How do we get our knowledge base set up? How do we create it, for us to have a QnA bot to answer users’ answers we need to have a knowledge base where we will store our questions and answers at. How do we get our questions and answers from SharePoint to the qnamaker.ai knowledge base?

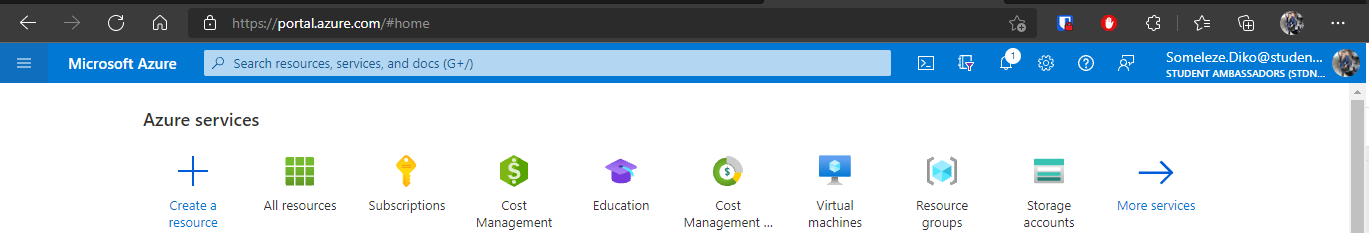
We can solve these questions by building a simple flow that will allow us to update our qnamaker.ai knowledge base with the latest questions and answers. Let’s build it!

**Note:** This assumes that you already have a SharePoint list built with the columns you will need (mostly Question column and Answers column).

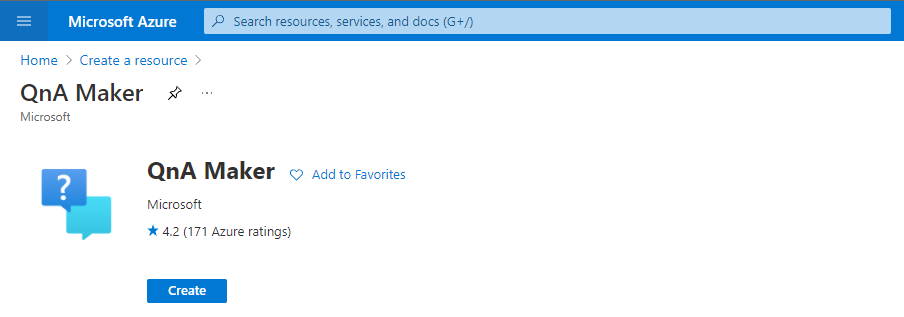
First things first…..Let’s build a flow that will allow us to update/add information on our knowledge base  
  
**QnA Maker App Service**

In order for you to be able to create/build your knowledge base you need a **QnA Maker App Service** from **Azure.  
  
Step 1:** Sign into **Azure** and click on **Create Resource** then look for **QnA Maker App** Service

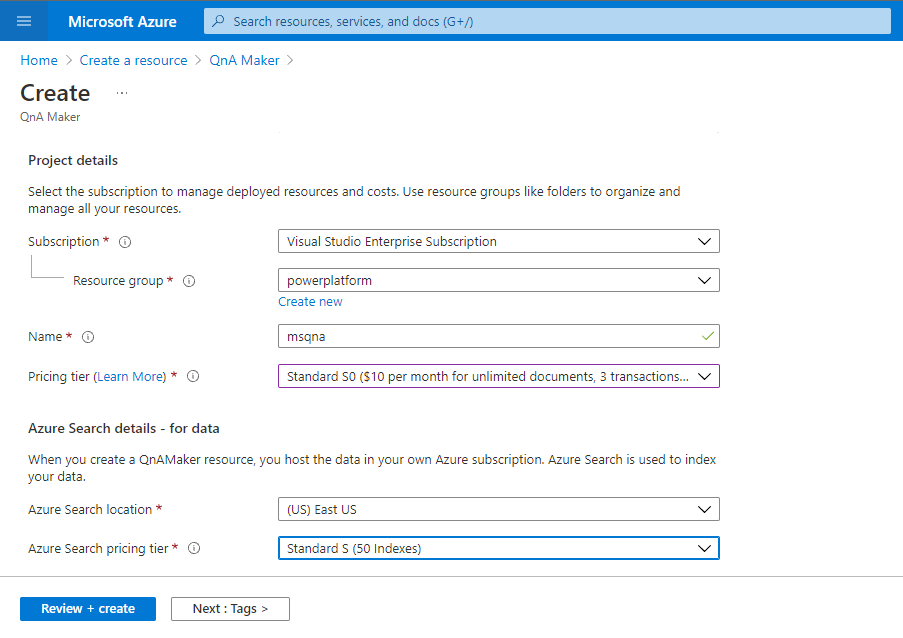
**Step 2:** Create a Resource



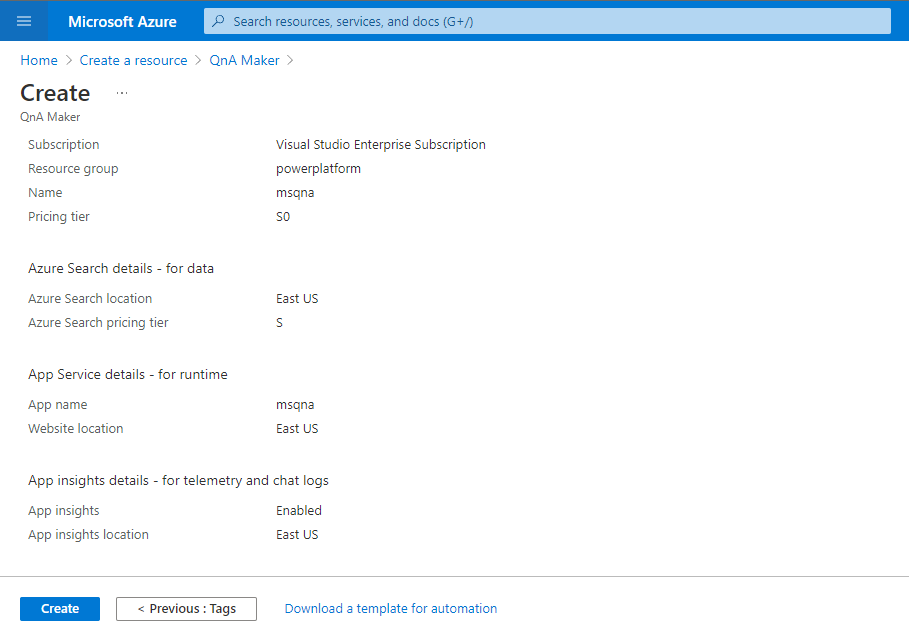
**Step 3:** Click on **Create** after you have found the QnaMaker App service.



**Step 4:** Fill in the necessary information about your QnA Maker App service based on your requirements, click on **Review + Create** once done.

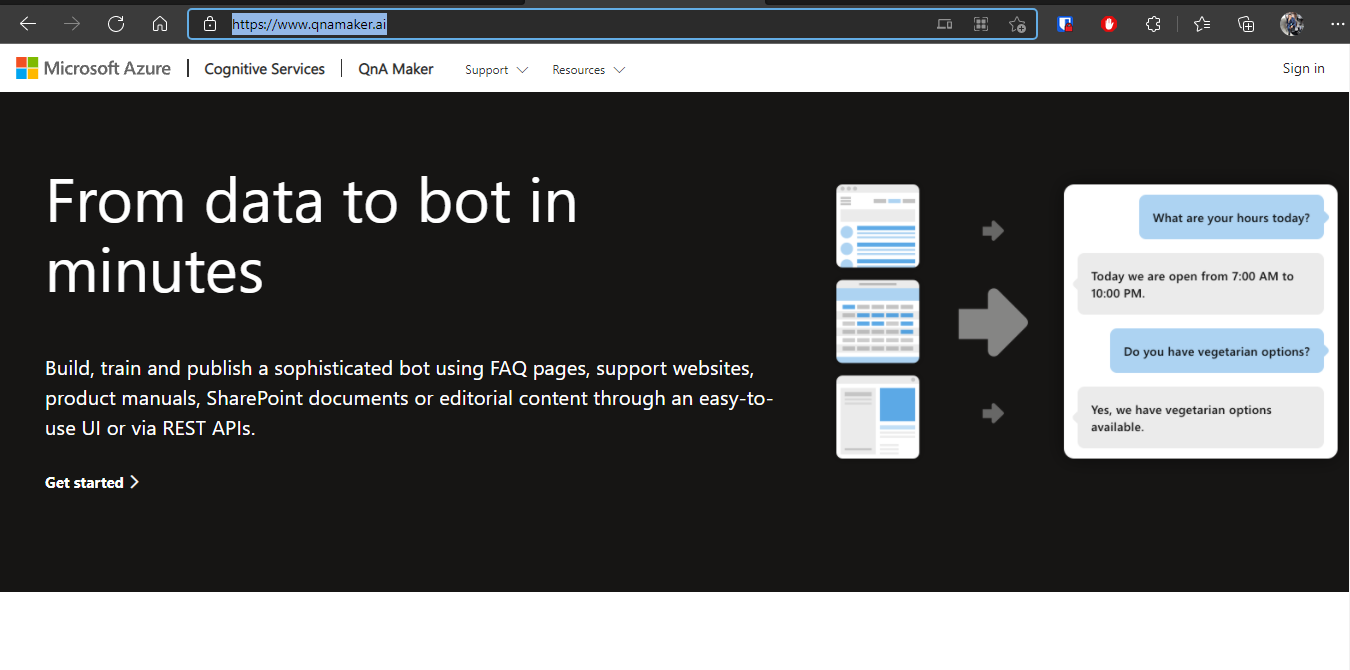


**Step 5:** Click on **Create** and your app service will be deployed

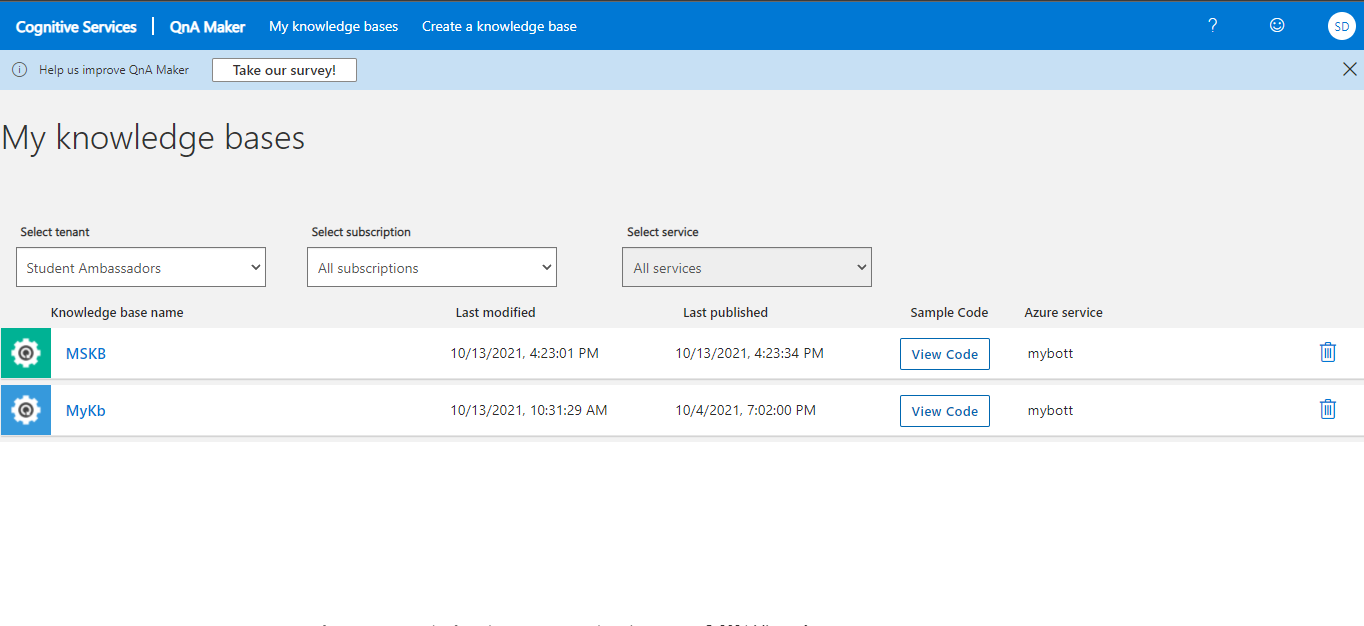


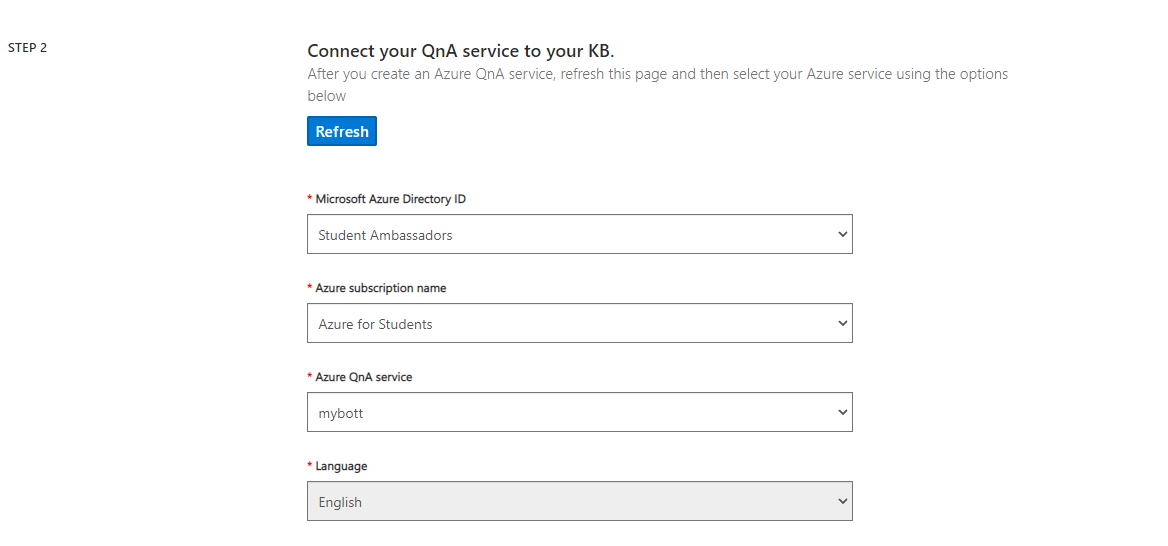
**QnA Maker Knowledge Base**

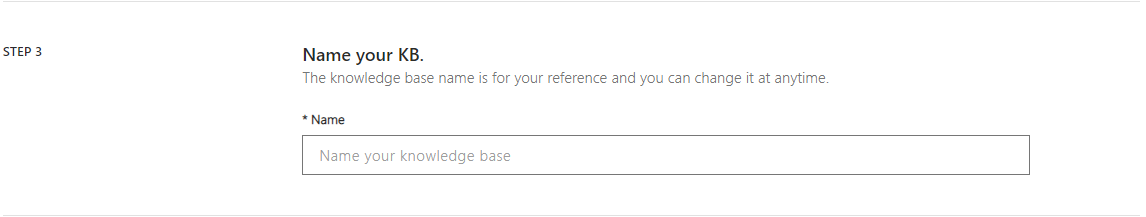
**Step 1:** Sign in into the [QnA Maker](https://www.qnamaker.ai/) platform with your Microsoft account.

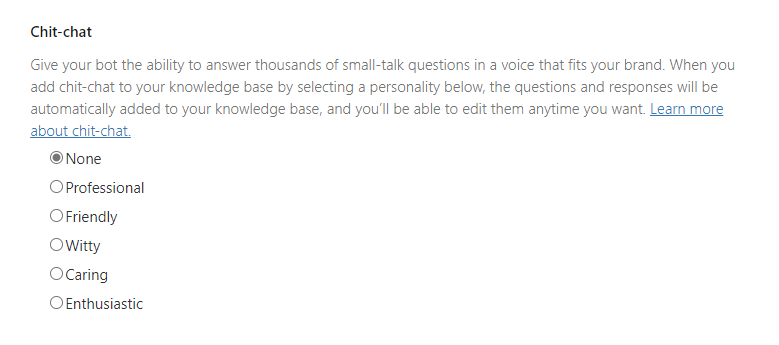


**Step 2:** Once signed in, click on **Create a Knowledge base** so that can have our knowledge base setup

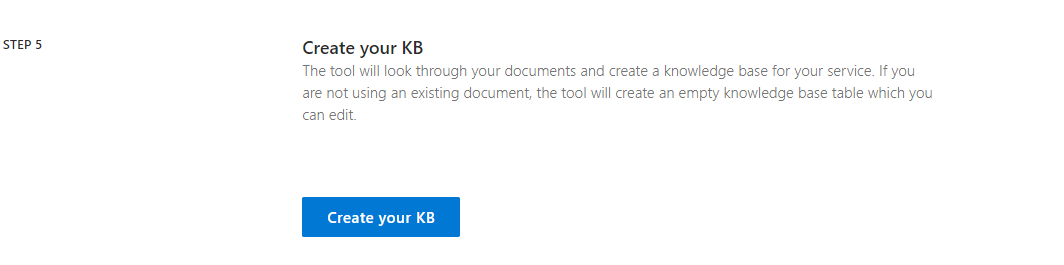


**Step 3:** Skip step 1 of the **Create Knowledge base** if you have your QnA maker app service set up and created. Fill in all the information about your knowledge base by pointing it to the right subscription and qna maker app service.  
  


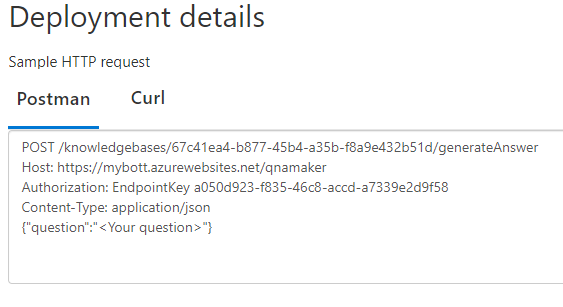
**Step 4:** Name your knowledge base   


**Step 5:** You can also populate your by using a **Url** or uploading a file (.csv for an example) but skip this part and choose **Professional** for the the **Chit-chat** part.  
  


**Step 6:** Now you can create your knowledge base by clicking on **Create KB**

****

**Note:** On your knowledge base click on **Settings** and navigate to the bottom of that page where you will find **Deployment Details** of your knowledge base. We will use this information in both of the flows we will be building in the next steps.



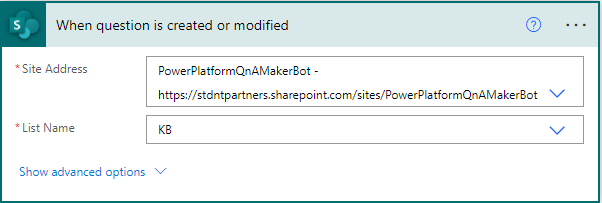
**Power Automate Flow - Update QnA maker knowledge base**

Now that we have our QnA knowledge base built, we have to allow the Admin/the user to be able to update it using the data that is stored in SharePoint (with Questions and Answers columns).

**Note:** You cannot use your SharePoint url when creating your knowledge base in qnamaker.ai it will give you an error, that is why we use Power Automate to update the knowledge base each time there is a new entry in SharePoint.

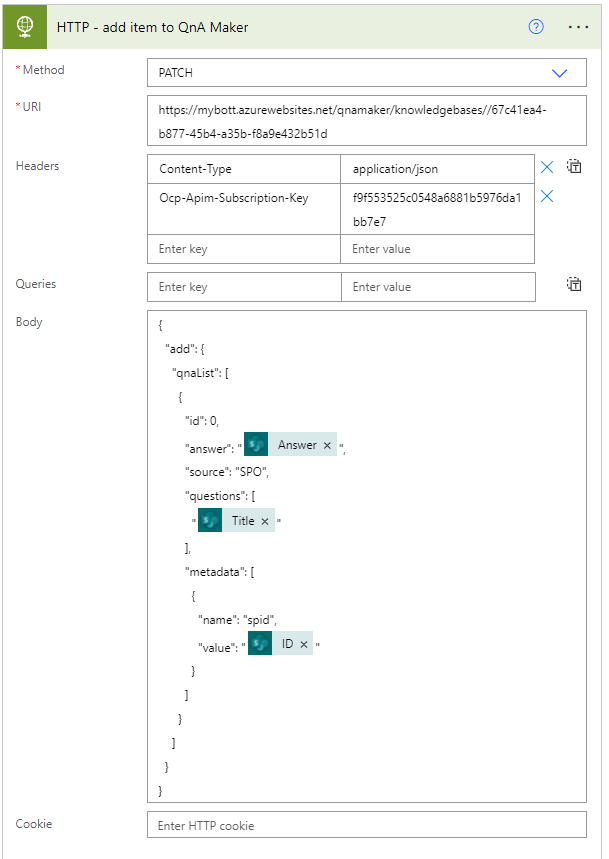
**Step 1:** When creating your flow, choose **Automated Cloud Flows** and choose SharePoint’s **When an item is created or modified** trigger. This will allow the flow to trigger each time the Admin adds/updates a question and answer.

1. Site Address - We need to specify the SharePoint site where our list resides in
2. List Name - We need to specify the SharePoint list where our question and answer columns are, so that we can pull the new information from there.

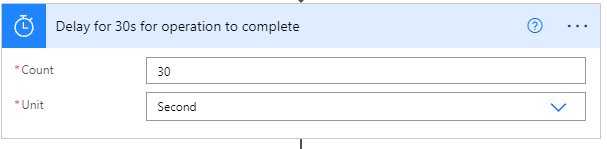


**Step 2:** Add an HTTP Request action so that we can be able to post the new item into our knowledge base.

1. Method - We use the PATCH method so that we can be able to update our knowledge base with the new information.
2. URI - Our knowledge base link with knowledge base ID
3. Headers - The subscription key (from your Azure subscription) and content-type goes in here.
4. Body - JSON that will contain the question and answer from our trigger **When an item is created or modified**, we will send or PATCH this to our knowledge base.

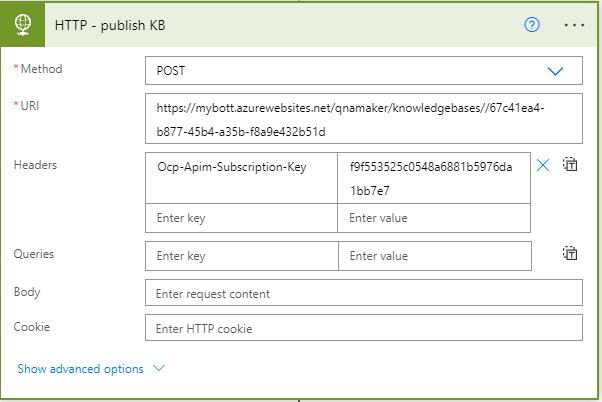


**Step 4:** Add a delay action that will wait for 30 seconds to help our knowledge get the data accurately. This caters for whenever there was an internet connectivity issue for the Admin.



**Step 5:** We add another HTTP Request action, which will allow us to publish our knowledge base with the new updated questions that include the recent one that was added from our Trigger.

1. Method - We use the POST method so that we can be able to update our knowledge base with the new information.
2. Our knowledge base link with knowledge base ID
3. Headers - The subscription key (from your Azure subscription) and content-type goes in here.



**Step 6:** Repeat step 4 for by adding another delay action wait for 30s after the changes have been published to the knowledge base.

**Power Automate Flow to Post an adaptive Card on teams as a bot**

Now that we have the QnA bot set up, it is time to make it work!

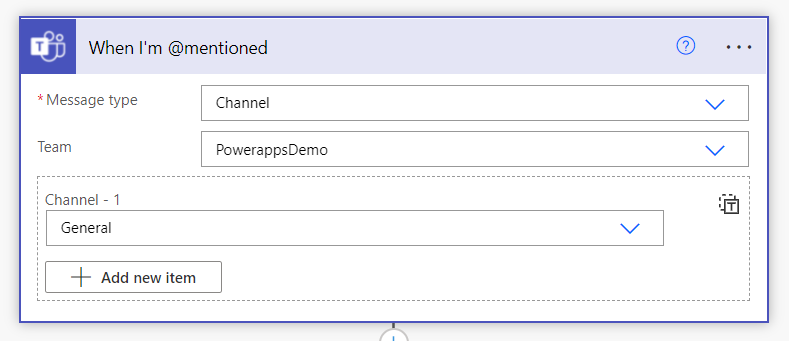
How do we use it in MS Teams? How do we link it with adaptive cards? What happens if there are questions that aren’t there in the Knowledge Base?

All of the above questions have been solved by a simple flow, let’s get started with building it.

**Step 1:** To call the bot, we need a trigger and for this case, our trigger is “When I’m @mentioned”.

When the admin is mentioned, we trigger the bot to work.

We need to mention the following things to get it running:

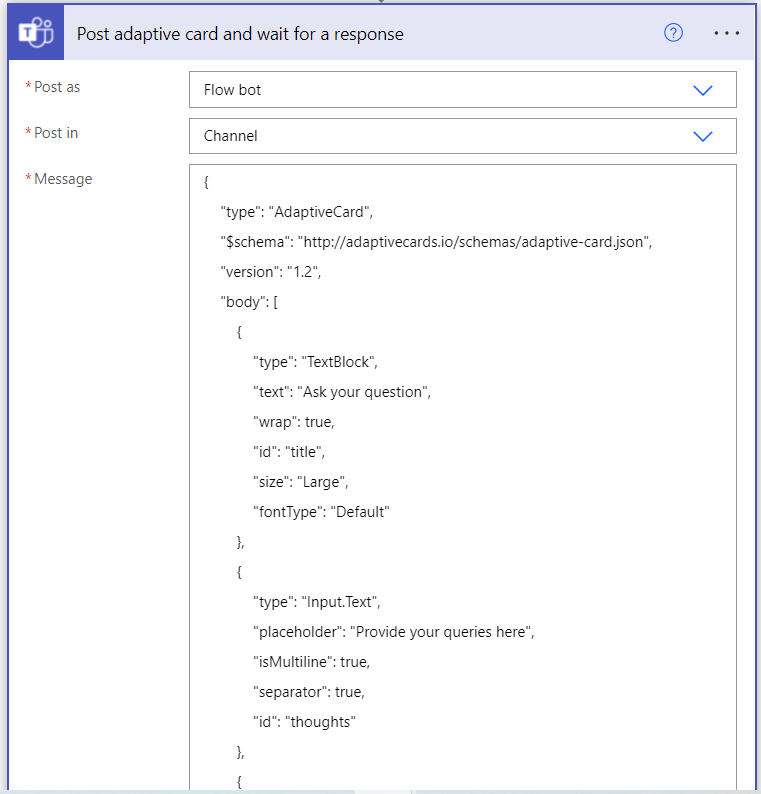
1. Message type (for our case it is channel).
2. Team- The team where we want the bot to function.
3. Channel- The channel where we want the bot to function.  
   

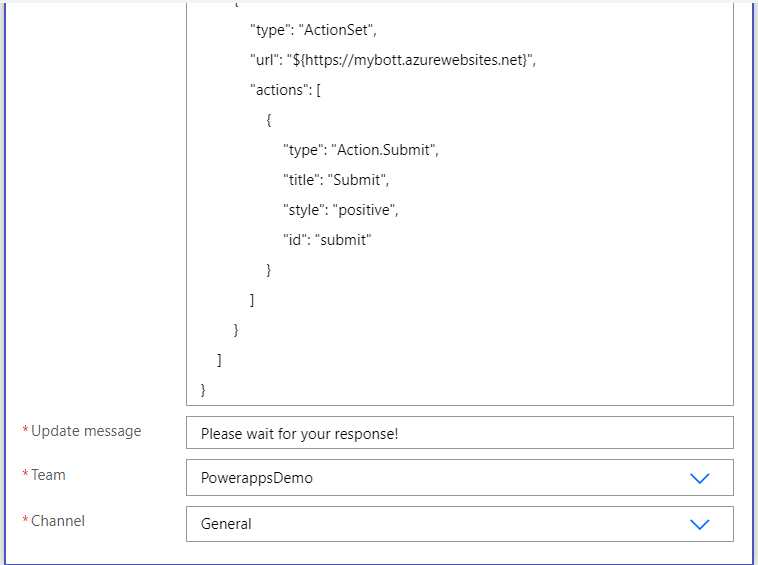
**Step 2:** We want users to ask questions, which can be sent to the QnA bot for answering. For the users to submit their answers we use an action “Post adaptive card and wait for response”.

To have an adaptive card that has a text block, and a submit button, we need to put in the relevant JSON, which can be drafted from the [adaptive card designer](https://adaptivecards.io/designer/).

We need to mention the following things to get it running:

1. Post as- How we want the adaptive card to pop up (flow bot, in our case)
2. Post in- Where we want to post it (channel, for our scenario)
3. Message- The JSON of the adaptive card
4. Update message- It can be some message that makes the user aware of the next step.
5. Team- The team we want to post in
6. Channel- The channel we want to post the adaptive card in





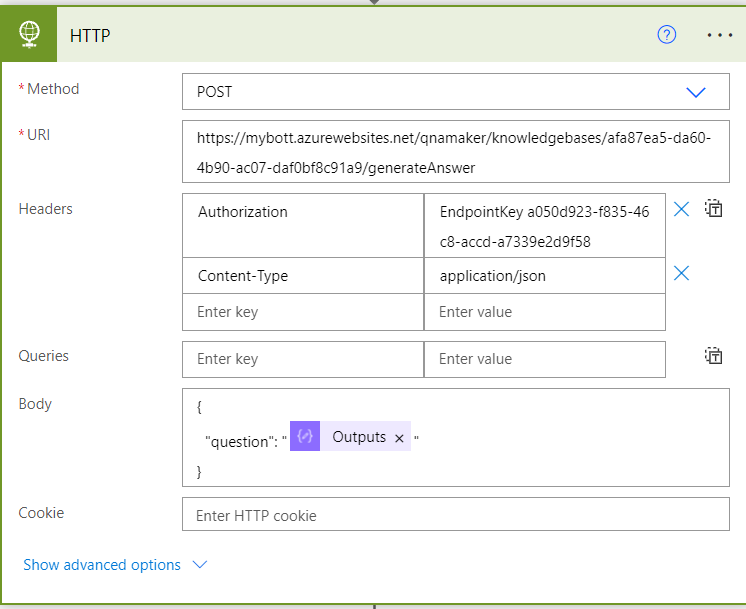
**Step 3:** Our next step is to get the question, for that we need Compose action. From our above JSON, we see that our question is stored in the dynamic content ‘thoughts’, we put this in our action.



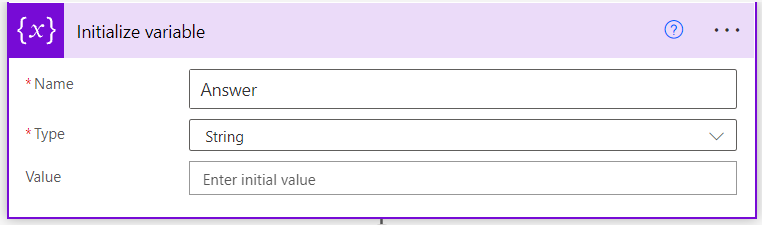
**Step 4:** To interact with our knowledge base which is fed into the bot, we need an HTTP response.

To get our HTTP response, we need to mention the following terms:

1. Method- What action do we want it to perform. (POST in our case, since we want the answers from the KB and post it back to our user).
2. URI- Our Knowledge Base link that generates the answer.
3. Header- The endpoint key and content-type goes in here.
4. Body- The question that we send to the KB needs to be mentioned here. For us, we send the output of the ‘Compose’ action, since that contains our question.

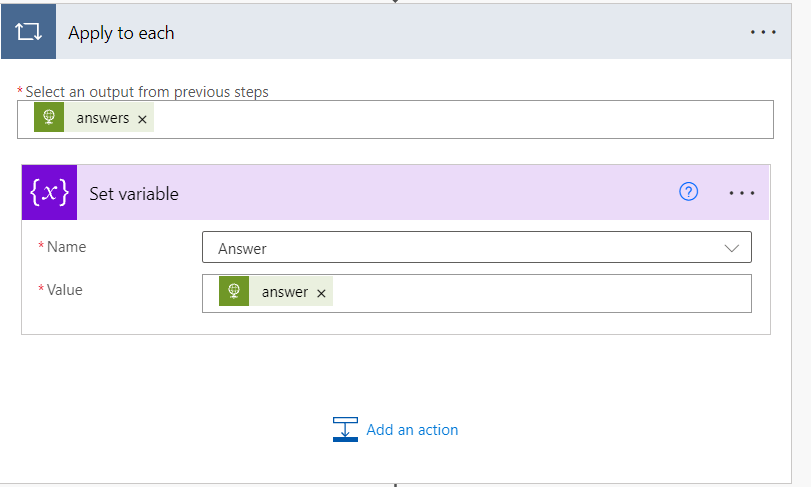


**Step 5:** We initialize a variable named ‘Answer’ that will contain the answer we receive back from the KB.

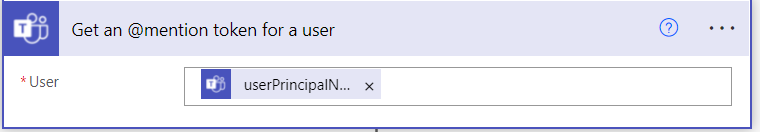


**Step 6:** To find the answer to our requested question, we must iterate it through all the questions, until it finds the one we requested. To complete this, we have to use the ‘Apply to each’ action.

It gets the ‘answer’ from our HTTP response and sets our ‘Answer’ variable to the received answer.



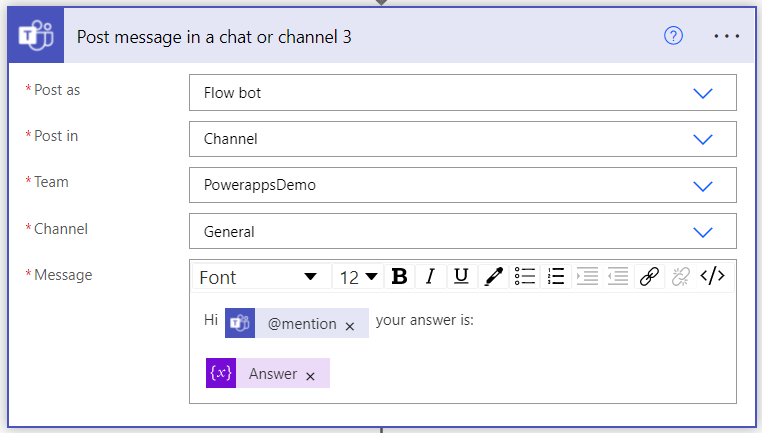
**Step 7:** For a busy channel that has too many queries, it will be easier for a user to find their answer when the answer is posted by mentioning their name. For this, we use the ‘Get an @mention token for a user’ action and mention the ‘userprincipalname’ in the user.



**Step 8:** We now need to post back the answer to the requested question. To achieve this, we use the ‘Post a message in a chat or channel’ as action.

We need to mention the following things to get it running:

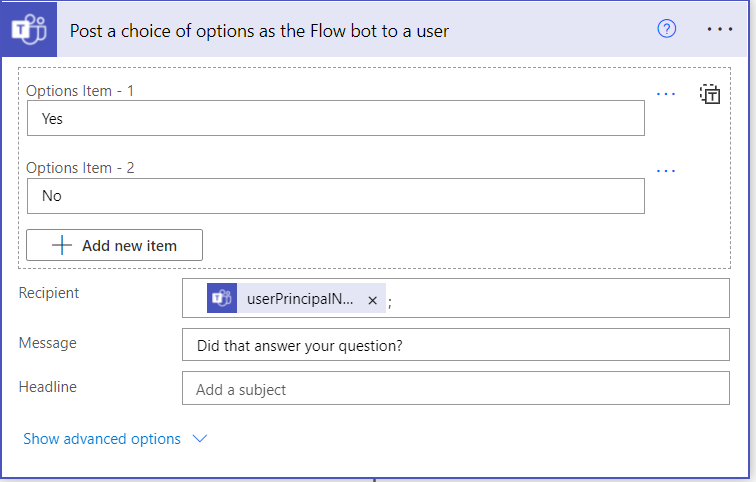
1. Post as- How we want the adaptive card to pop up (flow bot, in our case)
2. Post in- Where we want to post it (channel, for our scenario)
3. Team- The team we want to post in.
4. Channel- The channel we want to post the adaptive card in
5. Message- We mention the user and just share with them the answer

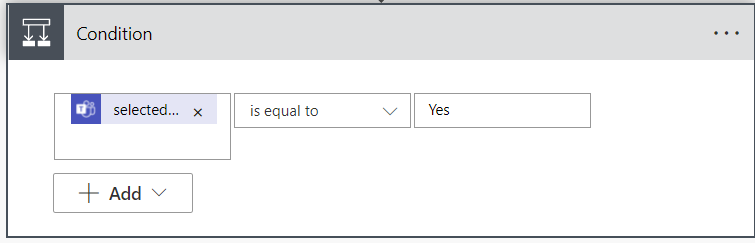


**Step 9:** To know if our answer satisfied their question or not, we sent them a poll.

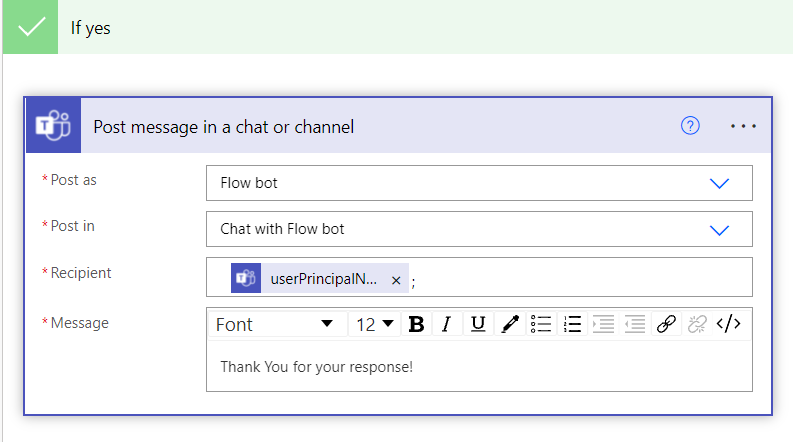
For running this successfully, we need to:

1. Give them a set of choices (Yes or No for our use-case)
2. Recipient- Mention who this poll needs to be sent to.
3. Message- Frame a message that goes with the choices.

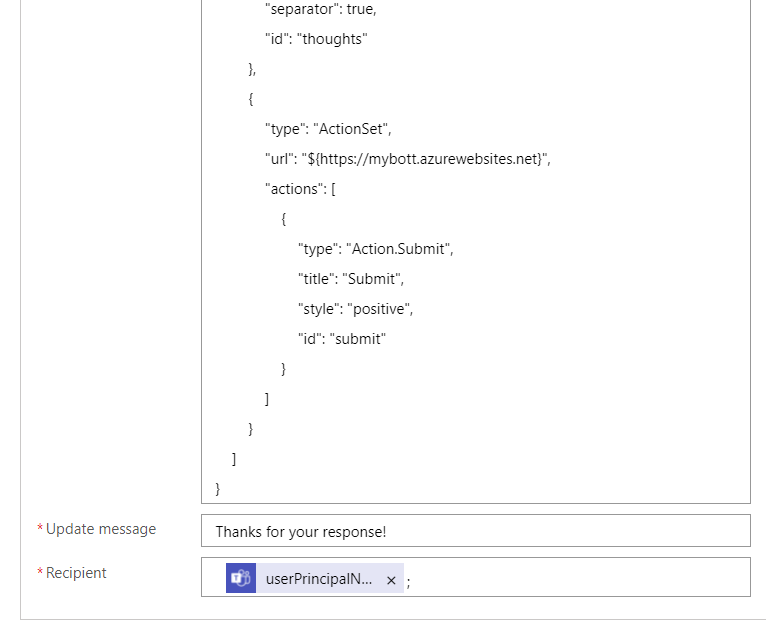
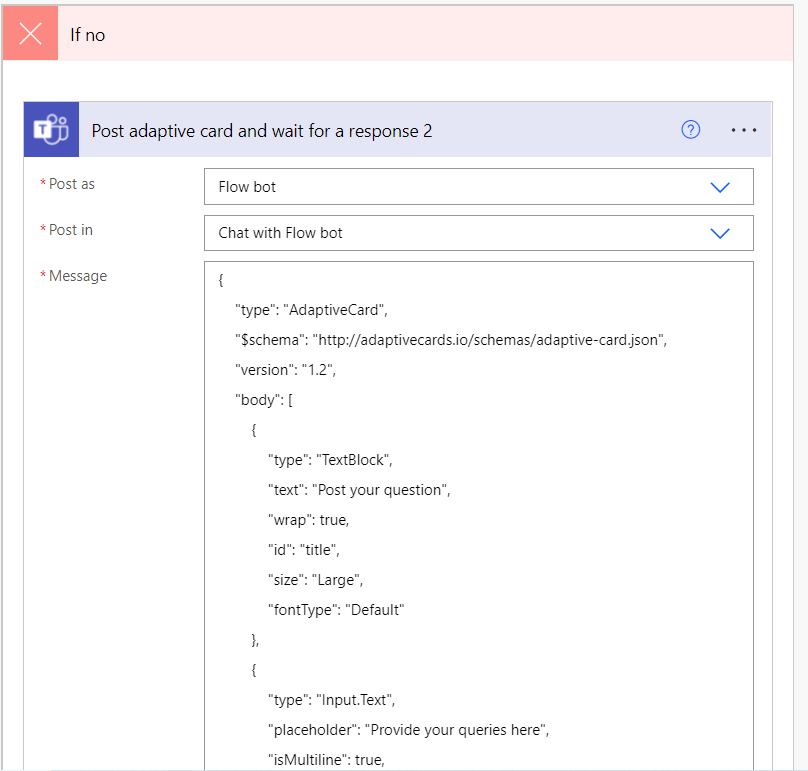


**Step 10**: Depending upon the choices selected, we follow a different course of action. To know which options were selected we use the ‘Condition’ action, which takes up the response from the above adaptive card.

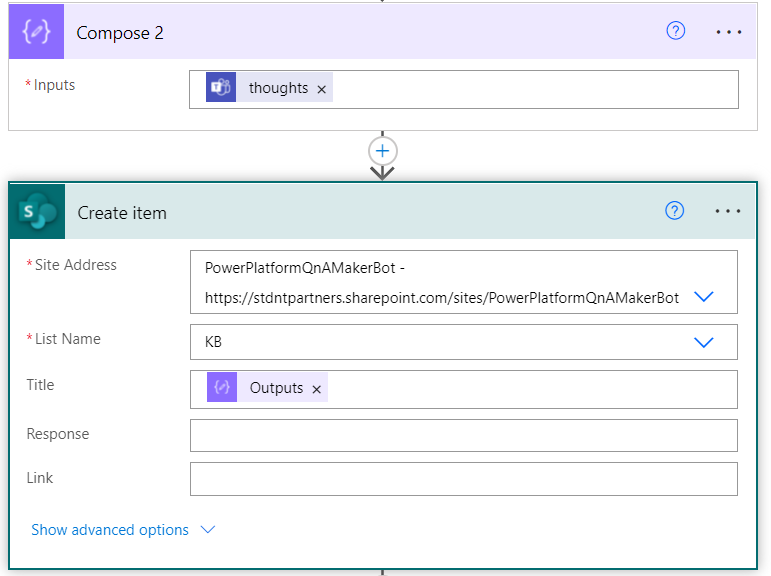
**Step 11:** If the selected response is “Yes”, it means the user is satisfied with the answer received. For this, we simply post a thank you message.



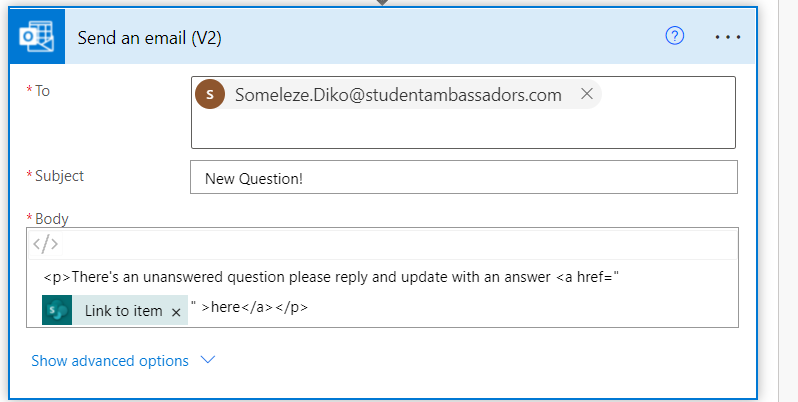
**Step 12:** If the selected response is “No”, it means the user did not get the required response. So we share with them an adaptive card that catches their response.



The response received is the question that was unanswered. Our next step is to store the question in our KB (which is a SharePoint list). For this, we use another ‘Compose’ action, that stores the question, and its output is sent to the ‘Create item in SharePoint’ action.



The last step is to inform the admin about the new question, so that they can answer it. For this, we use the ‘Send an email’ action.



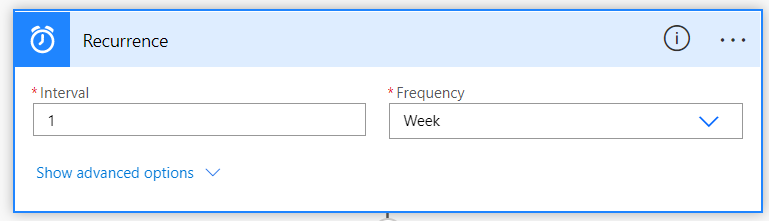
Lastly, an analysis of the most asked questions would help the admins understand the needs of the community better.

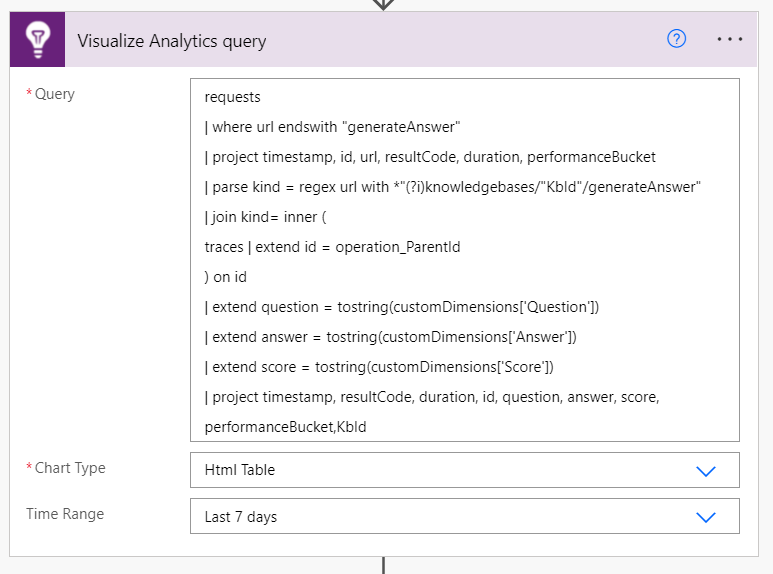
To get this, we followed two major steps:

1. Got the latest analysis using a query, details about which can be found [here](https://docs.microsoft.com/en-us/azure/cognitive-services/qnamaker/how-to/get-analytics-knowledge-base?tabs=v1).
2. Made a flow that captures the analysis and sends the admin an update after 7 days.

The flow goes like:

1. The flow runs after after week and it was achieved with the trigger “Recurrence”.



1. To get the analysis on the most asked question, we used the action ‘Visualize Analytics Query’. Added authentication token and API key to get the connection running. Once that is done, we just add the query, determine the chart type and the time range.  
   
2. To receive the analysis on email, we use the action “Send an email (V2)”. Add the recipient and the body of the response received from the above step as the email body.

