Microsoft Foundry Part 1

CSCI E-94

Fundamentals of Cloud Computing - Azure

Joseph Ficara

Copyright © 2013-2026
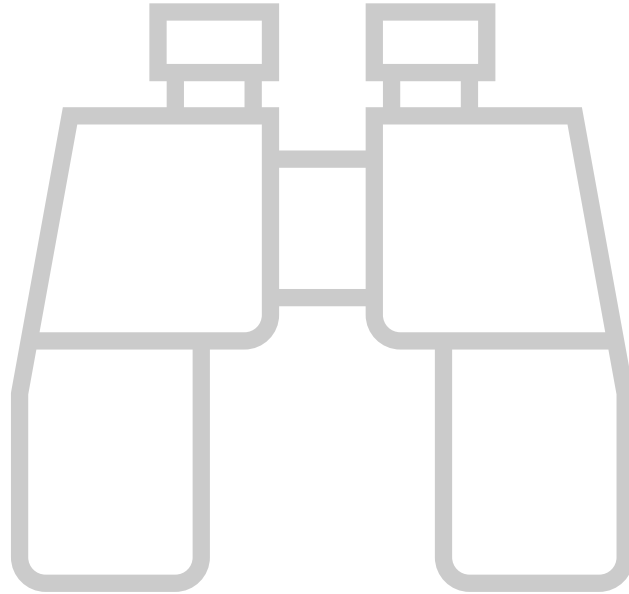
Agenda

- **Microsoft Foundry**
  - Overview
    - What is it?
    - Why do you care?
    - Essentials
  - Configuring in Azure
  - Simple Chatbot
  - Structured Output JSON

Overview

Microsoft Foundry – Overview

- **What is it?**
  - Unified AI Platform
    - End-to-end AI development
    - Low-code/no-code prompt flow workflows
    - SDKs
- **Experimentation & Deployment**
  - Test ideas in the playground
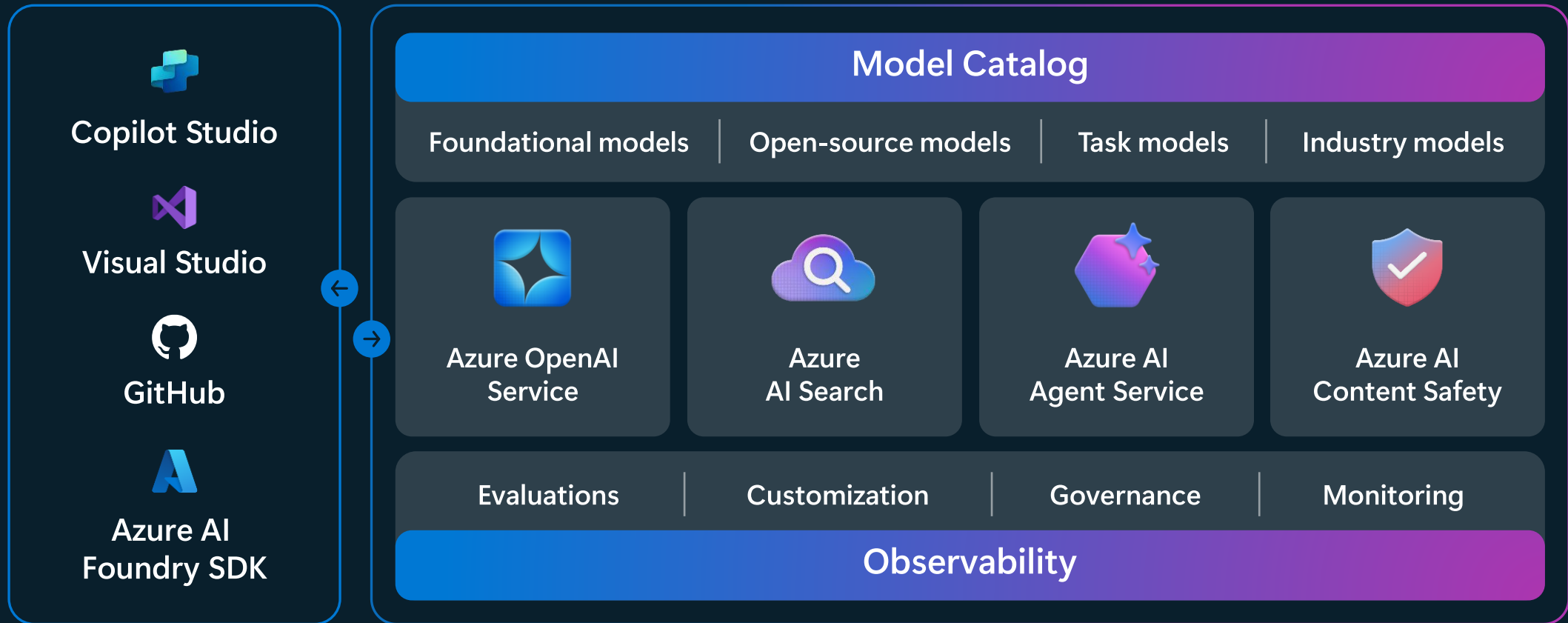 - Deploy to the cloud

Microsoft Foundry – Overview

- Why do you care?
  - Accelerate development of AI powered solutions
  - Highly available infrastructure
  - One stop place to organize, manage and maintain
    - Your Azure based AI solutions

Microsoft Foundry Overview

Copilot Studio

Visual Studio

GitHub

Azure AI Foundry SDK

Model Catalog

Foundational models	Open-source models	Task models	Industry models

Azure OpenAI Service

Azure AI Search

Azure AI Agent Service

Azure AI Content Safety

Evaluations	Customization	Governance	Monitoring

Observability

Microsoft Foundry – Overview
What is it

- **Collaborative Project Management**
  - Projects
    - Manage containers, resources, connectivity and security
- **Integration with Azure Services**
  - AI services
  - Azure Storage
  - Databases

Microsoft Foundry Overview

Ingest data

- Microsoft Fabric
- Azure SQL Database
- Cosmos DB
- Azure AI Search
- Azure Redis
- SharePoint
- Bing
- OSS Databases

Provision cloud resources

- Compute
- Storage
- Data
- Identity
- App Hosting
- AI Services

Open and Modular Platform

New Collaborations

accenture   gretel   KPMG

scale   STATSIG   Weights & Biases
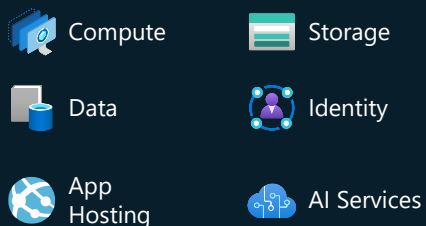
Orchestrate with OSS frameworks

- Semantic Kernel
- LangChain
- AutoGen
- LlamaIndex
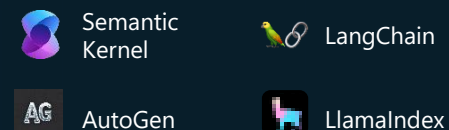
Leverage Azure App Services

- Functions
- Kubernetes
- Container Apps
- Logic Apps
- Dynamic Sessions
- PubSub

Monitor in production

- Azure Monitor
- Arize
- Azure App Config
- Weights & Biases
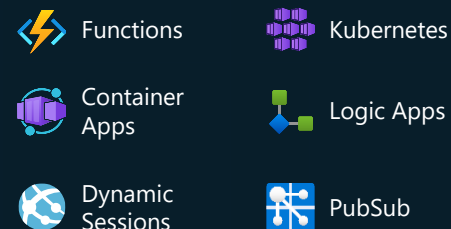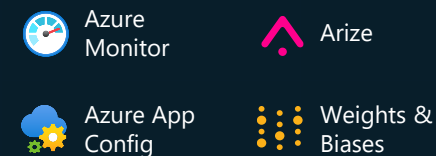
Microsoft Copilot Studio — **Microsoft Power Platform** — **Azure AI Foundry portal** — **Azure Machine Learning** — **GitHub Enterprise** — **Visual Studio Code** — **Azure DevOps** — **Microsoft Azure Portal**

Microsoft Foundry – Overview

- **Diverse AI models**
  - Over 11,000 models
    - Anthropic Claude
    - DeepSeek
    - Grok
    - Hugging Face
    - OpenAI
    - Microsoft
    - Mistral
    - Other open source and public models

Microsoft Foundry Overview

Support for diverse selection of models

Now 11,299 + Frontier, task, and Open Models

OpenAI
Model Family
(available day 1)

+

Phi SLM
Model Family

Mistral AI
Model Family

Meta Llama 2
Model Family

Jais G42
Model Family

Cohere
Model Family

Databricks
Model Family

Hugging Face
Collection

New

NTT DATA

New

BRIA

New

Industry
AI Models

Microsoft Foundry - Overview
What is it

- Responsible AI
  - Content Safety
  - Input & Output filters
  - Safety validation

Azure AI Content Safety

Harm categories

Groundedness detection

Prompt shields

Protected material detection

Microsoft Foundry - Overview
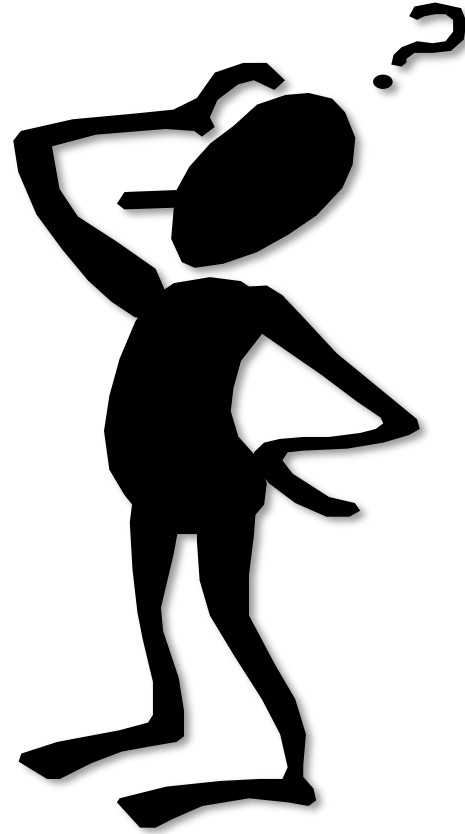
Creating a Foundry Resource

- **First** register the **microsoft.insights** resource provider

Creating a Foundry Resource

- **Next:** Navigate to **Home** and choose Foundry

Creating a Foundry Resource

- Fill out the card
  - Resource group,
  - Name
  - Region
    - East US2
      - Newer models
  - Default project name

Creating a Foundry Resource

- ## Setup KeyVault
  - ### Create new

Creating a Foundry Resource

■ Setup KeyVault

    ■ Create new

Creating a Foundry Resource

- Setup
  - App Insights
- Click
  - Review & create

Creating a Foundry Resource

- Review & Create - Accepts defaults for:
  - Network
  - Identify
  - Encryption
  - Tags

Creating a Foundry Resource

- Review Summary
  - Click Create

Create a Foundry resource ...

Basics   Storage   Network   Identity   Encryption   Tags   **Review + create**

⟲ View automation template

Basics

Subscription	2026-Lab_Joe_Ficara Demo Student
Resource group	rg_aifoundry
Name	ai-cscie94-foundry-demo-01
Region	East US
Default project name	01-lecture

Storage

Key Vault (preview)	kv-cscie94-demo-01
Application Insights	appi-cscie94-foundry-demo
Storage for Agents service	0 item(s)
Storage Account (preview)	

Network

Inbound Access	All networks, including the internet, can access this resource.

Identity

Identity type	System assigned

Previous   Next   Create ⑪

Configuring in Azure

- Once deployed navigate to the Foundry UI
  - Pick your foundry instance

Configuring in Azure

- ## Deploy a model
  - ### Pick **gpt-5-mini**
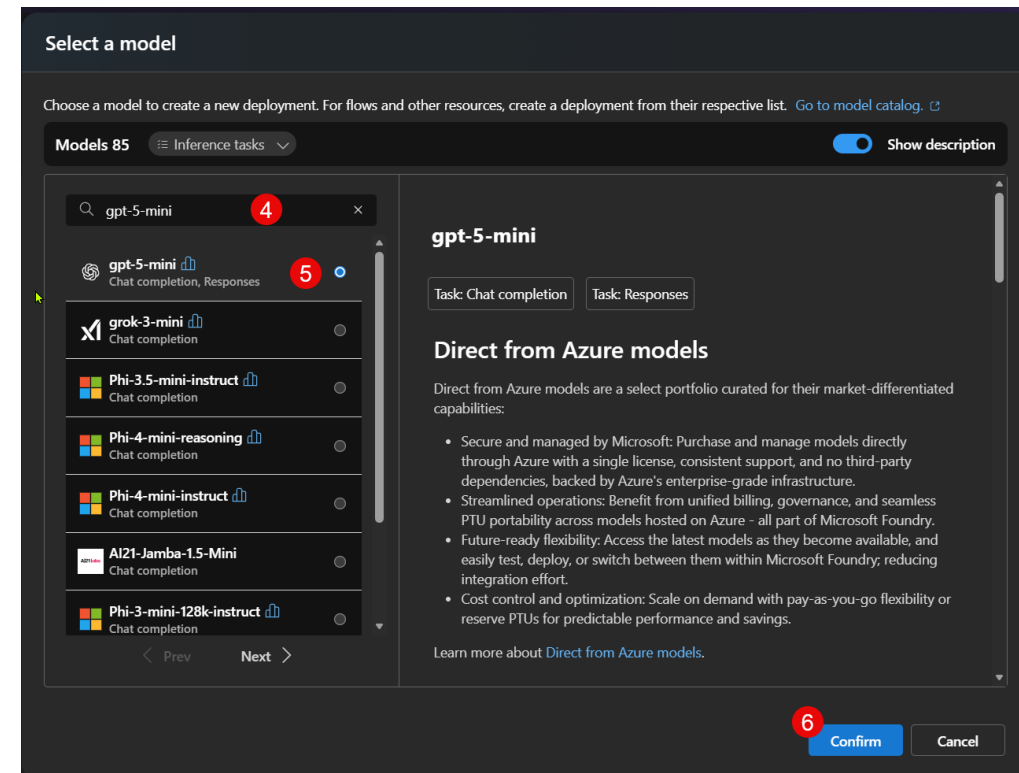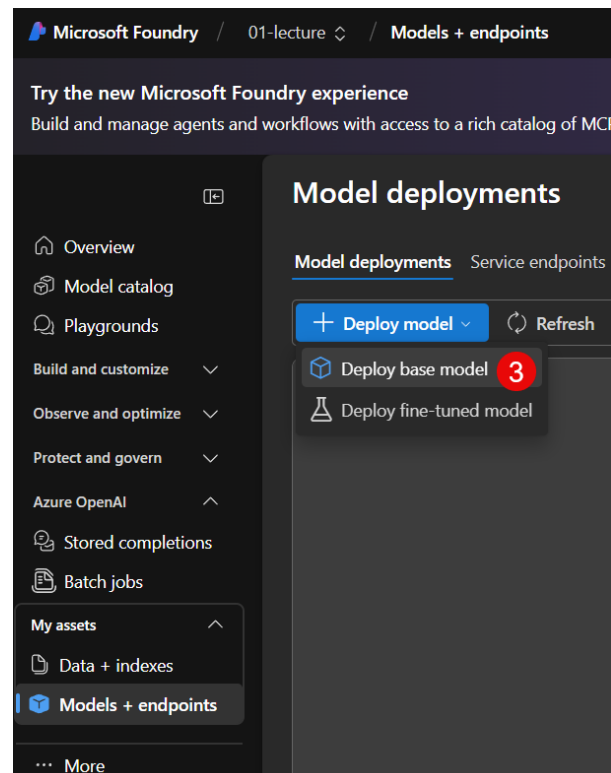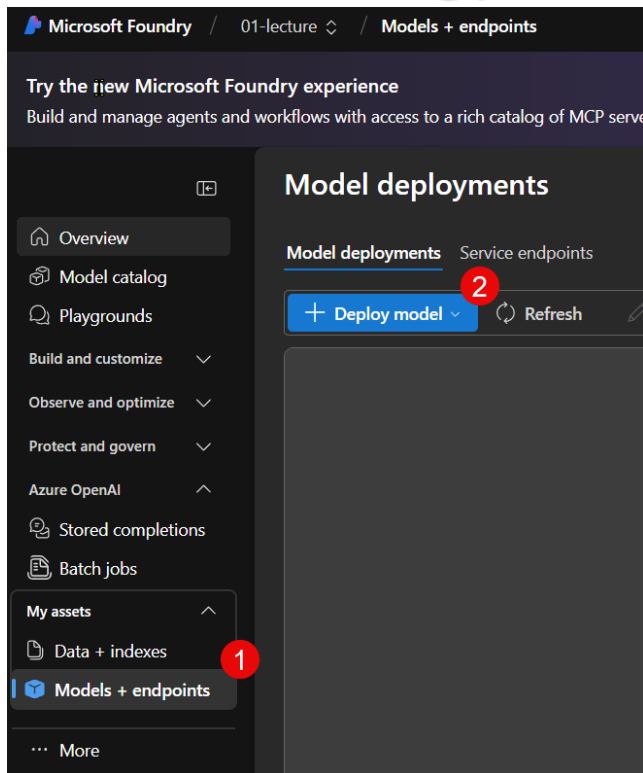
Configuring in Azure

- **Deploy a model**
  - Enter a Deployment Name
  - Choose a deployment type

Test in playground

- You can now experiment in the playground

Demo

Create Microsoft Foundry, Project & Test deployment

`Azure Portal`

Configuring in Azure

Let's Code!

Simple Chatbot

- **Basic Requirements:**

  - Allow a caller to pass in a prompt
    - Using a POST request

  - Use an **SLM** to process the content of the prompt
    - **SLM** -> **S**mall **L**anguage **M**odel

  - Return the result from the **SLM**
    - To the caller

Simple Chatbot

- Nuget packages needed

```xml
  <ItemGroup>
    <PackageReference Include="Microsoft.AspNetCore.OpenApi" Version="10.0.2" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="10.1.0" />
    <PackageReference Include="Azure.AI.OpenAI" Version="2.1.0" />
    <PackageReference Include="Microsoft.Extensions.AI" Version="10.2.0" />
    <PackageReference Include="Microsoft.Extensions.AI.OpenAI" Version="10.2.0-preview.1.26063.2" />
  </ItemGroup>
```

Simple Chatbot

- **How it works**
  - **Reads settings**
    - **Uri** to your Azure OpenAI Service in the connected resources
    - **ApiKey** to access the model
  - Connect to a deployed **gpt 5 mini SLM**
    - Using the settings defined by the **AISettings**
  - Passes the string provided in the body of the POST request
    - To the **SLM**
  - Returns result from the **SLM**
    - As the result of the **POST** request

Test in playground

- ## URL and API Key

  - ### Be sure to choose OpenAI to obtain the correct URL!
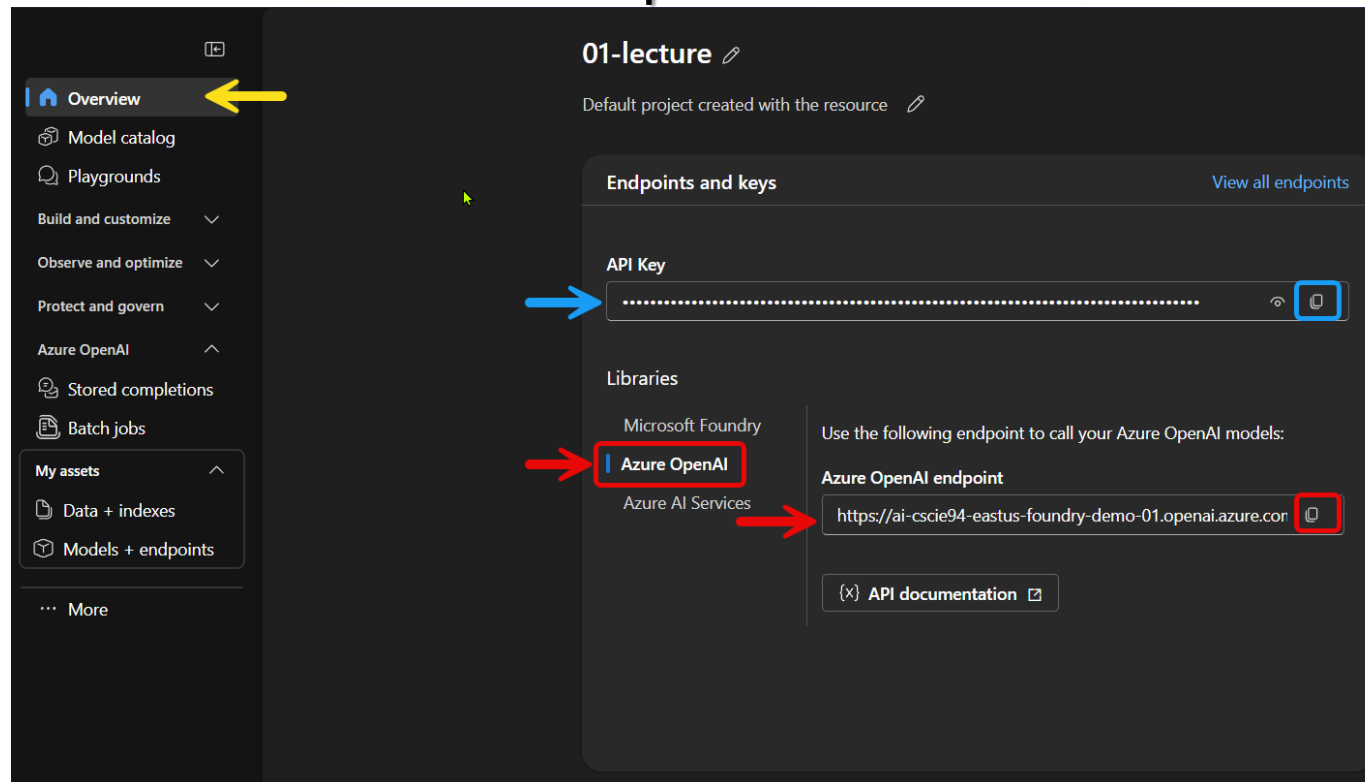
Simple Chatbot

- ## How do you do it?

  - ### Get an instance of `IChatClient`

    - Using dependency injection, in your controller's constructor

  - ### Create a `ChatOptions`

    - Include settings to direct the SLM

      - **Temperature (GPT 5 mini only accepts a Temperature of 1)**

        - The randomness of the SLM/LLM's responses

          - Lower values = more deterministic
          - Higher values = increase creativity

      - **MaxOutputTokens**

        - Maximum number of tokens the SLM/LLM can generate in a response

Simple Chatbot

- **Submit the prompt**
  - Using **`IChatClient`** instance & **`GetResponseAsync`** method
    - Provide
      - The **`ChatOptions`** instance
      - The **prompt** provided by the call to the REST API

- **Return the result from the SLM**
  - To the caller

Demo

Simple Chatbot

`SimpleWebAPIChatDemoSolution.sln`

`SimpleWebAPIChatDemo.csproj`

Structured Output JSON

- ## Why structured output?
  - ### Facilitates integration into your business logic
  - ### Makes it easier to deal with results
    - #### Strongly typed output
- ## How do you do it?
  - ### Get an instance of `IChatClient`
    - #### Using dependency injection
      - ##### In your controller's constructor

Structured Output JSON

- Nuget packages needed

```xml
  <ItemGroup>
    <PackageReference Include="Microsoft.AspNetCore.OpenApi" Version="10.0.2" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="10.1.0" />
    <PackageReference Include="Azure.AI.OpenAI" Version="2.1.0" />
    <PackageReference Include="Microsoft.Extensions.AI" Version="10.2.0" />
    <PackageReference Include="Microsoft.Extensions.AI.OpenAI" Version="10.2.0-preview.1.26063.2" />
  </ItemGroup>
```

Structured Output JSON
How do you do it ...

- **Create the shape you expect to be returned**
  - A public C# Class with public properties
    - Used to define the JSON structure to be returned

- **Generate the JSON Schema**
  - Using **Nuget** package `NJsonSchema`

- **Create a `ChatResponseFormatJson` instance**
  - Supply the schema
    - That defines the desired output format

Structured Output JSON
How do you do it ...

- Create a `ChatOptions`
  - Specify the `ResponseFormat`
    - With the `ChatResponseFormatJson` instance
  - Include settings to direct the SLM
    - Temperature
      - The randomness of the SLM/LLM's responses
        - Lower values = more deterministic
        - Higher values = increase creativity
    - MaxOutputTokens
      - Maximum number of tokens the SLM/LLM can generate in a response
    - Etc...

Structured Output JSON
How do you do it ...

- Submit the prompt using **GetResponseAsync**
  - Include the **ChatOptions** instance
  - Include an enhanced user **prompt**
    $"Identify and return a JSON list of the most important 3 key phrases from the following text: {prompt}";
    - The prompt in the interpolated C# string is the input from the user
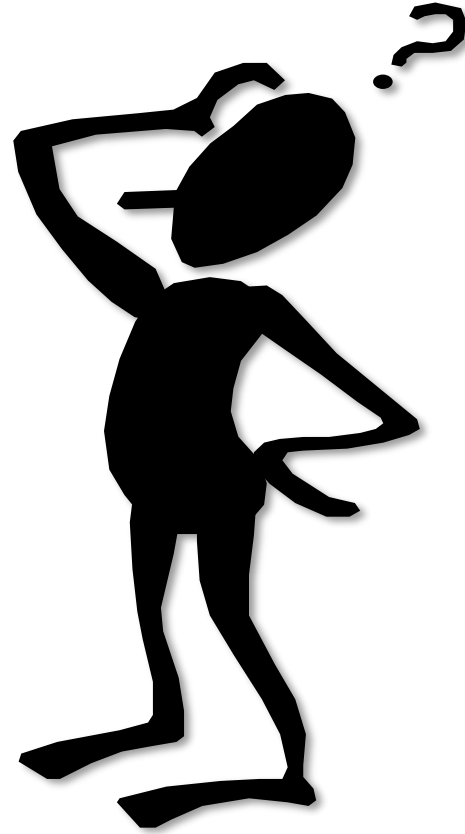
Demo

Structured Output JSON

```
StructuredOutputWebAPISolution.sln
StructuredOutputWebAPI.csproj
```

Questions

Further Reading

- Microsoft Foundry

  - [What is Microsoft Foundry? – Microsoft Foundry | Microsoft Learn](#)

  - [Microsoft Foundry architecture – Microsoft Foundry | Microsoft Learn](#)

  - [Microsoft Foundry frequently asked questions – Microsoft Foundry | Microsoft Learn](#)