Azure Language Service - Question Answering

In [1]:
```
# Importing Library to view image
from IPython.display import Image as img
```

Azure Language Services – Overview

- Azure offers powerful Language Services as part of Azure Cognitive Services and Developer Tools, serving two major areas:

Natural Language Processing (NLP) Services

These services are designed to help applications understand and process human language.

- **Conversational Language Understanding (CLU) :** Detects intents and entities from user utterances (e.g., chatbot inputs).Ideal for building intelligent assistants and domain-specific NLP models.
- **Text Analytics:** Performs sentiment analysis, key phrase extraction, language detection, and named entity recognition (NER).
- **Question Answering :** Builds Q&A systems using knowledge bases (FAQs, documents).
- **Text Translation :** Automatically translates text between languages using Microsoft Translator.
- **Text Summarization :** Extracts summaries from larger documents, useful in news, legal, and business content.
- **Language Detection :**Identifies the language of a given text input.
- **Speech-to-Text and Text-to-Speech:** (integrated with Azure Speech Service) Converts spoken language to written text and vice versa.

Programming Language–Related Services and Tools

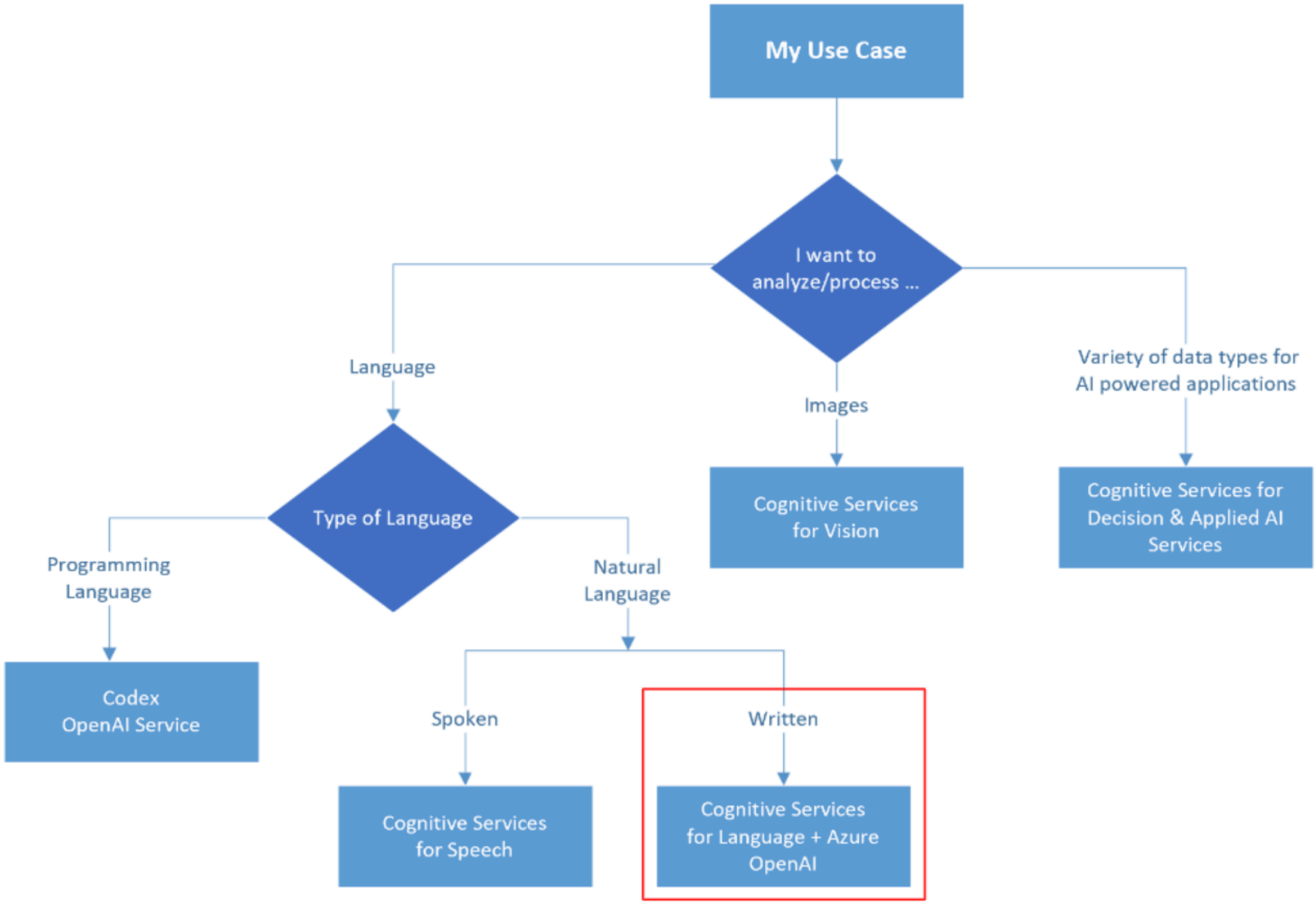
While not part of "Azure Language Services" specifically, Azure provides tools and integrations to support code understanding, analysis, and development:

- **GitHub Copilot (Powered by Azure OpenAI Service)** AI pair programmer that helps write, complete, and understand code in editors like VS Code.

- **Azure OpenAI Service :** Gives access to models like GPT-4, which can assist in code generation, debugging, documentation, and more. Supports both natural language and programming language use cases.
- **Azure DevOps & Visual Studio Code Extensions :** Tools for code collaboration, CI/CD, and AI-assisted development using extensions.

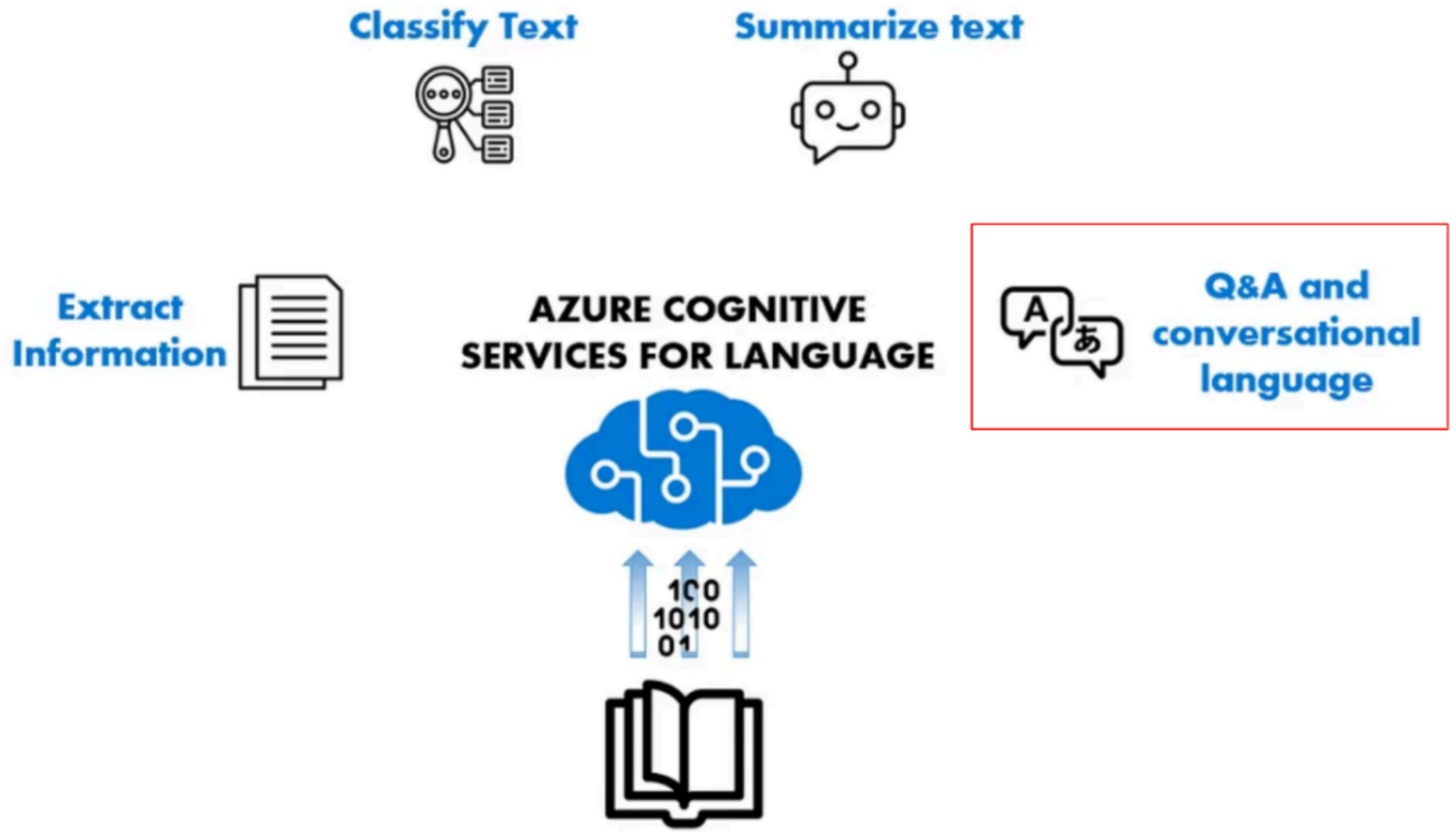
In [2]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\Swatimam hands on\Part 1\part1_pics\theory1.png")
```

Out[2]:



In [3]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\Swatimam hands on\Part 1\part1_pics\theory2.png" )
```

Out[3]:



Prerequisites:

Step 1

1. Installation of neccessary Library

In [4]:
```
# https://learn.microsoft.com/en-us/python/api/overview/azure/ai-language-questionanswering-readme?view=azure-python
```

In [2]:
```
# Install the azure question answering library
!pip install azure-ai-language-questionanswering

# doenv library
!pip install python-dotenv
```

```
Requirement already satisfied: azure-ai-language-questionanswering in c:\users\chirag pc\anaconda3\lib\site-packages (1.1.0)
Requirement already satisfied: azure-core<2.0.0,>=1.24.0 in c:\users\chirag pc\anaconda3\lib\site-packages (from azure-ai-langu
age-questionanswering) (1.32.0)
Requirement already satisfied: isodate<1.0.0,>=0.6.1 in c:\users\chirag pc\anaconda3\lib\site-packages (from azure-ai-language-
questionanswering) (0.7.2)
Requirement already satisfied: requests>=2.21.0 in c:\users\chirag pc\anaconda3\lib\site-packages (from azure-core<2.0.0,>=1.2
4.0->azure-ai-language-questionanswering) (2.32.3)
Requirement already satisfied: six>=1.11.0 in c:\users\chirag pc\anaconda3\lib\site-packages (from azure-core<2.0.0,>=1.24.0->a
zure-ai-language-questionanswering) (1.16.0)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\chirag pc\anaconda3\lib\site-packages (from azure-core<2.0.
0,>=1.24.0->azure-ai-language-questionanswering) (4.9.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\chirag pc\anaconda3\lib\site-packages (from requests>=2.21.
0->azure-core<2.0.0,>=1.24.0->azure-ai-language-questionanswering) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\chirag pc\anaconda3\lib\site-packages (from requests>=2.21.0->azure-cor
e<2.0.0,>=1.24.0->azure-ai-language-questionanswering) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\chirag pc\anaconda3\lib\site-packages (from requests>=2.21.0->azu
re-core<2.0.0,>=1.24.0->azure-ai-language-questionanswering) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\chirag pc\anaconda3\lib\site-packages (from requests>=2.21.0->azu
re-core<2.0.0,>=1.24.0->azure-ai-language-questionanswering) (2024.2.2)
Requirement already satisfied: python-dotenv in c:\users\chirag pc\anaconda3\lib\site-packages (0.21.0)
```

Step 2

2. Import the required library

In [3]:
```
# This section loads your Azure endpoint and subscription key from a .env file and prepares for secure API calls.
```

In [21]:
```
# Read secret keys and environment variables securely
import os
from dotenv import load_dotenv

# Load variables from .env file
load_dotenv()

# Azure authentication helper
from azure.core.credentials import AzureKeyCredential
```

In [5]:
```
# The AuthoringClient helps build and manage the Q&A project, while the QuestionAnsweringClient is used for querying that proj
```

In [23]:
```
# Authoring client – used to create and manage knowledge base projects
from azure.ai.language.questionanswering.authoring import AuthoringClient

# Runtime client – used to query your knowledge base at runtime
from azure.ai.language.questionanswering import QuestionAnsweringClient

# Import QnA models for structured interactions (e.g., QueryOptions, Answers)
from azure.ai.language.questionanswering import models as qna
```

Step 3:

Go to your resource group in Azure Portal https://portal.azure.com/

> ## Create new language service resource.
>
> ## Select both options for your language resource:

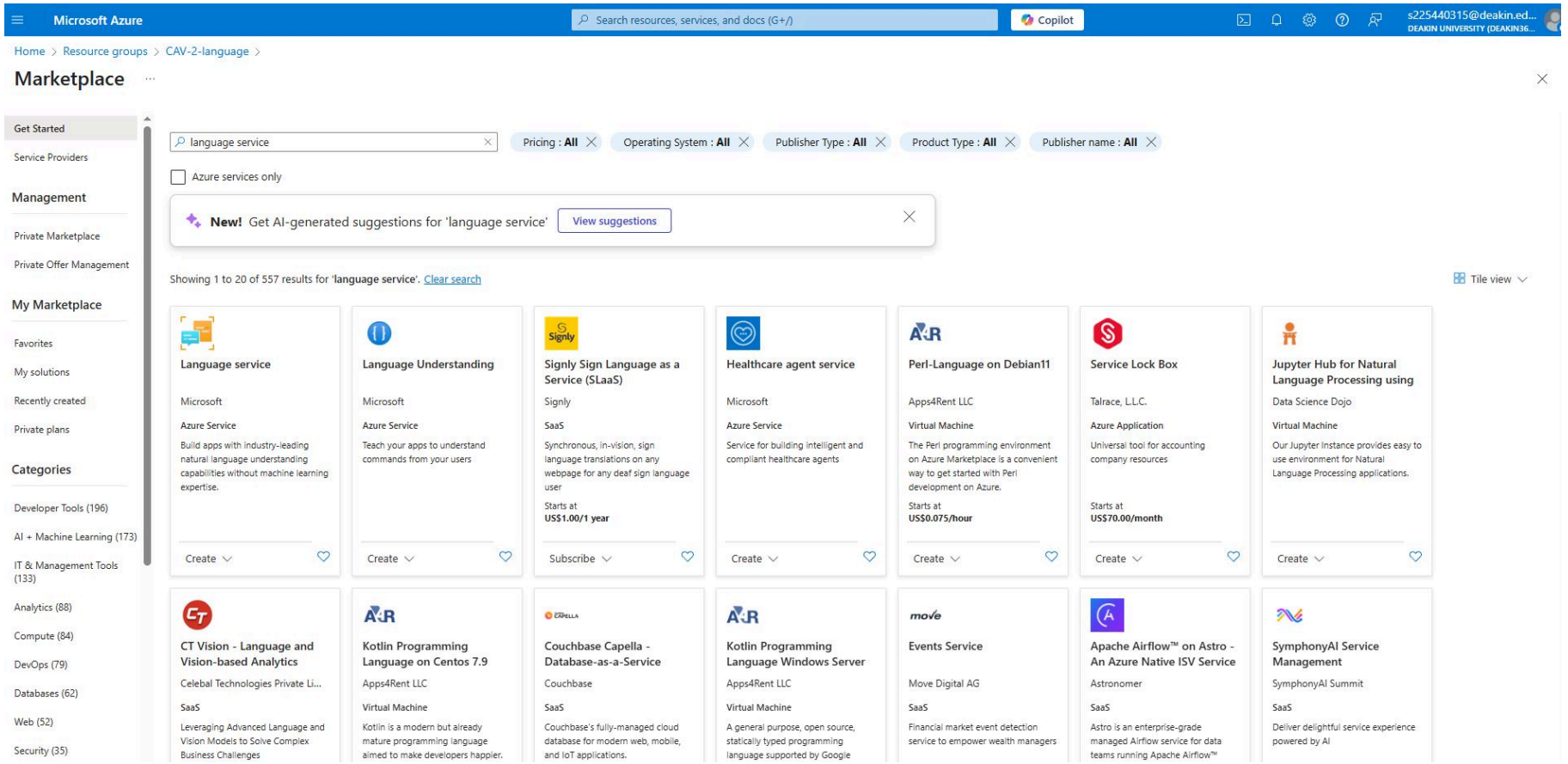
1. Custom Question Answering
2. Custom text classification etc.

Note:

1. To execute the Q&A maker, we just need the Custom Question answering service. However, as we will continue this case study in the next part to create a Conversational Language Understanding model, we are selecting both the options.
2. If you have an existing Language resource in the Free Tier (irrespective of the features selected), you cannot create a new resource in the same tier.

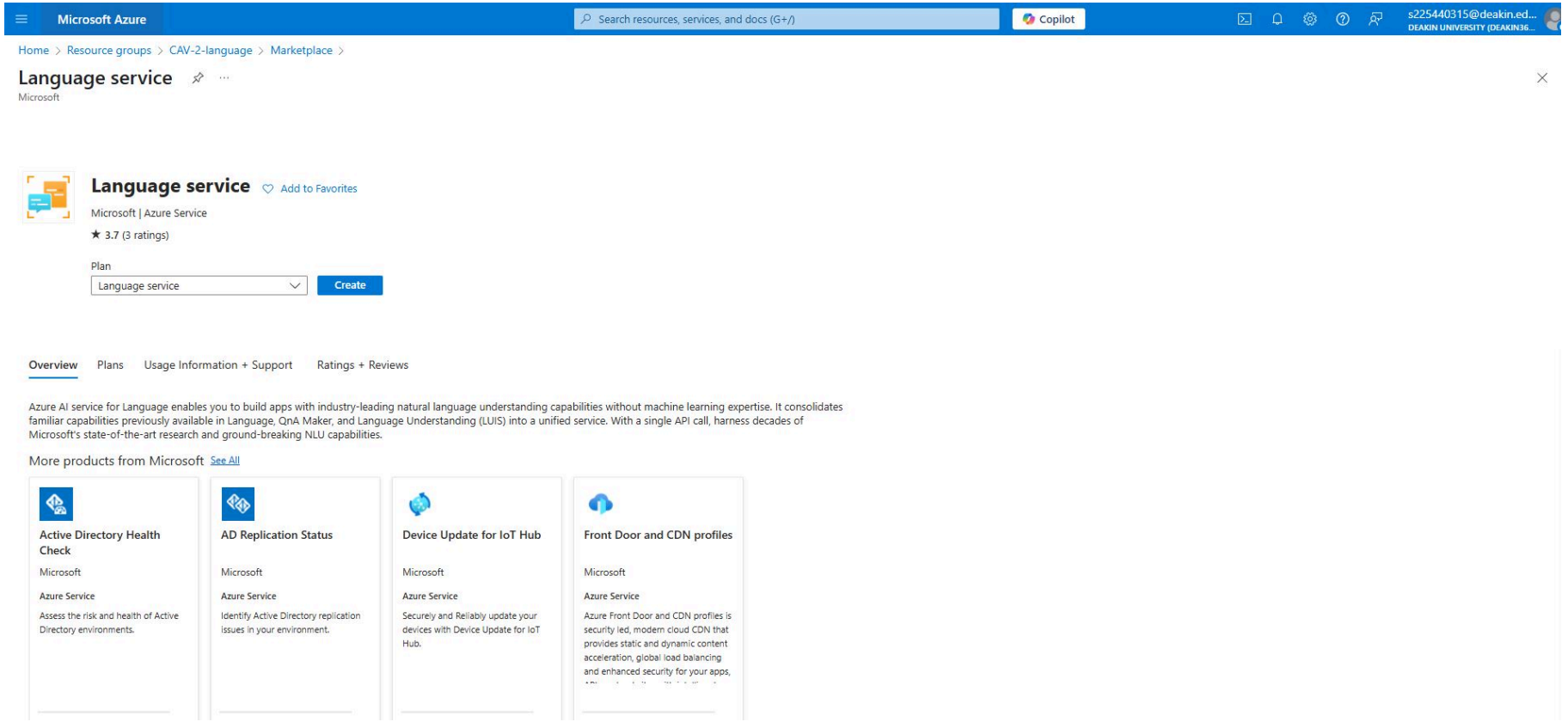
In [7]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\10.JPG")
```

Out[7]:



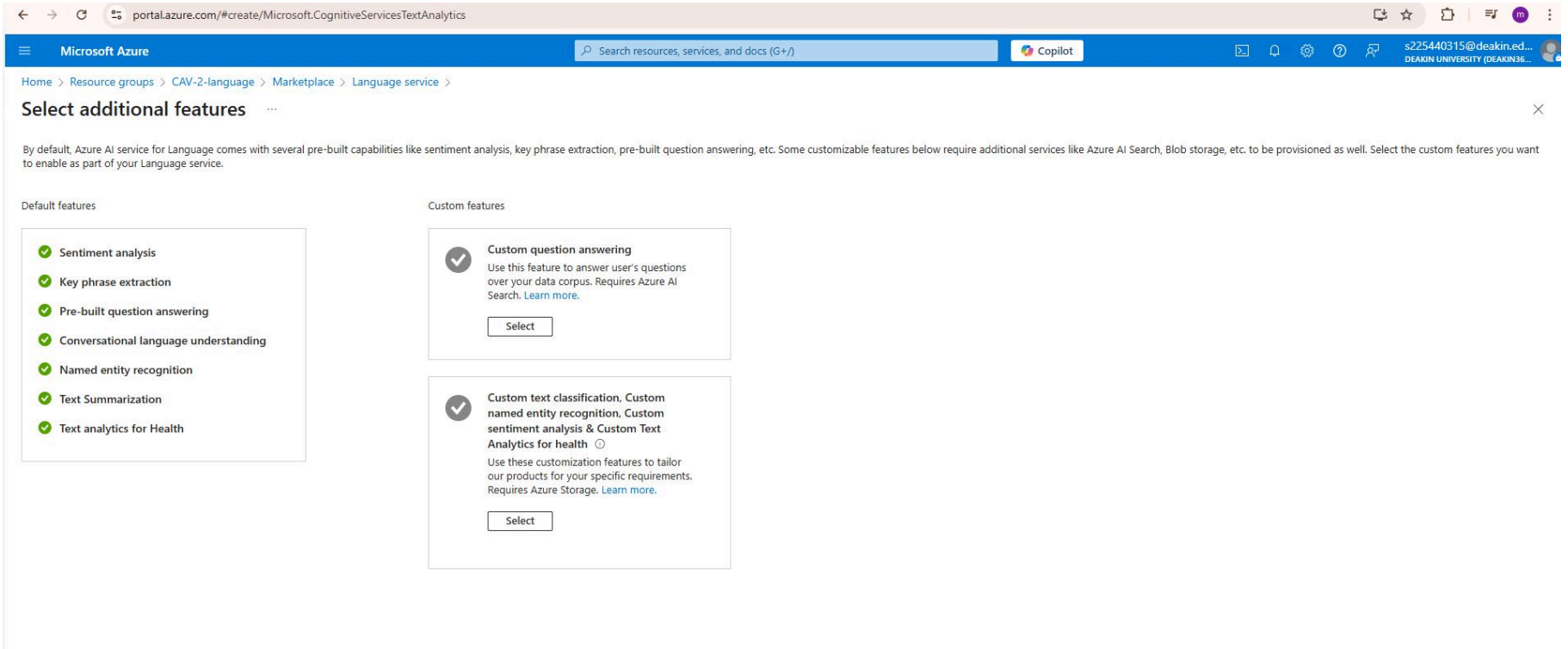
In [11]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\11.JPG")
```

Out[11]:



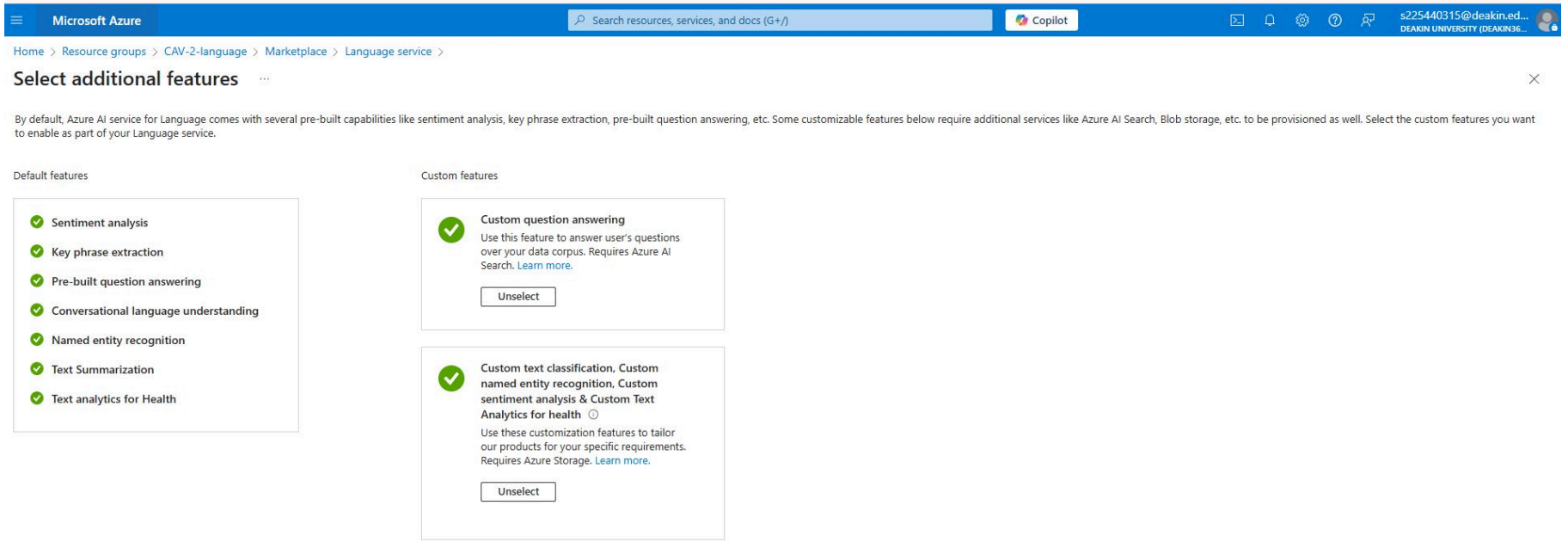
In [12]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\12.JPG")
```

Out[12]:



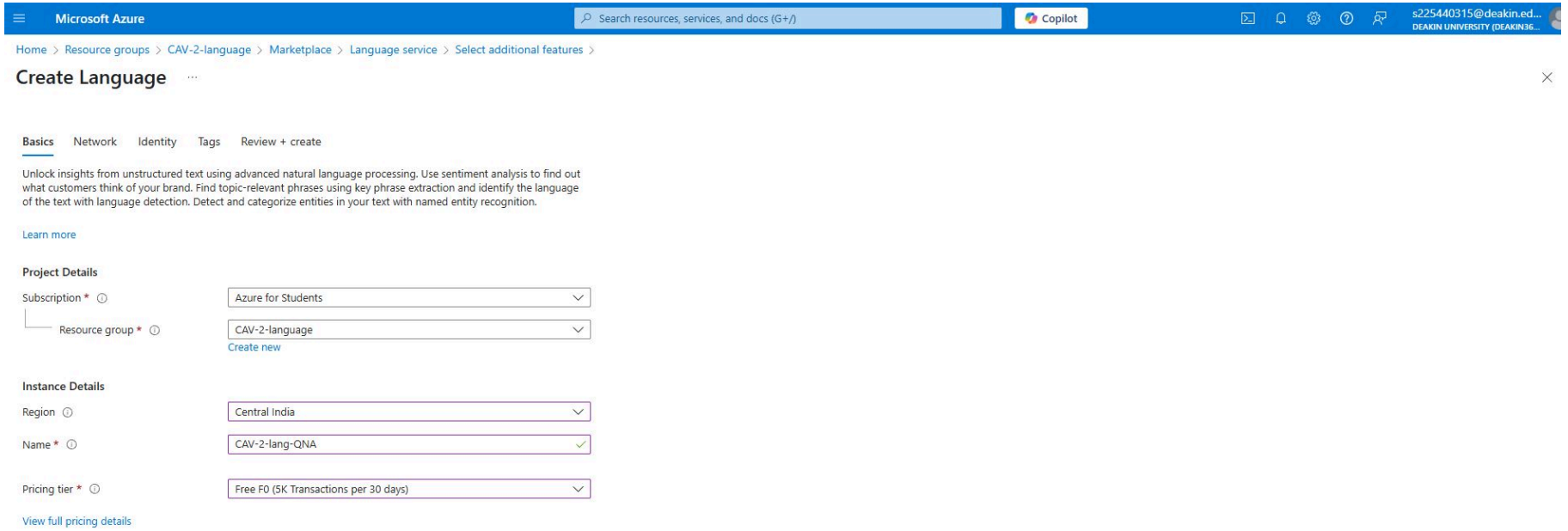
In [13]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\13.JPG")
```

Out[13]:



In [14]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\14.JPG")
```

Out[14]:



In [15]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\15.JPG")
```

Out[15]: View full Azure search pricing details

Custom text classification, Custom named entity recognition, Custom sentiment analysis & Custom Text Analytics for health

You need to create your own storage when using these customization features to host and upload all your training and labeled data and have it saved in your own Azure subscription. You get to associate only one storage account with one language resource that cannot be changed moving forward.
Learn more

New/Existing storage account *
- (•) New storage account
- ( ) Existing storage account

Storage account name * cavlangstorageqna ✓

Storage account type * Premium LRS

Learn more about storage account types

Responsible AI Notice

Microsoft provides technical documentation regarding the appropriate operation applicable to this Azure AI service that is made available by Microsoft. Customer acknowledges and agrees that they have reviewed this documentation and will use this service in accordance with it.

Responsible Use of AI documentation for Text Analytics for Health

Responsible Use of AI documentation for PII

Responsible Use of AI documentation for Language

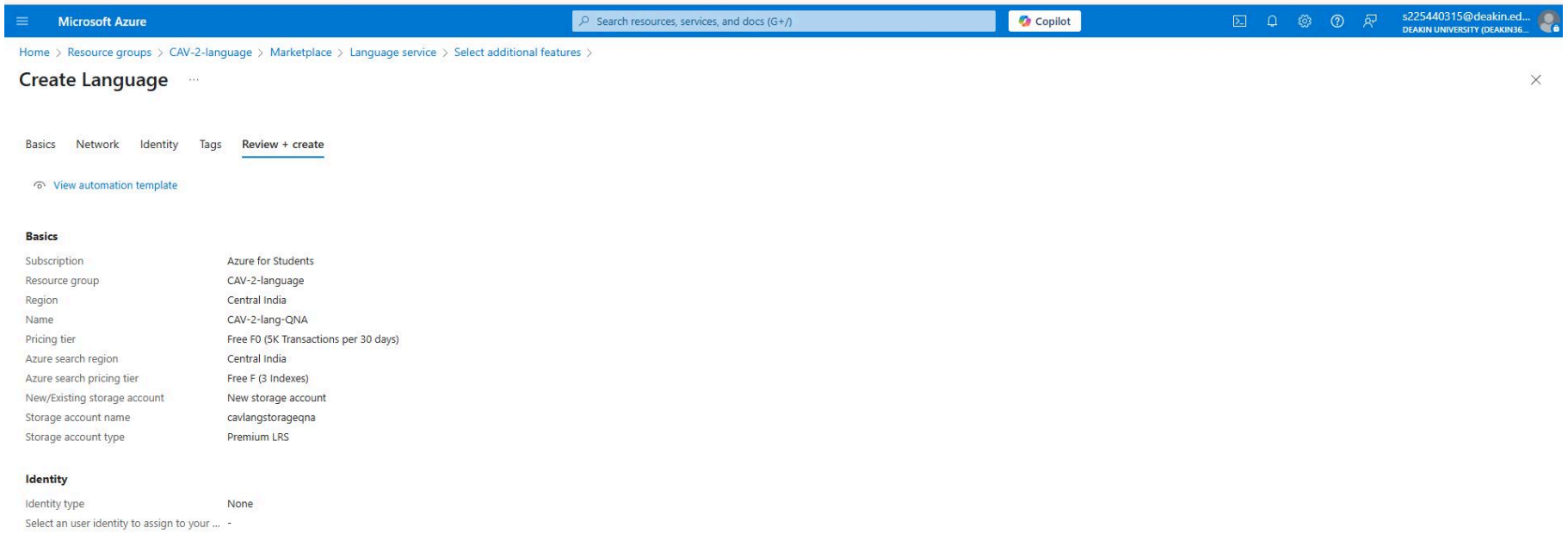
By checking this box I certify that I have reviewed and acknowledge the terms in the Responsible AI Notice. * ☑

Previous Next Review + create

> Please select the free pricing tier for your resource.
>
> Please select the Premium LRS for your storage account.

In [16]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\16.JPG")
```

Out[16]:



After executing the above steps, you will see three resources in your resource group:

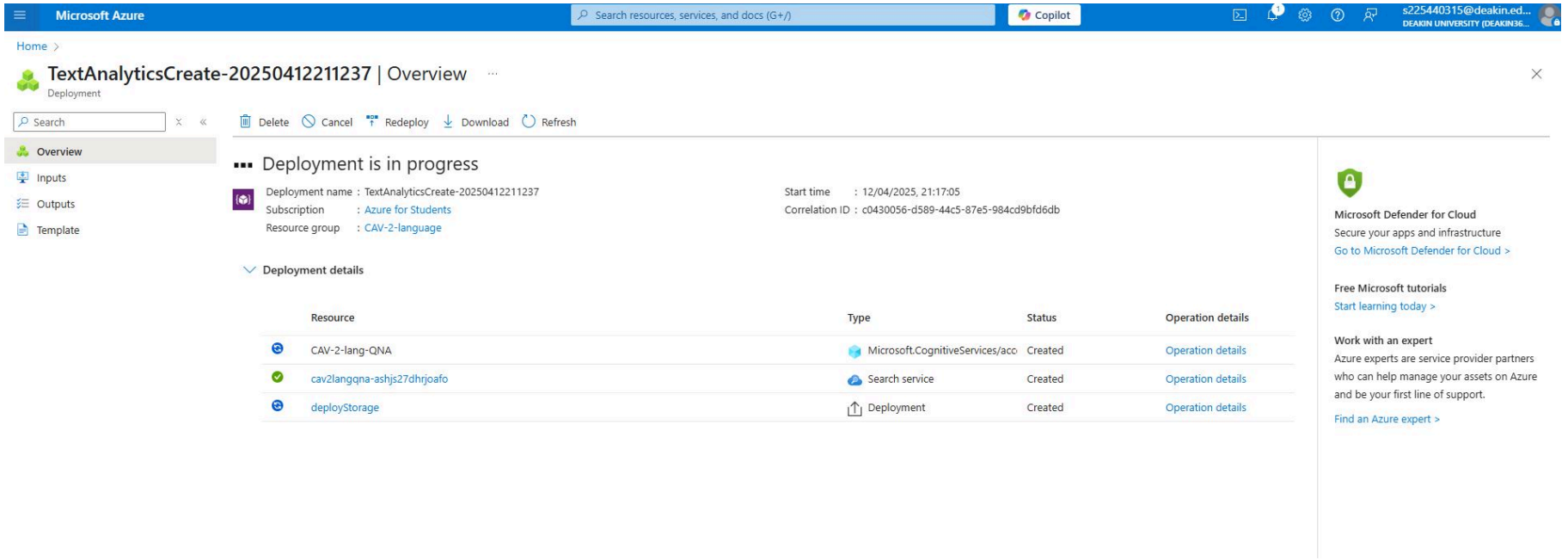
1. Language resource

2. Storage resource

3. Search resource

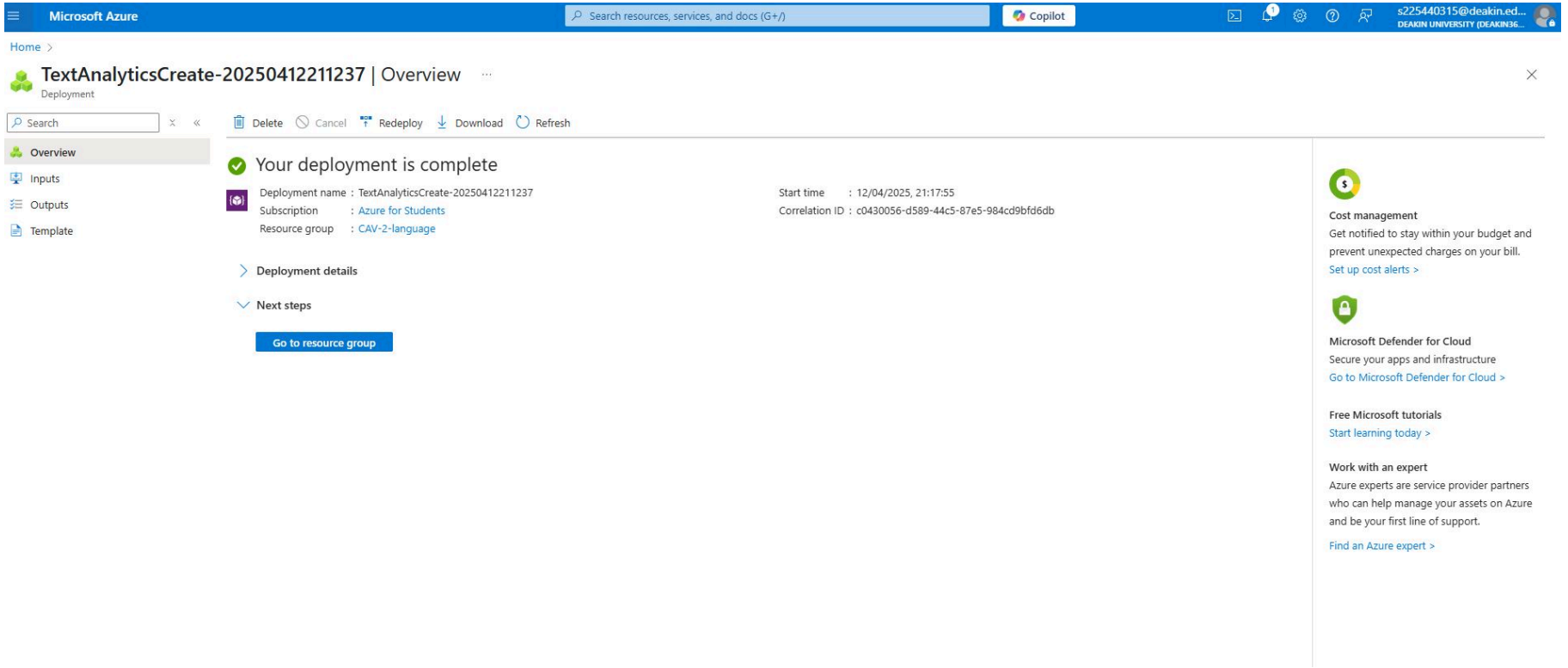
In [17]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\17.JPG")
```

Out[17]:



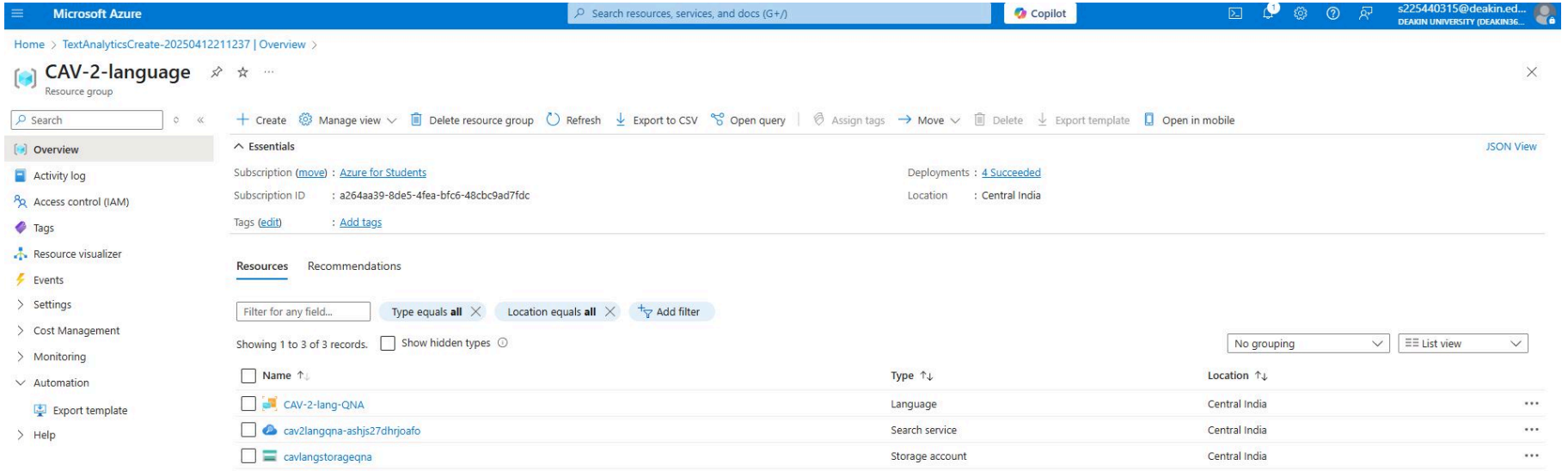
In [18]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\18.JPG")
```

Out[18]:



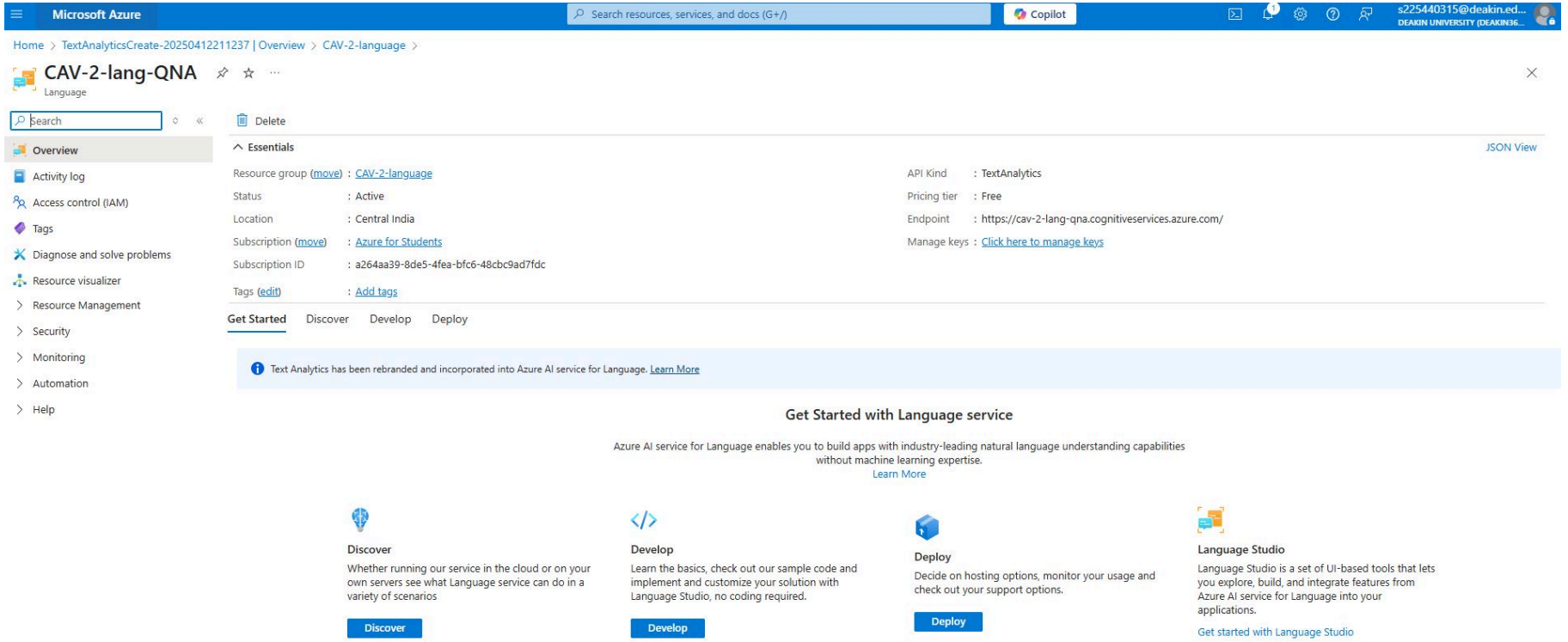
In [19]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\19.JPG")
```

Out[19]:



In [20]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\20.JPG")
```

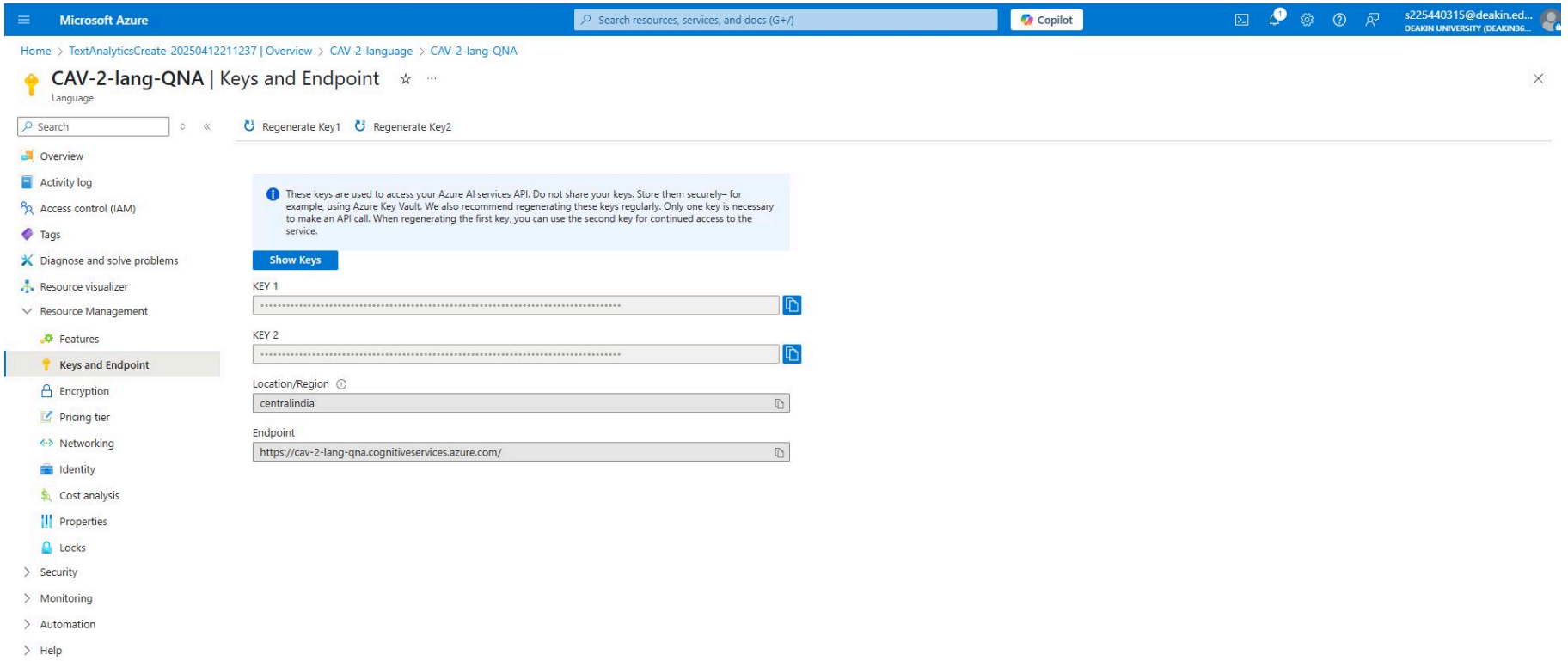
Out[20]:



Copy the subscription key and endpoint. Save them as environment variables, to authenticate and connect to the Language Service client.

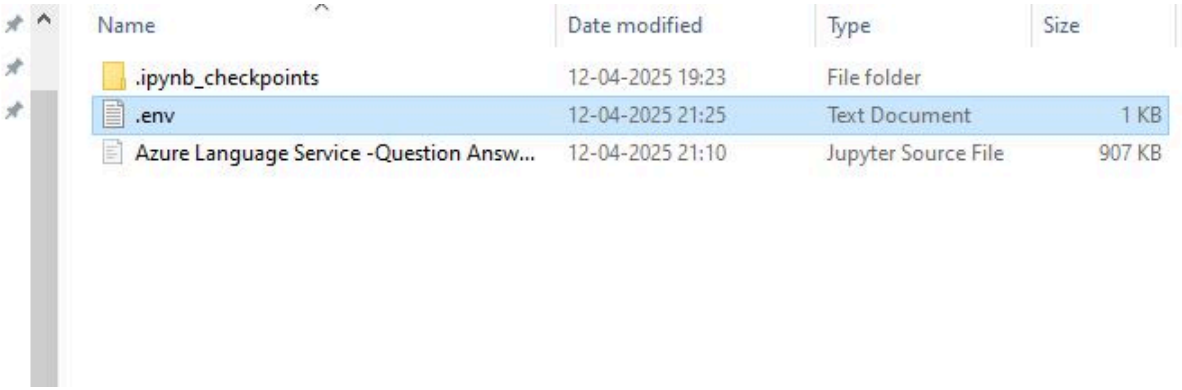
In [21]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\21.JPG")
```

Out[21]:



In [22]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\22.JPG")
```

Out[22]:



How to save the keys as environment variables?

1. Open a new instance of a word editor like Notepad.
2. Enter the names of your variables (subscription_key and endpoint) and save their values as strings.
3. Name the file as .env. Please note, this file will be hidden. To view it, kindly select the option to show hidden files.
4. For reference, here is a screenshot of an env file.
5. Import the file in your IDE, and read the values. For future runs, you will need to update this file with new values for subscription key and endpoint.

Note:

1. We are going to create several clients throughout the course of the case study in this session. We will use the same endpoint and key, and use them to create different clients.

Step 4 Initializing Azure QnA and Authoring Clients with Secure Credentials

In [34]:
```
# get service secrets
load_dotenv(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\Hands On Ipynb\.env.txt")
endpoint = os.environ.get("endpoint")
key = os.environ.get("subscription_key")
prediction=os.environ.get("prediction_url")

# Please note, we will create two clients, one for creating the project (Authoring) and one for querying it (Question Answerin

authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))
qna_client = QuestionAnsweringClient(endpoint, AzureKeyCredential(key))
```

https://olympus.mygreatlearning.com/mentorship_recordings/2669842

Step 5 Case Study

Now that your resource is created, go to the Language Studio https://language.cognitive.azure.com/home

Select your resource to access the studio.

Step 5.1 Creating new project, add knowledge base and deploy it using python sdk

we can also create it using azure language portal

In [4]:
```
# Create new project, add knowldege base and deploy it.

authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))

with authoring_client:

    # Step 1: create project

    print("\n********************* Creating a new project *********************")
    project_name = "RetailBot"
    project = authoring_client.create_project(
        project_name=project_name,
        options={
            "description": "Retail chatbot for assisting customers with product queries",
            "language": "en",
            "multilingualResource": True,
            "settings": {
                "defaultAnswer": "Sorry, I don't have an answer for that right now."
            }
        })

    # Output 1: View the project details

    print("view created project info:")
    print("\tname: {}".format(project["projectName"]))
    print("\tlanguage: {}".format(project["language"]))
```

```
    print("\tdescription: {}".format(project["description"]))
```

```
********************** Creating a new project **********************
view created project info:
        name: RetailBot
        language: en
        description: Retail chatbot for assisting customers with product queries
```

Verify from the azure language studio your app is created there with name as RetailBot

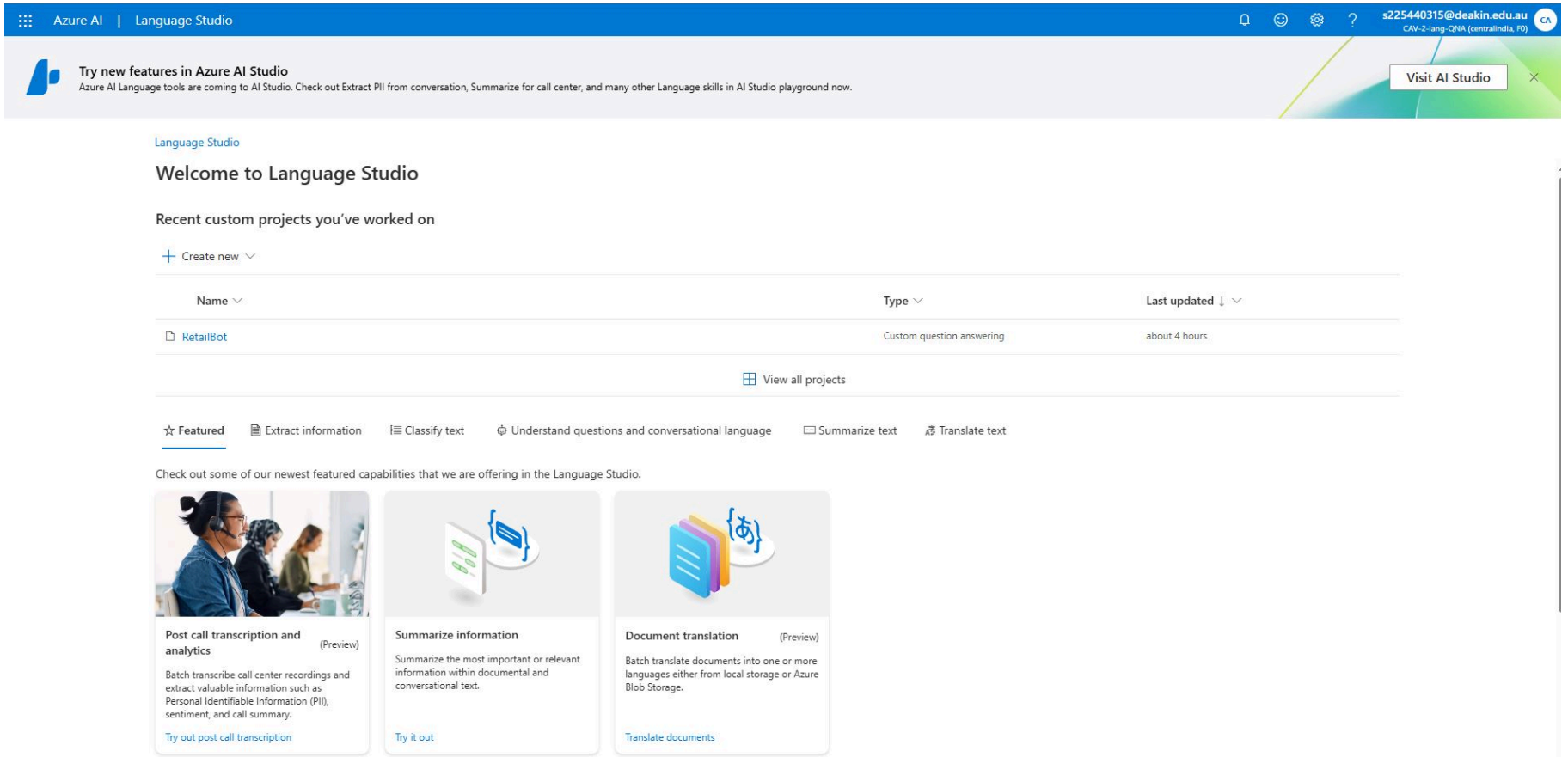
View the project on the Language Studio:

1. Navigate to https://language.cognitive.azure.com/home
2. Before executing the above steps, we can see that the Language Studio had no projects. Now, you can see your project created here.
3. Click on the project title to view the details like source name, url, and whether it is structured or unstructured.
4. Clicking on the source on this page shows you the questions and answers.

We can now see that a project has been created on the Language Studio.

In [28]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\33.JPG")
```

Out[28]:



Step 5.2: Add a knowledge base

> - We are adding two knowledge bases here. A minimum of one knowledge base is required.
> - Please note, the knowledge base cannot have more than 3 sources (urls, files) in Free Tier.
> - In other words, you can add upto three sources/indices in one project, under the free tier.

> ## Now to add knowledge base resource in our project first we have to create it

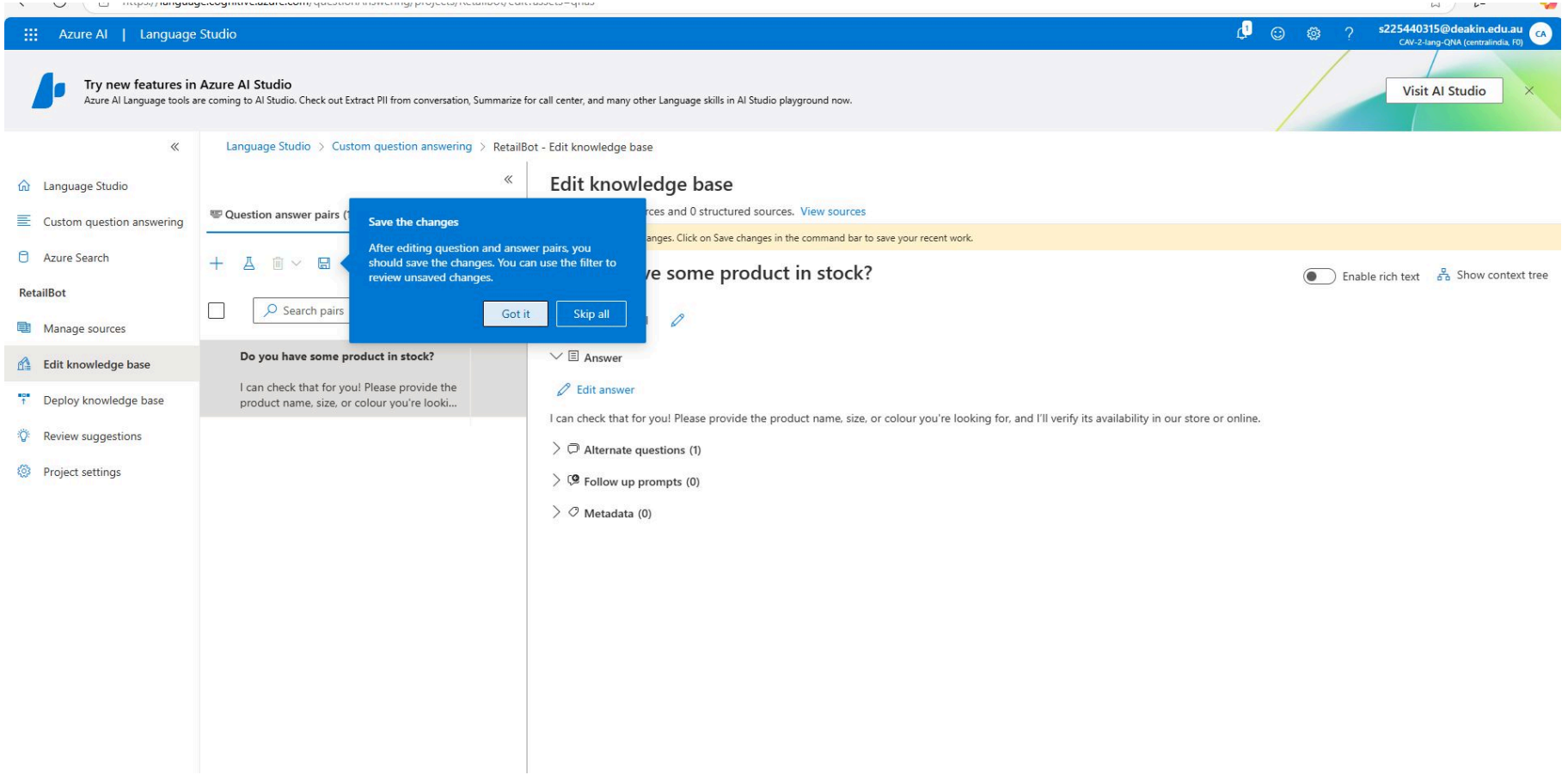
> # There are basically three ways of creating knowledge base
>
> ## Using Azure Language Studio UI
>
> - Manual but visual
> - Add sources by uploading files (TSV, Pdfs,Urls)
> - good for quick edits and test (this is correct and official")
>
> ## Using python SDK (code)
>
> - You generate .tsv files programmatically
> - then upload via portal OR host them and connect via code using sourceUri
> - Yes, this is alos fully supported and great for automation
>
> ## Using Rest API (Direct Http calls)
>
> - This is the third official method:
> - You call the Azure Language Service REST API to create projects, add/update sources, and deploy — without using Python SDK or the portal.
> - Great for use in systems where Python isn't available (e.g., Node.js, bash scripts, Postman, custom backend, etc.)

Now lets see first using Azure language studio

- click on project created on language studio we have just created
- Edit knowledge base
- now adding five custom base question answer as per problem state actually we can create as many as we can

In [52]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\35.JPG")
```

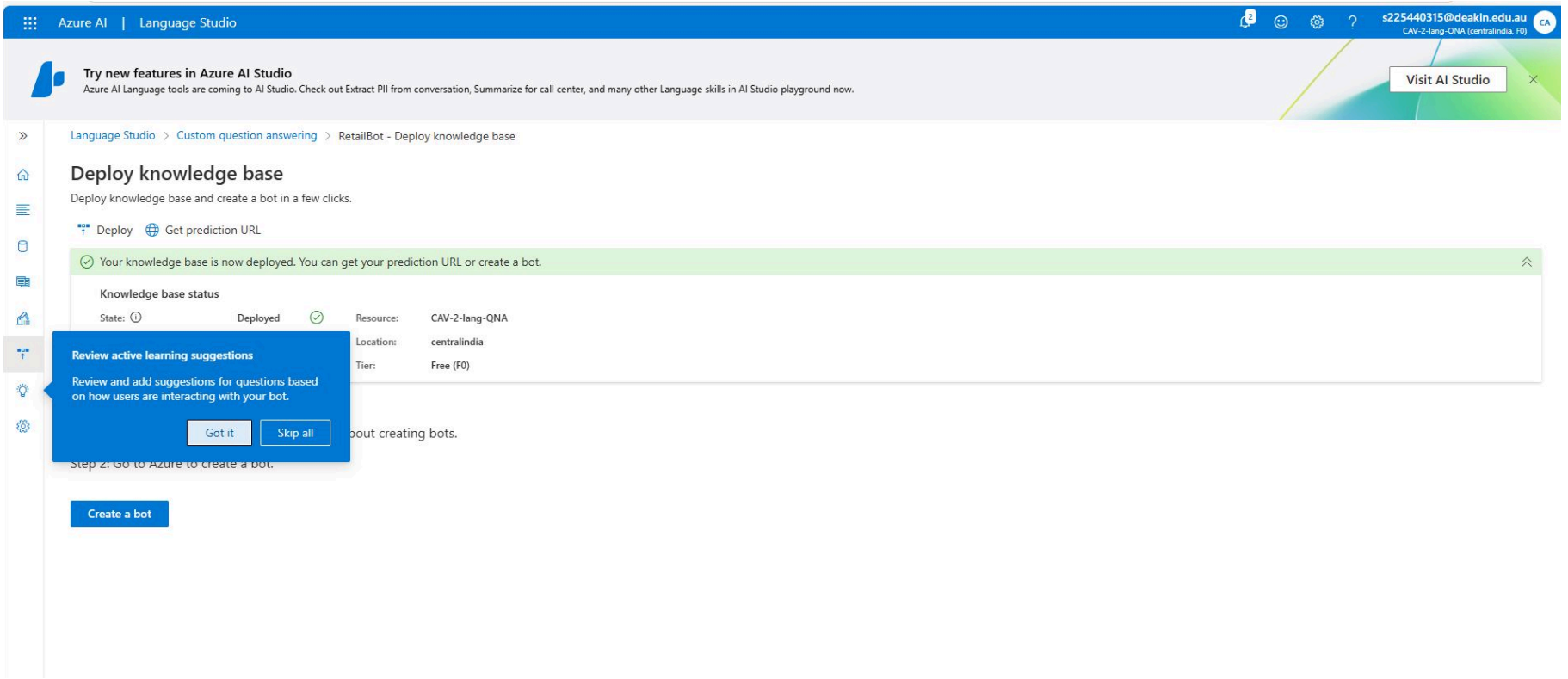
Out[52]:



On clicking on the source of the knowledge base, we can view the corpus of questions and answers. We can select a question from this base to test our program. We can also choose to rephrase the question. Alternatively, we can ask a question that is not present in the corpus, to test how well the answers are generated.

In [53]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\36.JPG")
```

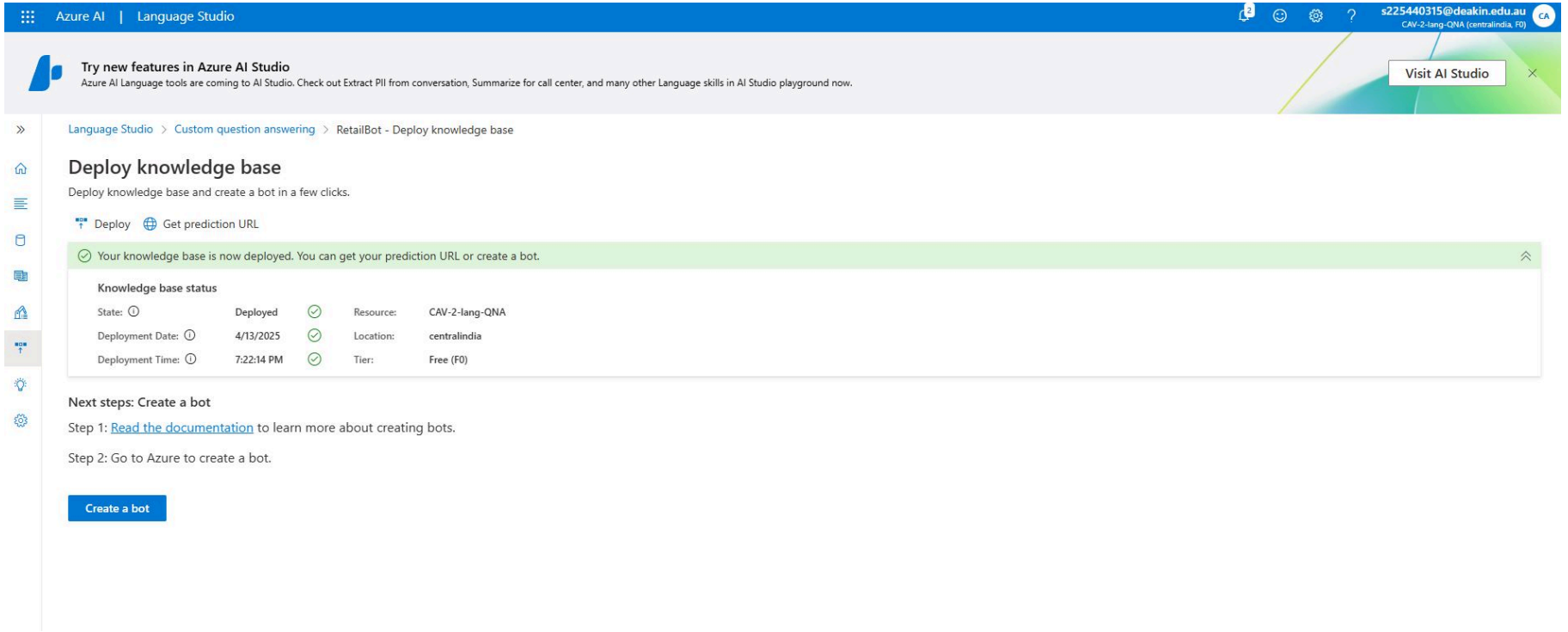
Out[53]:



After your knowledge base is deployed, you can see the deployment details in the Studio. As a next step, you can choose to create a chatbot using this corpus. In the current case study, we will not create a chatbot, but just access the corpus.

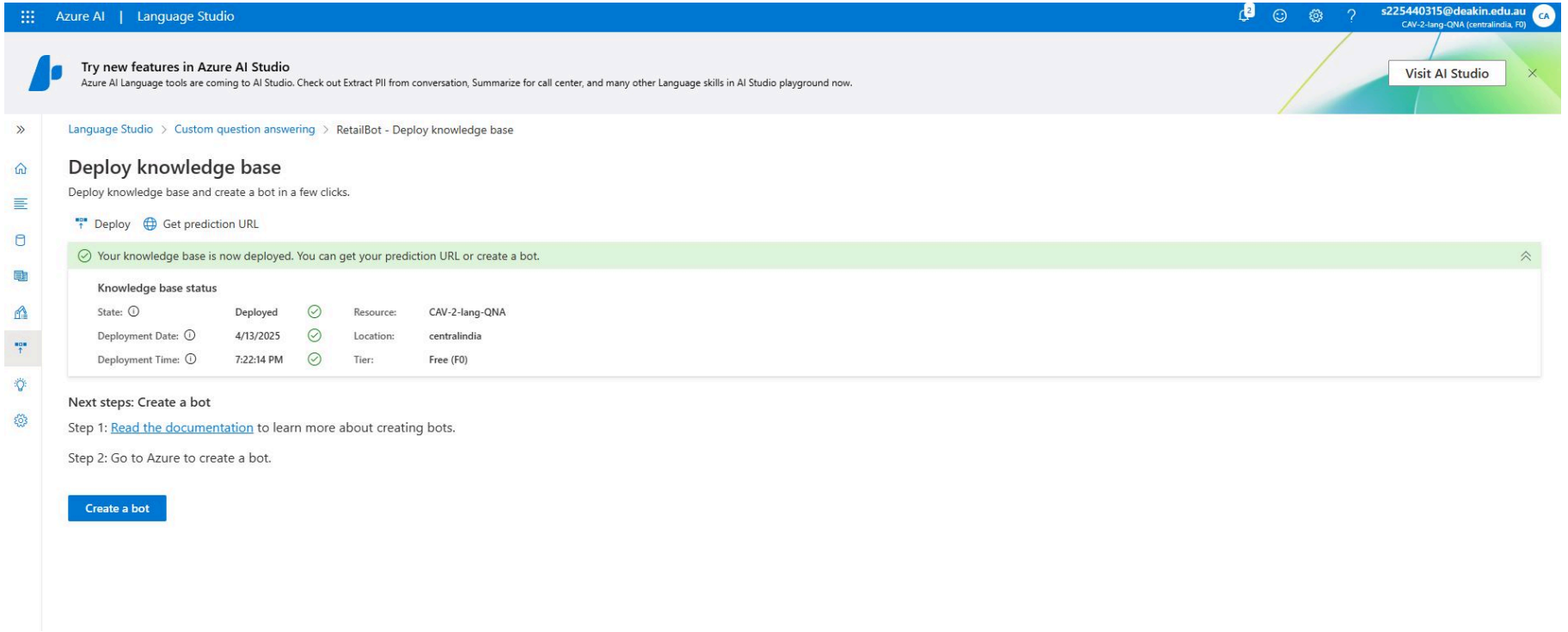
In [54]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\37.JPG")
```

Out[54]:



In [56]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\37.JPG")
```

Out[56]:



In [8]:

```
authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))
with authoring_client:
    # Step 2: Add a knowledge base
    print("\n********************* Adding a knowledge base **********************")

    # We are adding two knowledge bases here. A minimum of one knowledge base is required.
    # Please note, the knowledge base cannot have more than 3 sources (urls, files) in Free Tier.
    # In other words, you can add upto three sources/indices in one project, under the free tier.

    update_sources_poller = authoring_client.begin_update_sources(
        project_name=project_name,
        sources=[
           {
                "op": "add",
                "value": {
                    "displayName": "retail product FAQ-editorial",
                    "sourceKind": "url",
                    "sourceUri": " https://cav-2-lang-qna.cognitiveservices.azure.com/language/:query-knowledgebases?projectNa
```

```
                }
            }
        ]
    )
    update_sources_poller.result()

    # Output 2: list sources
    print("\nlist project sources")
    sources = authoring_client.list_sources(
        project_name=project_name
    )
    for source in sources:
         print("Knowledge base name: {}".format(source.get("displayName")))
         print("\tSource: {}".format(source.get("source")))
         print("\tSource kind: {}".format(source.get("sourceKind")))
         print("\tSource Uri: {}".format(source.get("sourceUri", "N/A")))
```

```
********** Adding a knowledge base **********

list project sources
Knowledge base name: productFAQ
        Source: Editorial
        Source kind: file
        Source Uri: N/A
Knowledge base name: retailQna
        Source: converted_qna.tsv
        Source kind: file
        Source Uri: https://qsincame.blob.core.windows.net/portal-files/356a6632-ed32-a48b-e290-5f388afa60a2-converted_qna.tsv
Knowledge base name: retail product FAQ-editorial
        Source: https://cav-2-lang-qna.cognitiveservices.azure.com/language/:query-knowledgebases?projectName=RetailBot&api-ver
sion=2021-10-01&deploymentName=production
        Source kind: url
        Source Uri: https://cav-2-lang-qna.cognitiveservices.azure.com/language/:query-knowledgebases?projectName=RetailBot&api
-version=2021-10-01&deploymentName=production
```

In [58]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\38.JPG")
```

Out[58]:

Step 5.3 deploy the project

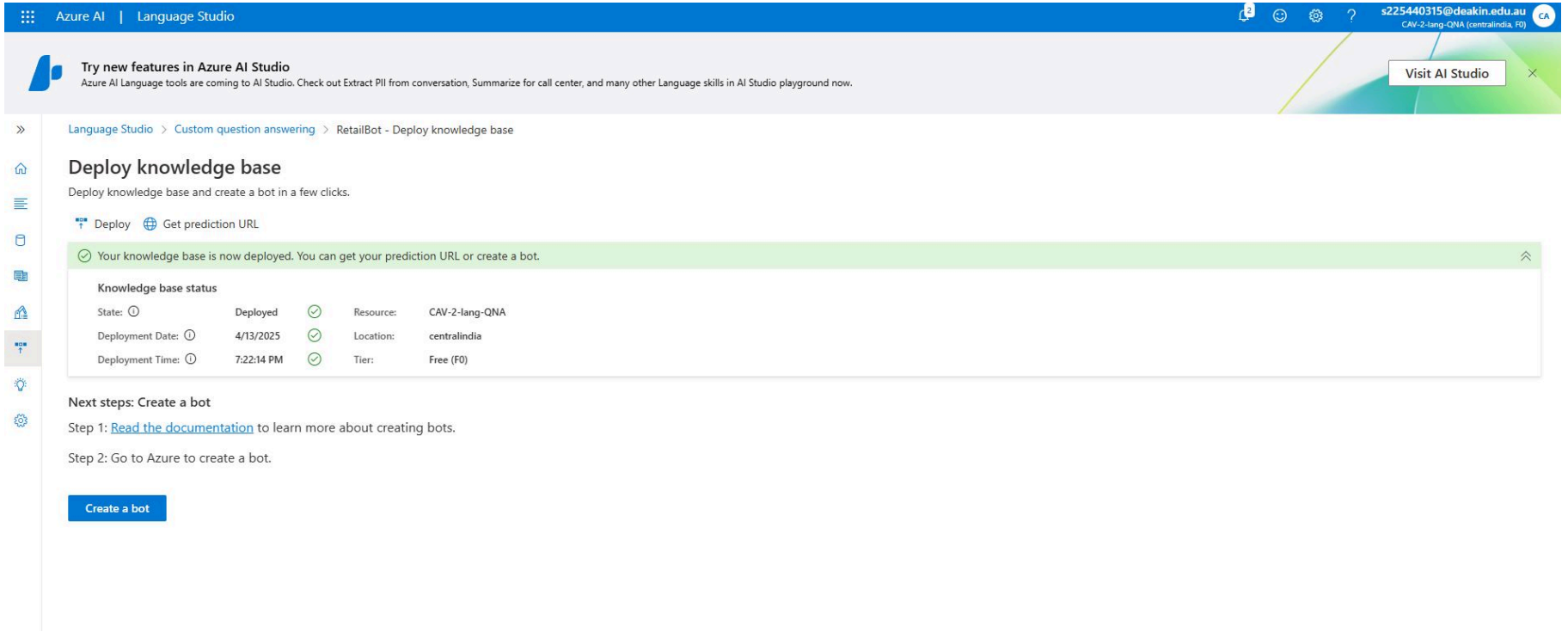
In [9]:

```
authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))
with authoring_client:
   # Step 3: deploy the project

    print("\n********************* Deploying the project **********************")

    deployment_poller = authoring_client.begin_deploy_project(
        project_name=project_name,
        deployment_name="production"
    )
    deployment_poller.result()

    # list all deployments
    deployments = authoring_client.list_deployments(
        project_name=project_name
    )

    print("view project deployments")
    for d in deployments:
        print(d)
```

```
********************* Deploying the project **********************
view project deployments
{'deploymentName': 'production', 'lastDeployedDateTime': '2025-04-15T13:10:27Z'}
```

In [61]:
```
img(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\screenshots of procedure\37.JPG")
```

Out[61]:



Testing the project:

1. We can test by asking just one question.

2. We can initiate a chit-chat with the project, in which you can ask additional questions. This method will be useful when you want to create a chatbot.

1. Test by asking one question:

In [11]:
```
# 1. Test by asking a question
qna_client = QuestionAnsweringClient(endpoint, AzureKeyCredential(key))
```

```
with qna_client:
    question= "What are your store hours?"
    output = qna_client.get_answers(
        question=question,
        top=3,
        confidence_threshold=0.5,
        include_unstructured_sources=True,
        short_answer_options=qna.ShortAnswerOptions(
            confidence_threshold=0.5,
            top=1
        ),
        project_name="RetailBot",
        deployment_name="test"
    )
    if output.answers:
        best_candidate = [a for a in output.answers if a.confidence and a.confidence > 0.5][0]
        print("Q: {}".format(question))
        print("A: {}".format(best_candidate.answer))
    else:
        print(f"No answers returned from question '{question}'")
```

```
Q: What are your store hours?
A: Our store is open from 9 AM to 9 PM, Monday to Saturday.
```

2. Test by creating a chit-chat:

In [12]:
```
# 2. Testing by initiating a chit-chat.

qna_client = QuestionAnsweringClient(endpoint, AzureKeyCredential(key))
with qna_client:
    first_question= "What is your return policy of that?"

    output = qna_client.get_answers(
        question=first_question,
        top=3,
        confidence_threshold=0.5,
        include_unstructured_sources=True,
        short_answer_options=qna.ShortAnswerOptions(
            confidence_threshold=0.5,
```

```
            top=1
        ),
        project_name="RetailBot",
        deployment_name="test"
    )
    if output.answers:
        best_candidate = [a for a in output.answers if a.confidence and a.confidence > 0.2][0]
        print(u"Q: {}".format(first_question))
        print(u"A: {}".format(best_candidate.answer))
        print('\n***************************************************************************\n')
    else:
        print(f"No answers returned from question '{first_question}'")


    if best_candidate.qna_id:
        followup_question = "Where is my order? Can I track it?"

        output = qna_client.get_answers(
            question=followup_question,
            top=3,
            confidence_threshold=0.5,
            answer_context=qna.KnowledgeBaseAnswerContext(
                previous_question=first_question,
                previous_qna_id=best_candidate.qna_id
            ),
            short_answer_options=qna.ShortAnswerOptions(
                confidence_threshold=0.5,
                top=1
            ),
            include_unstructured_sources=True,
            project_name="RetailBot",
            deployment_name="test"
        )
        if output.answers:
            print(u"Q: {}".format(followup_question))
            print(u"A: {}".format(output.answers[0].answer))
            print('\n***************************************************************************\n')
        else:
            print(f"No answers returned from question '{followup_question}'")
```

```
Q: What is your return policy of that?
A: You can return items within 30 days of purchase, provided they are unused and in original packaging.

********************************************************************************

Q: Where is my order? Can I track it?
A: You can track your order by visiting the 'My Orders' section on our website.

********************************************************************************
```

Closing the project - resource deletion

Please note, this notebook a part of a two-part series. If you wish to continue with the second part immediately, we would suggest that you keep the resource. If you intend to continue the second part at a later time, please proceed with the step of resource deletion now.

In [5]:

```python
import json
import pandas as pd

# Load your JSON file
with open(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\retail_qna.json", "r", encoding="utf-8") as file:
    data = json.load(file)

qna_list = data.get("qnaList", [])

# Use a set to track unique rows
unique_rows = set()
rows = []

for item in qna_list:
    id_ = item.get("id")
    answer = item.get("answer", "")
    category = item.get("metadata", [{}])[0].get("value", "")
    followups = " | ".join(item.get("followUpQuestions", []))
    source = item.get("source", "")
```

```
    for question in item.get("questions", []):
        row_tuple = (id_, question, answer, category, followups, source)
        if row_tuple not in unique_rows:
            unique_rows.add(row_tuple)
            rows.append({
                "ID": id_,
                "Question": question,
                "Answer": answer,
                "Category": category,
                "FollowUpQuestions": followups,
                "Source": source
            })

# Convert to DataFrame
df = pd.DataFrame(rows)

# Save to TSV
df.to_csv(r"E:\2. MDS DEAKIN UNIVERSITY\TASK\Task6\converted_qna.tsv", sep="\t", index=False)

print("TSV file created without duplicates.")
```

```
TSV file created without duplicates.
```

Again adding that .tsv file in project and perfoming deploy project

In [87]:

```
authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))
with authoring_client:
    # Step 2: Add a knowledge base
    print("\n********************* Adding a knowledge base *********************")

    # We are adding two knowledge bases here. A minimum of one knowledge base is required.
    # Please note, the knowledge base cannot have more than 3 sources (urls, files) in Free Tier.
    # In other words, you can add upto three sources/indices in one project, under the free tier.

    update_sources_poller = authoring_client.begin_update_sources(
        project_name=project_name,
        sources=[
           # {
           #      "op": "add",
```

```
        #       "value": {
        #           "displayName": "retail product FAQ-editorial",
        #           "sourceKind": "url",
        #           "sourceUri": "https://cav-2-lang-qna.cognitiveservices.azure.com/language/:query-knowledgebases?projectN
        #       }
        #   },
        # {
        #       "op": "add",
        #       "value": {
        #           "displayName": "retail product FAQ-editorial",
        #           "sourceKind": "url",
        #           "sourceUri": " " # <- add this empty string to avoid the BadArgument error
        #       }

        # }

    ]
)
update_sources_poller.result()

# Output 2: list sources
print("\nlist project sources")
sources = authoring_client.list_sources(
    project_name=project_name
)
for source in sources:
     print("Knowledge base name: {}".format(source.get("displayName")))
     print("\tSource: {}".format(source.get("source")))
     print("\tSource kind: {}".format(source.get("sourceKind")))
     print("\tSource Uri: {}".format(source.get("sourceUri", "N/A")))
```

```
********************* Adding a knowledge base *********************

list project sources
Knowledge base name: retailQna
        Source: converted_qna.tsv
        Source kind: file
        Source Uri: https://qsincame.blob.core.windows.net/portal-files/356a6632-ed32-a48b-e290-5f388afa60a2-converted_qna.tsv
Knowledge base name:
        Source: qna_chitchat_Caring
        Source kind: file
        Source Uri: N/A
Knowledge base name: zalando
        Source: https://www.zalando.co.uk/faq/Returns-and-Refunds
        Source kind: url
        Source Uri: https://www.zalando.co.uk/faq/Returns-and-Refunds
```

In [88]:

```python
authoring_client = AuthoringClient(endpoint, AzureKeyCredential(key))
with authoring_client:
   # Step 3: deploy the project

    print("\n********************* Deploying the project **********************")

    deployment_poller = authoring_client.begin_deploy_project(
        project_name=project_name,
        deployment_name="production"
    )
    deployment_poller.result()

    # list all deployments
    deployments = authoring_client.list_deployments(
        project_name=project_name
    )

    print("view project deployments")
    for d in deployments:
        print(d)
```

```
********************* Deploying the project **********************
view project deployments
{'deploymentName': 'production', 'lastDeployedDateTime': '2025-04-15T16:56:40Z'}
```

Testing by new question from updated knowledge base

In [90]:
```
# 1. Test by asking a question
qna_client = QuestionAnsweringClient(endpoint, AzureKeyCredential(key))

with qna_client:
    question= "what if my item is damaged or defective?"
    output = qna_client.get_answers(
        question=question,
        top=5,
        confidence_threshold=0.5,
        include_unstructured_sources=True,
        short_answer_options=qna.ShortAnswerOptions(
            confidence_threshold=0.5,
            top=1
        ),
        project_name="RetailBot",
        deployment_name="test"
    )
    if output.answers:
        best_candidate = [a for a in output.answers if a.confidence and a.confidence > 0.2][0]
        print("Q: {}".format(question))
        print("A: {}".format(best_candidate.answer))
    else:
        print(f"No answers returned from question '{question}'")
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[90], line 19
      6 output = qna_client.get_answers(
      7     question=question,
      8     top=5,
   (...)
     16     deployment_name="test"
     17 )
     18 if output.answers:
---> 19     best_candidate = [a for a in output.answers if a.confidence and a.confidence > 0.2][0]
     20     print("Q: {}".format(question))
     21     print("A: {}".format(best_candidate.answer))

IndexError: list index out of range
```

In [96]:

```python
qna_client = QuestionAnsweringClient(endpoint, AzureKeyCredential(key))

with qna_client:
    question = "What should I do if my item is damaged?"
    output = qna_client.get_answers(
        question=question,
        top=3,
        confidence_threshold=0.8,
        include_unstructured_sources=True,
        short_answer_options=qna.ShortAnswerOptions(
            confidence_threshold=0.8,
            top=1
        ),
        project_name="RetailBot",
        deployment_name="test"
    )

    if output.answers:
        filtered = [a for a in output.answers if a.confidence and a.confidence > 0.5]
        if filtered:
            best_candidate = filtered[0]
            print("Q: {}".format(question))
            print("A: {}".format(best_candidate.answer))
        else:
```

```
            print(f"No answers passed confidence threshold for question '{question}'")
    else:
        print(f"No answers returned from question '{question}'")
```

```
No answers passed confidence threshold for question 'What should I do if my item is damaged?'
```

In [5]:

```
import requests
import pandas as pd
from sklearn.metrics import precision_score, recall_score, f1_score, confusion_matrix, classification_report
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder

# Step 1: Test dataset (you should replace/add more real QnA pairs here)
test_data = [
    {"question": "What is your return policy?", "expected_answer": "You can return the product within 30 days."},
    {"question": "Where is my order?", "expected_answer": "You can track your order using the tracking ID sent to your email."
    {"question": "Suggest a good phone", "expected_answer": "We recommend the latest smartphones from Samsung and Apple."},
    {"question": "Tell me about shipping charges", "expected_answer": "Shipping is free for orders above $50."},
    {"question": "I want details of this product", "expected_answer": "This product includes a 1-year warranty and free access
]

# Step 2: Prediction config
prediction_url = "https://cav-cv-resource1.cognitiveservices.azure.com/language/:query-knowledgebases?projectName=RetailBot&ap
headers = {
    "Ocp-Apim-Subscription-Key": "9n4FvOvemhEAs1UKQYrJxA0KaJES6aiodEzND7zGBcQ2CHSZS5GjJQQJ99BDAC8vTInXJ3w3AAAaACOGZJAb",
    "Content-Type": "application/json"
}

# Step 3: Make predictions
y_true = []
y_pred = []

for item in test_data:
    question = item["question"]
    expected_answer = item["expected_answer"]

    payload = {
        "question": question,
        "top": 1
    }
```

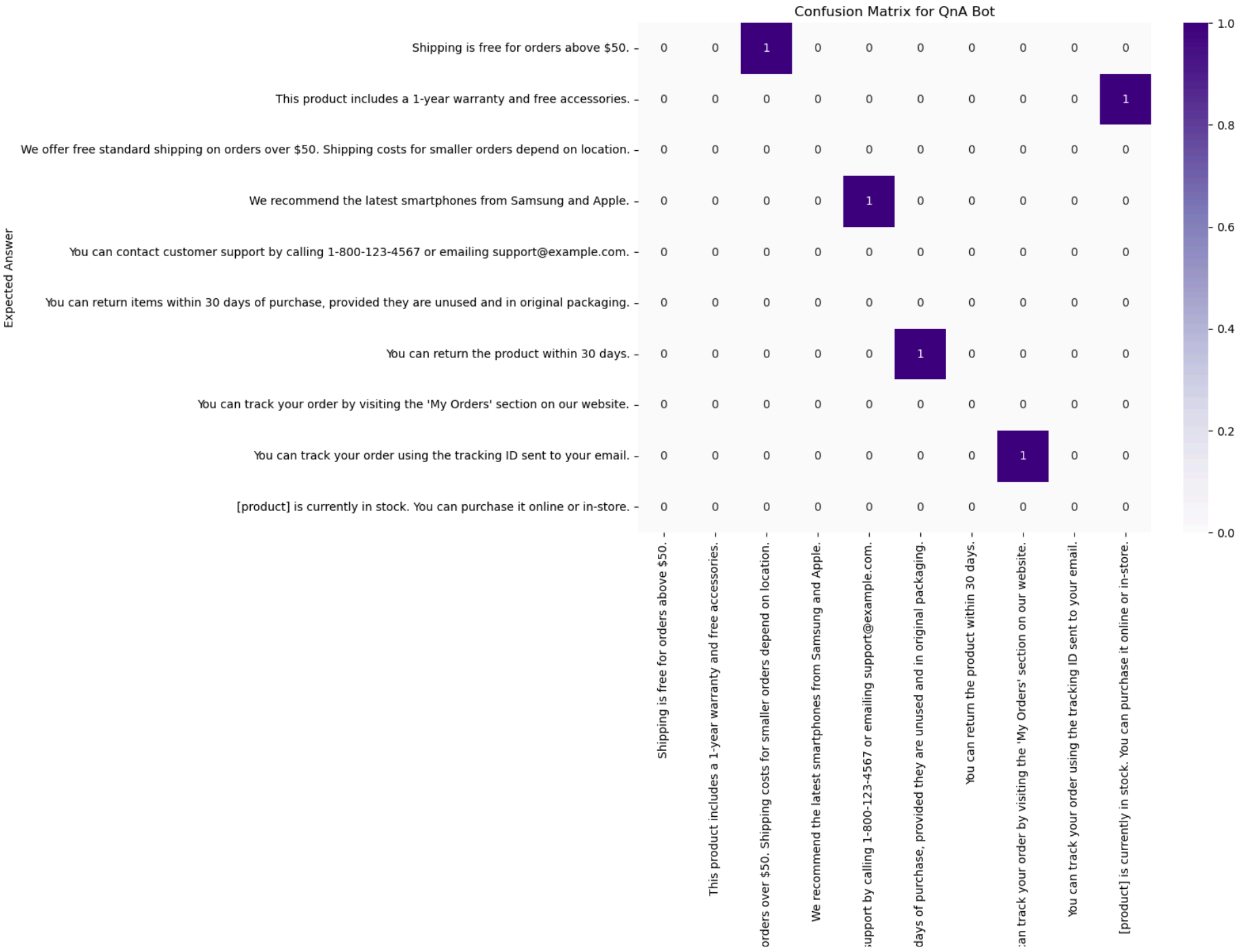
```
    response = requests.post(prediction_url, headers=headers, json=payload)
    result = response.json()

    # Extract predicted answer
    predicted_answer = result["answers"][0]["answer"] if result.get("answers") else "No Answer"

    y_true.append(expected_answer)
    y_pred.append(predicted_answer)

# Step 4: Convert to labels for classification metrics
label_encoder = LabelEncoder()
all_answers = list(set(y_true + y_pred))
label_encoder.fit(all_answers)

y_true_encoded = label_encoder.transform(y_true)
y_pred_encoded = label_encoder.transform(y_pred)

# Step 5: Calculate metrics
print("Precision:", precision_score(y_true_encoded, y_pred_encoded, average='macro'))
print("Recall:", recall_score(y_true_encoded, y_pred_encoded, average='macro'))
print("F1 Score:", f1_score(y_true_encoded, y_pred_encoded, average='macro'))

print("\nClassification Report:\n", classification_report(y_true_encoded, y_pred_encoded, target_names=label_encoder.classes_))

# Step 6: Confusion Matrix
cm = confusion_matrix(y_true_encoded, y_pred_encoded)
df_cm = pd.DataFrame(cm, index=label_encoder.classes_, columns=label_encoder.classes_)

plt.figure(figsize=(10, 8))
sns.heatmap(df_cm, annot=True, fmt='d', cmap='Purples')
plt.xlabel("Predicted Answer")
plt.ylabel("Expected Answer")
plt.title("Confusion Matrix for QnA Bot")
plt.show()
```

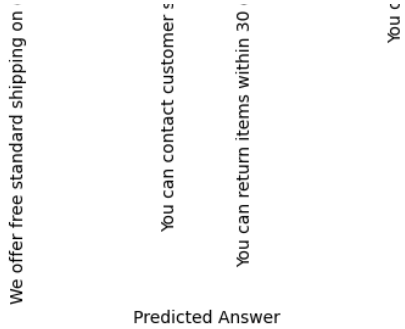
```
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Precision is il
l-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall is ill-d
efined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Precision and F
-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall and F-sc
ore are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavio
r.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Precision and F
-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall and F-sc
ore are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavio
r.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Precision and F
-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\Chirag Pc\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall and F-sc
ore are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavio
r.
  _warn_prf(average, modifier, msg_start, len(result))
```

Precision: 0.0
Recall: 0.0
F1 Score: 0.0

Classification Report:

	precision	recall	f1-score	support
Shipping is free for orders above $50.	0.00	0.00	0.00	1.0
This product includes a 1-year warranty and free accessories.	0.00	0.00	0.00	1.0
We offer free standard shipping on orders over $50. Shipping costs for smaller orders depend on location.	0.00	0.00	0.00	0.0
We recommend the latest smartphones from Samsung and Apple.	0.00	0.00	0.00	1.0
You can contact customer support by calling 1-800-123-4567 or emailing support@example.com.	0.00	0.00	0.00	0.0
You can return items within 30 days of purchase, provided they are unused and in original packaging.	0.00	0.00	0.00	0.0
You can return the product within 30 days.	0.00	0.00	0.00	1.0
You can track your order by visiting the 'My Orders' section on our website.	0.00	0.00	0.00	0.0
You can track your order using the tracking ID sent to your email.	0.00	0.00	0.00	1.0
[product] is currently in stock. You can purchase it online or in-store.	0.00	0.00	0.00	0.0
accuracy			0.00	5.0
macro avg	0.00	0.00	0.00	5.0
weighted avg	0.00	0.00	0.00	5.0


Confusion Matrix for QnA Bot
Expected Answer
Shipping is free for orders above $50.
This product includes a 1-year warranty and free accessories.
We offer free standard shipping on orders over $50. Shipping costs for smaller orders depend on location.
We recommend the latest smartphones from Samsung and Apple.
You can contact customer support by calling 1-800-123-4567 or emailing support@example.com.
You can return items within 30 days of purchase, provided they are unused and in original packaging.
You can return the product within 30 days.
You can track your order by visiting the 'My Orders' section on our website.
You can track your order using the tracking ID sent to your email.
[product] is currently in stock. You can purchase it online or in-store.
1.0
0.8
0.6
0.4
0.2
0.0



In [6]:
```
! pip install fuzzywuzzy
```

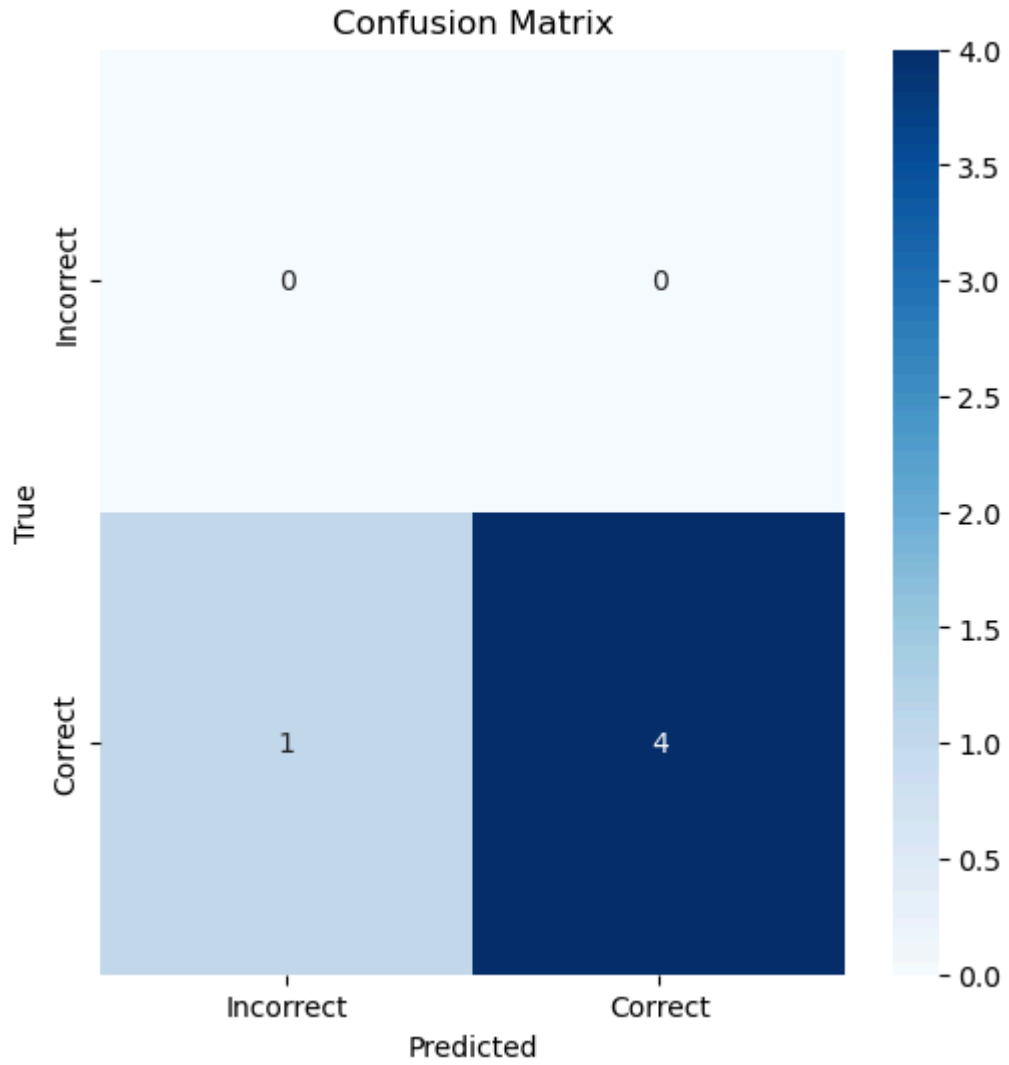
```
Collecting fuzzywuzzy
  Downloading fuzzywuzzy-0.18.0-py2.py3-none-any.whl.metadata (4.9 kB)
Downloading fuzzywuzzy-0.18.0-py2.py3-none-any.whl (18 kB)
Installing collected packages: fuzzywuzzy
Successfully installed fuzzywuzzy-0.18.0
```

In [13]:
```
import requests
from fuzzywuzzy import fuzz
from sklearn.metrics import precision_score, recall_score, f1_score

# ------------------------
# CONFIGURATION
# ------------------------
PREDICTION_URL = r"https://cav-cv-resource1.cognitiveservices.azure.com/language/:query-knowledgebases?projectName=RetailBot&a
SUBSCRIPTION_KEY = "9n4FvOvemhEAs1UKQYrJxA0KaJES6aiodEzND7zGBcQ2CHSZS5GjJQQJ99BDAC8vTInXJ3w3AAAaACOGZJAb"
SIMILARITY_THRESHOLD = 50  # Adjust as needed

HEADERS = {
    "Ocp-Apim-Subscription-Key": SUBSCRIPTION_KEY,
    "Content-Type": "application/json"
}

# ------------------------
# TEST QUESTIONS & ANSWERS
# ------------------------
test_data = [
    {"question": "What is your return policy?", "answer": "You can return the product within 30 days."},
    {"question": "can you track my order?", "answer": "You can track your order using the tracking ID sent to your email."},
```

```
    {"question": "How do I cancel an order?", "answer": "Orders can be canceled within 1 hour of placement by contacting custo
    {"question": "Tell me about shipping charges", "answer": "Shipping is free for orders above $50."},
    {"question": "What payment methods do you accept?", "answer": " any netbanking , gpay, Paypal, neft, credit, debit/credit
]

# ------------------------
# EVALUATION
# ------------------------
y_true_binary = []
y_pred_binary = []

for item in test_data:
    payload = {"question": item["question"], "top": 1}
    response = requests.post(PREDICTION_URL, headers=HEADERS, json=payload)
    result = response.json()

    predicted_answer = result.get("answers", [{}])[0].get("answer", "No Answer")
    true_answer = item["answer"]

    similarity = fuzz.token_set_ratio(predicted_answer, true_answer)
    is_correct = similarity >= SIMILARITY_THRESHOLD

    y_true_binary.append(1)
    y_pred_binary.append(1 if is_correct else 0)

    print(f"Q: {item['question']}")
    print(f"Answer: {true_answer}")
    print(f"Predicted: {predicted_answer}")
    print(f"Similarity: {similarity} → {'✅ Match' if is_correct else '❌ Mismatch'}")
    print("-" * 60)

# ------------------------
# METRICS
# ------------------------
print("\n🔍 Evaluation Metrics:")
print(f"Precision: {precision_score(y_true_binary, y_pred_binary):.2f}")
print(f"Recall:    {recall_score(y_true_binary, y_pred_binary):.2f}")
print(f"F1 Score:  {f1_score(y_true_binary, y_pred_binary):.2f}")


# CONFUSION MATRIX
```

```python
# ------------------------
conf_matrix = confusion_matrix(y_true_binary, y_pred_binary)
plt.figure(figsize=(6, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=["Incorrect", "Correct"], yticklabels=["Incorrect", "C
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

```
Q: What is your return policy?
Answer: You can return the product within 30 days.
Predicted: You can return items within 30 days of purchase, provided they are unused and in original packaging.
Similarity: 83 → ✅ Match
------------------------------------------------------------
Q: can you track my order?
Answer: You can track your order using the tracking ID sent to your email.
Predicted: You can track your order by visiting the 'My Orders' section on our website.
Similarity: 64 → ✅ Match
------------------------------------------------------------
Q: How do I cancel an order?
Answer: Orders can be canceled within 1 hour of placement by contacting customer support. After that, they cannot be canceled.
Predicted: Orders can be canceled within 1 hour of placement by contacting customer support. After that, they cannot be cancele
d.
Similarity: 100 → ✅ Match
------------------------------------------------------------
Q: Tell me about shipping charges
Answer: Shipping is free for orders above $50.
Predicted: We offer free standard shipping on orders over $50. Shipping costs for smaller orders depend on location.
Similarity: 86 → ✅ Match
------------------------------------------------------------
Q: What payment methods do you accept?
Answer:  any netbanking , gpay, Paypal, neft, credit, debit/credit card.
Predicted: We accept all major credit cards, PayPal, and online bank transfers.
Similarity: 47 → ❌ Mismatch
------------------------------------------------------------

🔍 Evaluation Metrics:
Precision: 1.00
Recall:    0.80
F1 Score:  0.89
```



In [39]:
```
PREDICTION_URLs= r"https://cav-cv-resource1.cognitiveservices.azure.com/language/:query-knowledgebases?projectName=RetailBot&a
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

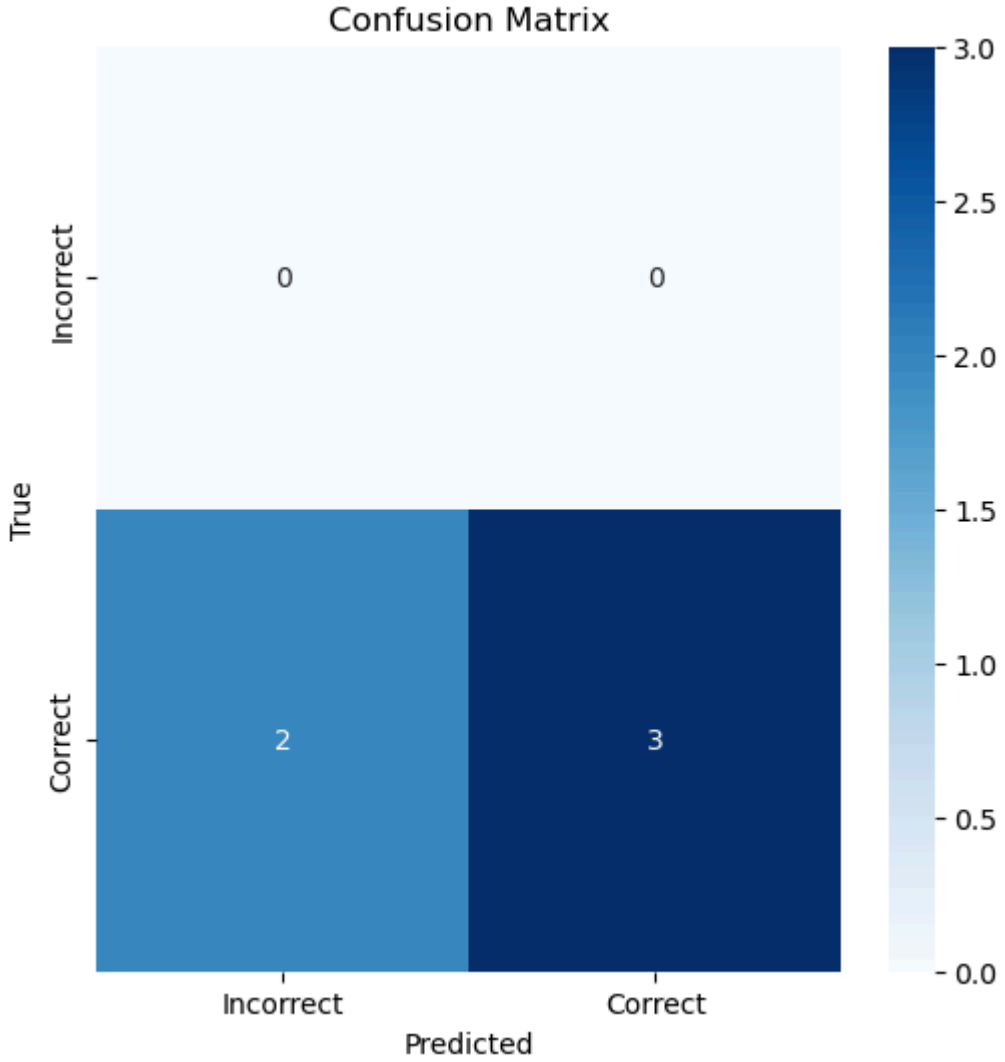
In [ ]:

In [ ]:

In [40]:
```
import requests
from fuzzywuzzy import fuzz
from sklearn.metrics import precision_score, recall_score, f1_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
# ------------------------
# CONFIGURATION
# ------------------------
PREDICTION_URL=PREDICTION_URLs
SUBSCRIPTION_KEY = key
SIMILARITY_THRESHOLD = 70 # Adjust as needed

HEADERS = {
    "Ocp-Apim-Subscription-Key": SUBSCRIPTION_KEY,
    "Content-Type": "application/json"
}
# ------------------------
# TEST QUESTIONS & ANSWERS
# ------------------------
test_data = [
    {"question": "What is your return policy?", "answer": "You can return the product within 30 days."},
    {"question": "can you track my order?", "answer": "You can track your order using the tracking ID sent to your email."},
    {"question": "How do I cancel an order?", "answer": "Orders can be canceled within 1 hour of placement by contacting custo
    {"question": "Tell me about shipping charges", "answer": "Shipping is free for orders above $50."},
    {"question": "What payment methods do you accept?", "answer": "any netbanking , gpay, Paypal, neft, credit, debit/credit c
]
# ------------------------
# EVALUATION
# ------------------------
y_true_binary = []
y_pred_binary = []
```

```python
for item in test_data:
    payload = {"question": item["question"], "top": 1}
    response = requests.post(PREDICTION_URL, headers=HEADERS, json=payload)
    result = response.json()

    predicted_answer = result.get("answers", [{}])[0].get("answer", "No Answer")
    true_answer = item["answer"]

    similarity = fuzz.token_set_ratio(predicted_answer, true_answer)
    is_correct = similarity >= SIMILARITY_THRESHOLD

    # Assign labels based on whether the prediction is correct or not
    y_true_binary.append(1)  # True answer is considered correct (True)
    y_pred_binary.append(1 if is_correct else 0)  # Predicted 1 if similar, else 0

    print(f"Q: {item['question']}")
    print(f"Answer: {true_answer}")
    print(f"Predicted: {predicted_answer}")
    print(f"Similarity: {similarity} → {'✅ Match' if is_correct else '❌ Mismatch'}")
    print("-" * 60)
# ------------------------
# METRICS
# ------------------------
print("\n🔍 Evaluation Metrics:")
precision = precision_score(y_true_binary, y_pred_binary)
recall = recall_score(y_true_binary, y_pred_binary)
f1 = f1_score(y_true_binary, y_pred_binary)
print(f"Precision: {precision:.2f}")
print(f"Recall:    {recall:.2f}")
print(f"F1 Score:  {f1:.2f}")
# ------------------------
# CONFUSION MATRIX
# ------------------------
conf_matrix = confusion_matrix(y_true_binary, y_pred_binary)
plt.figure(figsize=(6, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=["Incorrect", "Correct"], yticklabels=["Incorrect", "C
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

```
Q: What is your return policy?
Answer: You can return the product within 30 days.
Predicted: You can return items within 30 days of purchase, provided they are unused and in original packaging.
Similarity: 83 → ✅ Match
------------------------------------------------------------
Q: can you track my order?
Answer: You can track your order using the tracking ID sent to your email.
Predicted: You can track your order by visiting the 'My Orders' section on our website.
Similarity: 64 → ❌ Mismatch
------------------------------------------------------------
Q: How do I cancel an order?
Answer: Orders can be canceled within 1 hour of placement by contacting customer support. After that, they cannot be canceled.
Predicted: Orders can be canceled within 1 hour of placement by contacting customer support. After that, they cannot be cancele
d.
Similarity: 100 → ✅ Match
------------------------------------------------------------
Q: Tell me about shipping charges
Answer: Shipping is free for orders above $50.
Predicted: We offer free standard shipping on orders over $50. Shipping costs for smaller orders depend on location.
Similarity: 86 → ✅ Match
------------------------------------------------------------
Q: What payment methods do you accept?
Answer: any netbanking , gpay, Paypal, neft, credit, debit/credit card.
Predicted: We accept all major credit cards, PayPal, and online bank transfers.
Similarity: 47 → ❌ Mismatch
------------------------------------------------------------

🔍 Evaluation Metrics:
Precision: 1.00
Recall:    0.60
F1 Score:  0.75
```


Confusion Matrix
Incorrect
Correct
True
0
0
2
3
Incorrect
Correct
Predicted
3.0
2.5
2.0
1.5
1.0
0.5
0.0

In [ ]: