100 XP

Use Azure OpenAI SDK

7 minutes

In addition to REST APIs covered in the previous unit, users can also access Azure OpenAI models through C# and Python SDKs. The same functionality is available through both REST and these SDKs.

> **Note**
>
> Before interacting with the API using either SDK, you must create an Azure OpenAI resource in the Azure portal, deploy a model in that resource, and retrieve your endpoint and keys. Check out the Getting started with Azure OpenAI Service to learn how to do that.

For both SDKs covered in this unit, you need the endpoint and a key from your Azure OpenAI resource, and the name you gave for your deployed model. In the following code snippets, the following placeholders are used:

Expand table

Placeholder name	Value
`YOUR_ENDPOINT_NAME`	This base endpoint is found in the **Keys & Endpoint** section in the Azure portal. It's the base endpoint of your resource, such as `https://sample.openai.azure.com/`.
`YOUR_API_KEY`	Keys are found in the **Keys & Endpoint** section in the Azure portal. You can use either key for your resource.
`YOUR_DEPLOYMENT_NAME`	This deployment name is the name provided when you deployed your model in the Azure OpenAI Studio.

Install libraries

First, install the client library for your preferred language. The C# SDK is a .NET adaptation of the REST APIs and built specifically for Azure OpenAI, however it can be used to connect to Azure OpenAI resources or non-Azure OpenAI endpoints. The Python SDK is built and maintained by OpenAI.

Console

```
dotnet add package Azure.AI.OpenAI --prerelease
```

Configure app to access Azure OpenAI resource

Configuration for each language varies slightly, but both require the same parameters to be set. The necessary parameters are `endpoint`, `key`, and the name of your deployment, which is called the `engine` when sending your prompt to the model.

Add the library to your app, and set the required parameters for your client.

C#

```
// Add OpenAI library
using Azure.AI.OpenAI;

// Define parameters and initialize the client
string endpoint = "<YOUR_ENDPOINT_NAME>";
string key = "<YOUR_API_KEY>";
string deploymentName = "<YOUR_DEPLOYMENT_NAME>";

OpenAIClient client = new OpenAIClient(new Uri(endpoint), new
AzureKeyCredential(key));
```

Call Azure OpenAI resource

Once you've configured your connection to Azure OpenAI, send your prompt to the model.

C#

```
// Build completion options object
ChatCompletionsOptions chatCompletionsOptions = new ChatCompletionsOptions()
{
    Messages =
    {
        new ChatRequestSystemMessage("You are a helpful AI bot."),
```

```
            new ChatRequestUserMessage("What is Azure OpenAI?"),
        },
        DeploymentName = deploymentName
};

// Send request to Azure OpenAI model
ChatCompletions response = client.GetChatCompletions(chatCompletionsOptions);

// Print the response
string completion = response.Choices[0].Message.Content;
Console.WriteLine("Response: " + completion + "\n");
```

The response object contains several values, such as `total_tokens` and `finish_reason`. The completion from the response object will be similar to the following completion:

Console

```
"Azure OpenAI is a cloud-based artificial intelligence (AI) service that of-
fers a range of tools and services for developing and deploying AI ap-
plications. Azure OpenAI provides a variety of services for training and de-
ploying machine learning models, including a managed service for training and
deploying deep learning models, a managed service for deploying machine
learning models, and a managed service for managing and deploying machine
learning models."
```

In both C# and Python, your call can include optional parameters including `temperature` and `max_tokens`. Examples of using those parameters are included in this module's lab.

Next unit: Exercise - Integrate Azure OpenAI into your app

Continue >