Microsoft Microhacks

Hands-On Lab: Agentic AI

Contents

Overview

The Micohack event is designed to engage technical roles through a condensed, half-day hands-on hack experience. Leveraging the latest Microsoft technologies, this event provides participants with the opportunity to work on real-world problems, collaborate with peers, and explore innovative solutions.

The Microhack event is divided into several key challenges, each carefully crafted to test and expand the participants' proficiency with Microsoft's suite of tools. These challenges are not only technical in nature but also reflect real-world scenarios that businesses face, providing a comprehensive understanding of how to apply theoretical knowledge practically.

Hack Duration: 2 hours

The event kicks off with an initial overview of the customer scenario for the business problem the participants will solve by leveraging cutting-edge technology and services.

Following this, the team will complete the setup phase, where participants ensure that their development environments are correctly configured, and all necessary tools are ready for use.

Finally, they will tackle the first challenge, which involves identifying key ideas that underpin the implementation of Microsoft technologies in solving predefined problems.

Customer Scenario

Background

Contoso is a renowned company celebrated for its extensive catalog of outdoor camping products that span across various domains of the camping industry. With a history of innovation and excellence, Contoso has firmly established itself as a leader in providing cutting-edge camping solutions. The company offers a diverse range of products, including tents, sleeping bags, cooking equipment, and portable power devices, catering to both recreational campers and professional adventurers. Contoso's commitment to quality and performance is evident in its continuous development of state-of-the-art solutions that address the evolving needs of its global customer base.

The company's product portfolio is designed to enhance outdoor experiences, streamline camping operations, and foster a connection with nature. Contoso's flagship offerings, such as its robust all-season tents, advanced portable power generators, and innovative multi-functional tools, empower outdoor enthusiasts to harness the power of technology for an enhanced camping experience. Additionally, Contoso's camping gear, from lightweight backpacks to durable cookware, is engineered to deliver superior performance and reliability. With a customer-centric approach, Contoso not only provides top-tier products but also ensures exceptional support and services, solidifying its reputation as a trusted partner in the outdoor camping industry.

In addition to its impressive product offerings, Contoso also boasts an extensive and complex employee benefits program designed to attract and retain top talent. This comprehensive package includes competitive salaries, health and wellness programs, retirement savings plans, and professional development opportunities. The company's benefits are meticulously designed to address the diverse needs of its employees, fostering a supportive and engaging work environment. From flexible working arrangements to generous vacation policies, Contoso ensures that its workforce has the resources and support necessary to thrive both professionally and personally.

Business Problem

Despite its many successes, Contoso faces significant business challenges due to the need to leverage AI for understanding and extracting data quickly. The sheer volume and variety of products create complexity in managing and synchronizing information, updates, and support services. This complexity is further compounded by the integration of diverse sales data, each requiring meticulous coordination to ensure seamless interoperability and optimal performance.

Contoso aims to streamline its product catalog discover processes and modernize its product and sales navigation practices by leveraging AI-driven solutions for automating the understanding and extraction of data. This approach will ensure that information remains relevant and user-friendly. Furthermore, consolidating product information into a unified, easily navigable platform will enhance both customer experience and operational efficiency, enabling Contoso to maintain its reputation as a leader in customer satisfaction.

These efforts are part of Contoso's broader strategy to optimize its business operations and support its diverse product ecosystem. By tackling the complexities associated with its large catalog and legacy documentation through AI technologies, Contoso is poised to deliver even greater value to its customers and continue its legacy of excellence in the tech industry.

Technical Problem

To address Contoso's complex business challenges, Microsoft AI Agent technologies will play a pivotal role in transforming their operations and enhancing both customer and employee experiences. Contoso would like to explore low-code platforms such as Copilot Studio and pro-code platforms like Azure AI Foundry.

By integrating AI Agents into Contoso's systems, the company can significantly improve the management and synchronization of their extensive product catalog and sales data. These AI-driven solutions will automate the understanding and extraction of information from various unstructured documents, such as PDFs, and seamlessly integrate it with structured data stored in relational databases such as SQL. This will ensure that all information remains relevant, up-to-date, and easily accessible for both internal teams and customers.

In addition to improving employee support, the integration of AI Agents will streamline Contoso's product catalog discovery processes and modernize their product and sales navigation practices. By consolidating product information into a unified, easily navigable platform, AI Agents will enhance the customer experience and operational efficiency. Customers will be able to quickly find relevant products and receive personalized recommendations based on their preferences and needs. This will solidify Contoso's reputation as a leader in customer satisfaction and reinforce its commitment to leveraging cutting-edge technology for superior service delivery.

AI Agents are not always logical, and humans need to verify their work to ensure it coherent and valid answers. The AI agent enables the analyst to scale their work by processing the

data but the analyst should not be over reliant on the agent to assume the results are always accurate and need to do their own due diligence.

Goals

For today's hack, you are a business analyst working in the marketing department of Contoso Outdoor Company, a retail company who has adopted Digital transformation in the past. Your goal is to optimize campaigns and product purchases based on customer preferences by building a conversational AI agent as a tool integrated into Contoso's business applications to answer questions about sales data. However, the scale of the product catalog and wealth of data is challenging to query and blocking your ability to gain insights.

To address these challenges – Contoso has chosen you and your team, to help with a series of challenges. The goal is to deploy multiple Agent solutions to validate the use-case and technologies can address Contoso's business challenges.

Ultimately, the deployment of Microsoft AI Agent technologies will play a crucial role in Contoso's broader strategy to optimize its business operations and support its diverse product ecosystem. By addressing the complexities associated with their large catalog and legacy documentation, Contoso will be able to deliver even greater value to their customers and maintain their legacy of excellence. The adoption of these advanced AI solutions will position Contoso at the forefront of innovation in the outdoor camping industry, ensuring their continued success in meeting the evolving needs of their global customer base.

Challenges

Challenge 1: Copilot Studio (GUI)

Overview

Your challenge is to leverage Microsoft Copilot Studio to create an Agent that can provide information on Contoso's product catalog. This agent will also include automation orchestration to take in email questions sent to your email address around Contoso products.

As you embark on utilizing Copilot Studio and the larger Power Platform, remember that these tools are designed to be iterative, allowing continuous testing and improvement throughout the building process to ensure optimal performance and adaptability. We will test, adjust, and test some more.

Related Technologies

Microsoft Copilot Studio

- Copilot Studio is a graphical, low-code tool for both creating an agent—including building automations with Power Automate—and extending a Microsoft 365 Copilot with your own enterprise data and scenarios.
- One of the standout features of Copilot Studio is its ability to connect to other data sources using either prebuilt or custom plugins. This flexibility enables users to create and orchestrate sophisticated logic, ensuring that their agent experiences are both powerful and intuitive.
- The platform's low-code experience puts the power of AI at the user's fingertips, making it accessible even to people without extensive technical backgrounds.

Microsoft Power Platform

- Discover how to make the most of Microsoft Power Platform products with online training courses, docs, and videos covering product capabilities and how-to guides. Learn how to quickly and easily build custom apps using Power Apps, automate workflows to improve business productivity using Power Automate, analyze data for insights using Power BI, and rapidly design, configure, and publish modern websites using Power Pages.

- Use Microsoft Power Platform to build apps using Power Apps, automate tasks using Power Automate, analyze data using Power BI, and create websites using Power Pages.
- Enhance apps and business processes with AI Builder, connect to a variety of data sources through Connectors, use Copilot and generative AI capabilities to boost productivity, use powerful data service to quickly build enterprise-grade apps with Dataverse, and use low-code strongly typed declarative and functional language with Power Fx.

Microsoft Power Automate

- With its automation capabilities, Power Automate helps you streamline your business processes and automate repetitive tasks. Its intuitive interface and many connectors allow you to create workflows with little to no knowledge of coding. You can drag and drop components and set up workflows to save time and improve efficiency. Power Automate can handle simple tasks like sending notifications as well as more complex processes across multiple apps and services. It's flexible and scalable, making it useful for various automation needs in a modern workplace.
- Copilot in Power Automate accelerates your journey to adopting automation and transforming your processes. It enhances these scenarios by using the instructions you give Copilot written in natural language to surface possible solutions that can achieve desired results. Copilot stays with you all the way during creation to guide you through your entire process.

Azure Content Safety

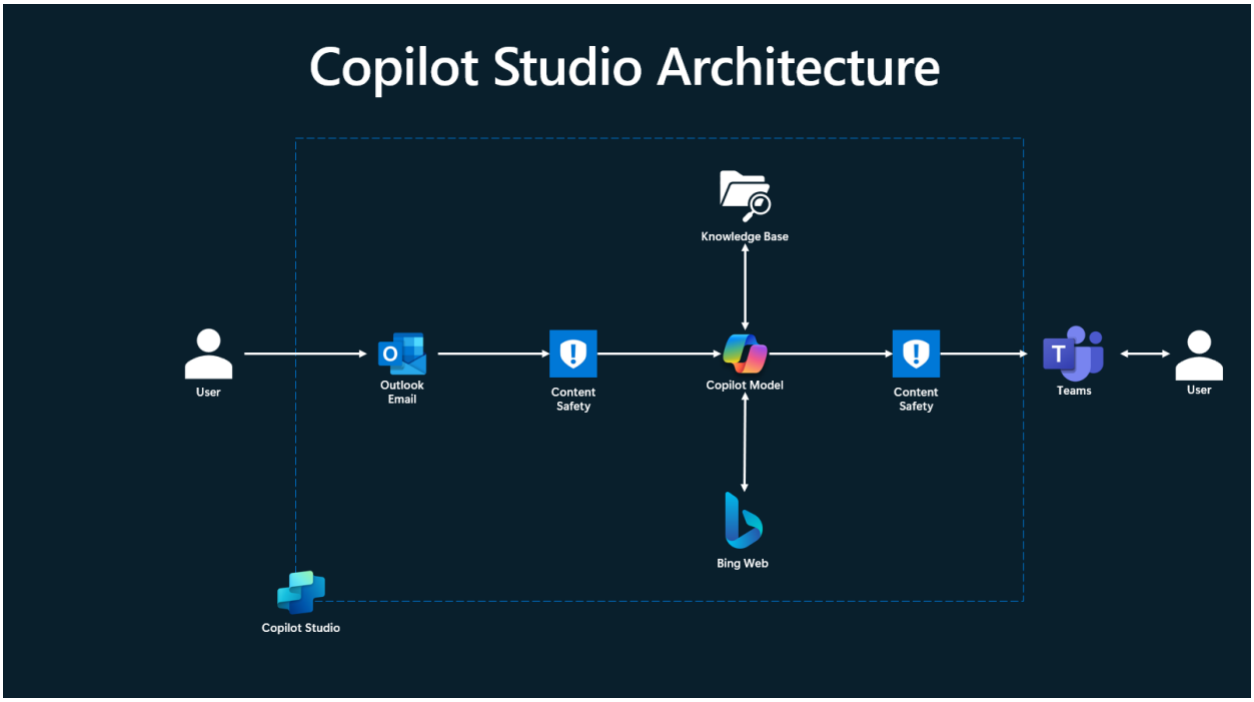
- Azure AI Content Safety is an AI service that detects harmful user-generated and AI-generated content in applications and services. Azure AI Content Safety includes text and image APIs that allow you to detect material that is harmful. The interactive Content Safety Studio allows you to view, explore, and try out sample code for detecting harmful content across different modalities.
- Content filtering software can help your app comply with regulations or maintain the intended environment for your users.
- Included features:
    - Prompt Shields
    - Grounded-ness Detection
    - Protected Material Text Detection

- Custom Categories
- Moderate Text Content
- Moderate Image Content
- Monitor Online Activity

Architecture

Basic Architecture

Copilot Studio – Agent Architecture

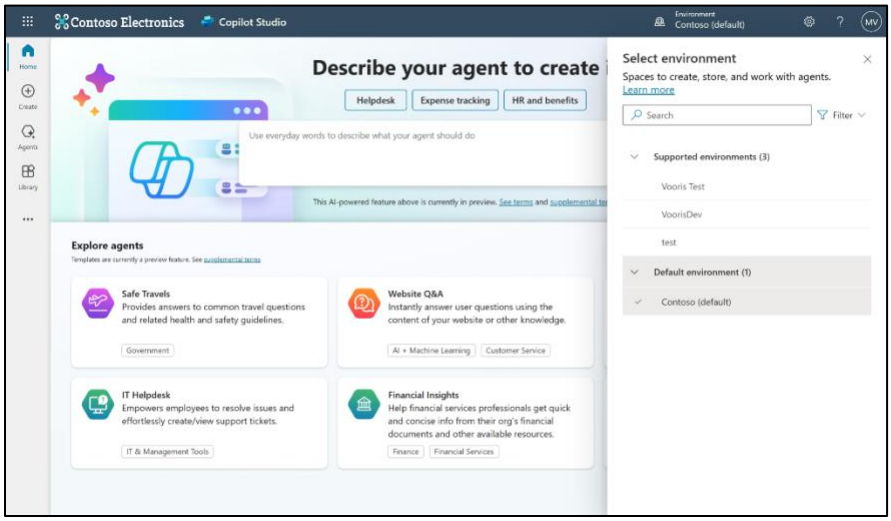


Prerequisites

- Have a subscription to Microsoft Copilot Studio

Steps

1. Review challenge Success Criteria before starting
2. Download the GitHub repository:
    a. https://github.com/Boykai/octo-microhack-agentic-ai
3. Locate **resources/contoso-tents-datasheet.pdf**
4. Navigate to Microsoft Copilot Studio
5. Select the **Environment** button, in the top right corner
6. Select **your environment** from the list



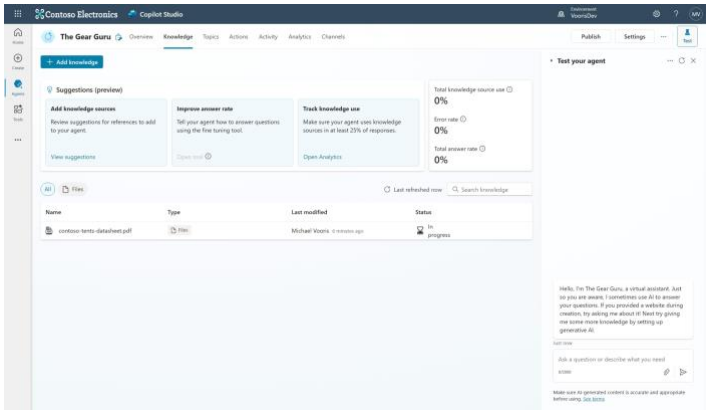
Creating an Agent:

7. Select **+ Create** button in the left navigation
8. Select **+ New agent** button
9. In the chat text box, enter the following:
    a. Use the Copilot Chat interface to describe your agent. Based off the use case provided you can use the following as an example "***I want to create an agent that will answer product questions and provide product recommendations***". Continue to chat with the copilot to build our out agent.
10. Select the **Create** button, top right corner
11. Select the **chat text box** in the testing window
12. Try the following prompts:
    a. What brands of tents do we sell?
    b. What beginner tents do our competitors sell? Include prices.

c. What are the specifications of the latest model of AlpineGear tents?
d. Can you recommend a product for light weight camping?
e. How does Alpine Explorer compare to Skyview 2-Person tent?
f. I am going camping in the Moab area around Thanksgiving, can you recommend a tent for that weather?
g. Show the tents we sell by region that are a similar price to our competitors beginner tents.
h. Show the tents we sell by region that are a similar price to our competitors beginner tents?
i. Show as a bar chart?

Knowledge Bases:

13. Select the **+ Add Knowledge** button
14. Select the **click to browse** button
15. Navigate to the **resources/contoso-tents-datasheet.pdf**
16. Select **Open** to upload the file
17. Select the **Add** button
18. You can check on the **Knowledge** tab to see if the document is still uploading



19. Select the **chat text box** in the testing window
20. Try the prompts from Step #12

Disabling General Knowledge:

21. Locate the **Allow the AI to use its own general knowledge**, in the Knowledge section
22. Select the toggle button to disable **General Knowledge**
23. Select the **Continue** button, in the Disabling the default AI knowledge pop-up
24. Select the **Refresh** button, in the test window
25. Select the **chat text box** in the testing window
26. Try the prompts from Step #12

Instructions:

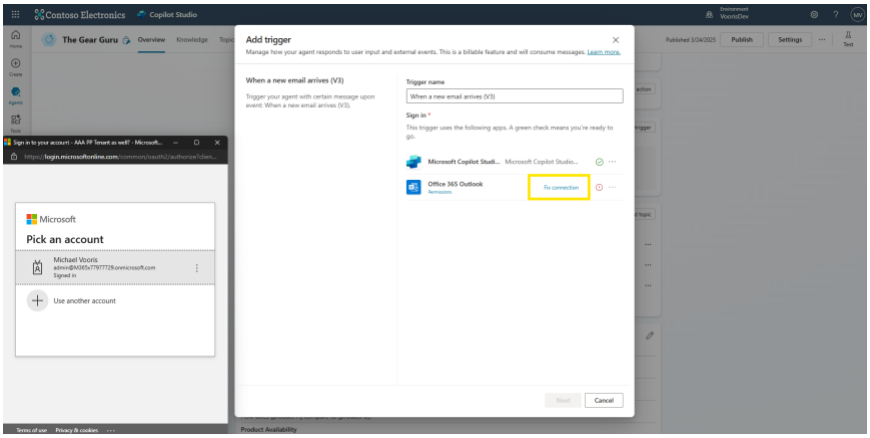
27. Locate the **Details** section in the **Overview Tab**
28. Select the **Edit** button
29. Update the **Instructions** by adding the following details:
    a. See **src/workshop/instructions/sales_agent_instructions.txt**
30. Select the **Save** button
31. Try the prompts from Step #12

Additional Enhancements:

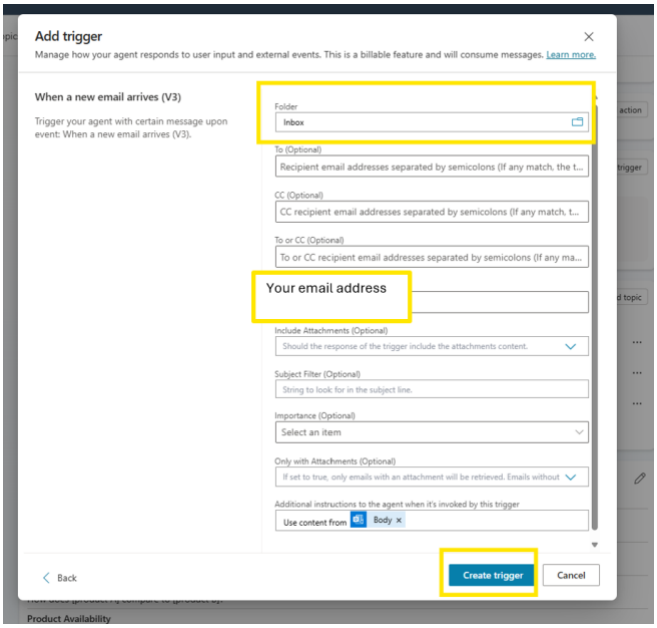
32. Locate the **Allow the AI to use its own general knowledge**, in the Knowledge section
33. Select the toggle button to disable **General Knowledge**
34. Select the **General Knowledge** toggle button to re-enable General Knowledge
35. Select the **Save** button
36. Select the **Settings** button
37. Update the **Name** and **Icon** to desired values
38. Select the **Save** button
39. Select the **Generative AI** button
40. Select the **Medium – More Balanced** radio button in the Content Moderation section
41. Select the **Save** button
42. Explore the remaining options then close the **Settings** window
43. Select the **Overview** tab
44. Locate the **Starter Prompt** section
45. Select the **Edit** button
46. Update the **Starter Prompts** with the prompts from Step #12, add **Titles**
47. Select the **Save** button

Triggers (Automation):

48. Locate the **Orchestration** toggle button
49. Select the **Orchestration** toggle button, to enable orchestration
50. Locate the **Triggers** section
51. Select the **+ Add trigger** button
52. Select **When a new email arrives (v3)** trigger button
    a. Note: you may two of them specifically look for the one that shows *Office 365 Outlook* that is the one that is tried to your Entra.
    b. Create or Fix connections, as needed:



53. Select the **Next** button
54. Update the following fields:
    a. Optional Fields: You can set the optional fields as you see fit. I have set the From field to my account. This will only set the trigger to fire if the email is from my account and no others.



55. Select the **Create trigger** button
56. Draft an email
    a. Tip: Try using the prompts from Step #12 as the email body, perhaps starting with “answer the following questions”
57. Send the email, so that the email triggers based on the agent trigger settings
58. Locate to the **Test your agent** pane
59. Select the **…** button
60. Select **Test trigger** from the drop-down menu
61. Select **When a new email arrives (V3)** from the options

62. Select the desired **date and time**
63. Select the **Start testing** button
64. Review the **Activity map** results
65. Review the results in the **Test your agent** pane

Publishing:

66. Select the **Channels** tab
67. Select the **Teams + Microsoft 365** button
68. Select desired **Teams channel** checkbox button
69. Select **Add channel** button
70. Select **Overview** tab
71. Select **Publish** button, top right
72. Open **Teams channel** in Microsoft Teams
73. Try prompts from Step #12
74. Done!

Success Criteria

☐ Successfully completed all Challenge 1 Steps without error

☐ Successfully create an agent in Copilot Studio

☐ Successfully add agent Knowledge Base

☐ Successfully disable/enable agent General Knowledge

☐ Successfully update and improve agent Instructions

☐ Successfully update additional agent settings

☐ Successfully add agent Automation

☐ Successfully add an agent with Teams integration

☐ Successfully publish an agent

Useful Resources

- Microsoft Copilot Studio documentation | Microsoft Learn
- Key concepts - Authoring agents - Microsoft Copilot Studio | Microsoft Learn
- Knowledge Bases - Microsoft Copilot Studio | Microsoft Learn
- Topics - Microsoft Copilot Studio | Microsoft Learn
- Triggers - Microsoft Copilot Studio | Microsoft Learn
- Actions - Microsoft Copilot Studio | Microsoft Learn
- Understand error codes - Microsoft Copilot Studio | Microsoft Learn
- Responsible AI FAQs - Microsoft Copilot Studio | Microsoft Learn
- Connectors overview | Microsoft Learn
- Azure AI Content Safety documentation | Microsoft Learn

Challenge 2: Azure AI Foundry (GUI)

Overview

We will set up the initial Azure environment for you to build on top of during your Microhack. This comprehensive setup includes configuring essential Azure services and ensuring access to all necessary resources. Participants will familiarize themselves with architecture, gaining insights into how various components interact to create a cohesive solution. With the foundational environment in place, the focus will shift seamlessly to the first Microhack Challenge endeavor.

Your challenge is to leverage Azure AI Foundry to create an Agent that can provide information on Contoso's product catalog. This agent will also include automation to take in email questions sent to your email address around Contoso products.

As you embark on utilizing Azure AI Foundry and the larger Azure platform, remember that these tools are designed to be iterative, allowing continuous testing and improvement throughout the building process to ensure optimal performance and adaptability. We will test and adjust and test some more.

Related Technologies

Azure CLI

- The Azure Command-Line Interface (CLI) is a cross-platform command-line tool to connect to Azure and execute administrative commands on Azure resources. It allows the execution of commands through a terminal using interactive command-line prompts or a script.
- For interactive use, you first launch a shell such as cmd.exe on Windows, or Bash on Linux or macOS, and then issue a command at the shell prompt. To automate repetitive tasks, you assemble the CLI commands into a shell script using the script syntax of your chosen shell, and then you execute the script.

Azure Bicep

- Bicep is a domain-specific language that uses declarative syntax to deploy Azure resources. In a Bicep file, you define the infrastructure you want to deploy to Azure and then use that file throughout the development lifecycle to repeatedly deploy that infrastructure. Your resources are deployed in a consistent manner.

- Bicep provides concise syntax, reliable type safety, and support for reusing code. Bicep offers a first-class authoring experience for your infrastructure-as-code solutions in Azure.
- Support for all resource types and API versions: Bicep immediately supports all preview and GA versions for Azure services. As soon as a resource provider introduces new resource types and API versions, you can use them in your Bicep file. You don't need to wait for tools to be updated before using the new services.
- Simple syntax: When compared to the equivalent JSON template, Bicep files are more concise and easier to read. Bicep doesn't require prior knowledge of programming languages. Bicep syntax is declarative and specifies which resources and resource properties you want to deploy.

Azure AI Foundry

- Azure AI Foundry provides a unified platform for enterprise AI operations, model builders, and application development. This foundation combines production-grade infrastructure with friendly interfaces, ensuring organizations can build and operate AI applications with confidence.
- Azure AI Foundry is designed for developers to:
    - Build generative AI applications on an enterprise-grade platform.
    - Explore, build, test, and deploy using cutting-edge AI tools and ML models, grounded in responsible AI practices.
    - Collaborate with a team for the full life-cycle of application development.
- With Azure AI Foundry, you can explore a wide variety of models, services and capabilities, and get to building AI applications that best serve your goals. Azure AI Foundry facilitates scalability for transforming proof of concepts into full-fledged production applications with ease. Continuous monitoring and refinement support long-term success.

Azure OpenAI

- Azure OpenAI Service provides REST API access to OpenAI's powerful language models including o3-mini, o1, o1-mini, GPT-4o, GPT-4o mini, GPT-4 Turbo with Vision, GPT-4, GPT-3.5-Turbo, and Embeddings model series. These models can be easily adapted to your specific task including but not limited to content generation,

summarization, image understanding, semantic search, and natural language to code translation. Users can access the service through REST APIs, Python/C#/JS/Java/Go SDKs.

- At Microsoft, we're committed to the advancement of AI driven by principles that put people first. Generative models such as the ones available in Azure OpenAI have significant potential benefits, but without careful design and thoughtful mitigations, such models have the potential to generate incorrect or even harmful content. Microsoft has made significant investments to help guard against abuse and unintended harm, which includes incorporating Microsoft's principles for responsible AI use, adopting a Code of Conduct for use of the service, building content filters to support customers, and providing responsible AI information and guidance that customers should consider when using Azure OpenAI.

Azure Grounding with Bing Search

- Grounding with Bing Search allows your Azure AI Agents to incorporate real-time public web data when generating responses. You need to create a Grounding with Bing Search resource, and then connect this resource to your Azure AI Agents. When a user sends a query, Azure AI Agents decide if Grounding with Bing Search should be leveraged or not. If so, it will leverage Bing to search over public web data and return relevant chunks. Lastly, Azure AI Agents will use returned chunks to generate a response.
- You can ask questions such as "what is the top news today" or "what is the recent update in the retail industry in the US?", which require real-time public data.
- Developers and end users don't have access to raw content returned from Grounding with Bing Search. The model response, however, includes citations with links to the websites used to generate the response, and a link to the Bing query used for the search. You can retrieve the model response by accessing the data in the thread that was created. These two references must be retained and displayed in the exact form provided by Microsoft, as per Grounding with Bing Search's Use and Display Requirements. See the how to display Grounding with Bing Search results section for details.

Azure AI Agent Service

- Azure AI Agent Service is a fully managed Azure service with SDKs for Python and C#. AI Agents are developed with Azure AI Agent Service and the SDK to enable

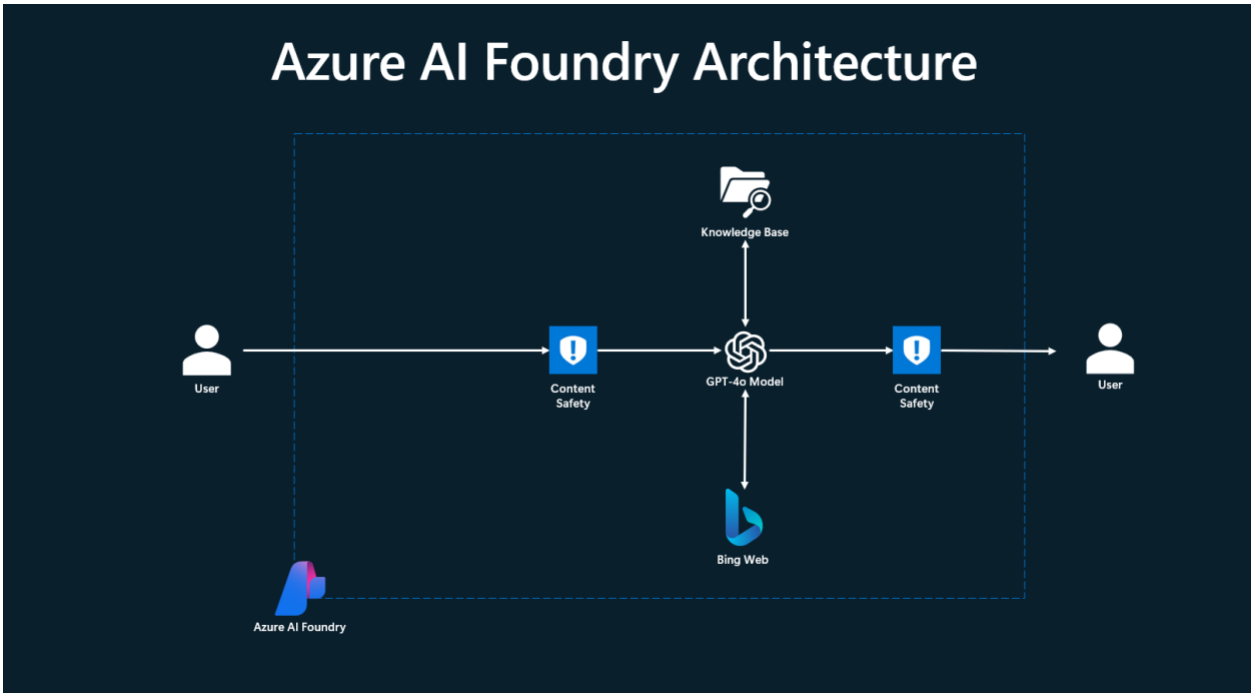
organizations to build tools into their applications for end-users to query data with NLP.

- The Azure AI Agent Service offers several advantages over traditional agent platforms:
    - **Rapid Deployment**: Optimized SDK for fast deployment, letting developers focus on building agents.
    - **Scalability**: Designed to handle varying user loads without performance issues.
    - **Custom Integrations**: Supports Function Calling for extending agent capabilities.
    - **Built-in Tools**: Includes Fabric, SharePoint, Azure AI Search, and Azure Storage for quick development.
    - **RAG-Style Search**: Features a built-in vector store for efficient file and semantic search.
    - **Conversation State Management**: Maintains context across multiple interactions.
    - **AI Model Compatibility**: Works with various AI models.
- Azure AI Agent Service is just one component of the AI Agent framework. There are many market leading agent frameworks like Semantic Kernel, LangChain, CrewAI. The Azure AI Agent Service is complementary to these frameworks. It accelerates the time to market and enables you to build robust business applications that scale.

Architecture

Basic Architecture

Azure AI Foundry – Agent Architecture



Prerequisites

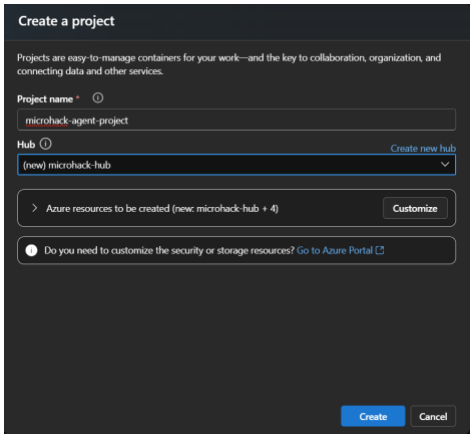
- To have a subscription in Azure
- Initiate an Azure AI services creation and agree to the Responsible AI terms **
    - ** If you have not created an Azure AI service resource in the subscription before
- To have an account in GitHub
- To have VS Code installed locally

Steps

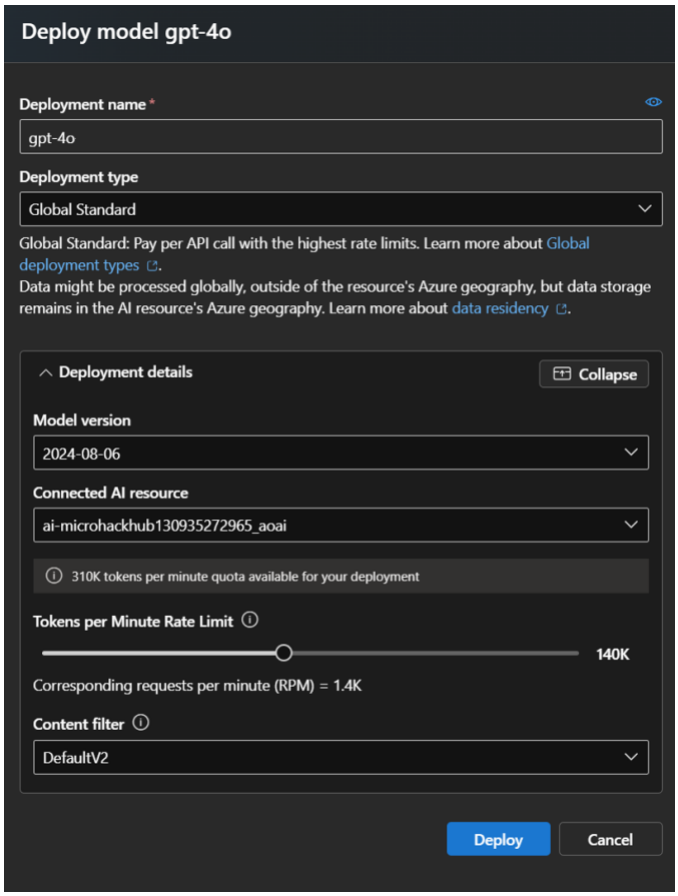
1. Review challenge Success Criteria before starting
2. Download the GitHub repository:
    - https://github.com/Boykai/octo-microhack-agentic-ai
3. Open the downloaded GitHub repository folder in VS Code

Option 1 (Portal - Recommended):

4. In a web browser, open Azure AI Foundry
5. Sign-in to Azure AI Foundry
6. Select **+Create Project** button
    - If you do not see **+Create Project**, select **Azure AI Foundry** button in the top left corner
7. Enter **microhack-agent-project** for Project Name
8. Select **Create new hub**
9. Enter **microhack-hub**
10. Select **Next** button
11. Select **Create** button
    - Recommended Regions: See gpt-4o, 2024-08-06 availability



12. Locate and copy **Project connection string**
13. Select **My Assets → Models + endpoints**
14. Select drop-down menu of **Deploy Model → Deploy base model**
15. Select **gpt-4o**
16. Select **Confirm** button
17. Select **Customize** button
18. Set Deployment name to **gpt-4o**
19. Select **Global Standard**
20. Select Model version **2024-08-06**
21. Set Tokens Per Minute Rate Limit to **140k**

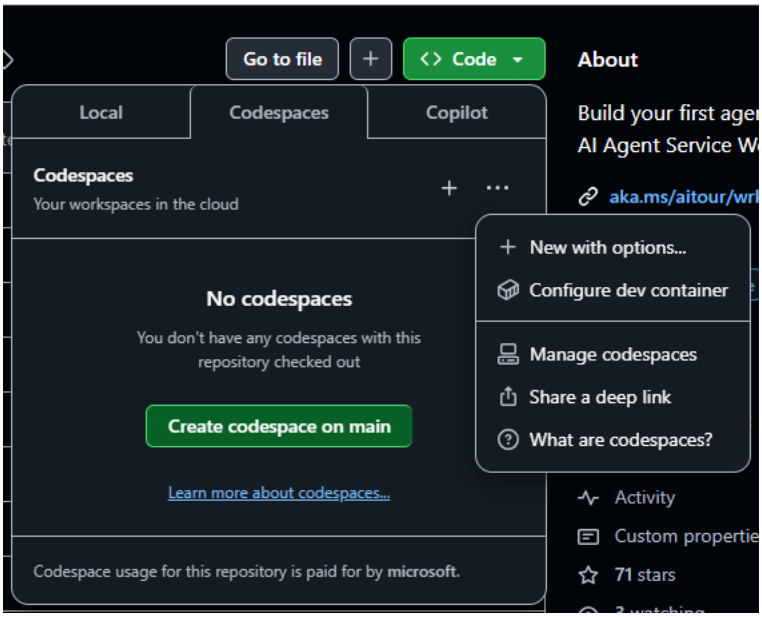


22. Select **Deploy** button
23. Skip to Step 46

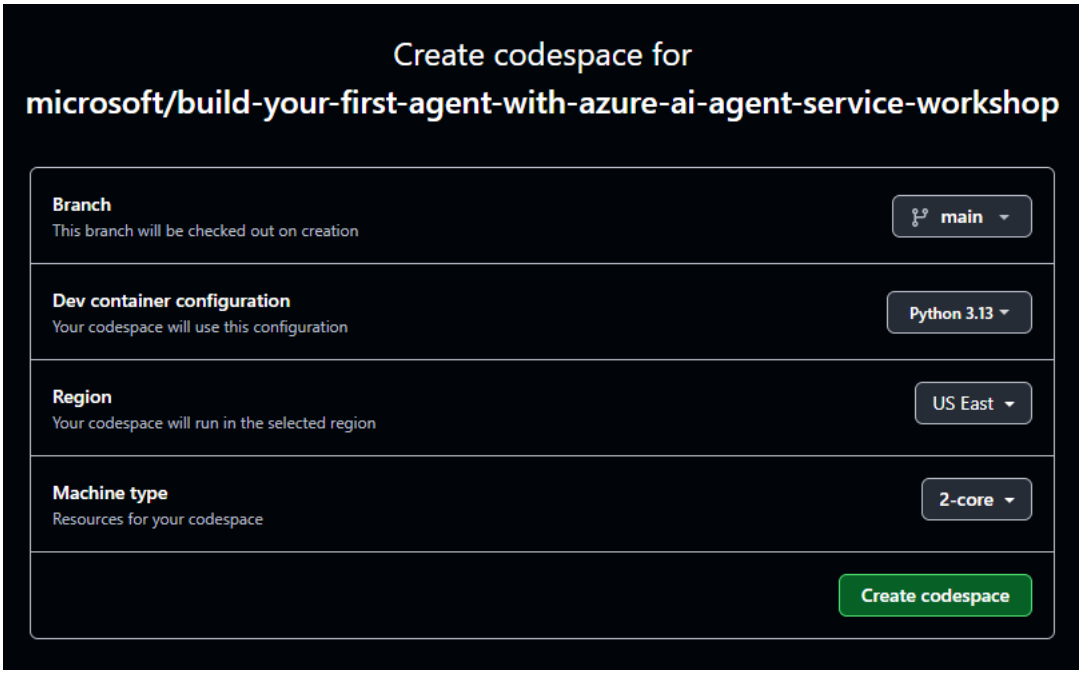
Option 2 (Codespaces):

24. In a browser navigate to the GitHub repository
    - https://github.com/Boykai/octo-microhack-agentic-ai

25. Select **Code** dropdown menu
26. Select **Codespaces** tab
27. Select **...** button
28. Select **+ New with options...**



29. Ensure the following options are selected:



30. Open terminal
31. Login to Azure CLI:

az login –use-device-code

- Possible Error: “Your sign-in was successful but your admin requires the device requesting access to be managed by your IT Admin to access this resource”.
  - Try Option #1 or #3
- Possible Error: Wrong tenant is selected by default, in which case try entering:
  - az login –user-device-code –tenant **your-tenant-id**

32. Select your Azure Subscription from the list
33. Upgrade Azure CLI Bicep package

az bicep upgrade

34. Deploy Azure resources

cd infra && ./deploy.sh

35. Skip to step 46

Option 3 (Local VS Code):

36. Install Azure CLI
37. Install Python 3.13
38. Install VS Code
39. Open VS Code
40. Open downloaded GitHub repository folder in VS Code
41. Open VS Terminal
42. Login to Azure CLI:

az login –use-device-code

- Possible Error: “Your sign-in was successful but your admin requires the device requesting access to be managed by your IT Admin to access this resource”.
  - Try Option #1

- Possible Error: Wrong tenant is selected by default, in which case try entering:
  - az login –user-device-code –tenant **your-tenant-id**

43. Select your Azure Subscription from the list
44. Upgrade Azure CLI Bicep package

az bicep upgrade

45. Deploy Azure resources

cd infra && ./deploy.sh

46. Validate resources were created in the Azure Portal

Create Grounding with Bing Search Service:

47. In a web browser, open Azure Portal
48. Select **hamburger** button in top left corner
49. Select **+ Create a resource** button
50. Select the **search box** and enter, **Grounding with Bing Search**
51. Select **Grounding with Bing Search** button
52. Select **Create** button
53. Select the following configuration parameters:

Subscription	**<your-azure-subscription-name>**
Resource group	**<your-resource-group>**
Name	**<your-resource-name>**
Region	Global
Pricing tier	Grounding with Bing Search ($35 per 1k transactions)
I confirm I have read and understood the notice above	Check

54. Select **Review + Create** button
55. Select **Create** button

Create Grounding with Bing Search Connection to Azure AI Foundry

56. In a web browser, open Azure AI Foundry
57. Ensure correct **Project** is selected
58. Select **Management Center** button, in bottom left corner
59. Select **New connection** button
60. Select **Grounding with Bing Search** button
61. Locate recently created **Grounding with Bing Search** service
62. Select **Add connection** button
63. Select **Close** button

Create Knowledge Base Vector Store

64. Select **Go to project** button
65. Select **Agents** button
66. Select **Select an Azure OpenAI Service resource** drop-down menu
67. Select the previously created Azure OpenAI Service from the drop-down menu
68. Select **Let's go** button
69. Select newly created agent **radio button**, left of agent name
70. Select **Try in playground** button
71. Try prompts
    - **Example prompts:**
        - What brands of tents do we sell?
        - What beginner tents do our competitors sell? Include prices.
        - What are the specifications of the latest model of AlpineGear tents?
        - Can you recommend a product for light weight camping?
        - How does Alpine Explorer compare to Skyview 2-Person tent?
        - I am going camping in the Moab area around Thanksgiving, can you recommend a tent for that weather?
        - Show the tents we sell by region that are a similar price to our competitors beginner tents.
        - Show the tents we sell by region that are a similar price to our competitors beginner tents?
        - Show as a bar chart?"
72. Update instructions with:
    - See **src/workshop/instructions/sales_agent_instructions.txt**
73. Try prompts from Step #71
74. Select **+ Add** button, next to Knowledge section
75. Select **Files** button

76. Select **Create vector store** drop-down menu
77. Select **Create a new vector store** option
78. Select **Select local files** button
79. Navigate to downloaded code repository
80. Select **resources/contoso-tents-datasheet.pdf**
81. Update **name** to **product-catalog**
82. Select **Upload and save** button
83. Try prompts from Step #71
84. Select **+ Add** button, next to Knowledge section
85. Select **Grounding with Bing Search** button
86. Select **radio button**, left of Grounding with Bing Search connection name
87. Select **Connect** button
88. Try prompts from Step #71
89. Done!

Success Criteria

☐ Successfully completed all Challenge 2 Steps without error

☐ Successfully deploy all Azure resources

☐ Successfully document Azure AI Foundry Project Connection String value

☐ Successfully document Grounding with Bing Search Service name

☐ Successfully document Azure OpenAI model name

☐ Successfully understand changes to agent responses to prompts

☐ Be aware of Azure AI Agent Service

☐ Be aware of Microsoft Semantic Kernel

☐ Be aware of Microsoft AutoGen

Useful Resources

- Azure Command-Line Interface (CLI) | Microsoft Learn
- Bicep | Microsoft Learn
- Azure AI Foundry | Microsoft Learn
- Azure AI Agent Service | Microsoft Learn
- Grounding with Bing Search | Microsoft Learn
- Azure OpenAI Service | Microsoft Learn
- Azure AI Agent Service | Microsoft Learn
- Semantic Kernel | Microsoft Learn
- AutoGen - Microsoft Research

Challenge 3: Azure AI Foundry (Code-First)

Overview

Next we will explore Actions (Function Calling) which enables Large Language Models (LLMs) to interact with external systems, execute tasks, and integrate with APIs. The LLM determines when to invoke a function based on user prompts and returns structured data for app use. Developers then implement the function logic within the app.

In this workshop, the function logic is used to execute the LLM dynamically generated SQL queries against the SQLite database.

If you're familiar with Azure OpenAI Function Calling, it requires defining a function schema for the LLM. Azure AI Agent Service supports this approach and also offers a more flexible option.

With the Azure AI Agent Service and its Python SDK, you can define the function schema directly within the Python function's docstring. This approach keeps the definition and implementation together, simplifying maintenance and enhancing readability.

For example, in the **src/workshop/sales_data.py** file,
the **async_fetch_sales_data_using_sqlite_query** function uses a docstring to specify its signature, inputs, and outputs. The SDK parses this docstring to generate the callable function for the LLM:

```
async def async_fetch_sales_data_using_sqlite_query(self: "SalesData", sqlite_query: str) -> str:
    """
    This function is used to answer user questions about Contoso sales data by executing SQLite queries against the database.

    :param sqlite_query: The input should be a well-formed SQLite query to extract information based on the user's question. The query result will be returned as a JSON object.
    :return: Return data in JSON serializable format.
    :rtype: str
    """
```

When the app starts, it incorporates the database schema and key data into the instructions for the Azure AI Agent Service. Using this input, the LLM generates SQLite-compatible SQL queries to respond to user requests expressed in natural language.

Related Technologies

Azure AI Foundry

- Azure AI Foundry provides a unified platform for enterprise AI operations, model builders, and application development. This foundation combines production-grade infrastructure with friendly interfaces, ensuring organizations can build and operate AI applications with confidence.
- Azure AI Foundry is designed for developers to:
    - Build generative AI applications on an enterprise-grade platform.
    - Explore, build, test, and deploy using cutting-edge AI tools and ML models, grounded in responsible AI practices.
    - Collaborate with a team for the full life-cycle of application development.
- With Azure AI Foundry, you can explore a wide variety of models, services and capabilities, and get to building AI applications that best serve your goals. Azure AI Foundry facilitates scalability for transforming proof of concepts into full-fledged production applications with ease. Continuous monitoring and refinement support long-term success.

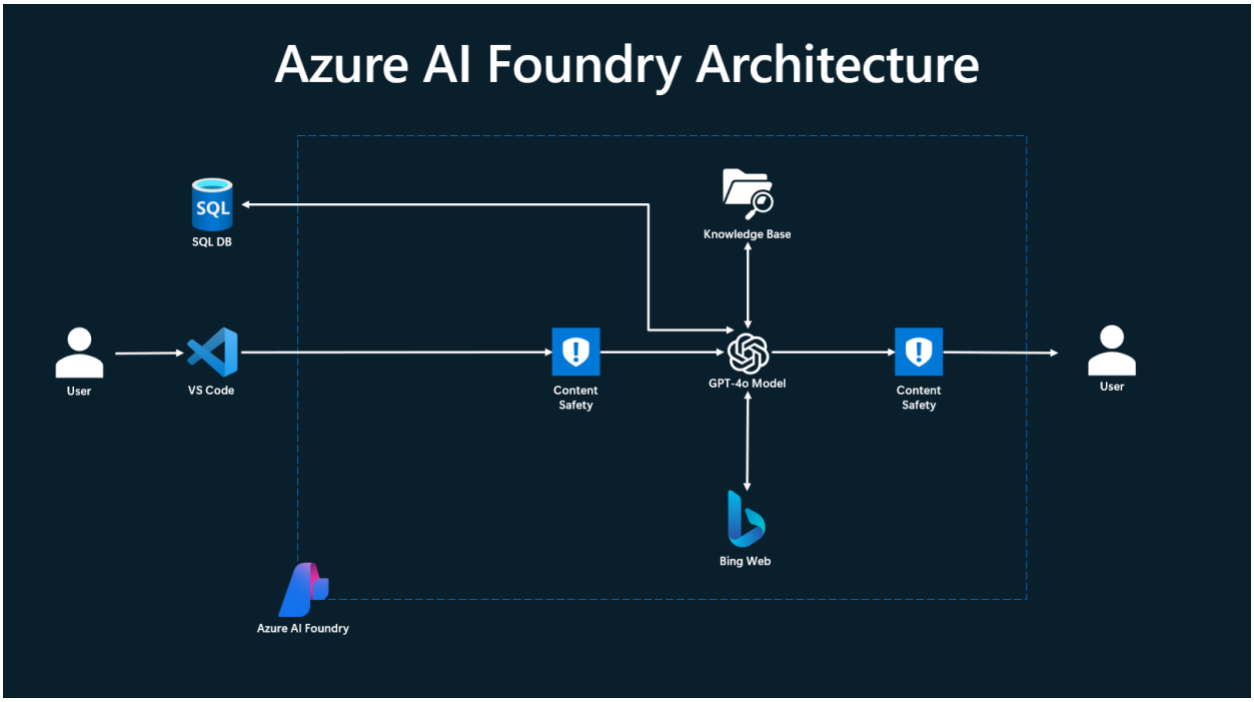
Azure OpenAI

- Azure OpenAI Service provides REST API access to OpenAI's powerful language models including o3-mini, o1, o1-mini, GPT-4o, GPT-4o mini, GPT-4 Turbo with Vision, GPT-4, GPT-3.5-Turbo, and Embeddings model series. These models can be easily adapted to your specific task including but not limited to content generation, summarization, image understanding, semantic search, and natural language to code translation. Users can access the service through REST APIs, Python/C#/JS/Java/Go SDKs.
- At Microsoft, we're committed to the advancement of AI driven by principles that put people first. Generative models such as the ones available in Azure OpenAI have significant potential benefits, but without careful design and thoughtful mitigations, such models have the potential to generate incorrect or even harmful content. Microsoft has made significant investments to help guard against abuse and unintended harm, which includes incorporating Microsoft's principles for responsible AI use, adopting a Code of Conduct for use of the service, building content filters to support customers, and providing responsible AI information and guidance that customers should consider when using Azure OpenAI.

Architecture

Basic Architecture

Azure AI Foundry – Agent Architecture



Prerequisites

- To have a subscription in Azure
- Initiate an Azure AI services creation and agree to the Responsible AI terms **
    - ** If you have not created an Azure AI service resource in the subscription before
- To have an account in GitHub
- To have VS Code installed locally
- You can reuse AI Hub and project for Challenge 3, created in Challenge 2.

Steps:

1. Open the downloaded GitHub repository in VS Code
2. Validate the file **src/workshop/.env** is created
    - If not created, copy **src/workshop/.env.sample** and rename the copy to **.env**
3. Update and save the **.env** file with the **Project Connection String** value from the AI Foundry resource project created
    - You can find this string in the AI Foundry portal in the Overview page for your Project **microhack-project** (look in the Project details section)
4. Save the changes

Actions (Function Calling):

5. Open **src/workshop/main.py**
6. Uncomment the following lines by removing the # characters

 # INSTRUCTIONS_FILE = "instructions/instructions_function_calling.txt"

 # toolset.add(functions)

7. The code should now appear as:

```
INSTRUCTIONS_FILE = "instructions/instructions_function_calling.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_code_interpreter.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_file_search.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_bing_grounding.txt"
async def add_agent_tools():
"""Add tools for the agent."""
```

```
    # Add the functions tool
    toolset.add(functions)

    # Add the code interpreter tool
    # code_interpreter = CodeInterpreterTool()
    # toolset.add(code_interpreter)

    # Add the tents data sheet to a new vector data store
    # vector_store = await utilities.create_vector_store(
    #     project_client,
    #     files=[TENTS_DATA_SHEET_FILE],
    #     vector_name_name="Contoso Product Information Vector Store",
    # )
    # file_search_tool = FileSearchTool(vector_store_ids=[vector_store.id])
    # toolset.add(file_search_tool)

    # Add the Bing grounding tool
    # bing_connection = await
project_client.connections.get(connection_name=BING_CONNECTION_NAME)
    # bing_grounding = BingGroundingTool(connection_id=bing_connection.id)
    # toolset.add(bing_grounding)
```

8. **Save** the changes
9. Open and review **src/workshop/instructions/instructions_function_calling.txt**
10. Enable Codespaces in VS Code
11. Press **F5** to run the app

12. In the terminal, you'll see the app start, and the agent app will prompt you to enter your query.
13. Enter the following prompts in the terminal:
    - **Example prompts:**
        - Help
        - What were the sales by region?
        - What was last quarter's revenue?
        - Which products sell best in Europe?
        - Total shipping costs by region?
        - What regions have the highest sales?
        - What were the sales of tents in the United States in April 2022?
    - **Explanation:**
        - The LLM generates an SQL query to answer the user's question. For the question **"What are the sales by region?"**, the following SQL query is generated:
            - SELECT region, SUM(revenue) AS total_revenue FROM sales_data GROUP BY region;
14. To stop the agent – press **Shift + F5** or type **exit**

Code Interpreter:

15. Open **src/workshop/main.py**
16. Uncomment the following lines by removing the # characters

```
# INSTRUCTIONS_FILE = "instructions/instructions_code_interpreter.txt"

# code_interpreter = CodeInterpreterTool()
# toolset.add(code_interpreter)
```

17. The code should now appear as:

```
INSTRUCTIONS_FILE = "instructions/instructions_function_calling.txt"
INSTRUCTIONS_FILE = "instructions/instructions_code_interpreter.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_file_search.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_bing_grounding.txt"
async def add_agent_tools():
"""Add tools for the agent."""
```

```
    # Add the functions tool
    toolset.add(functions)

    # Add the code interpreter tool
    code_interpreter = CodeInterpreterTool()
    toolset.add(code_interpreter)

    # Add the tents data sheet to a new vector data store
    # vector_store = await utilities.create_vector_store(
    #     project_client,
    #     files=[TENTS_DATA_SHEET_FILE],
    #     vector_name_name="Contoso Product Information Vector Store",
    # )
    # file_search_tool = FileSearchTool(vector_store_ids=[vector_store.id])
    # toolset.add(file_search_tool)

    # Add the Bing grounding tool
    # bing_connection = await
project_client.connections.get(connection_name=BING_CONNECTION_NAME)
    # bing_grounding = BingGroundingTool(connection_id=bing_connection.id)
    # toolset.add(bing_grounding)
```

18. **Save** the changes
19. Open and review **src/workshop/instructions/instructions_code_interpreter.txt**
20. Press **F5** to run the app
21. In the terminal, you'll see the app start, and the agent app will prompt you to enter your query.
22. Try prompts from Step #13
23. To stop the agent – press **Shift + F5** or type **exit**

Knowledge Base:

24. Open **src/workshop/main.py**
25. Uncomment the following lines by removing the # characters

```
# INSTRUCTIONS_FILE = "instructions/instructions_file_search.txt"

# vector_store = await utilities.create_vector_store(
#    project_client,
#    files=[TENTS_DATA_SHEET_FILE],
#    vector_name_name="Contoso Product Information Vector Store",
# )
# file_search_tool = FileSearchTool(vector_store_ids=[vector_store.id])
# toolset.add(file_search_tool)
```

26. The code should now appear as:

```
INSTRUCTIONS_FILE = "instructions/instructions_function_calling.txt"
INSTRUCTIONS_FILE = "instructions/instructions_code_interpreter.txt"
INSTRUCTIONS_FILE = "instructions/instructions_file_search.txt"
# INSTRUCTIONS_FILE = "instructions/instructions_bing_grounding.txt"
async def add_agent_tools():
    """Add tools for the agent."""


    # Add the functions tool

    toolset.add(functions)


    # Add the code interpreter tool

    code_interpreter = CodeInterpreterTool()

    toolset.add(code_interpreter)


    # Add the tents data sheet to a new vector data store

    vector_store = await utilities.create_vector_store(

        project_client,
```

```
        files=[TENTS_DATA_SHEET_FILE],
        vector_name_name="Contoso Product Information Vector Store",
    )
    file_search_tool = FileSearchTool(vector_store_ids=[vector_store.id])
    toolset.add(file_search_tool)


    # Add the Bing grounding tool
    # bing_connection = await
project_client.connections.get(connection_name=BING_CONNECTION_NAME)
    # bing_grounding = BingGroundingTool(connection_id=bing_connection.id)
    # toolset.add(bing_grounding)
```

27. **Save** the changes
28. Open and review **src/workshop/instructions/instructions_code_file_search.txt**
29. Press **F5** to run the app
30. In the terminal, you'll see the app start, and the agent app will prompt you to enter your query
31. Try prompts from Step #13
32. To stop the agent – press **Shift + F5** or type **exit**

Grounding with Bing Search

33. Open **src/workshop/main.py**
34. Uncomment the remain lines of code by removing the # characters
35. **Save** the changes
36. Open and review **src/workshop/instructions/instructions_bing_grounding.txt**
37. In the terminal, you'll see the app start, and the agent app will prompt you to enter your query
38. Try prompts from Step #13
39. To stop the agent – press **Shift + F5** or type **exit**
40. Done!

Success Criteria

☐ Successfully completed all Challenge 3 Steps without error

☐ Successfully deployed all resources in architecture

☐ Understanding of architecture and its components

☐ Understanding GitHub code repository and its components

☐ Be aware of Microsoft Semantic Kernel

☐ Be aware of Microsoft AutoGen

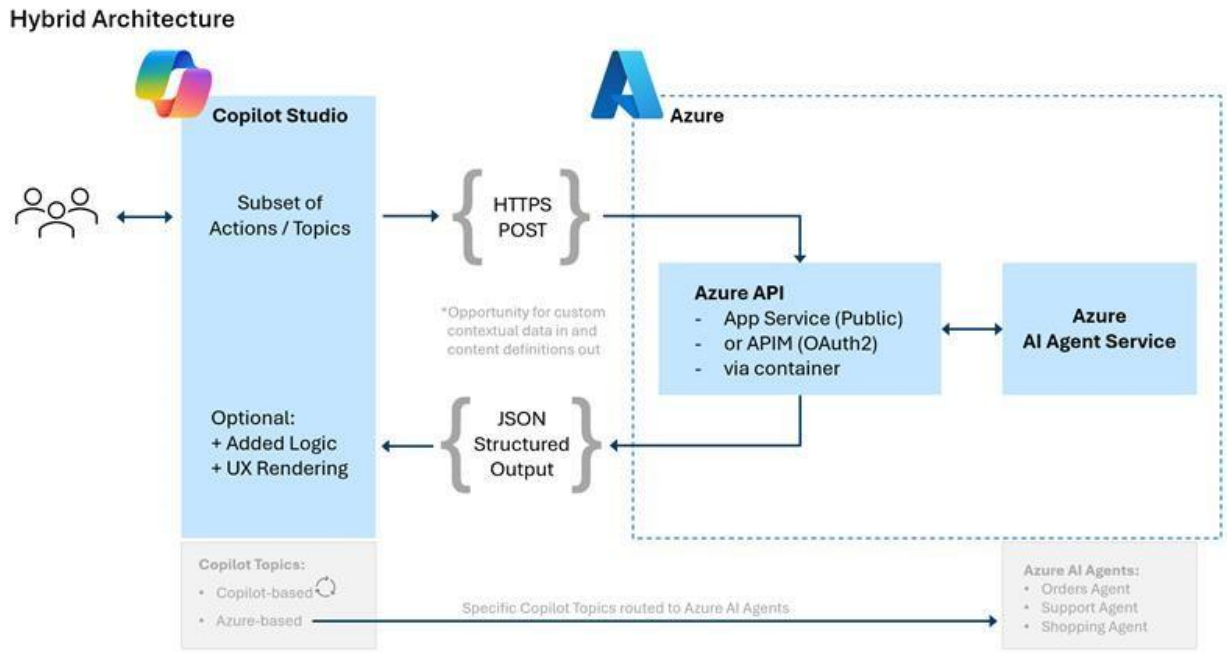
Useful Resources

- Azure Command-Line Interface (CLI) | Microsoft Learn
- Bicep | Microsoft Learn
- Azure AI Foundry | Microsoft Learn
- Azure AI Agent Service | Microsoft Learn
- Grounding with Bing Search | Microsoft Learn
- Azure OpenAI Service | Microsoft Learn
- Semantic Kernel | Microsoft Learn
- AutoGen - Microsoft Research

Challenge 4: Azure AI Foundry and CoPilot Studio – Better Together

Overview

Stand up API wrapping Azure AI Agent Service functionality which can be used as a Skill from within Co-pilot Studio.



Steps

API installation from https://github.com/Boykai/octo-microhack-agentic-ai/tree/main/src/workshop:

Initial Deploy to Azure

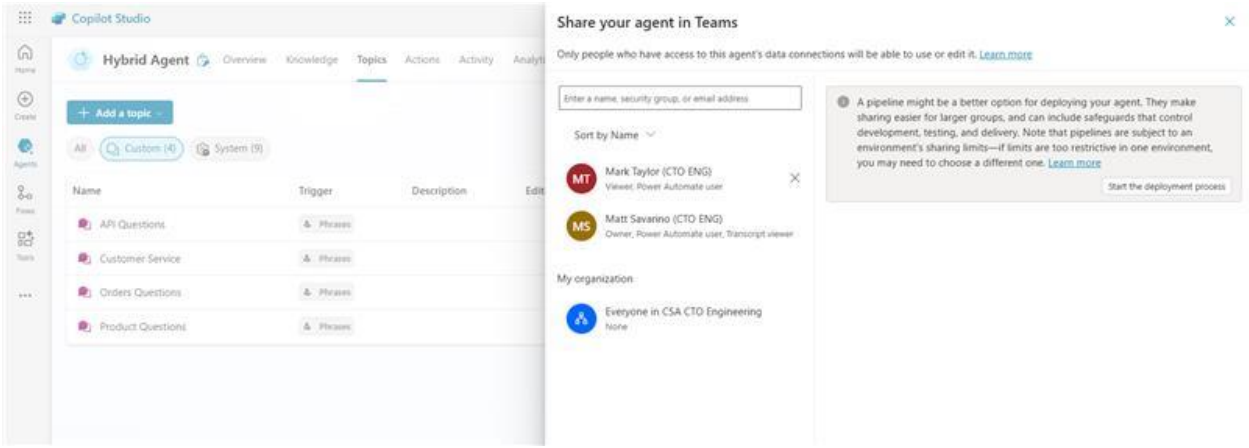
1. Run the deploy script:

   ./deploy.ps1

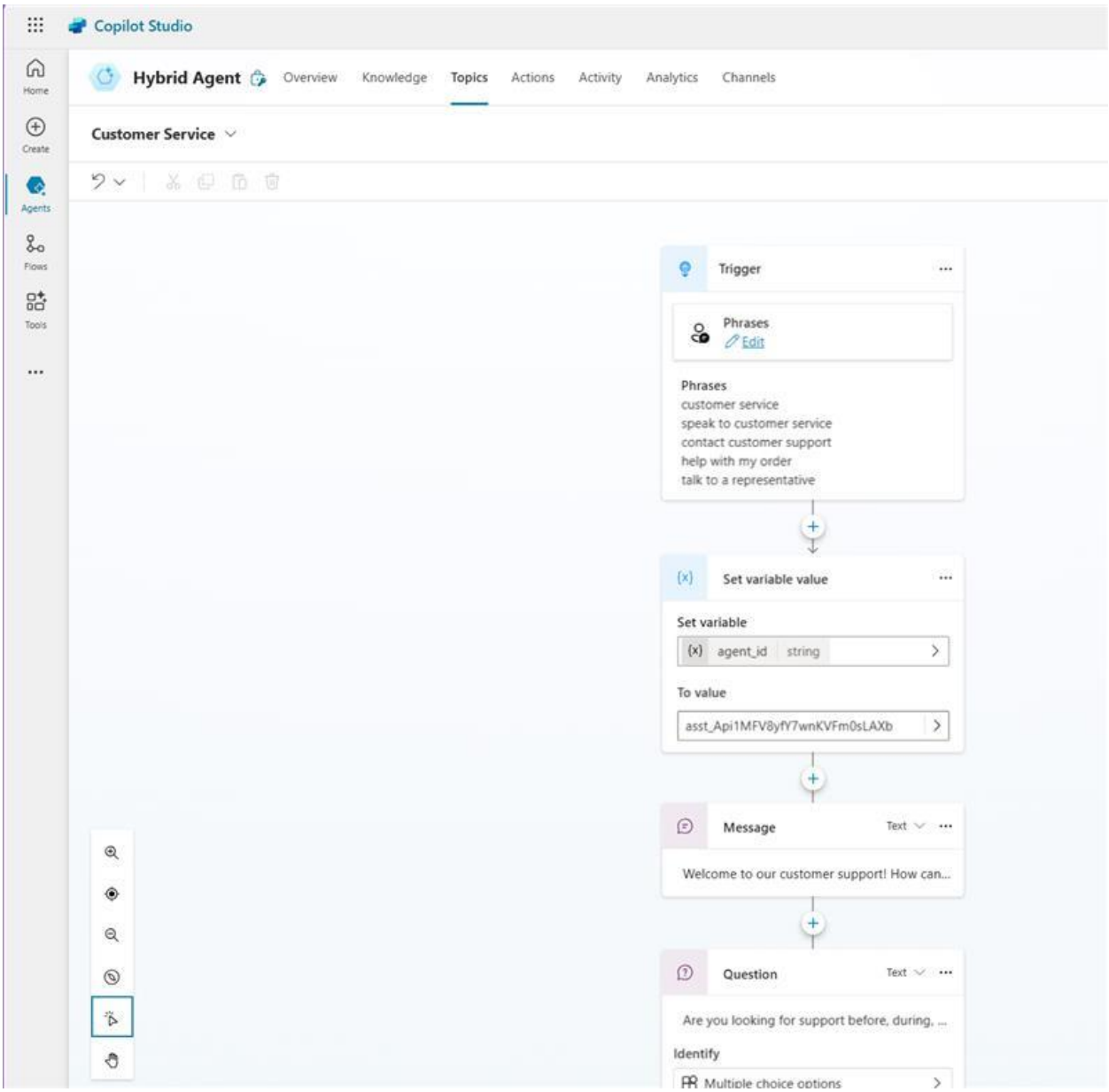
2. In Azure Portal, navigate to App > Settings > Identity
3. Enable system assigned identity
4. Add Azure AI Developer role assignments

CoPilot Studio:

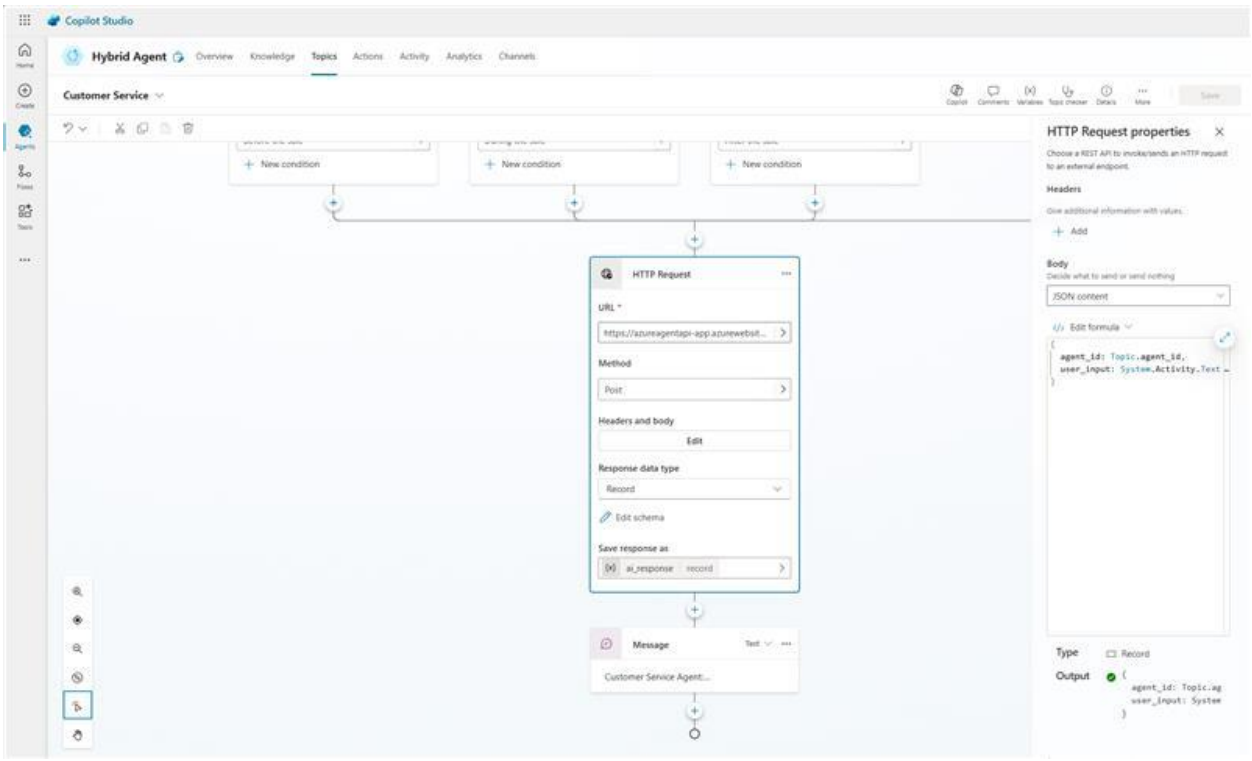
1.



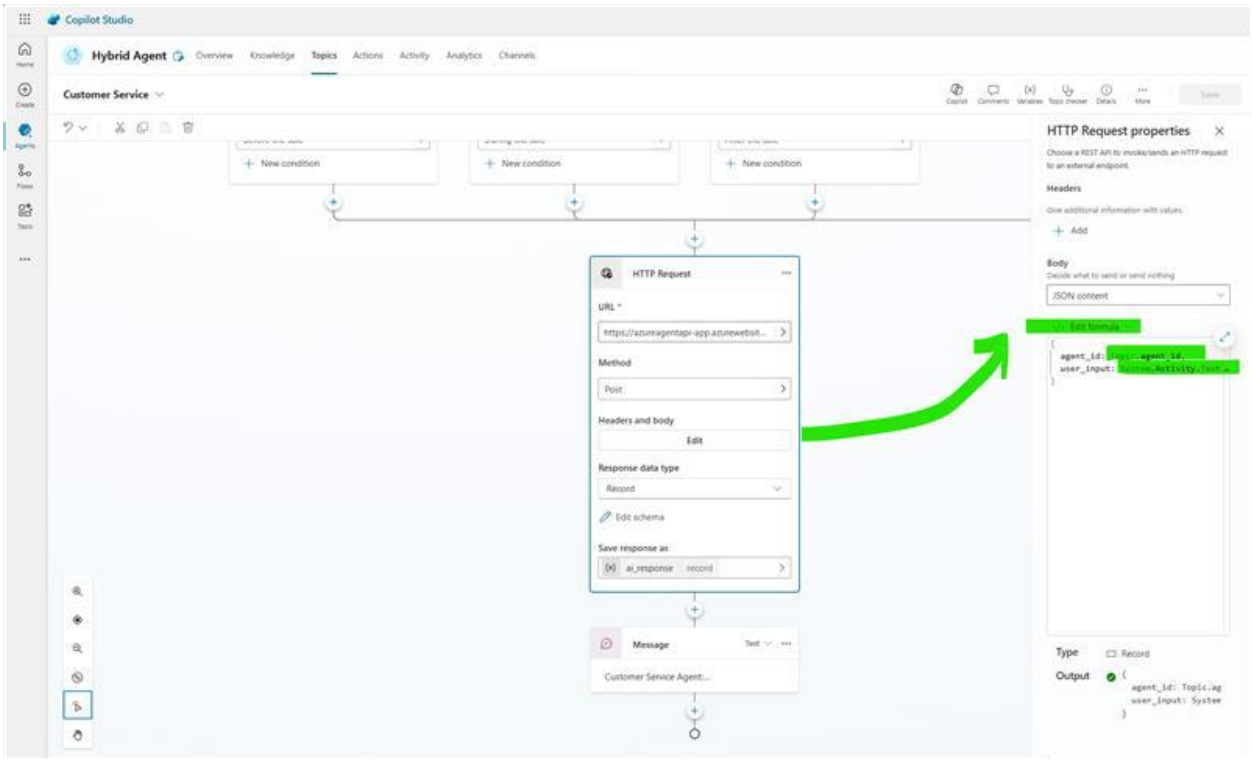
2.


Copilot Studio
Hybrid Agent
Overview
Knowledge
Topics
Actions
Activity
Analytics
Channels
Home
Create
Agents
Flows
Tools
Customer Service
Trigger
Phrases
Edit
Phrases
customer service
speak to customer service
contact customer support
help with my order
talk to a representative
Set variable value
Set variable
agent_id
string
To value
asst_Api1MFV8yfY7wnKVFm0sLAXb
Message
Text
Welcome to our customer support! How can...
Question
Text
Are you looking for support before, during, ...
Identify
Multiple choice options

3.


Copilot Studio
Hybrid Agent
Overview
Knowledge
Topics
Actions
Activity
Analytics
Channels
Customer Service
New condition
HTTP Request
URL *
https://azureagentapi-app.azurewebsit...
Method
Post
Headers and body
Edit
Response data type
Record
Edit schema
Save response as
ai_response
record
Message
Text
Customer Service Agent:...
HTTP Request properties
Choose a REST API to invoke/sends an HTTP request to an external endpoint.
Headers
Give additional information with values.
Add
Body
Decide what to send or send nothing
JSON content
Edit formula
{
agent_id: Topic.agent_id,
user_input: System.Activity.Text
}
Type
Record
Output
{
agent_id: Topic.ag
user_input: Syste
}

4.


Copilot Studio
Hybrid Agent
Overview
Knowledge
Topics
Actions
Activity
Analytics
Channels
Customer Service
New condition
HTTP Request
URL *
https://azureagentapi-app.azurewebsit...
Method
Post
Headers and body
Edit
Response data type
Record
Edit schema
Save response as
ai_response
record
Message
Text
Customer Service Agent:...
HTTP Request properties
Choose a REST API to invoke/sends an HTTP request to an external endpoint.
Headers
Add
Body
Decide what to send or send nothing
JSON content
Edit formula
Type
Record
Output
{
agent_id: Topic.ag
user_input: Syste
}

Congratulations!

Cleaning Up Environment

1. Navigate to Copilot Studio – delete agents created
2. Navigate to Azure Portal – delete Azure resource group created

Troubleshooting

Quota Availability

If you have issues with Azure OpenAI quota, you can create a support ticket for requesting more quota for a given region. Please see Azure OpenAI quota documentation for more details.

Codespaces Authentication

If you receive the following error while utilizing Codespaces, this could be due to Codespaces not being a managed device by your Azure environment's IT organization. In this case, please utilize a local environment on your device, which is managed by your IT organization.