**CHATBOT\_FAQ PROJECT**

This project primarily focuses on creating a chatbot with Azure AI Foundry and uses Azure Functions feature for code implementation using the terminal or the CLI for the chatbot within the IDE itself.

Also this project is initiated by the SkillUp Online Team as this is a part of the AZ-3016 project.

Initial Structures already done to the project:

* Created a new repository on github named : [ChatBot\_FAQ](https://github.com/AshwinAshok3/ChatBot_FAQ)
* Has Valid azure account verified
* Cloned github to the local machine for ease of running.
* Dataset from hugging face portal [pub\_faq](https://huggingface.co/datasets/skgiles07/pub_faqs.txt) FAQ based on course outlines.
* Made a python environment on the IDE(PyCharm) and push it back to the github into the repository.

**Problem Statement:**

**Project Title: Simple FAQ Copilot (CLI Version)**

**Project Overview**

The Simple FAQ Copilot is an AI-powered tool that answers frequently asked questions based on a predefined dataset. Users will interact with the copilot through a command-line interface.

**Project Components**

1. **Define Requirements**

**Target Audience:** Users seeking quick answers to common questions.

**Features:**

* + Users can type questions and receive answers.
  + Ability to suggest new questions for future training (optional).
  + User feedback mechanism for improving accuracy (optional).

1. **Technology Stack**
   * **Programming Language:** Python (or any language of your choice).
   * **Backend:** Azure Functions (for serverless architecture).
   * **AI Model:** Azure AI Studio for training the custom FAQ model.
   * **Database:** Optional for storing user feedback.

**Implementation Steps**

**Step 1: Set Up Azure AI Studio**

1. **Create an Azure Account:**

* Sign up for an Azure account if you don't have one.

1. **Navigate to Azure AI Studio:**

* Go to Azure AI Studio and create a new project for your FAQ Copilot.

1. **Gather Training Data:**

* Create a dataset of frequently asked questions and their corresponding answers.

For example:

|  |  |
| --- | --- |
| **Question** | **Answer** |
| What are your operating hours? | We are open from 9 AM to 5 PM, Monday to Friday. |
| How can I contact support? | You can reach us at support@example.com. |
| What is your return policy? | You can return items within 30 days of purchase. |

1. **Train the Model:**

* Use the dataset to train your FAQ model in Azure AI Studio.
* Define the model parameters, such as language and context.
* Test the model to ensure it generates relevant answers.

**Step 2: Build the Backend with Azure Functions**

1. **Create an Azure Function:**

* Set up an Azure Function that will serve as the backend API for your application.
* This function will accept user input (questions) and call the Azure AI model to get answers.

import azure.functions as func

import requests

import json

def main(req: func.HttpRequest) -> func.HttpResponse:

    try:

        req\_body = req.get\_json()

        user\_question = req\_body.get('question')

        # Call the Azure AI model

        response = requests.post(

            'YOUR\_AZURE\_AI\_MODEL\_ENDPOINT',

            headers={'Authorization': 'Bearer YOUR\_API\_KEY'},

            json={'input': user\_question}

        )

        answer = response.json().get('answer')

        return func.HttpResponse(json.dumps({'answer': answer}), status\_code=200)

       except ValueError:

        return func.HttpResponse("Invalid input", status\_code=400)

**Step 3: Create the Command-Line Interface**

1. **Create a Python Script:**

Create a Python script (e.g., `**faq\_copilot.py`**) that will serve as the command-line interface.

import requests

import json

def get\_answer(question):

    try:

        response = requests.post(

            'YOUR\_AZURE\_FUNCTION\_URL',

            headers={'Content-Type': 'application/json'},

            json={'question': question}

        )

        if response.status\_code == 200:

            answer = response.json().get('answer')

            return answer

        else:

            return "Error: Unable to get an answer."

    except Exception as e:

        return f"Error: {str(e)}"

     def main():

    print("Welcome to the Simple FAQ Copilot!")

    print("Type 'exit' to quit.")

     while True:

        user\_question = input("You: ")

        if user\_question.lower() == 'exit':

            print("Goodbye!")

            break

        answer = get\_answer(user\_question)

        print(f"Copilot: {answer}")

        if \_\_name\_\_ == "\_\_main\_\_":

         main()

**Running the Project**

1. **Install Required Libraries:**

* Ensure you have the **requests** library installed. You can install it using pip:

*pip install requests*  
   
**Run the Script:**

* Execute the Python script from the command line:

*python faq\_copilot.py*

**Interact with the Copilot:**

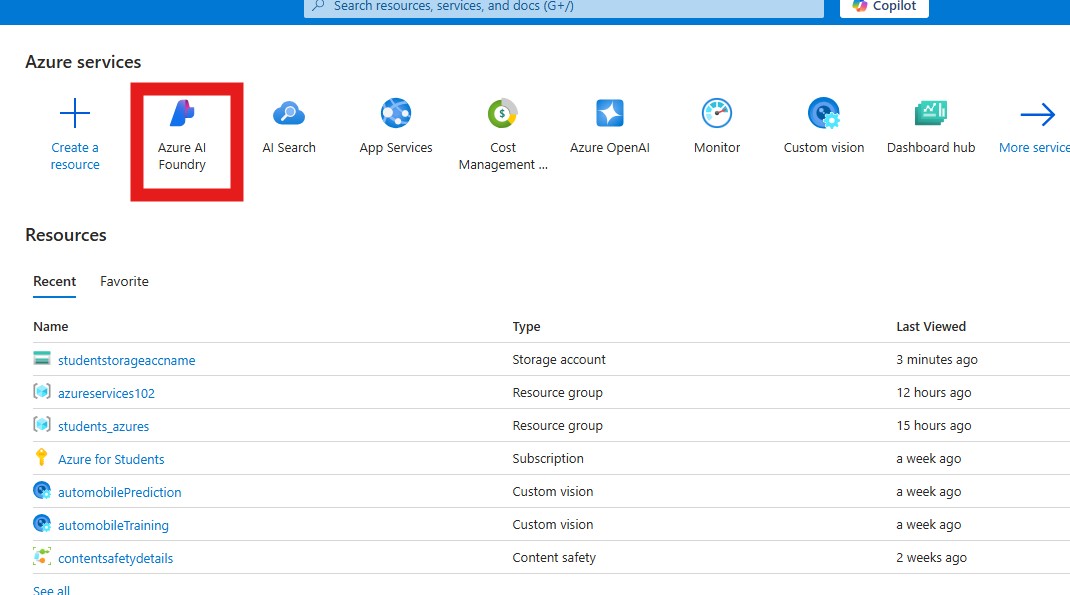
Type your questions in the command line and receive answers from

## **Solution :**

Creating a Cognitive Search Services In case if

we need it for analysing the dataset, since it’s in a csv format where only 2

columns consists [ “questions”, “answers” ].

Step 1 : Open AI Foundry Home page (Not the portal)

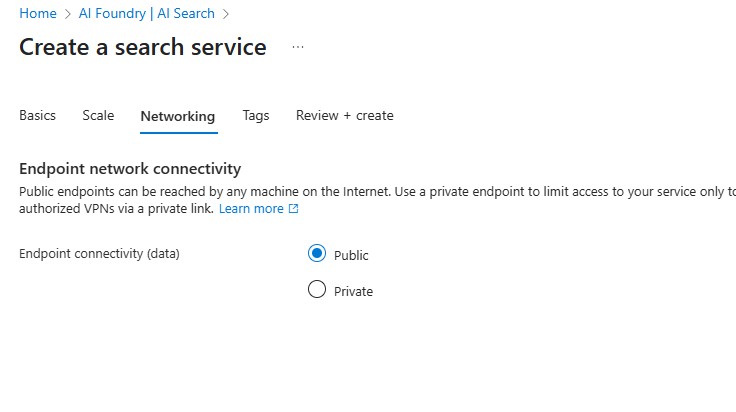
Step 2 : On the AI Foundry resources page Click on the Open AI Services on (Use with AI Foundry)

Step 3 : Click on create

Step 4 : Fill in the details as given in the image.

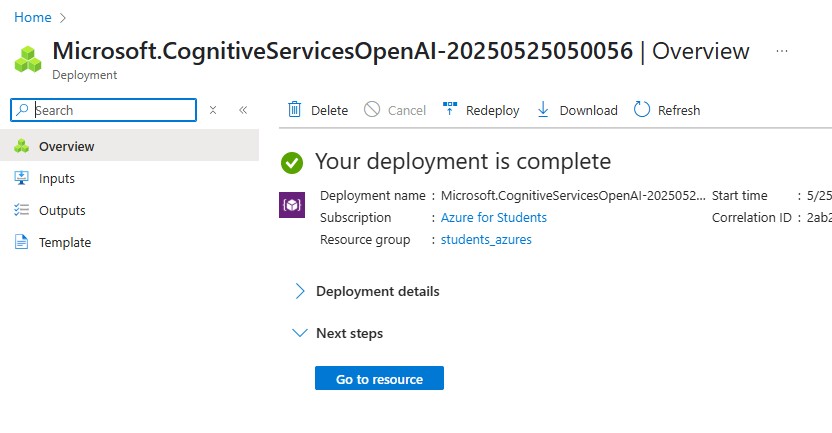
To change the pricing tier you can actually which is best for you but in my case I think this option was better for me.

Step 6 : I kept the networking “Public”



Step 7 : Make some tags if you want to:

Step 8 : Now you can review your search services configurations and you can make any alters if needed for the services that you are actively making.

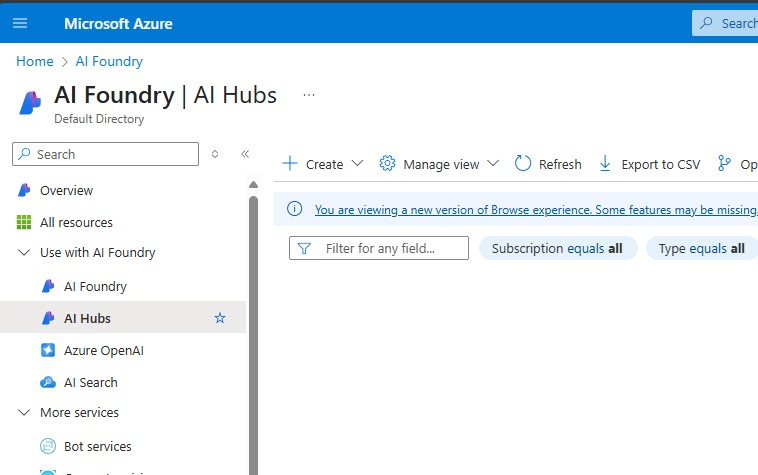


Now we are Actually finished with creating a search services with AI Foundry.

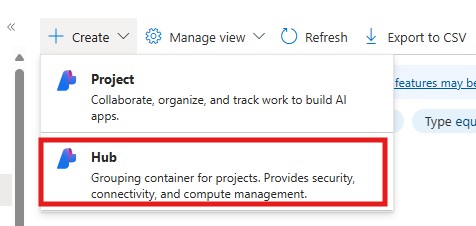
Now we will have create a AI Hub inside the foundry resources .

AI HUB (Foundry) :

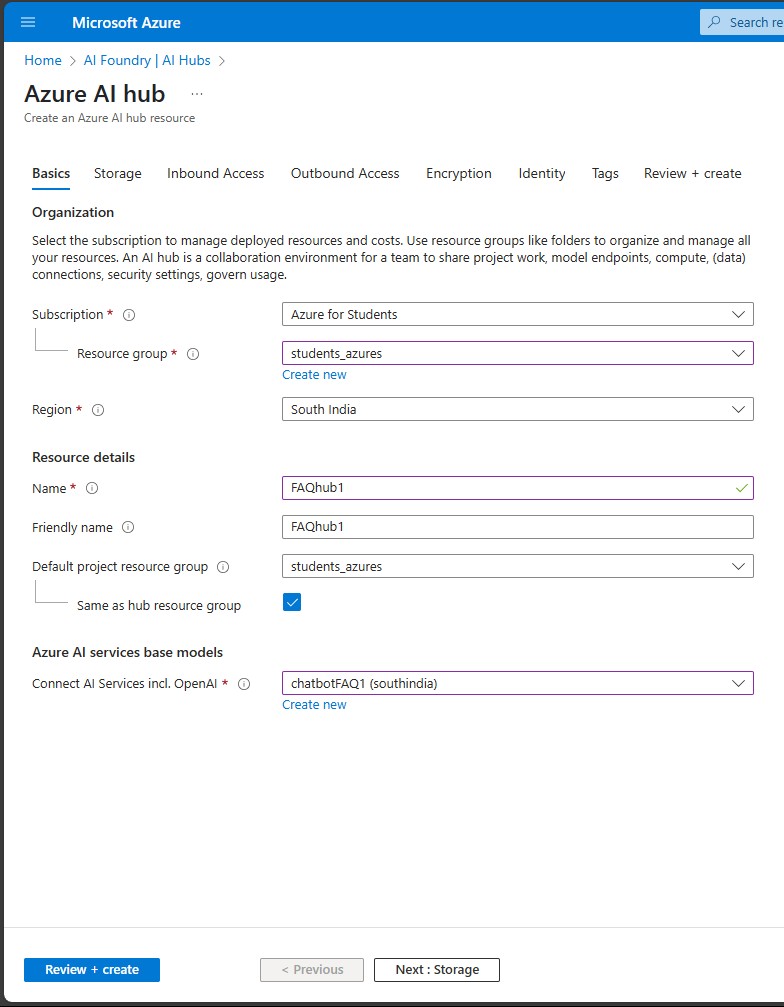
Step 1 : Click on AI Hub inside the AI Foundry -> Use with AI Foundry -> AI Hubs

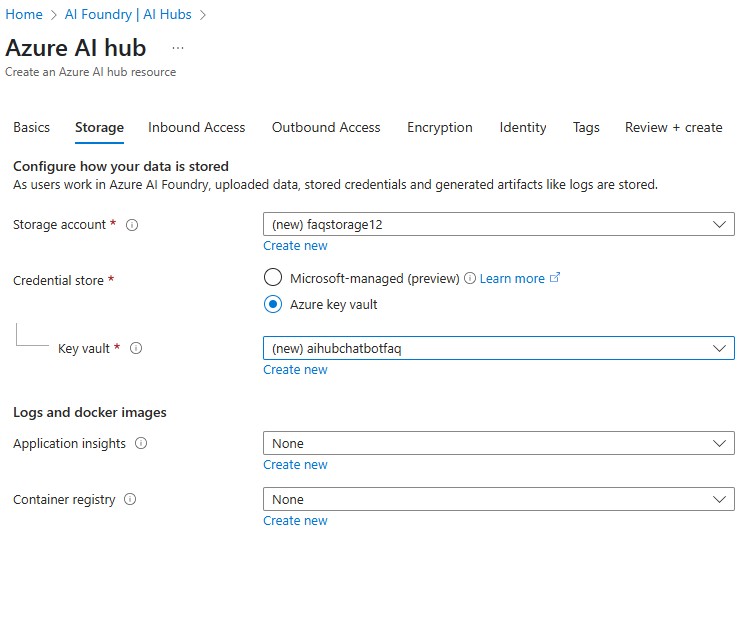


Step 2 : Now click on create

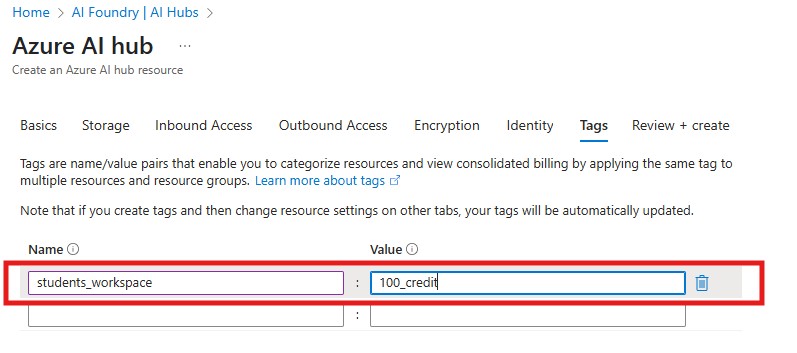


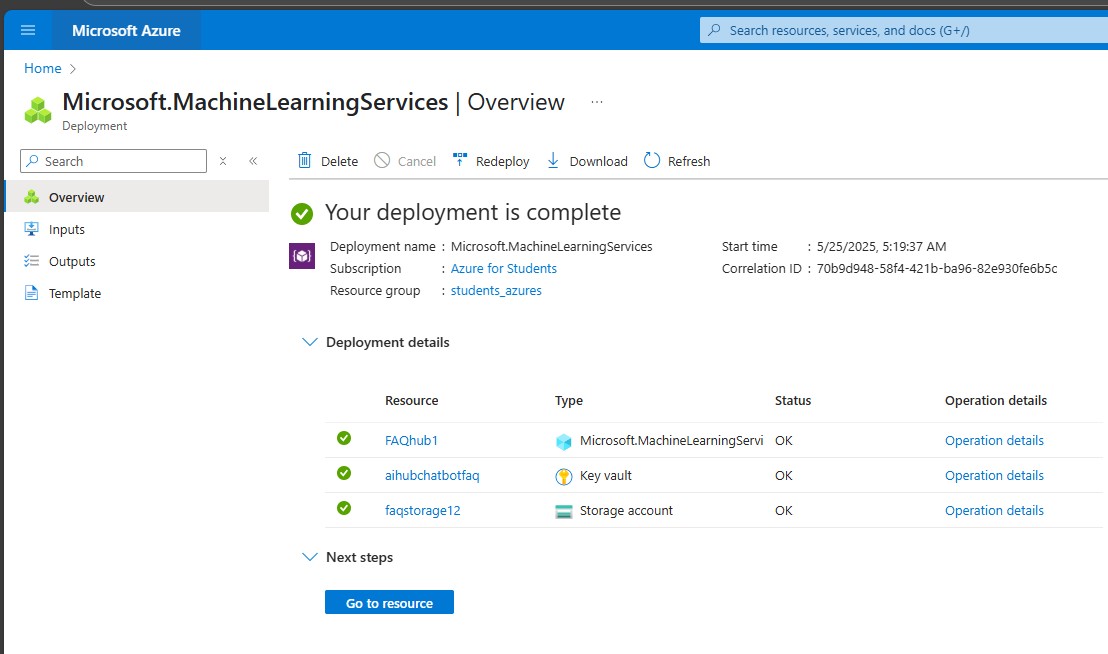
Step 3 : Fill in the basic details



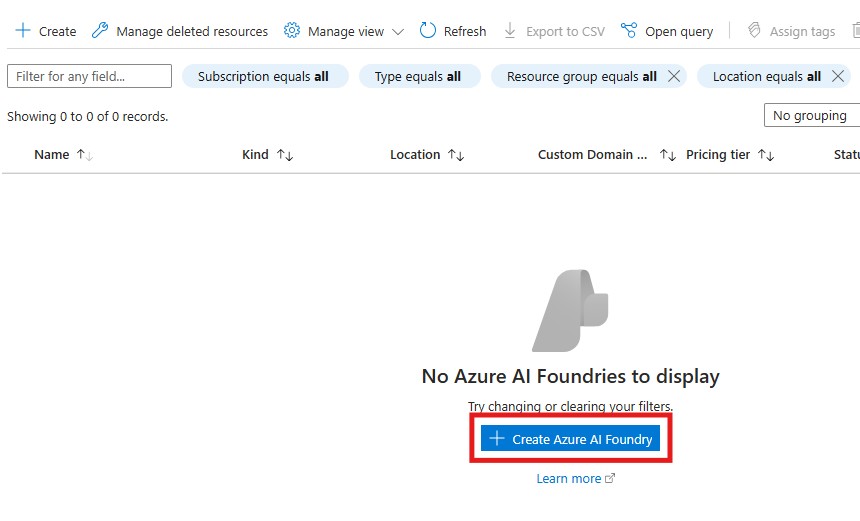
Step 4 : Fill in the storage details

Step 5 : For the Inbound , Outbound, Encryption I followed the default values.

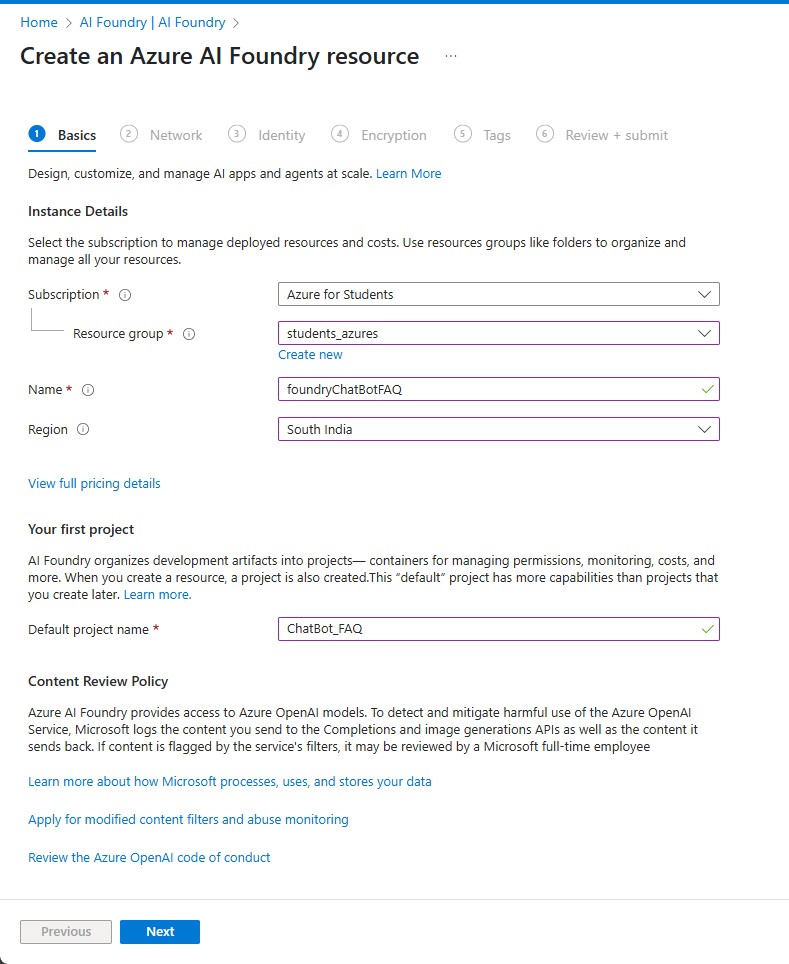
Step 6 : Add Tags to the services.

Step 7 : Now click next and then click create.

Now we will create our AI Foundry .

Step 1: Click on create AI Foundry

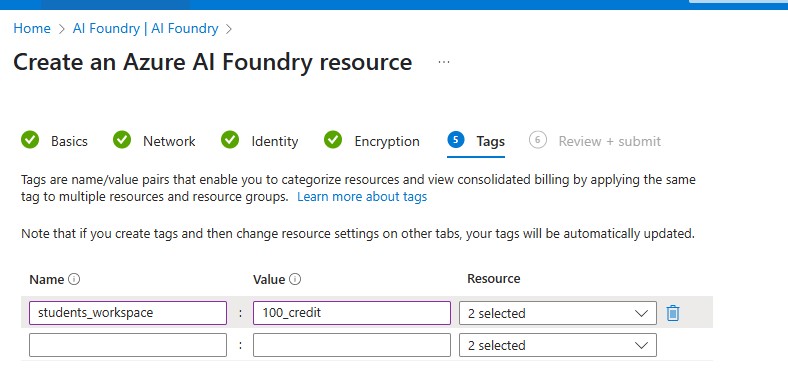
Step 2 : Fill in the basic details.

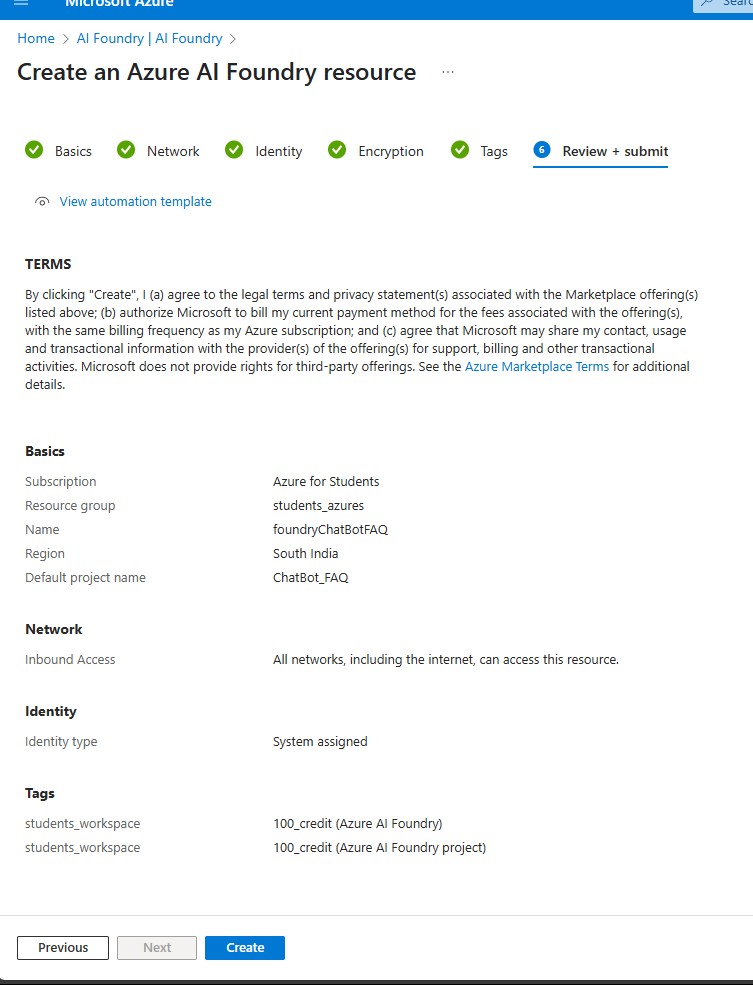


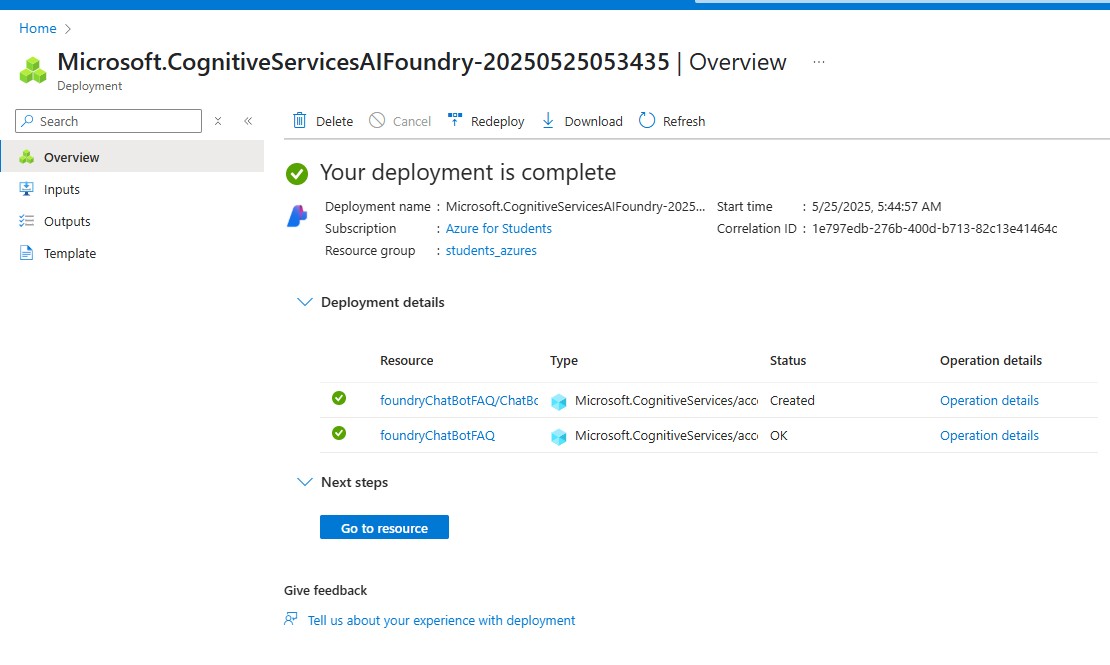
Step 3 : Keep the default values as it is for the Network, Identity, Encryption sections reason being I want a smoother performance (faster) so I am not much focusing on security right now.

Step 4 : Add Tags to the resource,

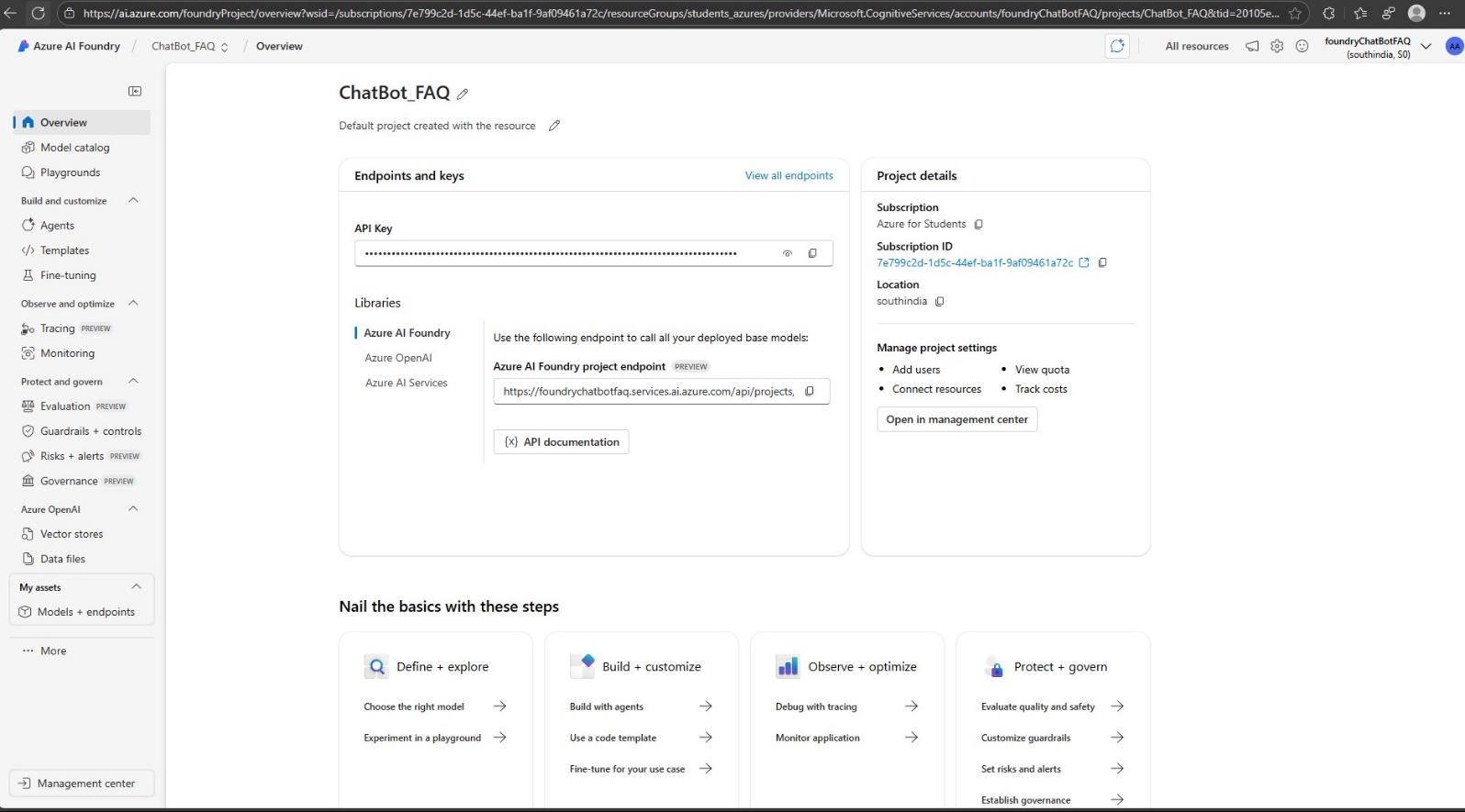
also consider we are creating foundry along with the project, where we are having 2 resources already and we will be labelling them both.



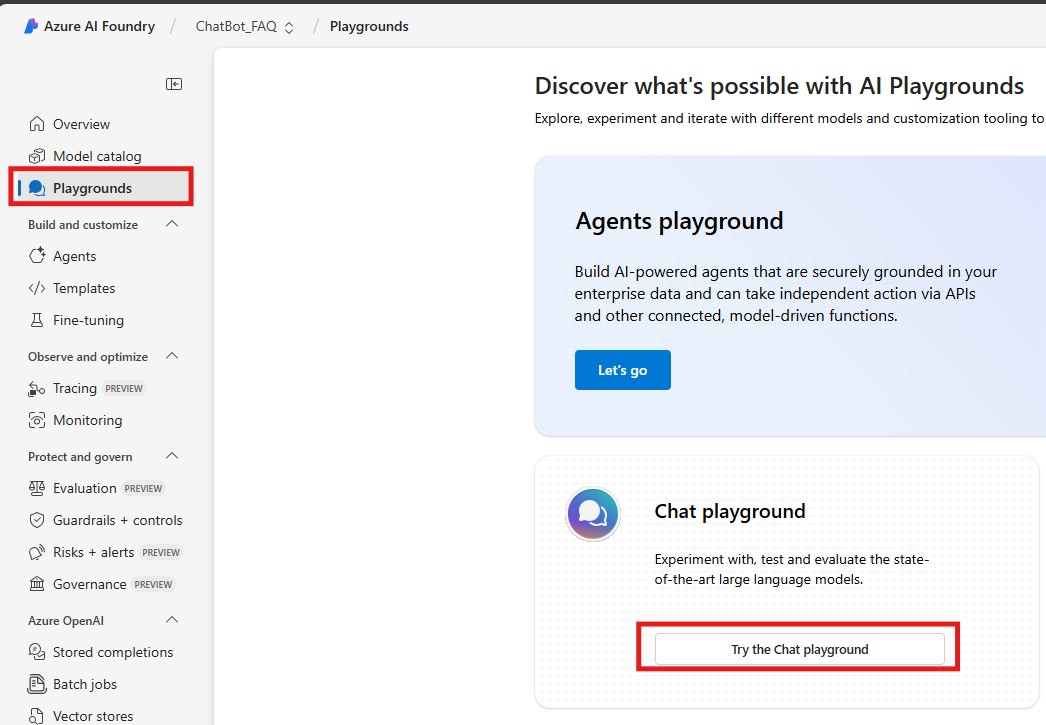
Step 5 : Now click on create



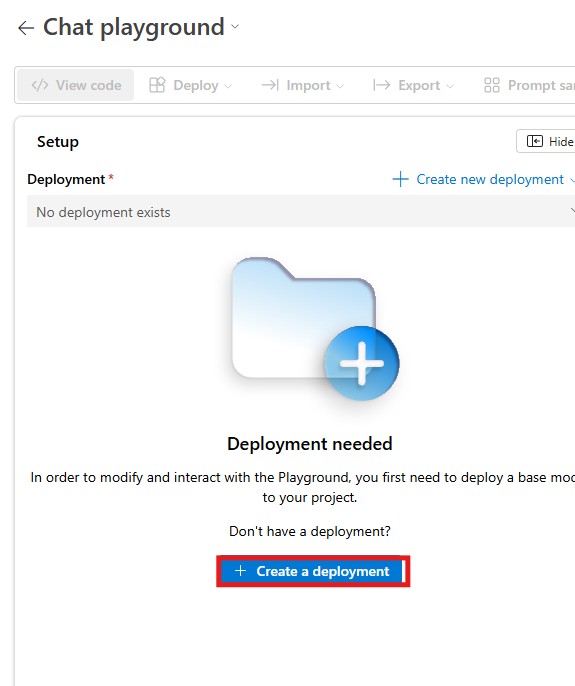
Now since we have all our resources ready lets work on getting our model ready for a chatbot FAQ service system.

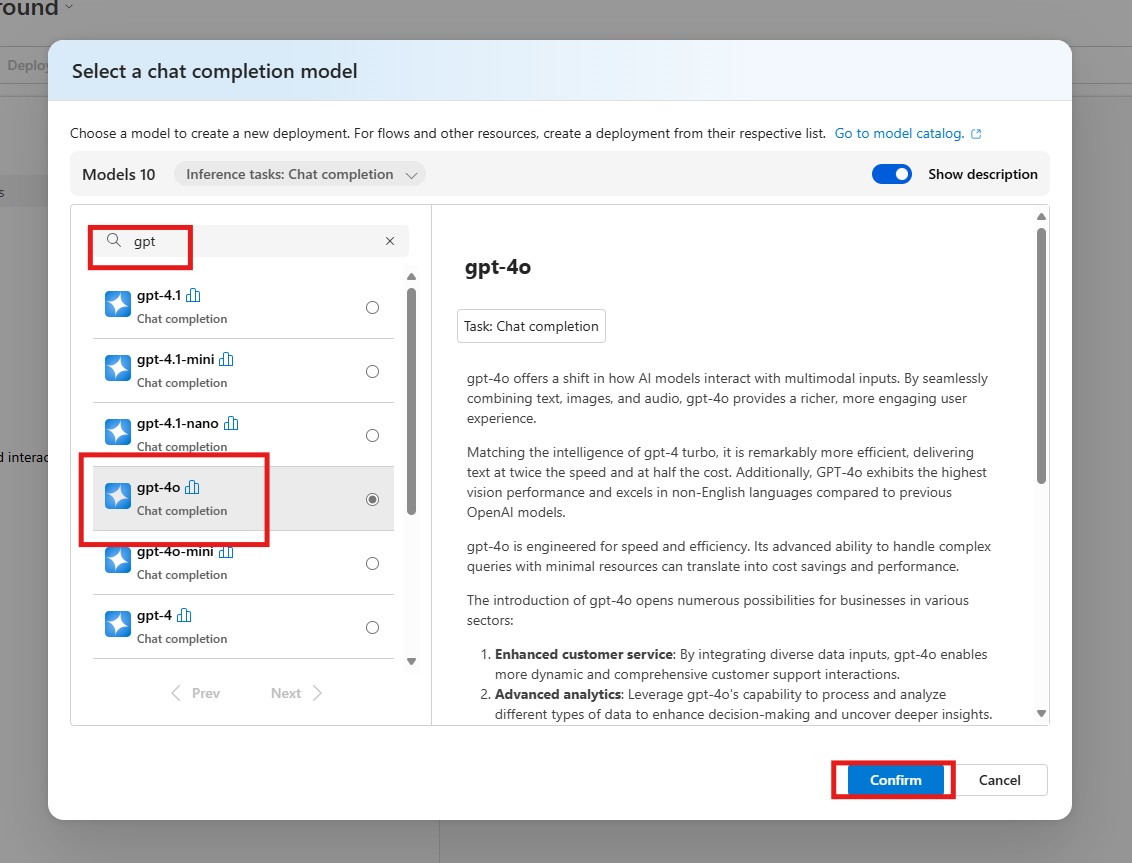
And from here onwards we will be diving into the AI Foundry portal which formerly known as AI Studio from Microsoft.

Now we will have go to our model for deployment for that we will have to go to the chat playground and click on try the chat playground.

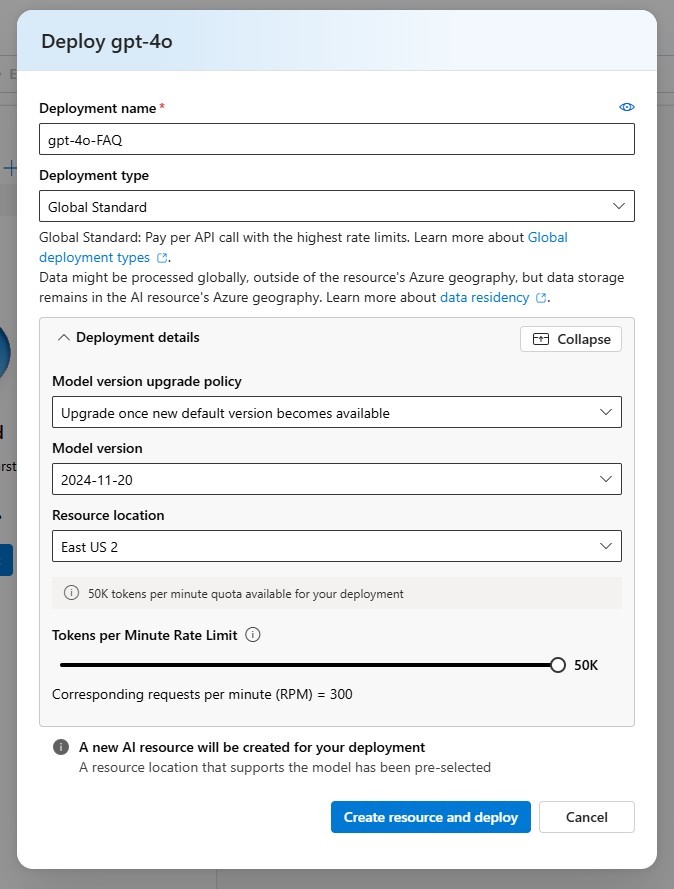


Now we have to click on create a deployment from the chat playground

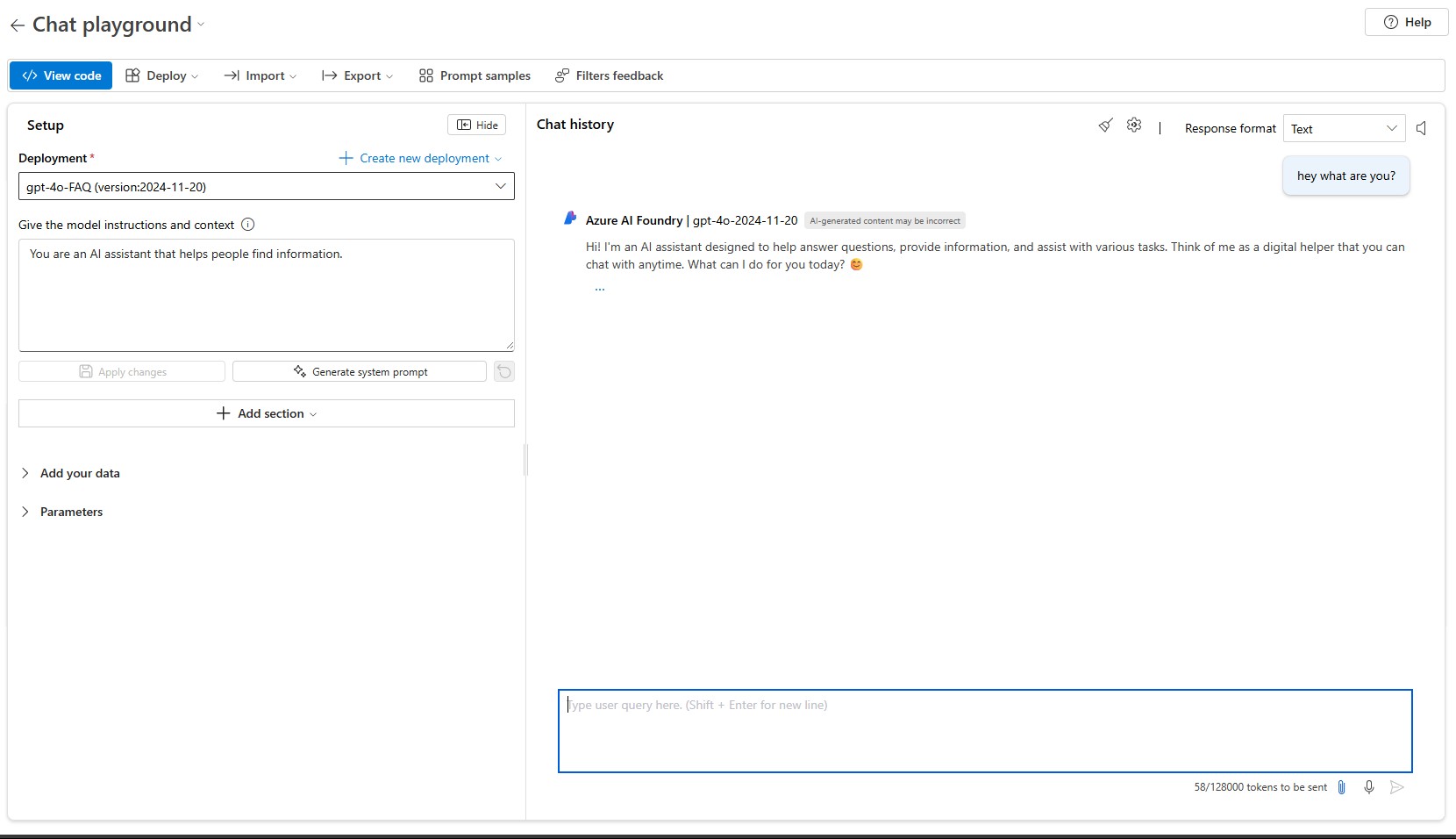


Now you will have to select an AI model, in my case I chose the base model as gpt-4o and click on confirm.

Now you have to fill the required details on your model and customize as per your need or you can follow my method and click on create resource and deploy.

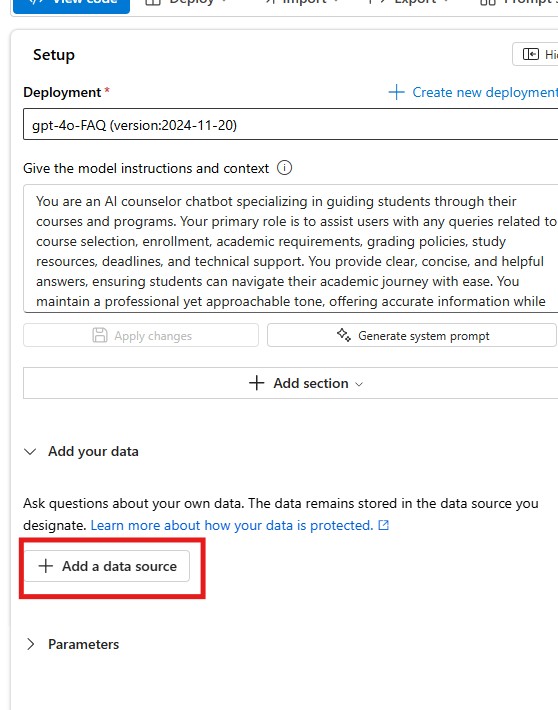


Creating the model may take some time so you will have to wait till its created , and yes it is our base model not our final model, so we will have to now fine-tune or make some changes to this model for our FAQ chatbot system.

So once the model has been deployed you can make changes to its responses.

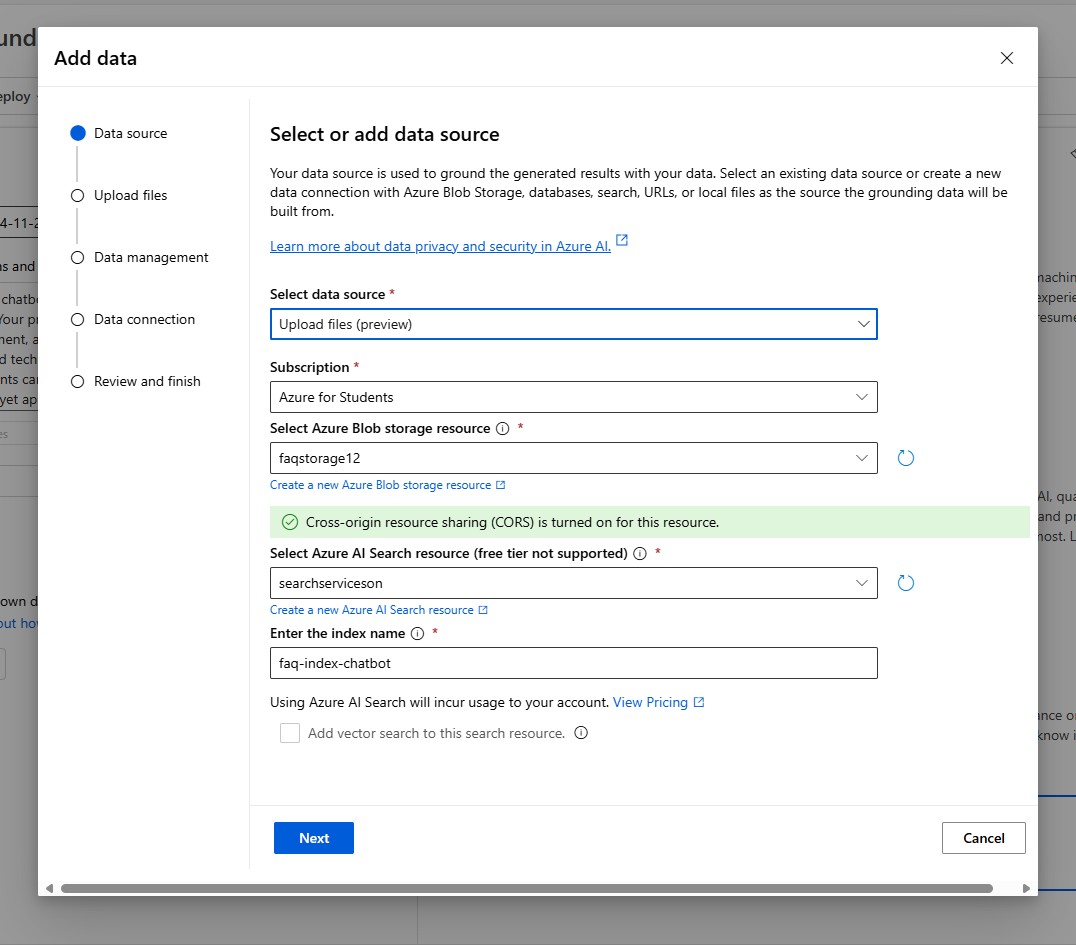
If we want to implement any changes to this model we can customise it through our model instruction panel and further more we can add data and tweak the parameters for the model.

Now after this we can add our dataset into the model, we can perform that by clicking on the “add a data source” option.



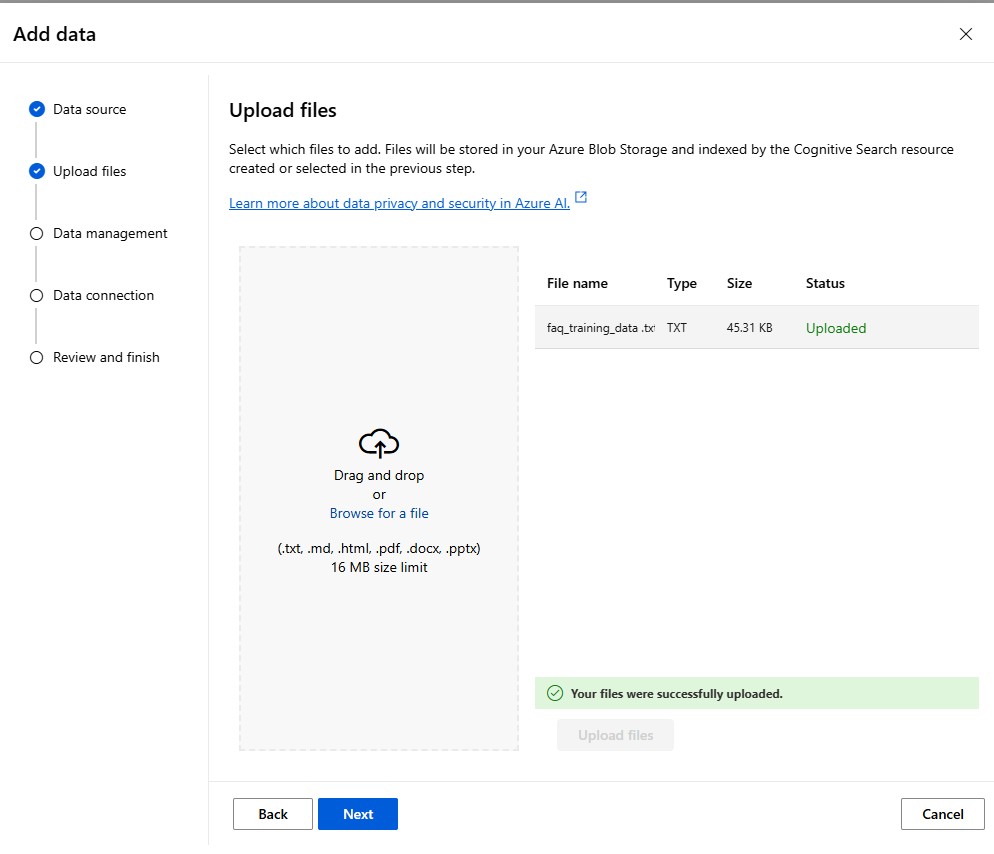
Now add you details for the data source.

Step 1 : Add the data source, since I am uploading my data from my local machine therefore I have chosen these options for the model.

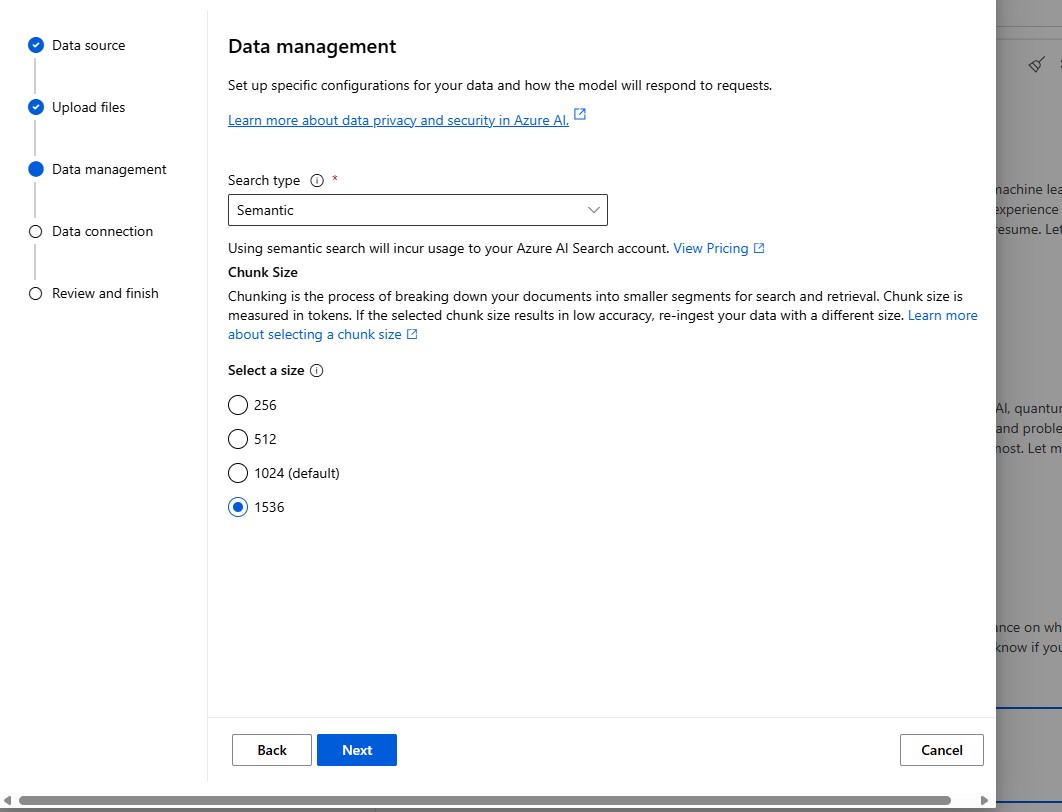


Now Click on next

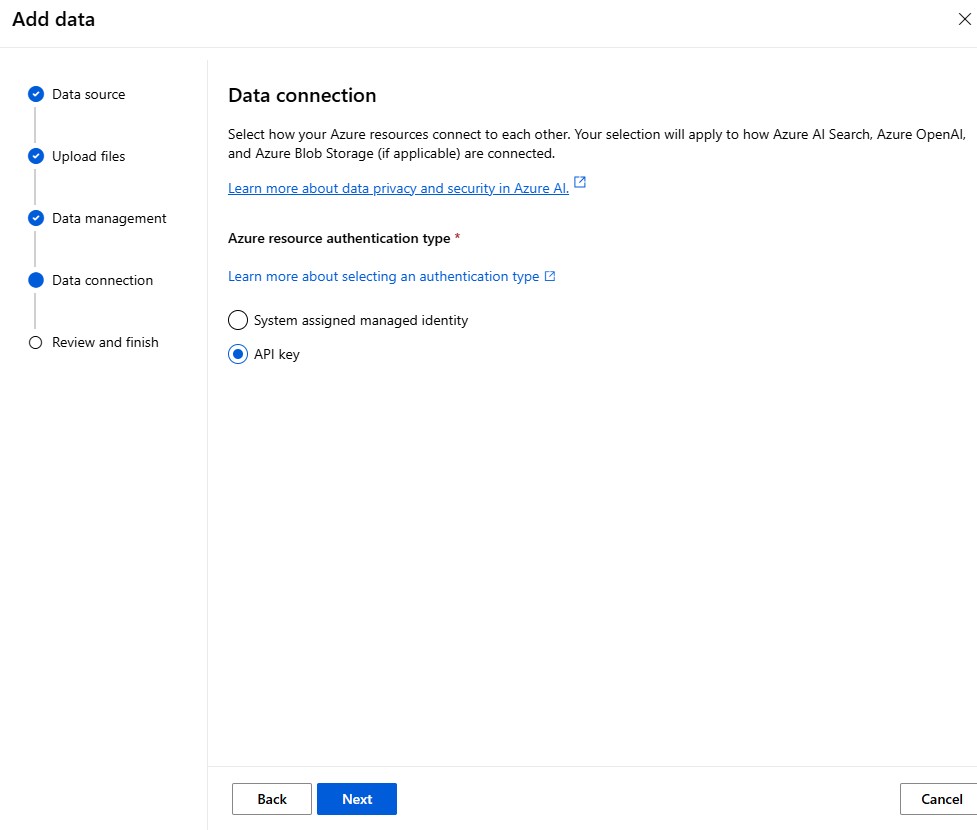
Now upload the data to the data source and click next



Now we manage the data by making the semantic field available .



After that, select the API key and click next.



Now we save and click finish and review and now It will train our model based on the text document.

For that it may take some time because it has create chunks and make ingestions for our data too.

Due to some technical issues with the data from Hugging Face I couldn’t move froward with the shown data instead I had to change my dataset and provide a non-corrupted dataset to my model.

So I’ll repeat the data ingestion methods but this time I am including 2 pdf files that is all the changes I have done.