Use Azure AI Services to Process and Analyze Survey Results

A guide for beginners who want to learn how to use Azure AI Services for text analysis.

# Introduction

In this lab, you will learn how to use Azure AI Services to process an array of survey results and do sentiment analysis on the results. By the end of this lab, you will have a report that contains the survey results, as well as a summary of the sentiment.

# Prerequisites

To complete this lab, you will need:

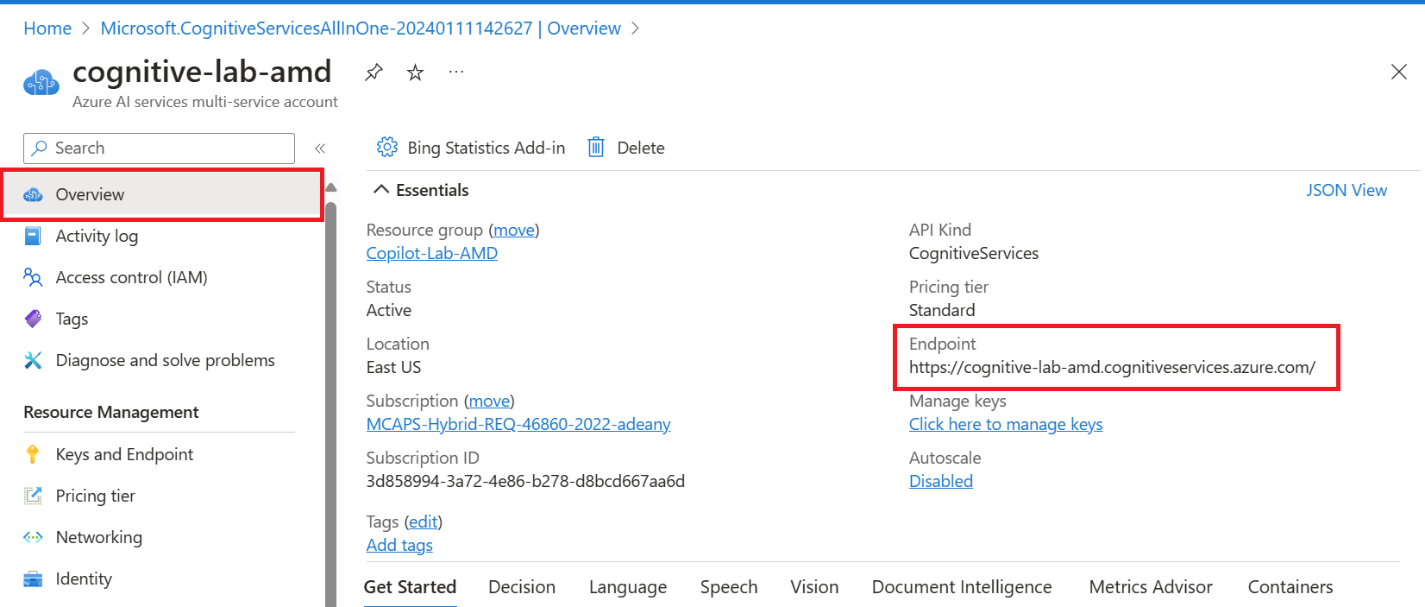
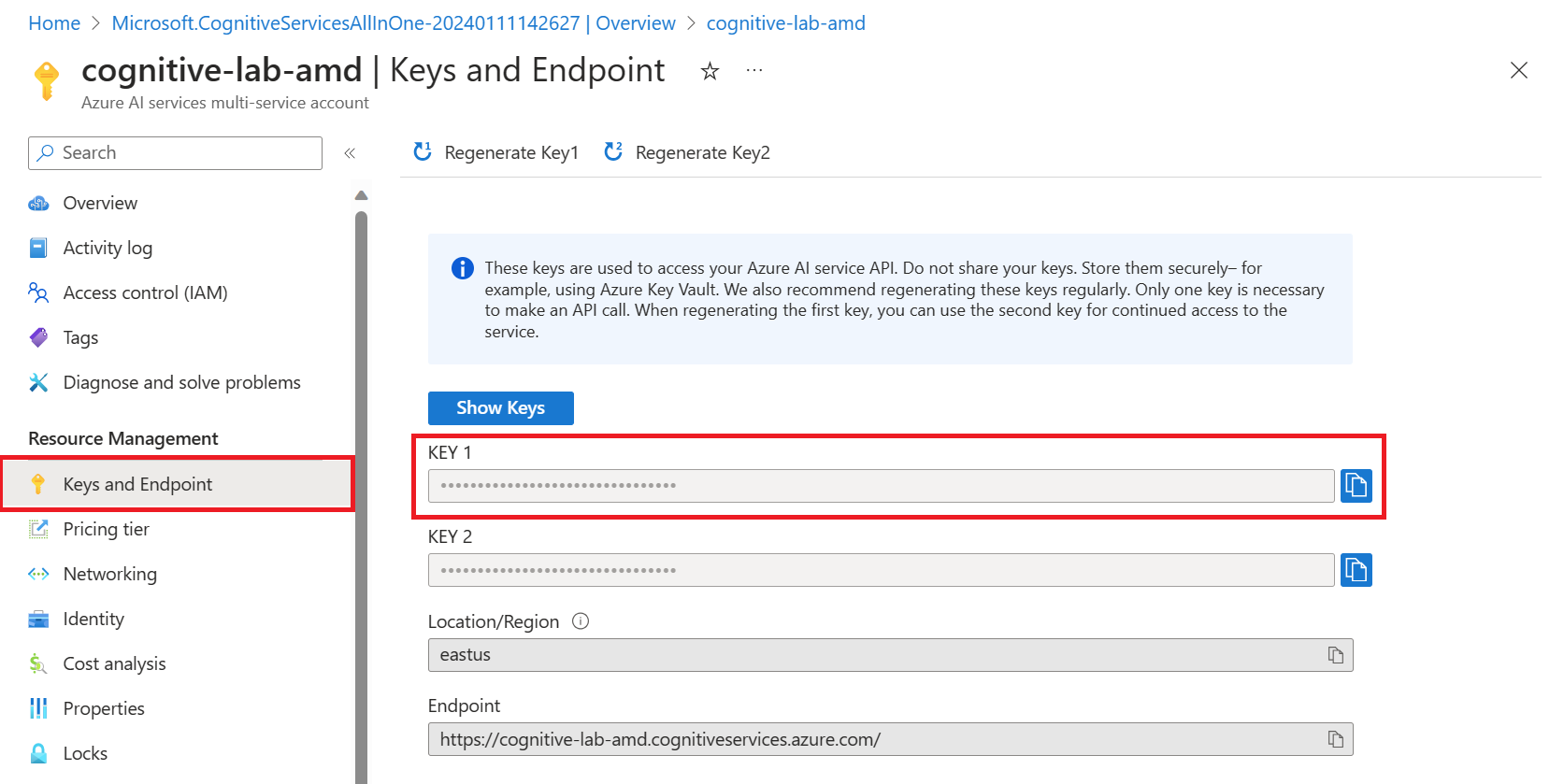
* An Azure subscription
* Visual Studio Code with dotnet

# Step 1: Create an Azure AI Services Resource

The first step is to create an Azure Cognitive Services resource that will provide access to the Text Analytics and Translator APIs. To do this, follow these steps:

1. Log in to the **Azure portal** (https://portal.azure.com) and click on the "**Create a resource**" button on the top left corner.
2. Search for "**Azure AI Services**" and select the option with the same name. **A screenshot of a chat

   Description automatically generated**
3. Click on the "**Create**" button and fill in the required fields, such as the resource name, the subscription, the resource group, the location, and the pricing tier. You can choose any name and location, but make sure to select the "S0" pricing tier, which gives you 5,000 transactions per month for free.  
     
   A screenshot of a computer

   Description automatically generated
4. Click on the "**Review + create**" button and then on the "**Create**" button to confirm the creation of the resource.
5. Once the resource is created, click “**Go to resource**”
6. On the **Overview page**, copy the endpoint URL. You will need this later to connect to the APIs.  
     
   
7. On the **Keys and** Endpoints page, copy one of the keys. You will need this later to connect to the APIs.  
     
   

# Step 2: Create a C# Console Application Project

Follow these steps to complete the lab:

1. Open VS Code and navigate to a new folder.
2. Open a new terminal and type: **dotnet new console --framework net8.0**. A new console app template should be created.  
     
   A screen shot of a computer program

   Description automatically generated
3. Let’s pretend we just completed a customer survey that asked participants to provide comments on their thoughts about Github Copilot and we have the results that we want to perform sentiment analysis on. **Paste** the following code in the generated **Program.cs** file and click **Ctrl + S to** save:

|  |
| --- |
| string[] customerFeedback = [  "I like it!",    "It doesn't work for me, but it seems like it would benefit others",   "I didn't find it very useful",   "My company will be using this immediately. It's amazing!",   "I can't unsee this. Sad.",  "I wish I had discovered this sooner.",  ];  foreach (string comment in customerFeedback) {  Console.WriteLine(“comment: “ + comment);  } |

# Step 3: Use the Text Analytics API to Analyze the Results

1. Now that we have our survey results in our application, let’s ask Copilot to analyze the results leveraging Azure AI Services. You can type something like: **// Use the Azure AI Text Analytics API and a Text Analytics client to analyze the customer feedback array and output the positive/neutral/negative results for each to the command line.  
     
   A black background with white text

   Description automatically generated**
2. Copilot should return something like:  
     
   A screenshot of a computer program

   Description automatically generated  
   A screenshot of a computer program

   Description automatically generated
3. If we look at Copilot’s suggestion, it looks like in order to use the Azure AI Text Analytics API we’ll first need to install the NuGet package. Let’s run **dotnet add package Azure.AI.TextAnalytics** in the command line
4. The package should install successfully. Now, our .csproj file should have been updated with the package reference and should look something like this:  
     
   A screen shot of a computer program

   Description automatically generated
5. Next, Copilot suggested to use the TextAnalyticsClient to analyze the sentiment of the customer feedback. Let’s **Copy** the code that Copilot provided and **paste** it into our Program.cs file. You should have something like this:  
     
   A screen shot of a computer program

   Description automatically generated
6. Next, we will make sure **to replace the analytics key and endpoint** with the values you copied in Step 1 and **Ctrl + S** to save.
7. Now let’s run our console app by running the command: **dotnet run.** The console should output the results of the sentiment analysis. You should see a response like this:  
     
   A screenshot of a computer error

   Description automatically generated
8. Next, let’s prompt Copilot to summarize the results. You can type something like: **// After writing the sentiments to the console, summarize the sentiments.  
     
   A black background with white text

   Description automatically generated**
9. Copilot will return something like this:  
     
   A screenshot of a computer program

   Description automatically generated  
   A screen shot of a computer program

   Description automatically generated
10. Let’s modify our code based on Copilot’s suggestion. **Copy** the code modifications that your pair programmer suggested and **paste** it in your Program.cs file. **Ctrl + S** to save.
11. In the terminal, run the command **dotnet run.** We can now see that after the sentiment analysis has been done on each of the comments, it outputs the number of comments for each sentiment type. The output should look something like this:  
      
    A screenshot of a computer

    Description automatically generated

# Conclusion

In this lab, you learned how to use Azure AI Services to process an array of survey results to do sentiment analysis on the results and output a report that contains the survey results, as well as a sentiment summary. You can use these skills to create similar reports for other types of forms and documents.