

**Lab Manual- Azure Language AI services Lab using SDK**

**Prepared for**:

**Date:** 18th Dec 2023

**Prepared by:**

Document Name: Lab Manual **Document Number** AZLabn916

**Contributor:**

Contents

[1. Objective 3](#_Toc167362836)

[2. Provision an Azure AI Services resource 3](#_Toc167362837)

[3. Copy Endpoint and Key 7](#_Toc167362838)

[4. Clone the repository in Visual Studio Code 8](#_Toc167362839)

[5. Install SDK Library 9](#_Toc167362840)

[6. Configure Endpoint and Key 10](#_Toc167362841)

[7. Use SDK 12](#_Toc167362842)

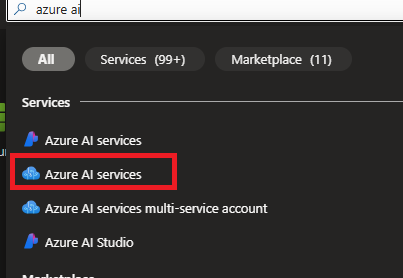
# Objective

In this exercise, you'll get started with Azure AI Services by creating an **Azure AI Services** resource in your Azure subscription and using it from a client application. The goal of the exercise is not to gain expertise in any particular service, but rather to become familiar with a general pattern for provisioning and working with Azure AI services as a developer.

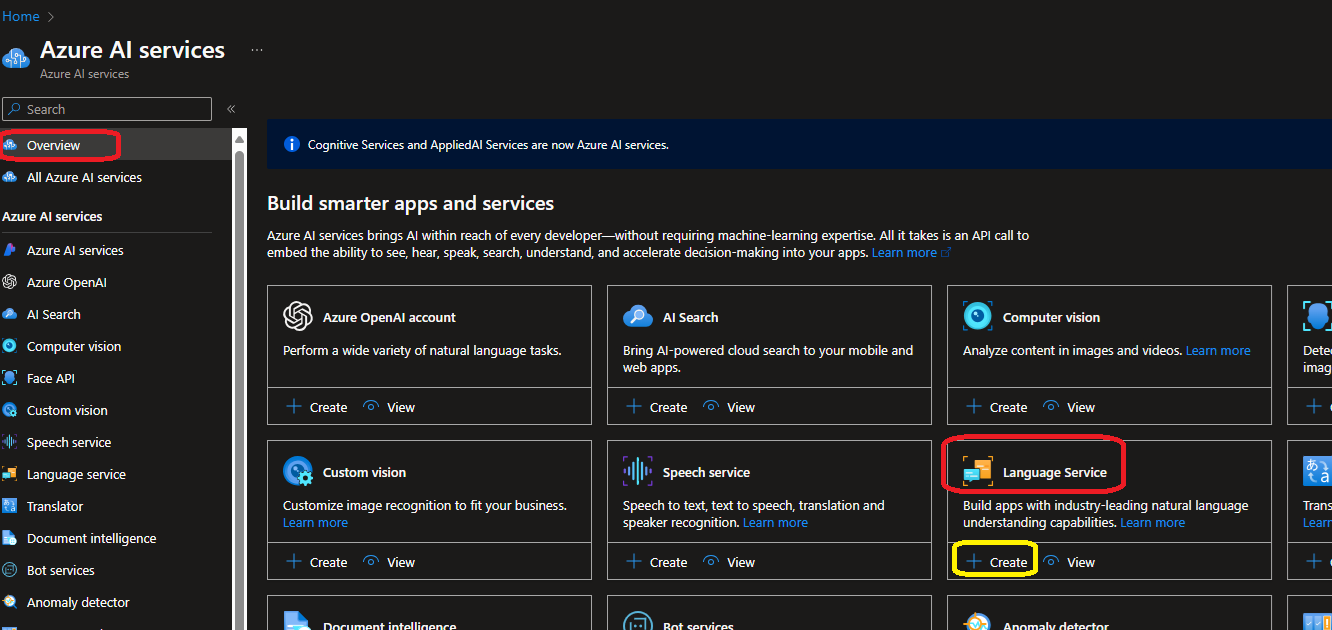
# Provision an Azure AI Services resource

Azure AI Services are cloud-based services that encapsulate artificial intelligence capabilities you can incorporate into your applications. You can provision **individual Azure AI services** resources for **specific APIs** (for example, **Language** or **Vision**), or you can provision a single Azure AI Services resource that provides access to multiple Azure AI services APIs through a single **endpoint** and **key**. In this case, you'll use a single Azure AI Services resource

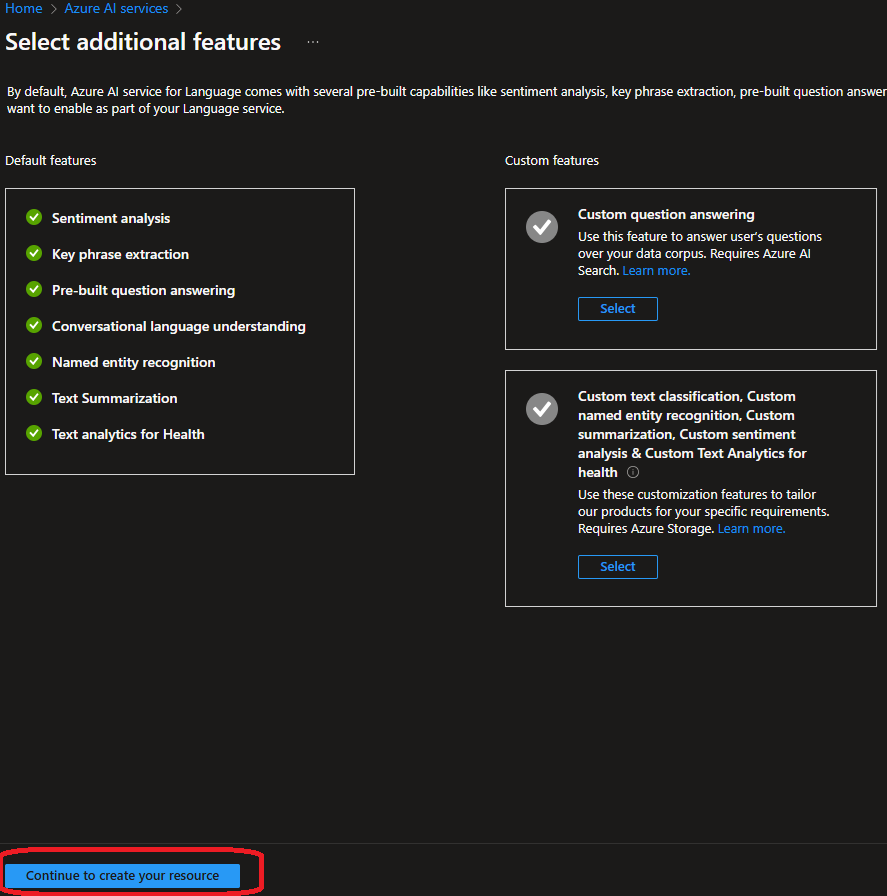
* Serach AI Service and select **Azure AI Services**



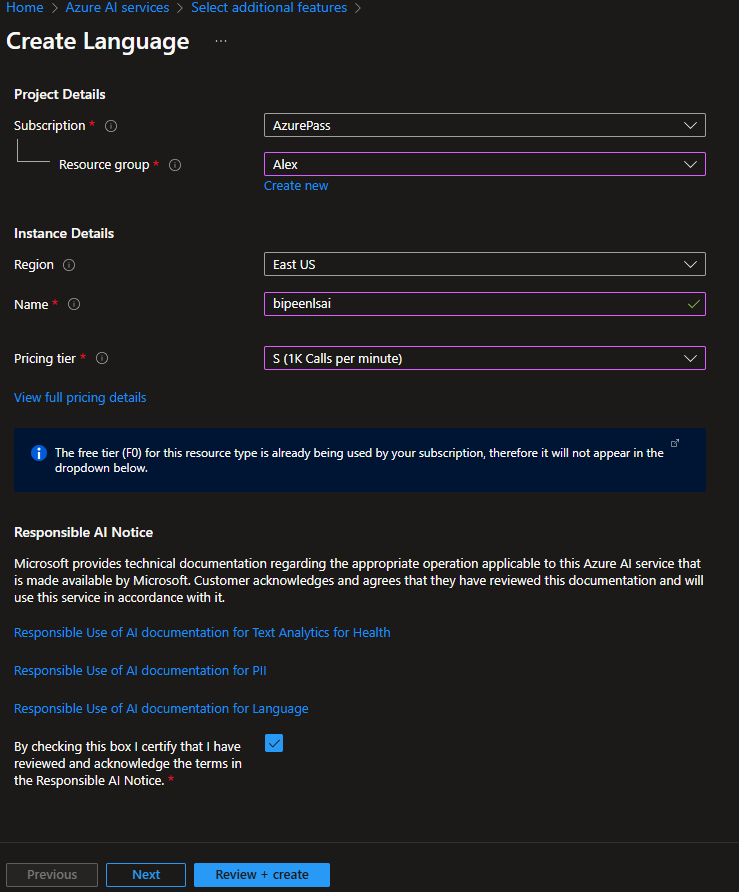
* Select **Language** and Click **create**



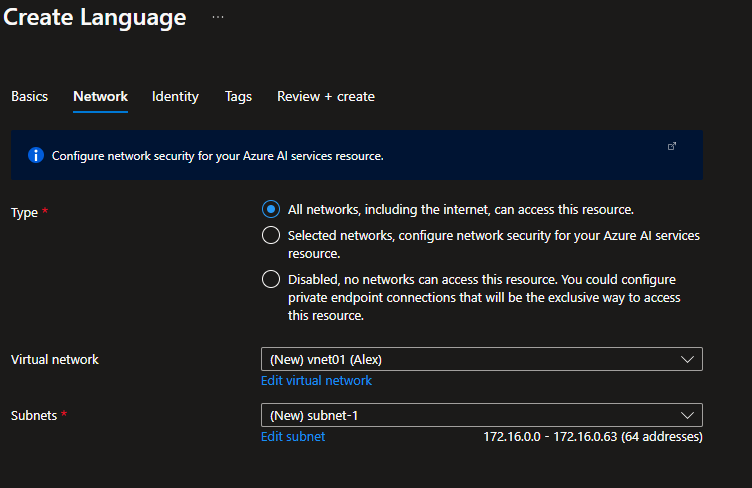
* Click **Continue to create your resource**



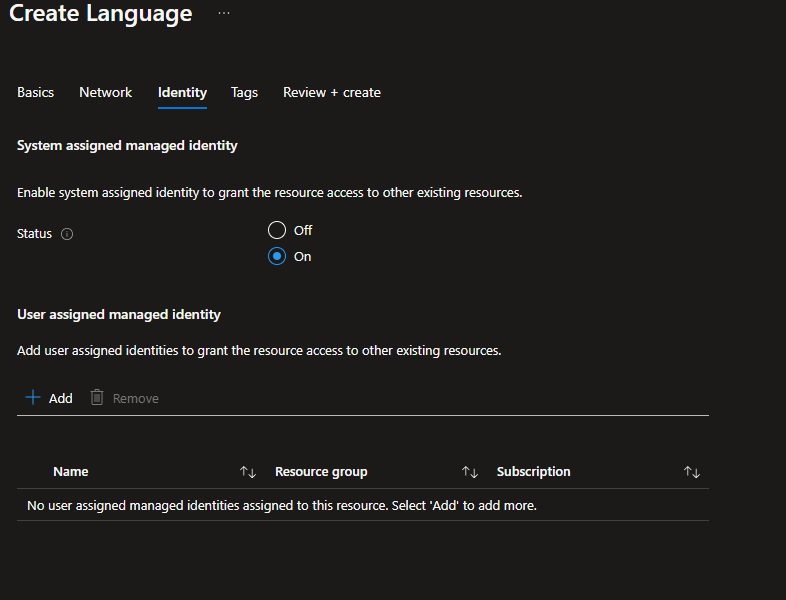
* Select your
  + Resource Group
  + Name of AI Instance
  + Pricing should be **FREE** ( Select S-Standard if free is not abalable)



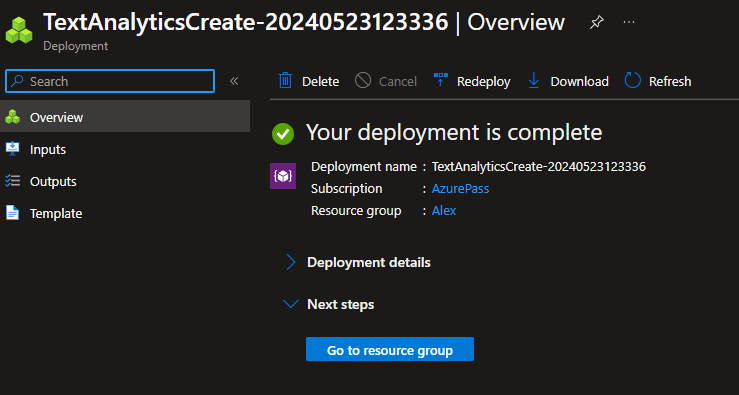
* Click **Next and Leave the default**



* Click **Next and Leave the default**

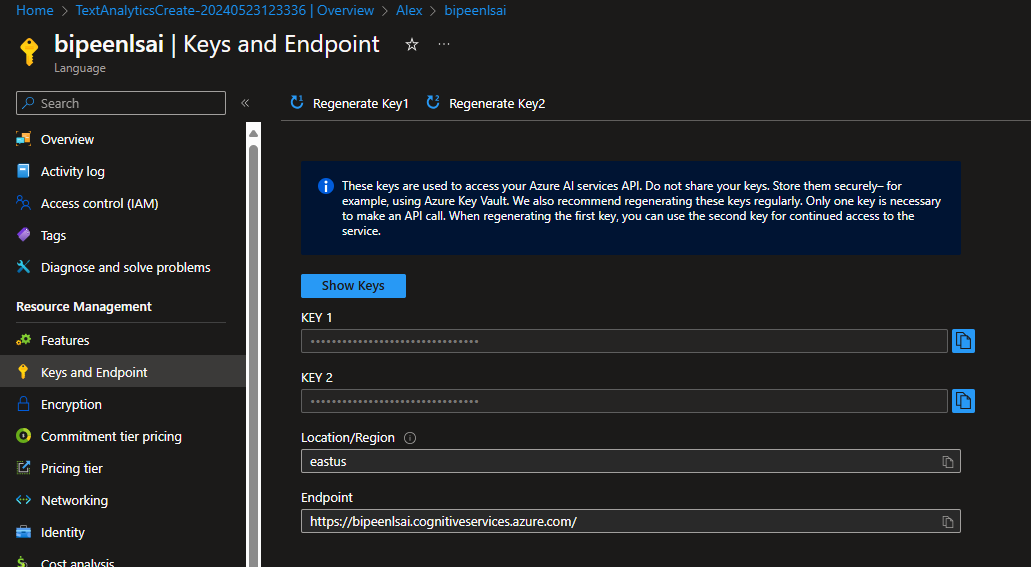


* Click **Review and Create**



# Copy Endpoint and Key

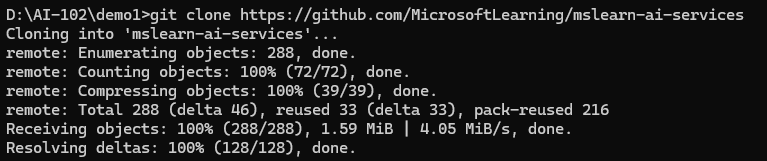
1. Wait for deployment to complete, and then view the deployment details.
2. Go to the resource and view its **Keys and Endpoint** page. This page contains the information that you will need to connect to your resource and use it from applications you develop. Specifically:
   * An HTTP *endpoint* to which client applications can send requests.
   * Two *keys* that can be used for authentication (client applications can use either key to authenticate).
   * The *location* where the resource is hosted. This is required for requests to some (but not all) APIs.



# Clone the repository in Visual Studio Code

1. Start Visual Studio Code.
2. Open the palette (SHIFT+CTRL+P) and run a **Git: Clone** command to clone the https://github.com/MicrosoftLearning/mslearn-ai-services repository to a local folder (it doesn't matter which folder).

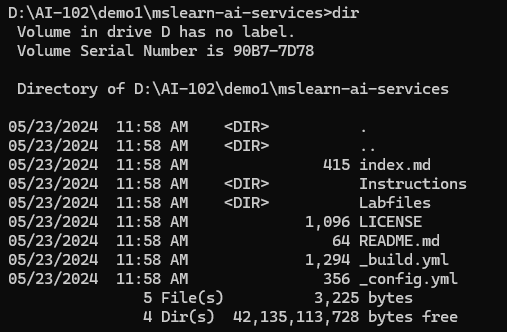
**git clone https://github.com/MicrosoftLearning/mslearn-ai-services**



1. Go Inside **mslearn-ai-services** directory

**cd mslearn-ai-services**

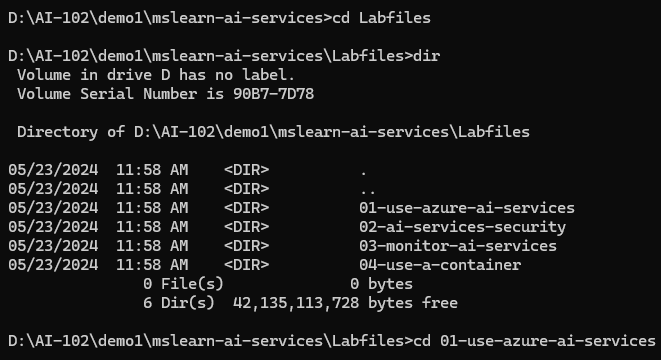
****

****

1. Go Inside **01-use-azure-ai-services** directory under **Labfiles**

**Cd Labfiles**

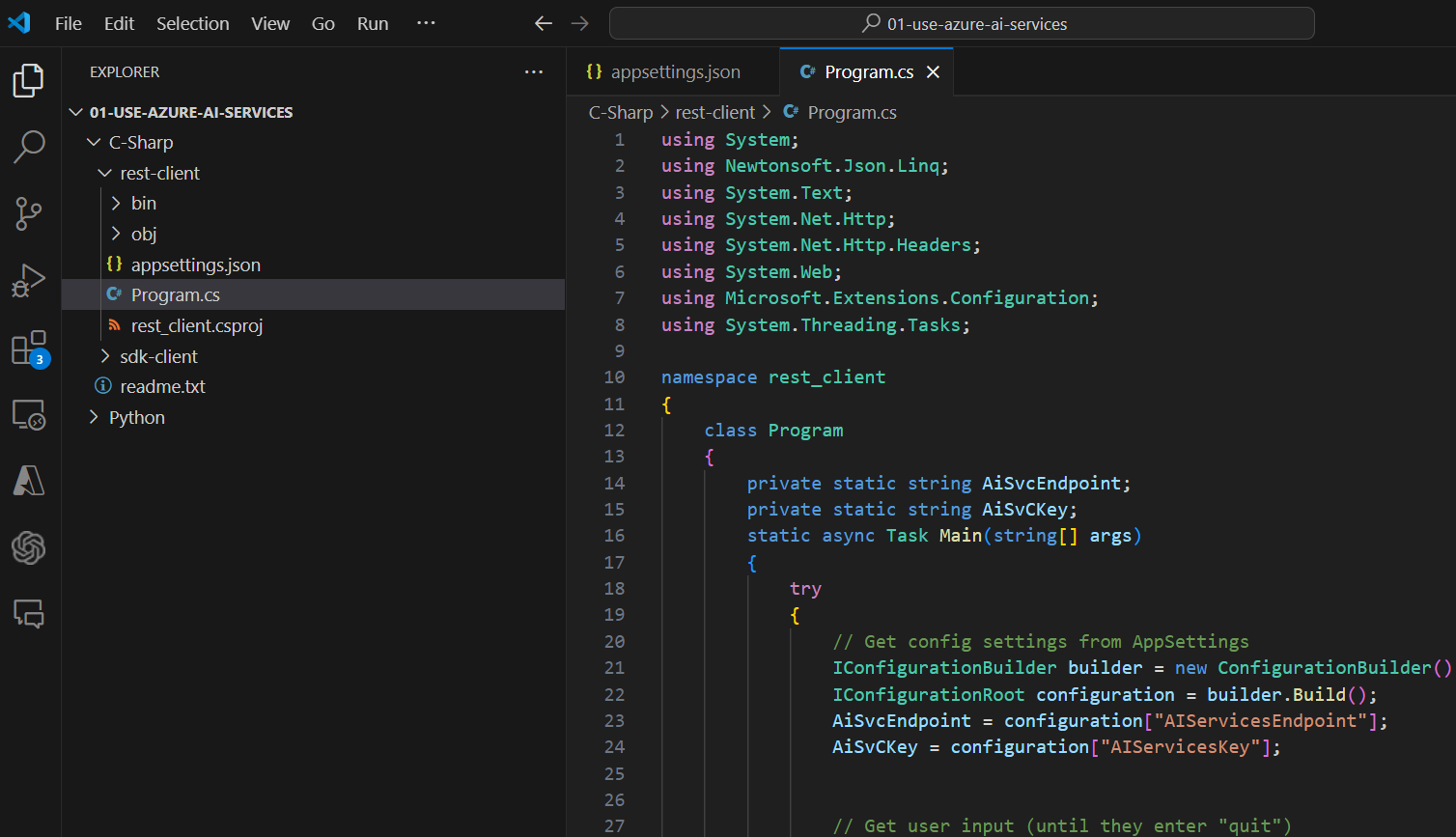
**cd 01-use-azure-ai-services**

****

1. Open the folder in Visual Studio Code.

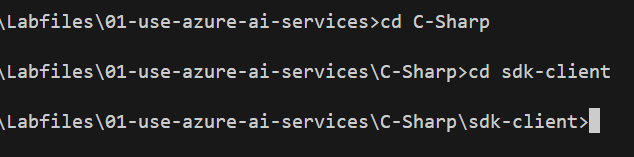
**Code .**

****

****

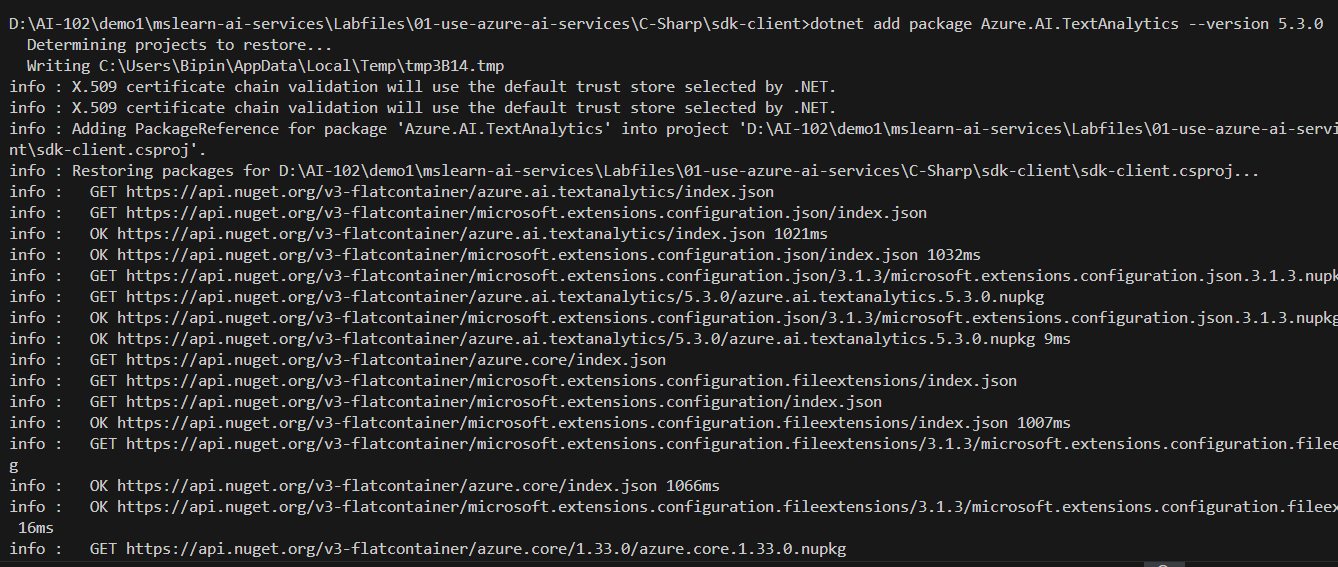
# Install SDK Library

* In Visual Studio Code, Open Terminal
* Go Inside  **sdk-client** folder under the **C-Sharp**

****

* Install the Text Analytics SDK package by running the appropriate command .

**dotnet add package Azure.AI.TextAnalytics --version 5.3.0**

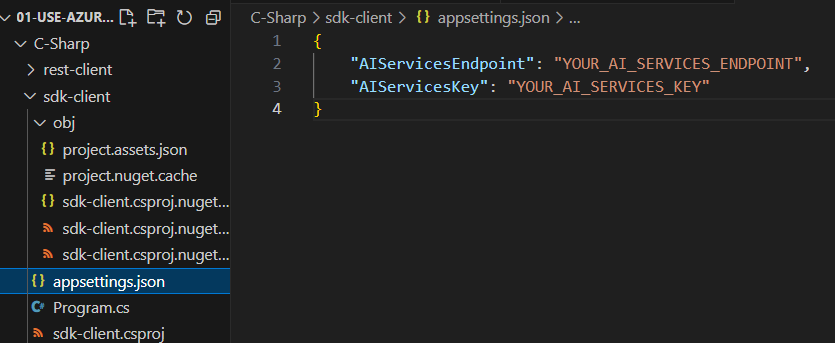


# Configure Endpoint and Key

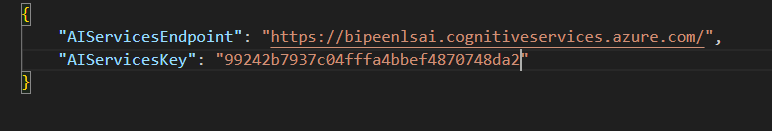
The Azure AI services APIs are REST-based, so you can consume them by submitting JSON requests over HTTP. In this example, you'll explore a console application that uses the **Language** REST API to perform language detection; but the basic principle is the same for all of the APIs supported by the Azure AI Services resource.

* In Visual Studio Code, expand the **C-Sharp**
* View the contents of the **rest-client** folder, and note that it contains a file for configuration settings:

appsettings.json

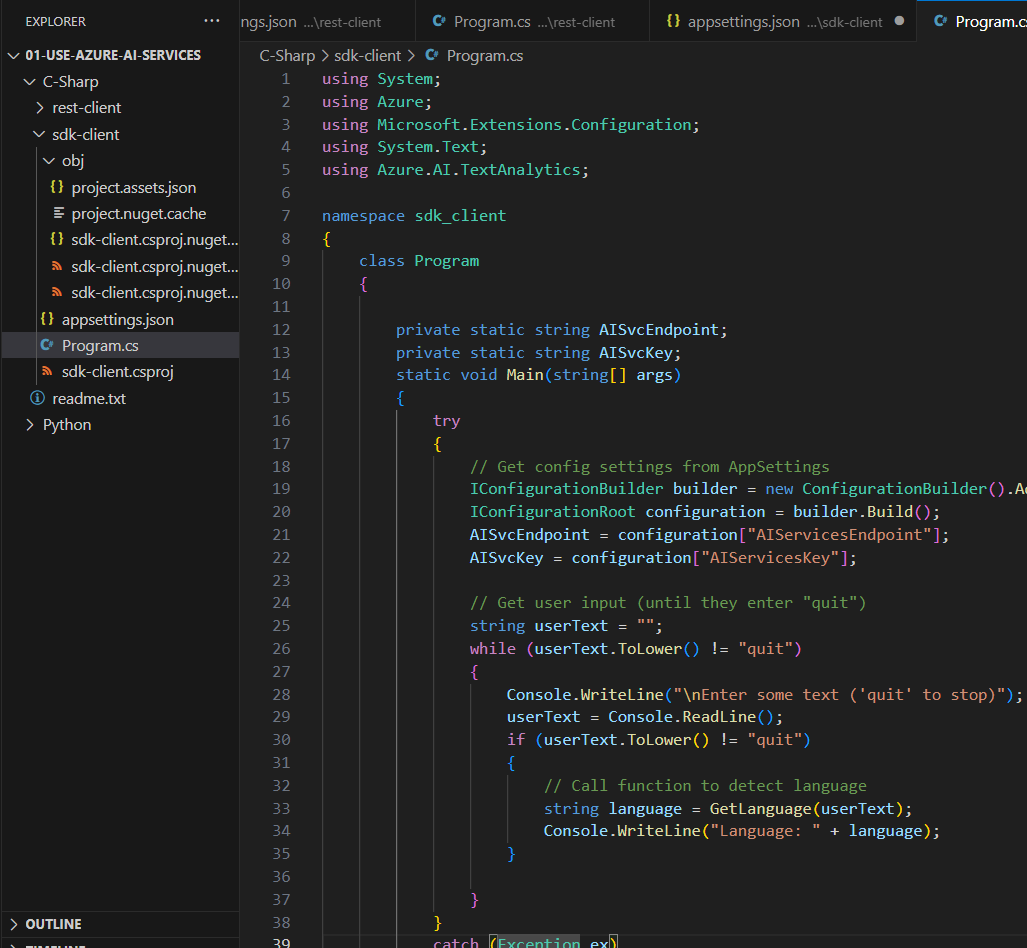


* Open the configuration file and update the configuration values it contains to reflect the **endpoint** and an authentication **key** for your Azure AI services resource. **Save your** changes.



Open the code file **Program.cs** and review the code it contains, noting the following details:

* The namespace for the SDK you installed is imported
* Code in the **Main** function retrieves the endpoint and key for your Azure AI services resource - these will be used with the SDK to create a client for the Text Analytics service.
* The **GetLanguage** function uses the SDK to create a client for the service, and then uses the client to detect the language of the text that was entered.

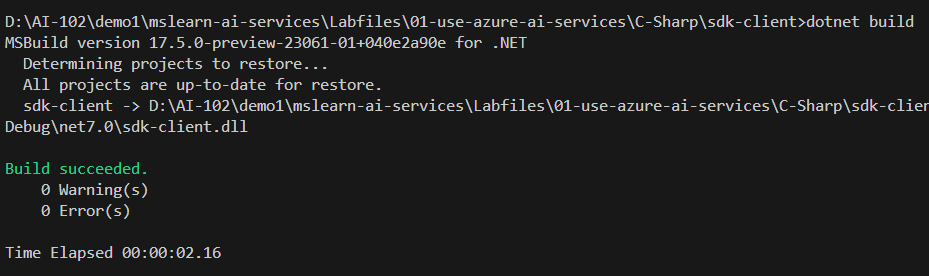


# Use SDK

You can write code that consumes Azure AI services REST APIs directly, but there are software development kits (SDKs) for many popular programming languages, including Microsoft C#, Python, Java, and Node.js. Using an SDK can greatly simplify development of applications that consume Azure AI services.

* In VS Code terminal type

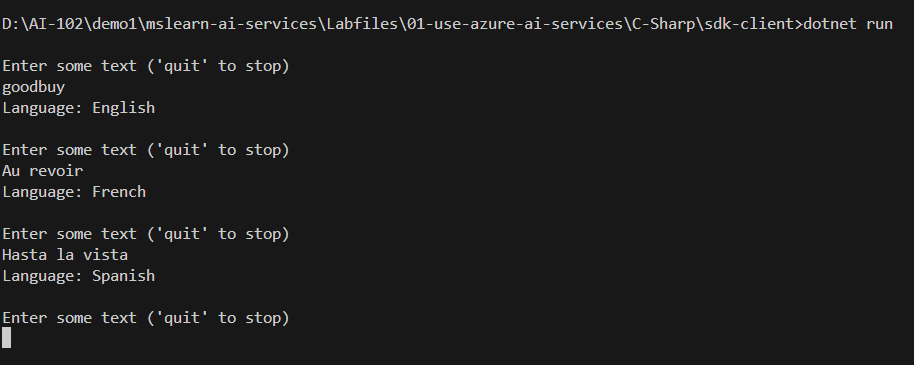
**Dotnet build**



**Dotnet Run**

****

1. When prompted, enter some text and review the language that is detected by the service,  try entering "**Goodbye**", "**Au revoir**", and "**Hasta la vista**"..



Enter quit to exit

