# Develop an AI agent with VS Code extension

In this exercise, you'll use the Azure AI Foundry VS Code extension to create an agent that can use Model Context Protocol (MCP) server tools to access external data sources and APIs. The agent will be able to retrieve up-to-date information and interact with various services through MCP tools.

This exercise should take approximately **30** minutes to complete.

> **Note**: Some of the technologies used in this exercise are in preview or in active development. You may experience some unexpected behavior, warnings, or errors.

## Prerequisites

Before starting this exercise, ensure you have:

- Visual Studio Code installed
- An active Azure subscription

## Install the Azure AI Foundry VS Code extension

Let's start by installing and setting up the VS Code extension.

1. Open Visual Studio Code.
2. Select **Extensions** from the left pane (or press **Ctrl+Shift+X**).
3. In the search bar, type **Azure AI Foundry** and press Enter.
4. Select the **Azure AI Foundry** extension from Microsoft and click **Install**.
5. After installation is complete, verify the extension appears in the primary navigation bar on the left side of Visual Studio Code.

## Sign in to Azure and create a project

Now you'll connect to your Azure resources and create a new AI Foundry project.

1. In the VS Code sidebar, select the **Azure AI Foundry** extension icon.
2. In the Azure Resources view, select **Sign in to Azure…** and follow the authentication prompts.
3. After signing in, select your Azure subscription from the dropdown.
4. Create a new Azure AI Foundry project by selecting the **+** (plus) icon next to **Resources** in the Azure AI Foundry Extension view.
5. Choose whether to create a new resource group or use an existing one:

   **To create a new resource group:**

   - Select **Create new resource group** and press Enter
   - Enter a name for your resource group (e.g., "rg-ai-agents-lab") and press Enter
   - Select a location from the available options and press Enter

   **To use an existing resource group:**

   - Select the resource group you want to use from the list and press Enter
6. Enter a name for your Azure AI Foundry project (e.g., "ai-agents-project") in the textbox and press Enter.

7. Wait for the project deployment to complete. A popup will appear with the message “Project deployed successfully.”

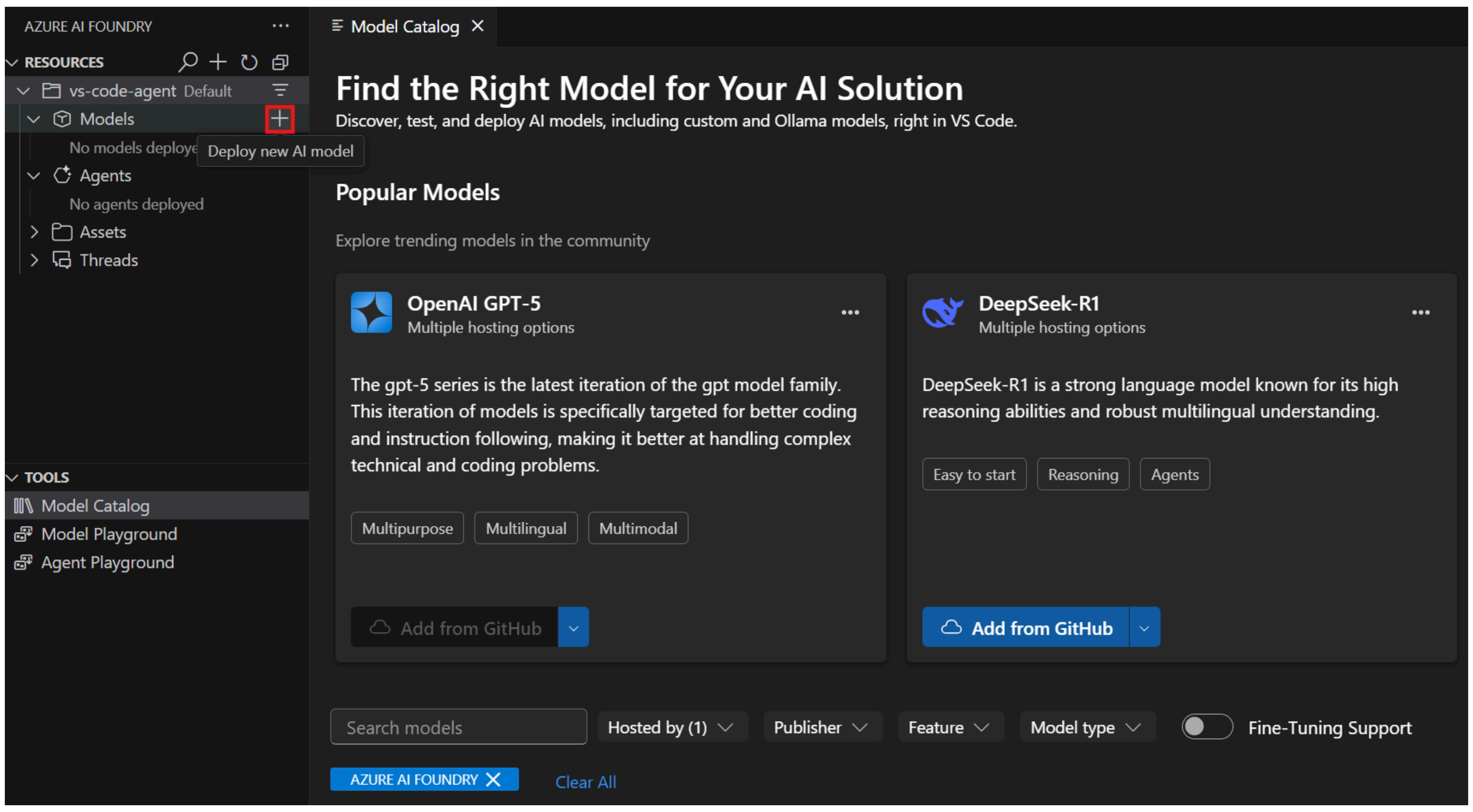
## Deploy a model

You’ll need a deployed model to use with your agent.

1. When the “Project deployed successfully” popup appears, select the **Deploy a model** button. This opens the Model Catalog.

    > **Tip**: You can also access the Model Catalog by selecting the **+** icon next to **Models** in the Resources section, or by pressing **F1** and running the command **Azure AI Foundry: Open Model Catalog**.

2. In the Model Catalog, locate the **gpt-4o** model (you can use the search bar to find it quickly).



Find the Right Model for Your AI Solution

Discover, test, and deploy AI models, including custom and Ollama models, right in VS Code.

Explore trending models in the community

The gpt-5 series is the latest iteration of the gpt model family. This iteration of models is specifically targeted for better coding and instruction following, making it better at handling complex technical and coding problems.

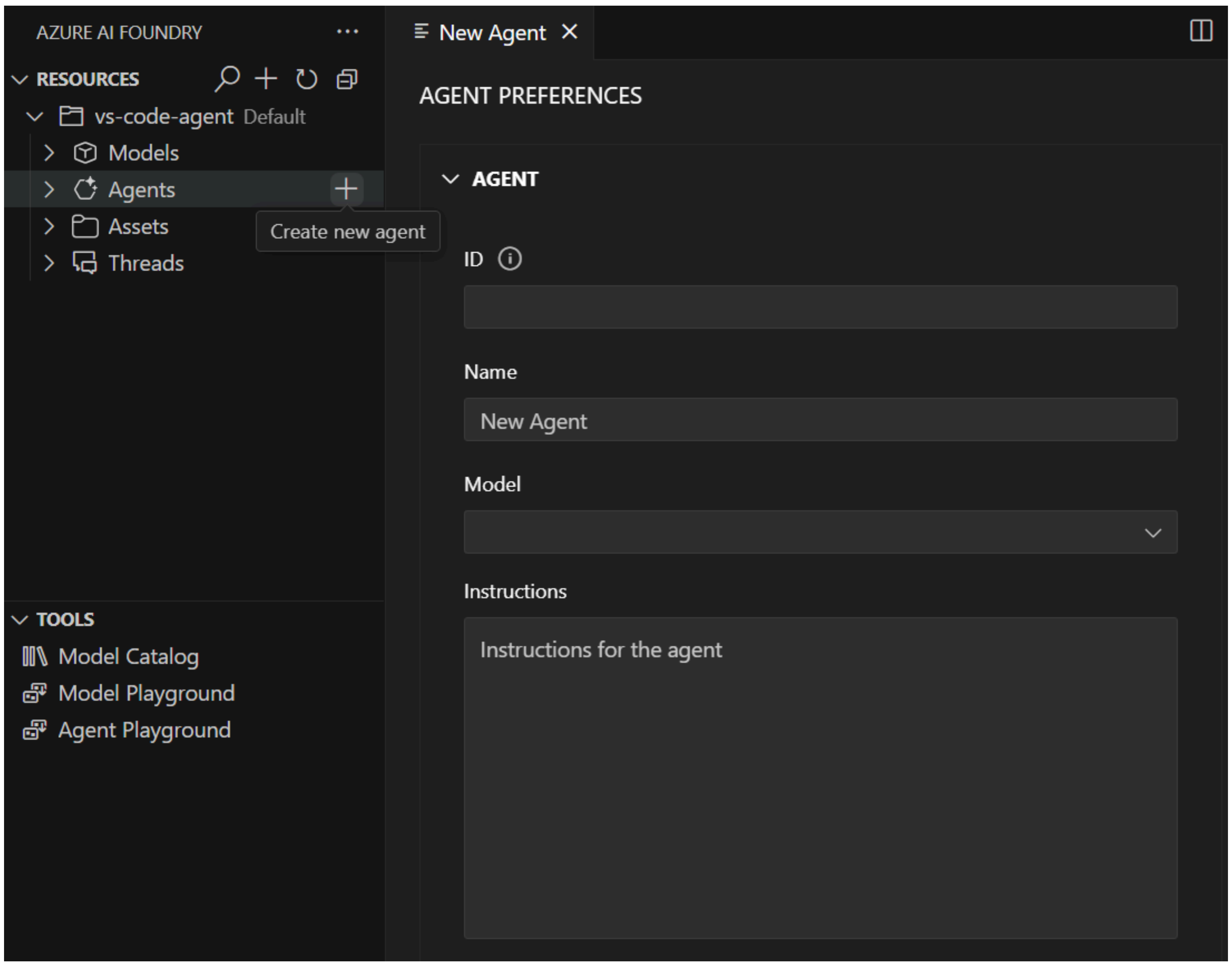
DeepSeek-R1 is a strong language model known for its high reasoning abilities and robust multilingual understanding.

3. Select **Deploy in Azure** next to the gpt-4o model.
4. Configure the deployment settings:
    - **Deployment name**: Enter a name like “gpt-4o-deployment”
    - **Deployment type**: Select **Global Standard** (or **Standard** if Global Standard is not available)
    - **Model version**: Leave as default
    - **Tokens per minute**: Leave as default
5. Select **Deploy in Azure AI Foundry** in the bottom-left corner.
6. In the confirmation dialog, select **Deploy** to deploy the model.
7. Wait for the deployment to complete. Your deployed model will appear under the **Models** section in the Resources view.

## Create an AI agent with the designer view

Now you’ll create an AI agent using the visual designer interface.

1. In the Azure AI Foundry extension view, find the **Resources** section.
2. Select the **+** (plus) icon next to the **Agents** subsection to create a new AI Agent.



3. Choose a location to save your agent files when prompted.

4. The agent designer view will open along with a `.yaml` configuration file.

### Configure your agent in the designer

1. In the agent designer, configure the following fields:

   - **Name**: Enter a descriptive name for your agent (e.g., “data-research-agent”)
   - **Description**: Add a description explaining the agent’s purpose
   - **Model**: Select your GPT-4o deployment from the dropdown
   - **Instructions**: Enter system instructions such as:

   Code Copy

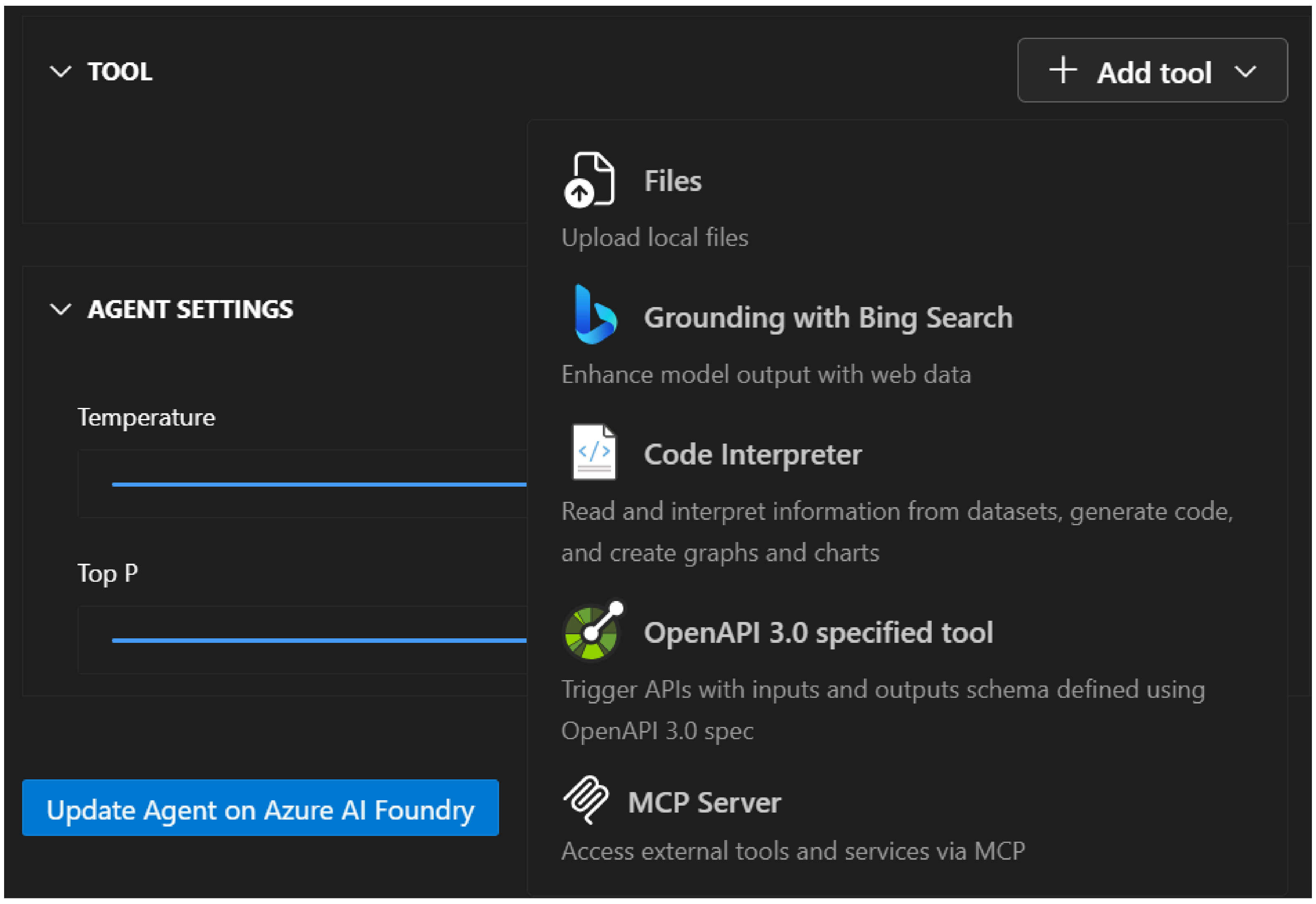
   ```
   You are an AI agent that helps users research information from various sources.
   Use the available tools to access up-to-date information and provide
   comprehensive responses based on external data sources.
   ```

2. Save the configuration by selecting **File > Save** from the VS Code menu bar.

## Add an MCP Server tool to your agent

You’ll now add a Model Context Protocol (MCP) server tool that allows your agent to access external APIs and data sources.

1. In the **TOOL** section of the designer, select the **Add tool** button in the top-right corner.


TOOL
Add tool
Files
Upload local files
Grounding with Bing Search
Enhance model output with web data
Code Interpreter
Read and interpret information from datasets, generate code, and create graphs and charts
OpenAPI 3.0 specified tool
Trigger APIs with inputs and outputs schema defined using OpenAPI 3.0 spec
MCP Server
Access external tools and services via MCP
AGENT SETTINGS
Temperature
Top P
Update Agent on Azure AI Foundry

1. From the dropdown menu, choose **MCP Server**.
2. Configure the MCP Server tool with the following information:

   - **Server URL**: Enter the URL of an MCP server (e.g., `https://gitmcp.io/Azure/azure-rest-api-specs`)
   - **Server Label**: Enter a unique identifier (e.g., “github_docs_server”)

3. Leave the **Allowed tools** dropdown empty to allow all tools from the MCP server.
4. Select the **Create tool** button to add the tool to your agent.

## Deploy your agent to Azure AI Foundry

1. In the designer view, select the **Create on Azure AI Foundry** button in the bottom-left corner.

2. Wait for the deployment to complete.

3. In the VS Code navbar, refresh the **Azure Resources** view. Your deployed agent should now appear under the **Agents** subsection.

## Test your agent in the playground

1. Right-click on your deployed agent in the **Agents** subsection.

2. Select **Open Playground** from the context menu.

3. The Agents Playground will open in a new tab within VS Code.

4. Type a test prompt such as:

   Code

   ```
   Can you help me find documentation about Azure Container Apps and provide an example
   of how to create one?
   ```

5. Send the message and observe the authentication and approval prompts for the MCP Server tool:

   - For this exercise, select **No Authentication** when prompted.
   - For the MCP Tools approval preference, you can select **Always approve**.

6. Review the agent’s response and note how it uses the MCP server tool to retrieve external information.

7. Check the **Agent Annotations** section to see the sources of information used by the agent.

## Generate sample code for your agent

1. Right-click on your deployed agent and select **Open Code File**, or select the **Open Code File** button in the Agent Preferences page.
2. Choose your preferred SDK from the dropdown (Python, .NET, JavaScript, or Java).
3. Select your preferred programming language.
4. Choose your preferred authentication method.
5. Review the generated sample code that demonstrates how to interact with your agent programmatically.

You can use this code as a starting point for building applications that leverage your AI agent.

## View conversation history and threads

1. In the **Azure Resources** view, expand the **Threads** subsection to see conversations created during your agent interactions.
2. Select a thread to view the **Thread Details** page, which shows:
   - Individual messages in the conversation
   - Run information and execution details
   - Agent responses and tool usage
3. Select **View run info** to see detailed JSON information about each run.

## Summary

In this exercise, you used the Azure AI Foundry VS Code extension to create an AI agent with MCP server tools. The agent can access external data sources and APIs through the Model Context Protocol, enabling it to provide up-to-date information and interact with various services. You also learned how to test the agent in the playground and generate sample code for programmatic interaction.

## Clean up

When you've finished exploring the Azure AI Foundry VS Code extension, you should clean up the resources to avoid incurring unnecessary Azure costs.

### Delete your agents

1. In the Azure AI Foundry portal, select **Agents** from the navigation menu.
2. Select your agent and then select the **Delete** button.

### Delete your models

1. In VS Code, refresh the **Azure Resources** view.
2. Expand the **Models** subsection.
3. Right-click on your deployed model and select **Delete**.

### Delete other resources

1. Open the [Azure portal](https://portal.azure.com).
2. Navigate to the resource group containing your AI Foundry resources.
3. Select **Delete resource group** and confirm the deletion.