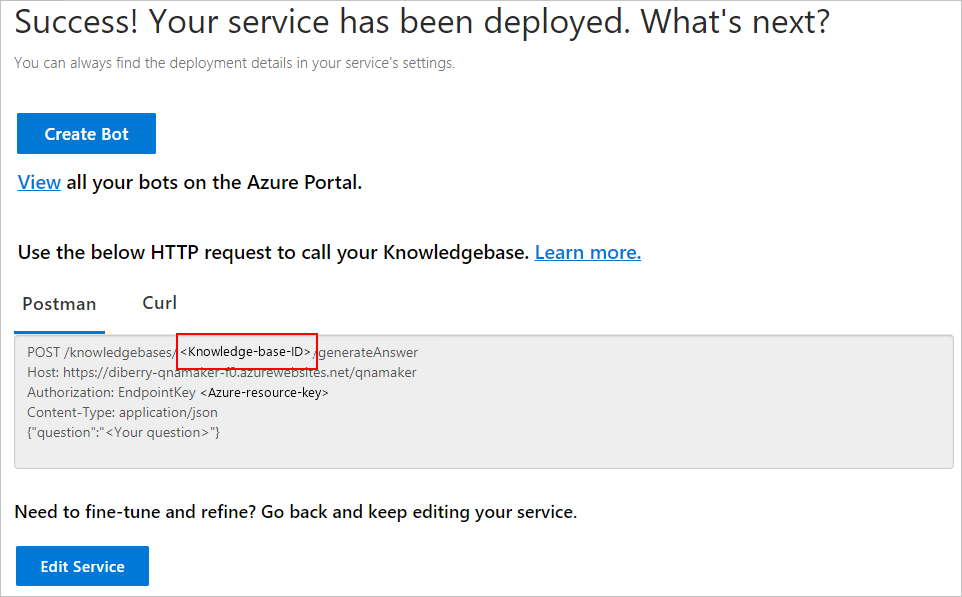
* **Microsoft Azure Directory ID**: The tenant associated with the Azure subscription selected in Step 2.1.
* **Azure subscription name**: The Azure subscription to which the ARM template was deployed.
* **Azure QnA service**: The QnA service created during the deployment. This is the same as the "Base resource name"; for example, if you chose "contosofaqplus" as the base name, the QnA Maker service will be named contosofaqplus.

**Multi-Turn Enablement**

With the new updates to the FAQ Plus app template, the knowledge base can now support multi-turn conversations. To understand the basics of multi-turn conversations, navigate to the [QnA Maker documentation](https://docs.microsoft.com/en-us/azure/cognitive-services/QnAMaker/how-to/multiturn-conversation" \l "what-is-a-multi-turn-conversation) to understand about multi-turn conversations. To enable multi-turn on the newly created knowledge base, go to this [link](https://docs.microsoft.com/en-us/azure/cognitive-services/QnAMaker/how-to/multiturn-conversation#create-a-multi-turn-conversation-from-a-documents-structure) to enable multi-turn extraction on the knowledge base.

* Note: For best practices with regards to formatting and document structure, please follow these [guidelines](https://docs.microsoft.com/en-us/azure/cognitive-services/QnAMaker/how-to/multiturn-conversation#building-your-own-multi-turn-document).

After [publishing the knowledge base](https://docs.microsoft.com/en-us/azure/cognitive-services/qnamaker/tutorials/create-publish-query-in-portal#publish-to-get-knowledge-base-endpoints), note the knowledge base ID (see screenshot).



Remember the knowledge base ID: we will need it in the next step.

**Step 6: Deployment steps for updating existing application**

Prerequisites:

1. Application is installed in the tenant using the ARM template mentioned in Catalog.
2. Access to resource group with application.
3. Visual Studio to publish the code.
4. Please navigate to app service with **[Base Resource Name]** as name, on the left blade click on Configuration and check whether all these values are present.

|  |
| --- |
| [  {  "name": "AccessCacheExpiryInDays",  "value": "5",  "slotSetting": false  },  {  "name": "AppBaseUri",  "value": " https://<< appdomain >>",  "slotSetting": false  },  {  "name": "APPINSIGHTS\_INSTRUMENTATIONKEY",  "value": "",  "slotSetting": false  },  {  "name": "ApplicationInsightsLogLevel",  "value": "Information",  "slotSetting": false  },  {  "name": "AzureAd:ApplicationIdURI",  "value": " https://<< appdomain >>",  "slotSetting": false  },  {  "name": "AzureAd:ClientId",  "value": "<< tabclientid >>",  "slotSetting": false  },  {  "name": "AzureAd:ClientSecret",  "value": "<< tabclientsecret >>",  "slotSetting": false  },  {  "name": "AzureAd:GraphScope",  "value": "https://graph.microsoft.com/User.Read.All openid profile",  "slotSetting": false  },  {  "name": "AzureAd:Instance",  "value": "https://login.microsoftonline.com",  "slotSetting": false  },  {  "name": "AzureAd:TenantId",  "value": "<<Tenantid>>",  "slotSetting": false  },  {  "name": "AzureAd:ValidIssuers",  "value": "",  "slotSetting": false  },  {  "name": "DIAGNOSTICS\_AZUREBLOBRETENTIONINDAYS",  "value": "1",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:0:LanguageCode",  "value": "en",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:0:QnAMakerHostUrl",  "value": "",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:0:QnAMakerSubscriptionKey",  "value": "",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:1:LanguageCode",  "value": "es",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:1:QnAMakerHostUrl",  "value": "",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:1:QnAMakerSubscriptionKey",  "value": "",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:2:LanguageCode",  "value": "pt",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:2:QnAMakerHostUrl",  "value": "",  "slotSetting": false  },  {  "name": "LanguageQnAMakerSubscriptionKeyJson:2:QnAMakerSubscriptionKey",  "value": "",  "slotSetting": false  },  {  "name": "ManifestAppId",  "value": "<<TabClientid>>",",  "slotSetting": false  },  {  "name": "MicrosoftAppId",  "value": "<<botid>>",  "slotSetting": false  },  {  "name": "MicrosoftAppPassword",  "value": "<<bot secret>>",”,  "slotSetting": false  },  {  "name": "QnAMakerApiEndpointUrl",  "value": "",  "slotSetting": false  },  {  "name": "QnAMakerHostUrl",  "value": "",  "slotSetting": false  },  {  "name": "QnAMakerSubscriptionKey",  "value": "",  "slotSetting": false  },  {  "name": "ScoreThreshold",  "value": "0.5",  "slotSetting": false  },  {  "name": "SearchIndexingIntervalInMinutes",  "value": "10",  "slotSetting": false  },  {  "name": "SearchServiceAdminApiKey",  "value": "",  "slotSetting": false  },  {  "name": "SearchServiceName",  "value": "",  "slotSetting": false  },  {  "name": "SearchServiceQueryApiKey",  "value": "",  "slotSetting": false  },  {  "name": "SITE\_ROLE",  "value": "bot",  "slotSetting": false  },  {  "name": "StorageConnectionString",  "value": "",  "slotSetting": false  },  {  "name": "TenantId",  "value": "<<Tenantid>>",",  "slotSetting": false  },  {  "name": "WEBSITE\_HTTPLOGGING\_RETENTION\_DAYS",  "value": "1",  "slotSetting": false  }  ] |

1. Replace Tenant id, appdomain, tab client id and tab client secret in the configuration with the values from Azure AD applications above.
2. If QnAMakerApiHostUrl/ LanguageQnAMakerSubscriptionKeyJson:0:QnAMakerSubscriptionKey is missing, please add them,
   1. Go to cognitive service which is created in the respective resource group with **[Base Resource Name]**. It will correspond to the default language (English).
   2. Navigate to ‘**Keys and Endpoint**’ under **Resource Management** in the left blade.
   3. Copy the values under **Key 1** to QnAMakerSubscriptionKey Graphical user interface, application

      Description automatically generated
   4. And for the QnAMakerApiEndpointUrl you need to get the APP Service for each language HOST URL.

Graphical user interface, text, application, email

Description automatically generated

1. If LanguageQnAMakerSubscriptionKeyJson:1:QnAMakerSubscriptionKey is missing, follow the above steps for Spanish language Qna maker cognitive service.
2. If LanguageQnAMakerSubscriptionKeyJson:2:QnAMakerSubscriptionKey is missing, follow the above steps for Portuguese language Qna maker cognitive service.
3. Please navigate to app service with **[Base Resource Name]-config** as name, on the left blade click on Configuration and add the following Name and value if missing –

Name - LanguageQnAMakerSubscriptionKeyJson

Value –

|  |
| --- |
| [{"LanguageCode":"en","LanguageName":"English","Default":true,"QnAMakerSubscriptionKey":"<<ENSubscriptionKey>>"},{"LanguageCode":"es","LanguageName":"Español","QnAMakerSubscriptionKey":"<<ESSubscriptionKey>>"},{"LanguageCode":"pt","LanguageName":"Portugués","QnAMakerSubscriptionKey":"<<PTSubscriptionKey>>"}] |

1. Please navigate to Azure Function app with **[Base Resource Name]-function** as name, on the left blade click on Configuration and add the following key and value if missing –

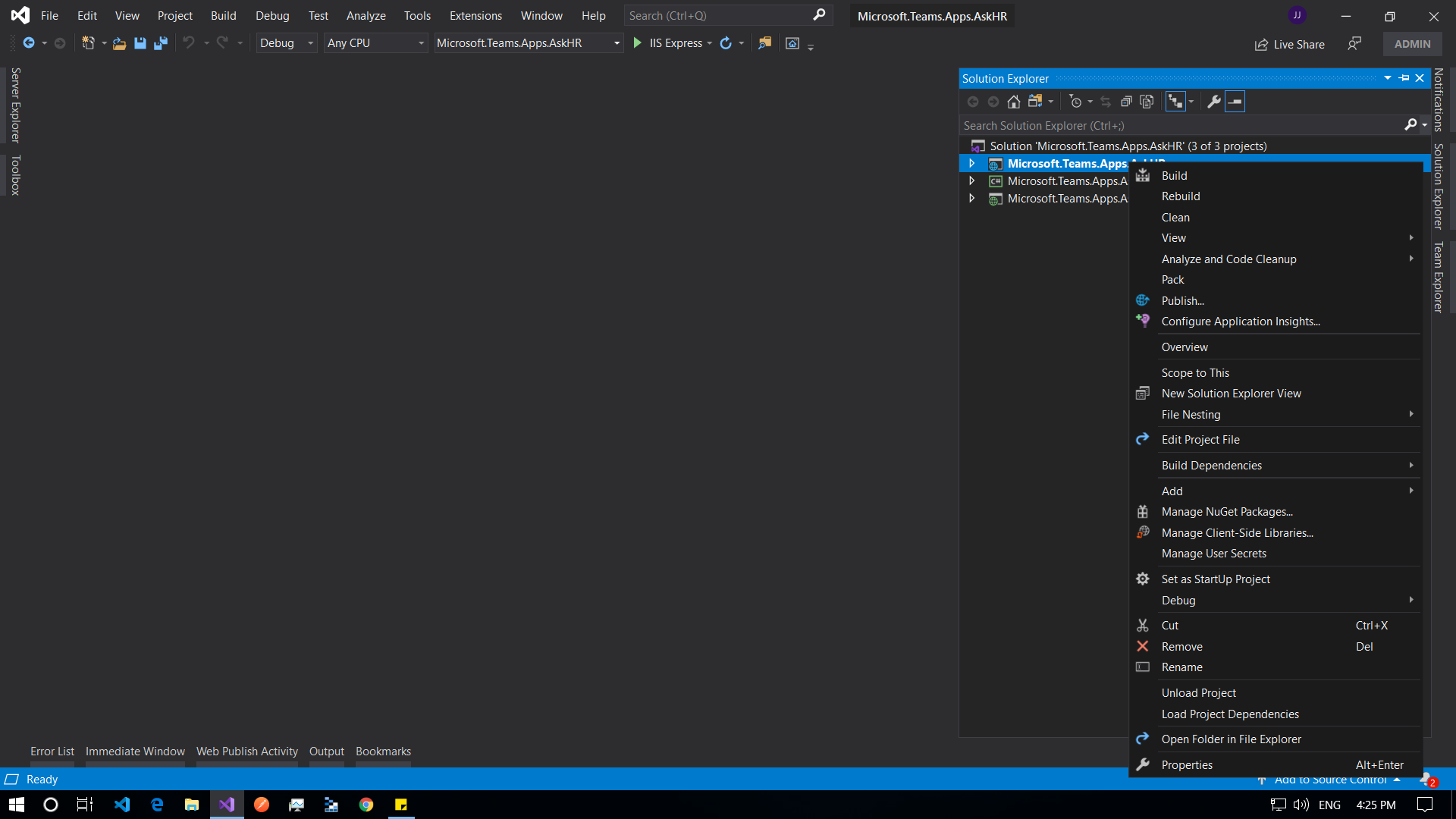
Name – LanguageQnAMakerSubscriptionKeyJson

Value –

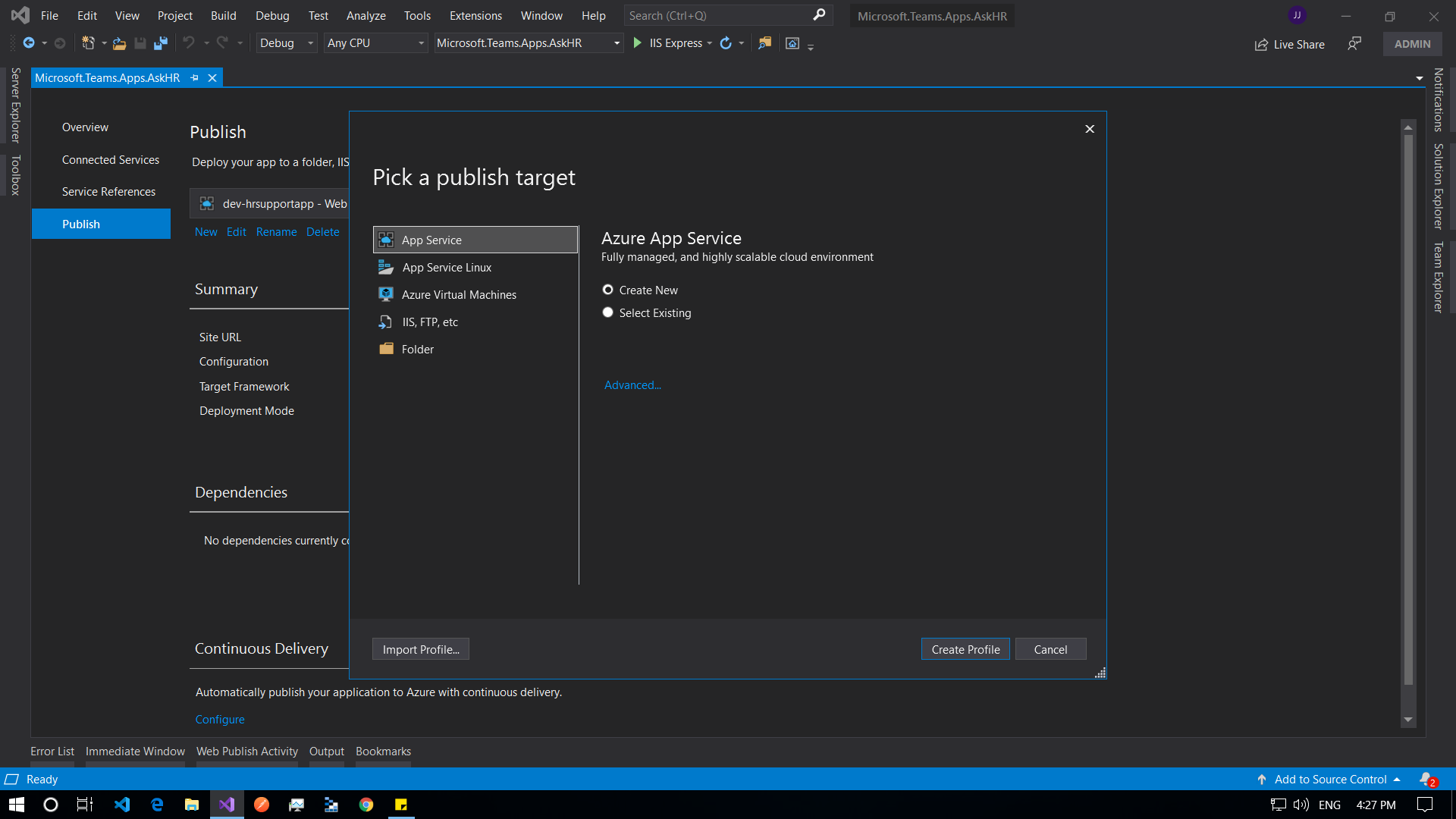
|  |
| --- |
| [{"LanguageCode":"en","LanguageName":"English","Default":true,"QnAMakerSubscriptionKey":"<<ENSubscriptionKey>>"},{"LanguageCode":"es","LanguageName":"Español","QnAMakerSubscriptionKey":"<<ESSubscriptionKey>>"},{"LanguageCode":"pt","LanguageName":"Portugués","QnAMakerSubscriptionKey":"<<PTSubscriptionKey>>"}] |

### Deploying bot changes:

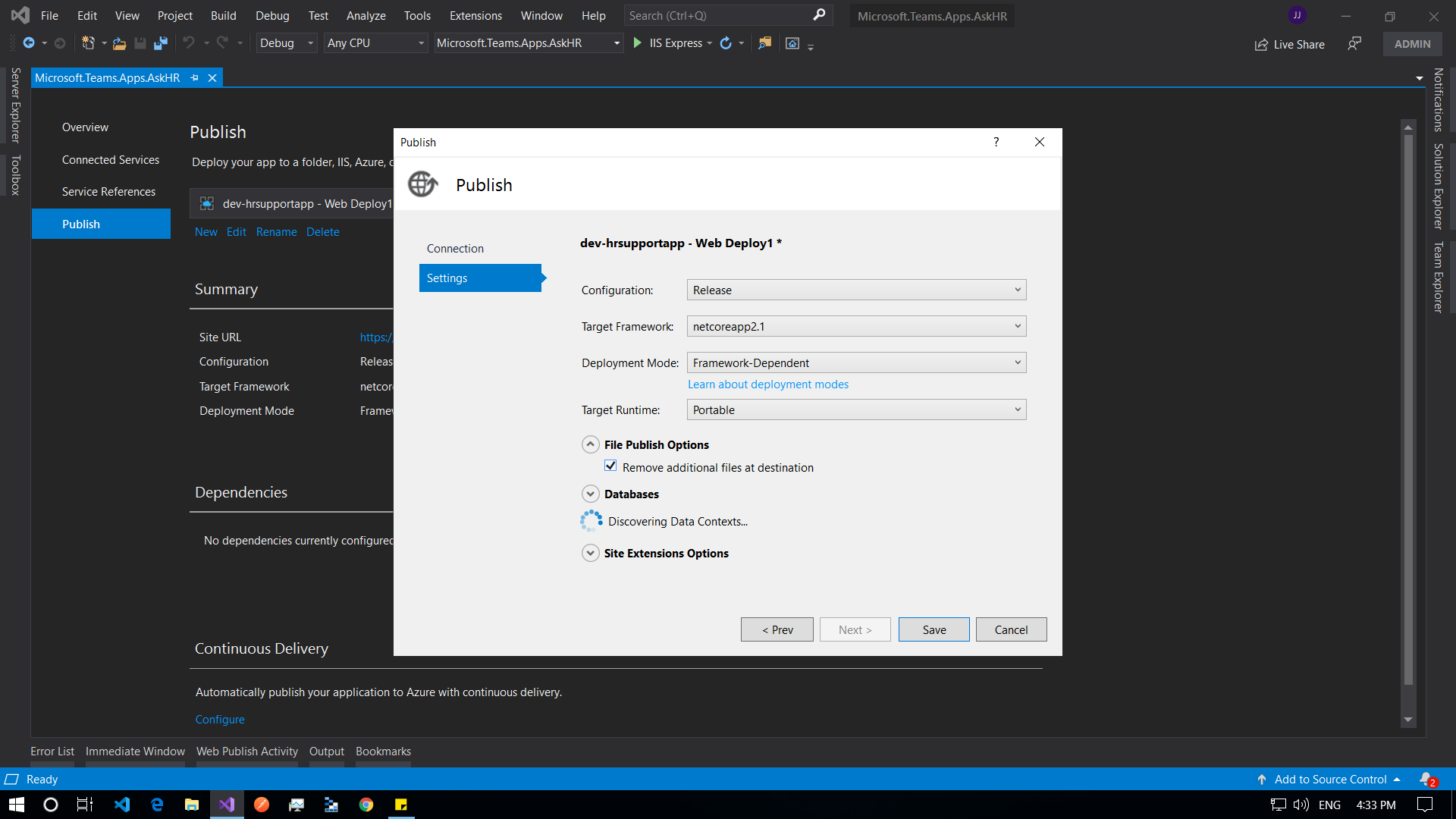
1. Go to [azure portal](https://www.portal.azure.com/).
2. Navigate to the resource group.
3. Click on the –type App Service with **[Base Resource Name]** as name.
4. Click on ‘**Get publish profile**’ and wait until the profile gets downloaded.
5. Open the updated solution.
6. Right click on ‘**Microsoft.Teams.Apps.** **FAQPlusPlus**’ in the solution explorer.



1. Click on ‘**Publish’**.
2. Click on ‘**New**’. Choose ‘**Import Profile**’ from left bottom of the dialog box. Select the publish profile downloaded earlier in Step 4.



1. Click on ‘**Edit**’. Select ‘**Settings**’ from the side menu in the dialog. Under ‘**File Publish Options**’ check ‘**Remove additional files as destination**’ on.



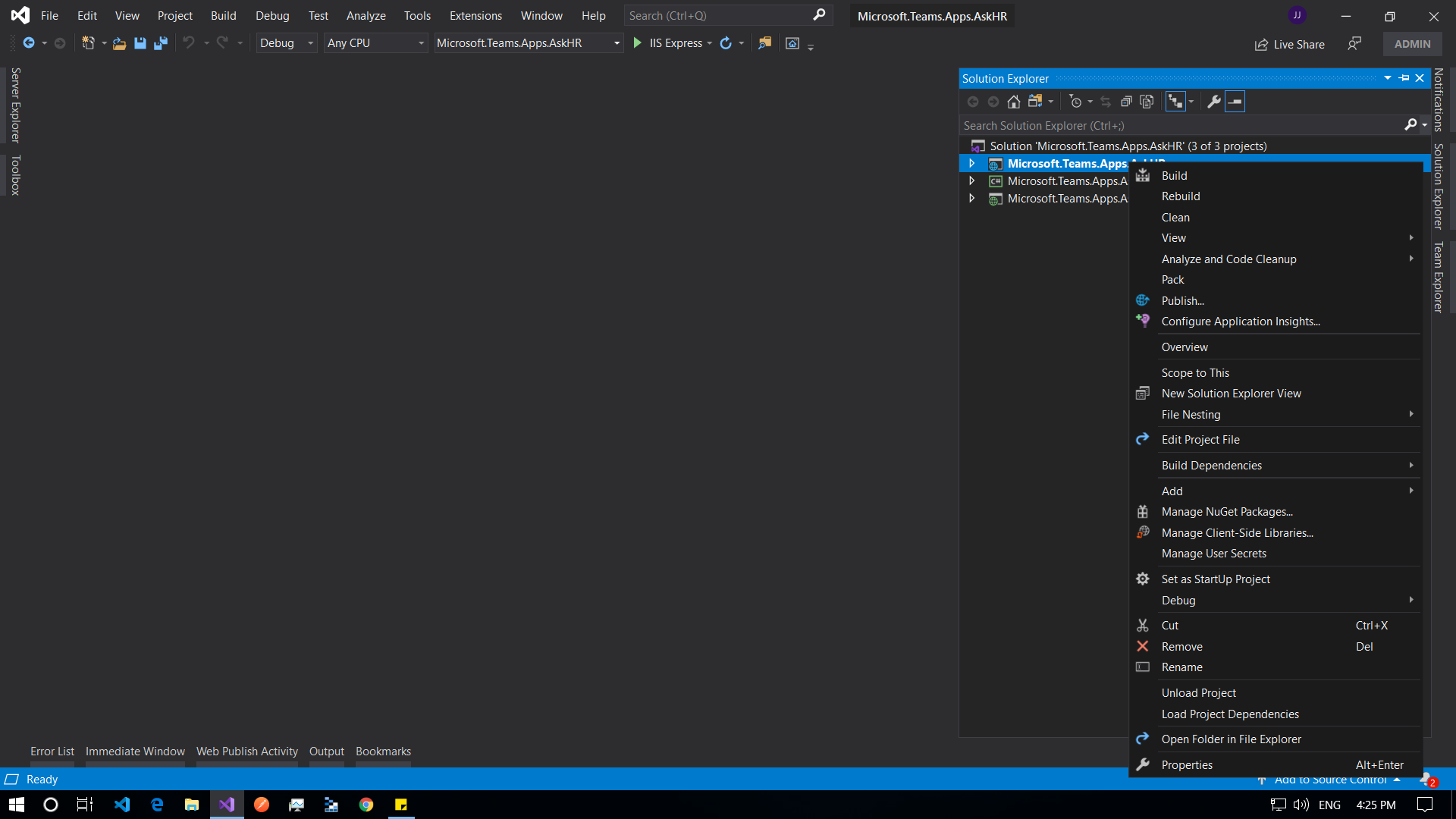
Click on ‘**Save**’.

1. Click on ‘**Publish**’ and wait for the process to get over. Check the application for verify the changes.

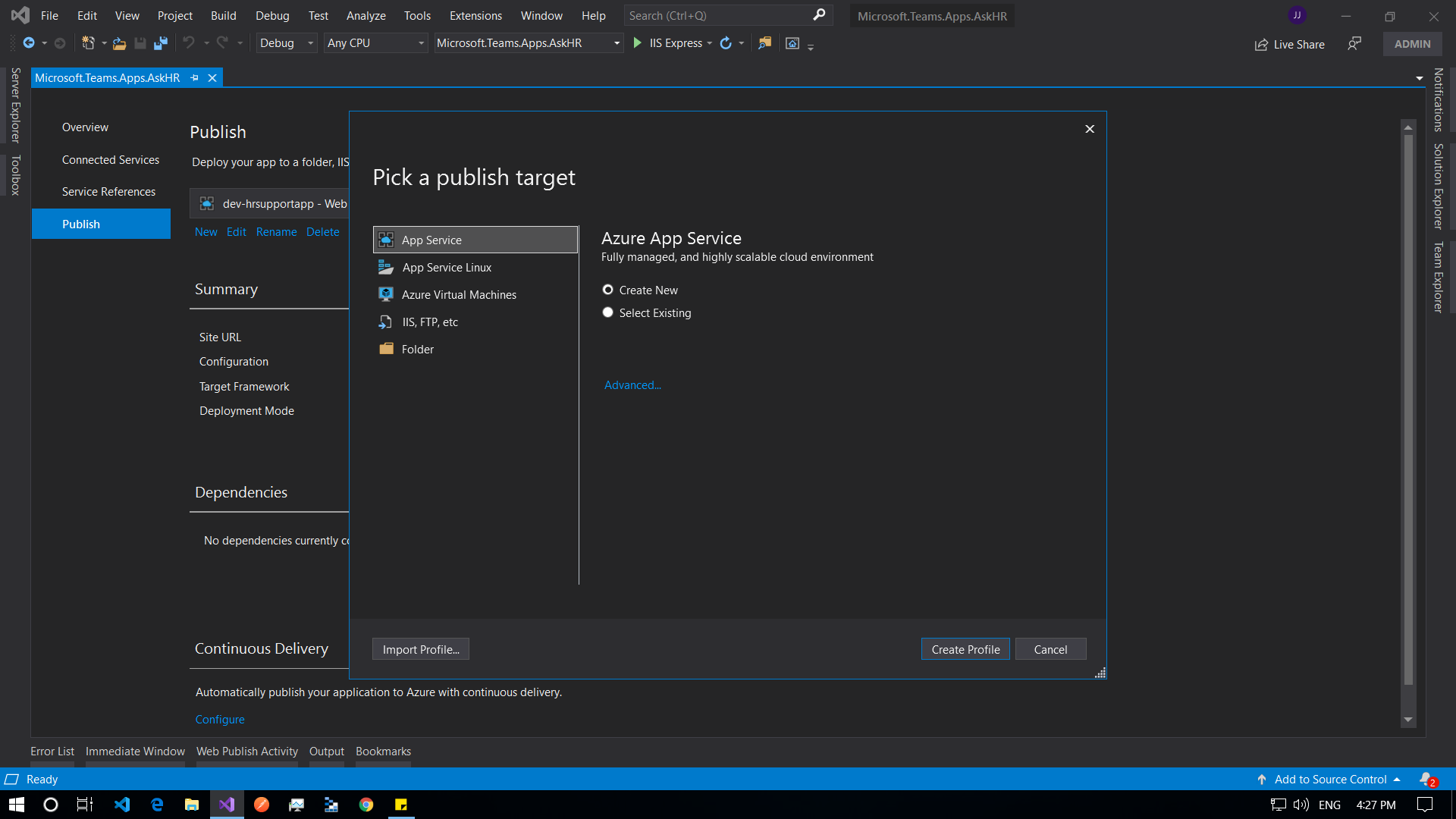
### Deploying config app changes:

Following are the steps to deploy config app changes –

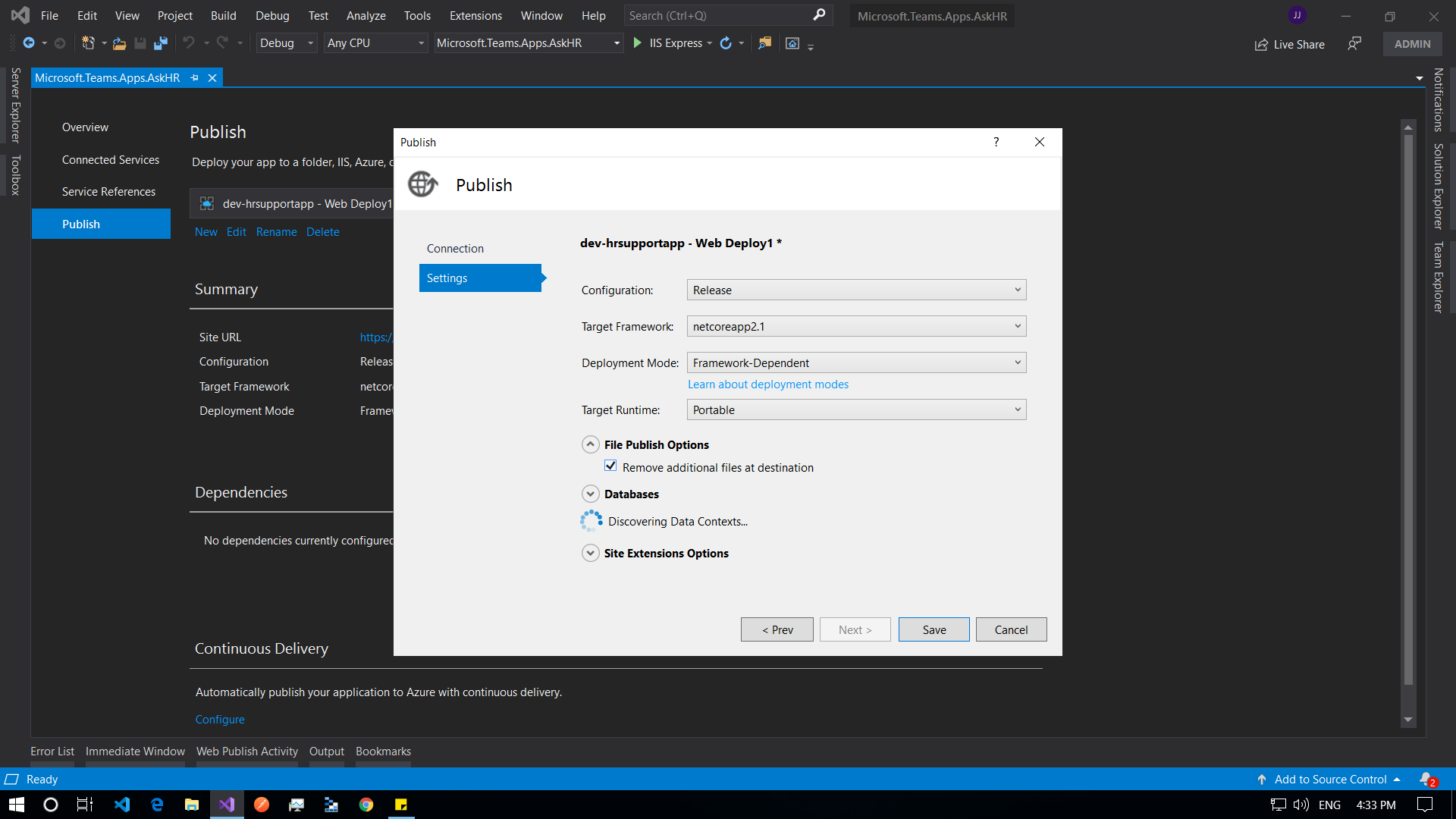
1. Go to [azure portal](https://www.portal.azure.com/).
2. Navigate to the resource group.
3. Click on the –type App Service with **[Base Resource Name]-config** as name.
4. Click on ‘**Get publish profile**’ and wait until the profile gets downloaded.
5. Open the updated solution.
6. Right click on ‘**Microsoft.Teams.Apps.** **FAQPlusPlus.Configuration**’ in the solution explorer.



1. Click on ‘**Publish’**.
2. Click on ‘**New**’. Choose ‘**Import Profile**’ from left bottom of the dialog box. Select the publish profile downloaded earlier in Step 4.



1. Click on ‘**Edit**’. Select ‘**Settings**’ from the side menu in the dialog. Under ‘**File Publish Options**’ check ‘**Remove additional files as destination**’ on.



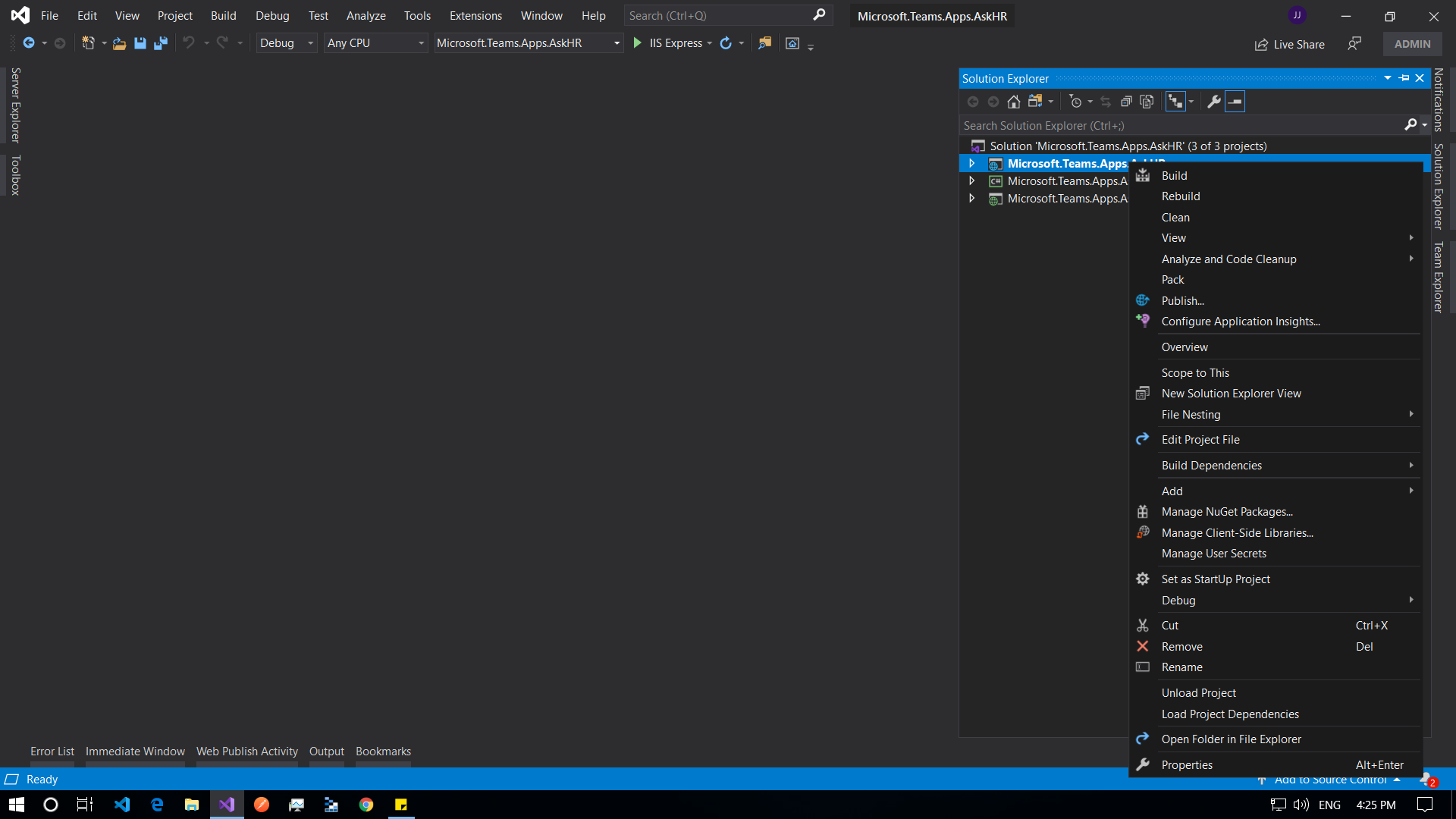
Click on ‘**Save**’.

1. Click on ‘**Publish**’ and wait for the process to get over. Check the application for verify the changes.

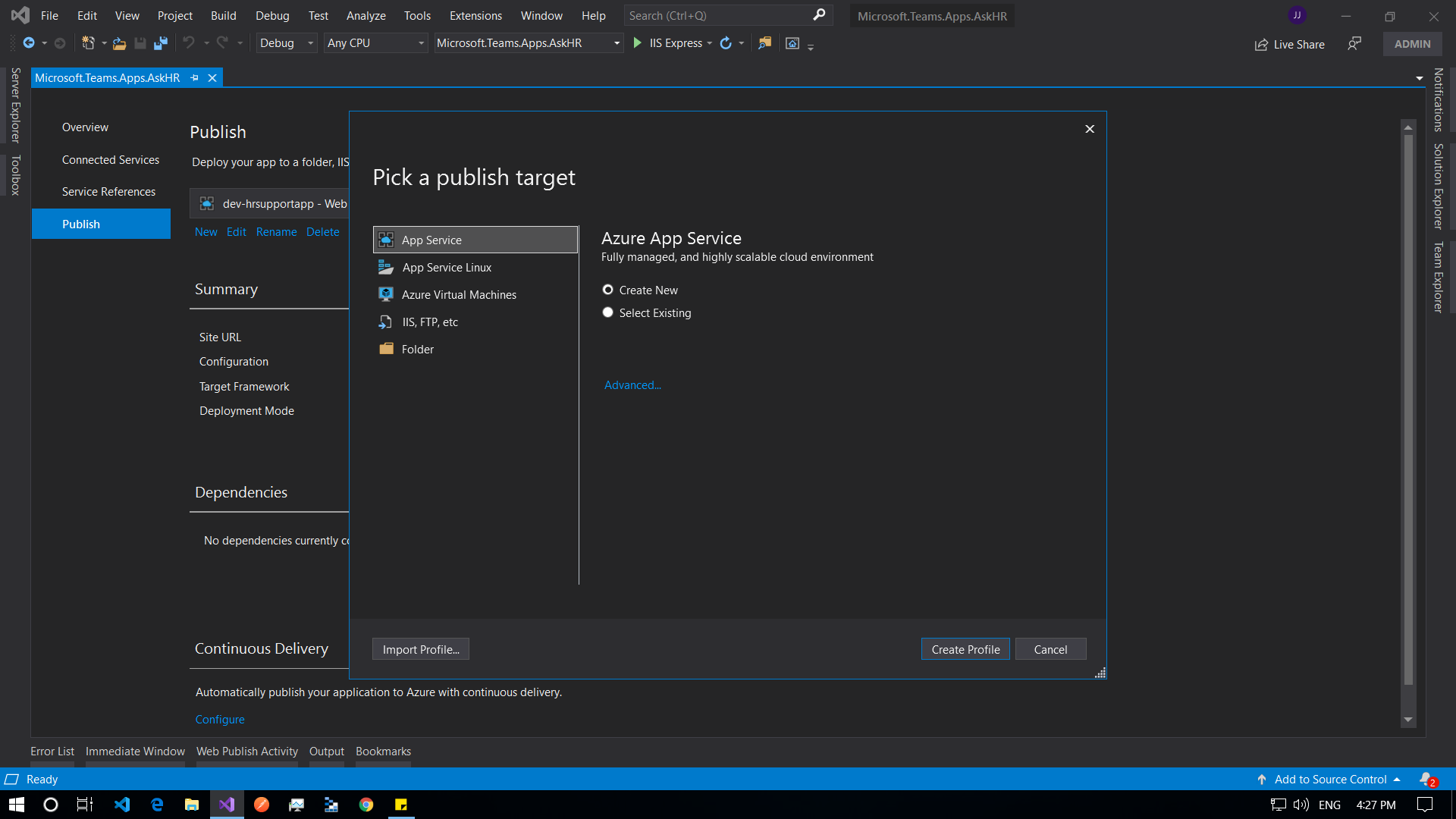
### Deploying azure function changes:

Following are the steps to deploy the Azure function changes –

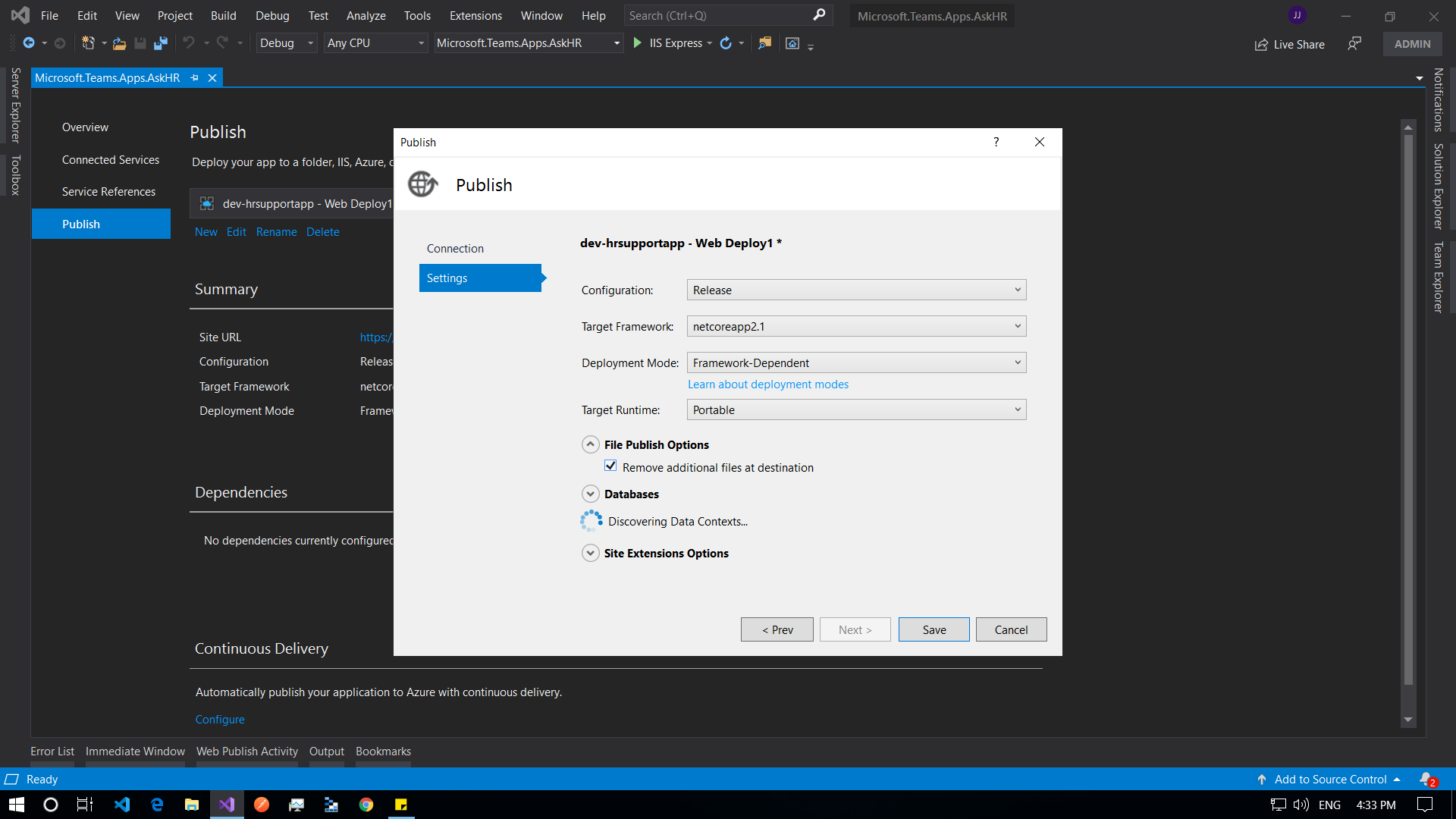
1. Go to [azure portal](https://www.portal.azure.com/).
2. Navigate to the resource group.
3. Click on the –type Function App with **[Base Resource Name]-function** as name.
4. Click on ‘**Get publish profile**’ and wait until the profile gets downloaded.
5. Open the updated solution.
6. Right click on ‘**Microsoft.Teams.Apps.** **FAQPlusPlus.AzureFunction**’ in the solution explorer.



1. Click on ‘**Publish’**.
2. Click on ‘**New**’. Choose ‘**Import Profile**’ from left bottom of the dialog box. Select the publish profile downloaded earlier in Step 4.



1. Click on ‘**Edit**’. Select ‘**Settings**’ from the side menu in the dialog. Under ‘**File Publish Options**’ check ‘**Remove additional files as destination**’ on.



Click on ‘**Save**’.

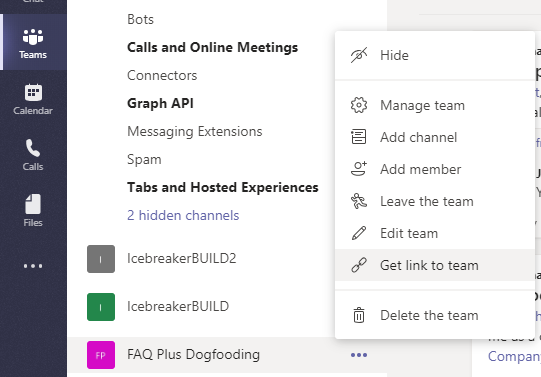
1. Click on ‘**Publish**’ and wait for the process to get over. Check the application for verify the changes.

**Step 7: Finish configuring the FAQ Plus app**

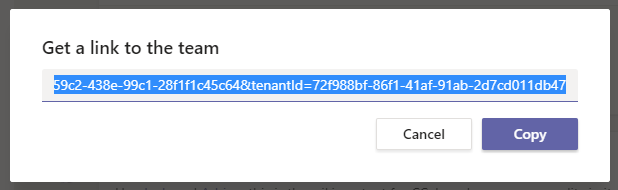
1. Go to the configuration app, which is at

https://[BaseResourceName]-config.azurewebsites.net/configuration.

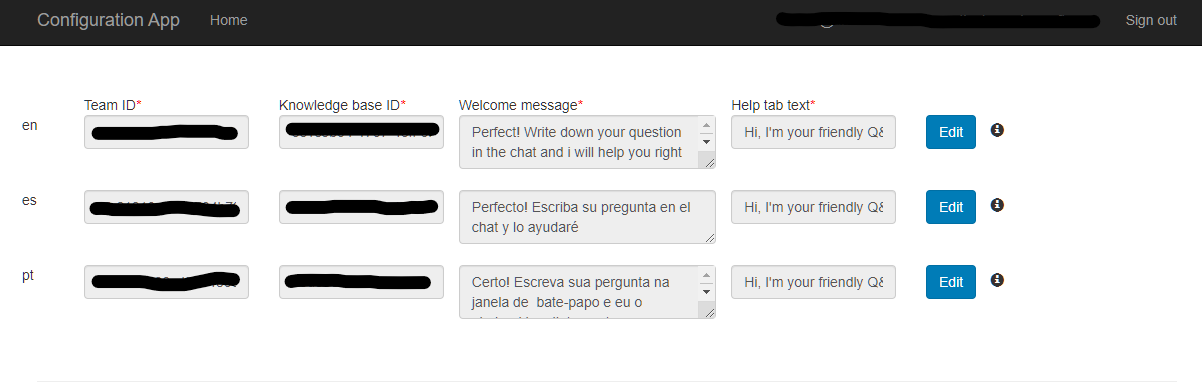
1. For example, if you chose “contosofaqplus” as the base name, the configuration app will be at https://contosofaqplus-config.azurewebsites.net/configuration.
2. You will be prompted to login with your credentials. Make sure that you log in with an account that is in the list of users allowed to access the configuration app.
3. Get the link to the team with your experts from the Teams client. To do so, open Microsoft Teams, and navigate to the team. Click on the "..." next to the team name, then select "Get link to team".



Cick on "Copy" to copy the link to the clipboard.



1. Paste the copied link into the "Team Id" field, then press "OK".



1. Enter the QnA Maker knowledge base ID into the "Knowledge base ID" field, then press "OK".
2. Customize the "Welcome message" that's sent to your End users when they install the app. This message supports basic markdown, such as bold, italics, bulleted lists, numbered lists, and hyperlinks. See [here](https://docs.microsoft.com/en-us/adaptive-cards/authoring-cards/text-features#markdown) for complete details on what Markdown features are supported.
3. Repeat these steps for each language supported by the bot.

**Notes**

Remember to click on "OK" after changing settings for a language. To edit the setting later, click on "Edit" to make the text box editable.

**Step 8: Create the Teams app packages**

Create two Teams app packages: one for end-users to install personally, and one to be installed to the experts team.

1. Open the Manifest\manifest\_enduser.json file in a text editor.
2. Change the placeholder fields in the manifest to values appropriate for your organization.

* developer.name ([What's this?](https://docs.microsoft.com/en-us/microsoftteams/platform/resources/schema/manifest-schema#developer))
* developer.websiteUrl
* developer.privacyUrl
* developer.termsOfUseUrl

1. Change the <<botId>> placeholder to your Azure AD application's ID from above. This is the same GUID that you entered in the template under "Bot Client ID".
2. Change the <<tabClientId>> placeholder to your Azure AD application's ID which was created for your tab from above.
3. In the "validDomains" section, replace the <<appDomain>> with your Bot App Service's domain. This will be [BaseResourceName].azurewebsites.net. For example if you chose "contosofaqplus" as the base name, change the placeholder to contosofaqplus.azurewebsites.net.
4. Save and Rename manifest\_enduser.json file to a file named manifest.json.
5. 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 faqplus-enduser.zip, so you know that this is the app for end-users.
* Make sure that the 3 files are the *top level* of the ZIP package, with no nested folders.

1. Delete the manifest.json file.

Repeat the steps above but with the file Manifest\manifest\_sme.json. Name the resulting package faqplus-experts.zip, so you know that this is the app for experts.

**Step 9: Run the apps in Microsoft Teams**

1. If your tenant has sideloading apps enabled, you can install your app by following the instructions [here](https://docs.microsoft.com/en-us/microsoftteams/platform/concepts/apps/apps-upload#load-your-package-into-teams)
2. You can also upload it to your tenant's app catalog, so that it can be available for everyone in your tenant to install. See [here](https://docs.microsoft.com/en-us/microsoftteams/tenant-apps-catalog-teams)
3. Install the experts app (the faqplus-experts.zip package) to your team of subject-matter experts. This **MUST** be the same team that you selected in Step 5.3 above.

* We recommend using [app permission policies](https://docs.microsoft.com/en-us/microsoftteams/teams-app-permission-policies) to restrict access to this app to the members of the experts team.

1. Install the end-user app (the faqplus-enduser.zip package) to your users.

Deployment steps for updating existing application

Please follow ***Step 6: Deployment steps for updating existing application*** to update existing application after doing code changes.