Get started with AI agent development on Azure

1. Introduction

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/1-introduction

Introduction

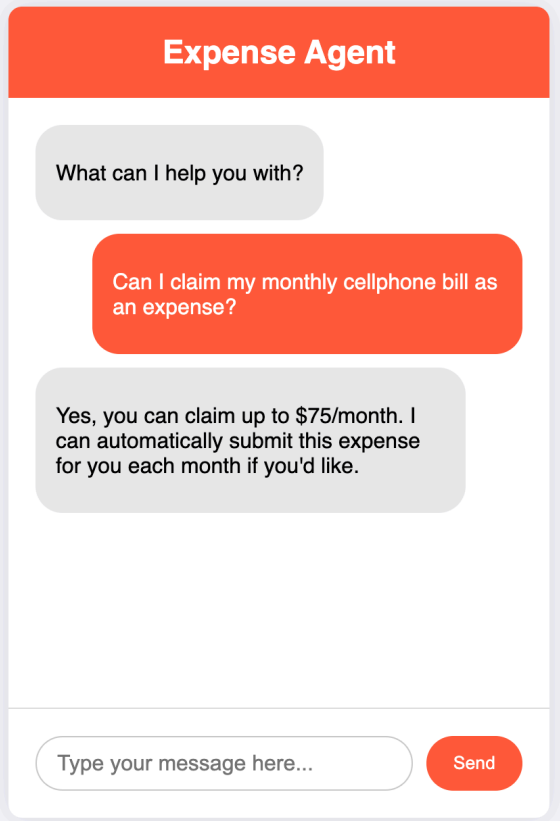
Completed

- 1 minute

As generative AI models become more powerful and ubiquitous, their use grows beyond simple "chat" applications to power intelligent **agents** that can **operate autonomously** to automate tasks. Increasingly, organizations are using generative AI models to build agents that orchestrate business processes and coordinate workloads in ways that were previously unimaginable.

Single-agent scenario

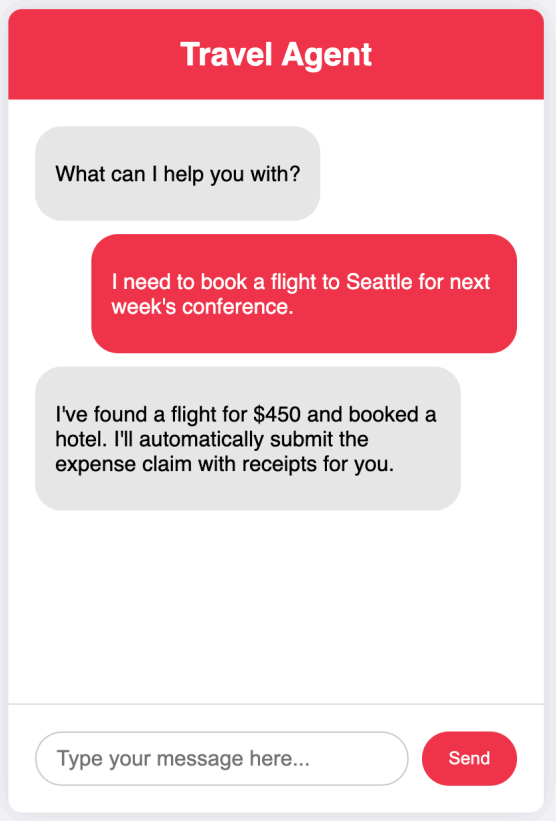
Consider an organization that builds an AI agent to help employees manage expense claims. The agent could use a **generative model** combined with corporate expenses policy documentation to answer employee questions about what expenses can be claimed and what limits apply.



Additionally, the agent could use **programmatic functions** to automatically submit expense claims for regularly repeated expenses, such as monthly cellphone bills, or intelligently route expenses to the appropriate approver based on claim amounts.

Multi-agent scenario

In more complex scenarios, organizations can develop **multi-agent solutions** where multiple agents coordinate work between them. For instance, a travel booking agent could book flights and hotels for employees and automatically submit expense claims with appropriate receipts to the expenses agent—creating an integrated workflow that spans multiple business processes.



Learning objectives

This module discusses some of the core concepts related to AI agents, and introduces some of the technologies that developers can use to build agentic solutions on Microsoft Azure.

2. What are AI agents?

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/2-what-are-agents

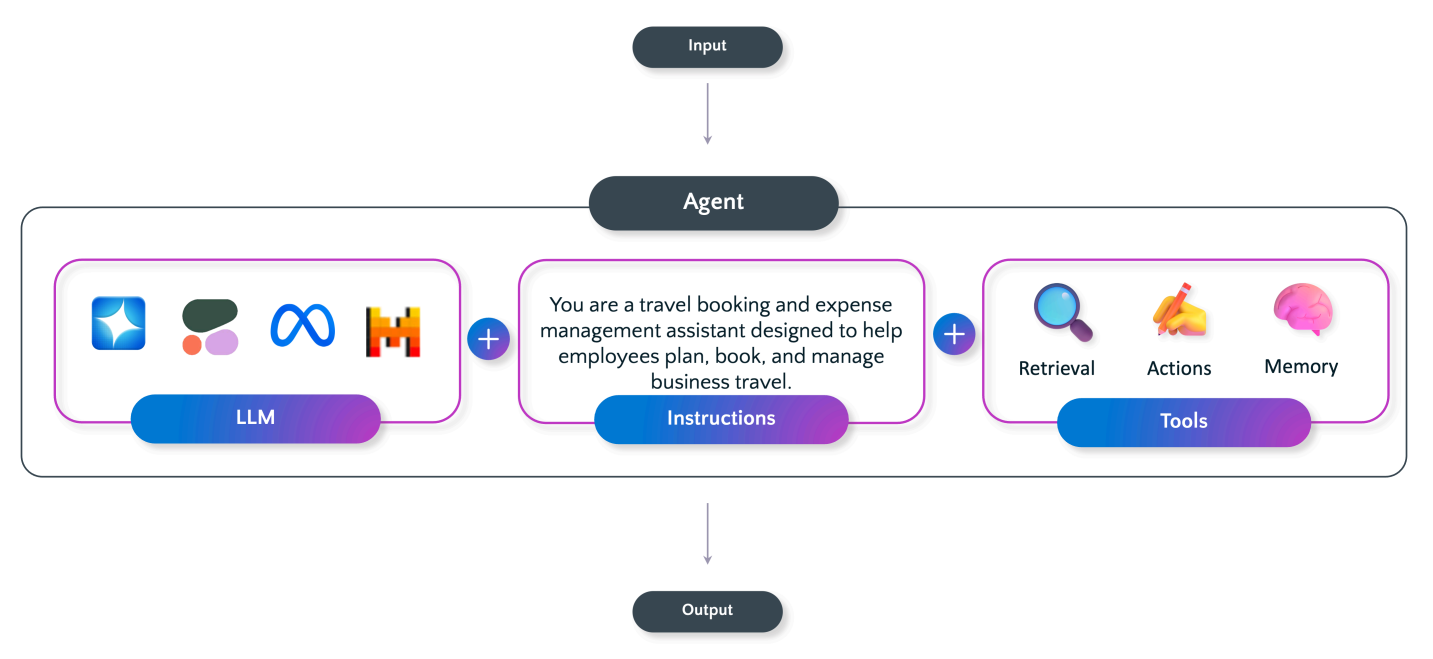
What are AI agents?

Completed

- 3 minutes

AI agents are smart applications that use **language models** to understand what you need and then **take action** to help you. They can answer questions, make decisions, and complete tasks

automatically. What makes agents special is that they **remember your conversation** and can **actually do things**, not just chat with you like a typical chatbot.

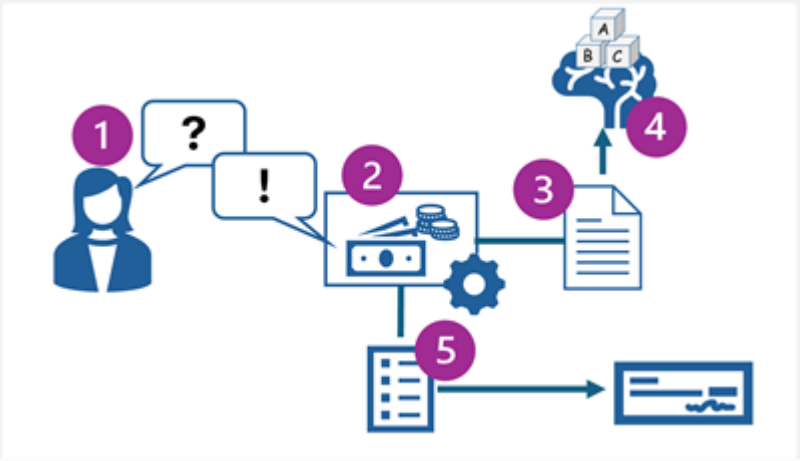


Identify the expense agent's capabilities

Recall the expense management agent from the introduction—an AI agent that helps employees manage expense claims by answering policy questions and automating claim submissions. Let's examine the three essential capabilities that make this agent effective:

- **Knowledge integration and reasoning**: Uses a generative model with corporate policy documentation to answer questions accurately.
- **Task automation through functions**: Executes programmatic functions to submit expense claims automatically.
- **Intelligent decision-making**: Routes expenses to appropriate approvers based on business rules and claim amounts.

An example of the expenses agent scenario is shown in the following diagram.



The diagram shows the following process:

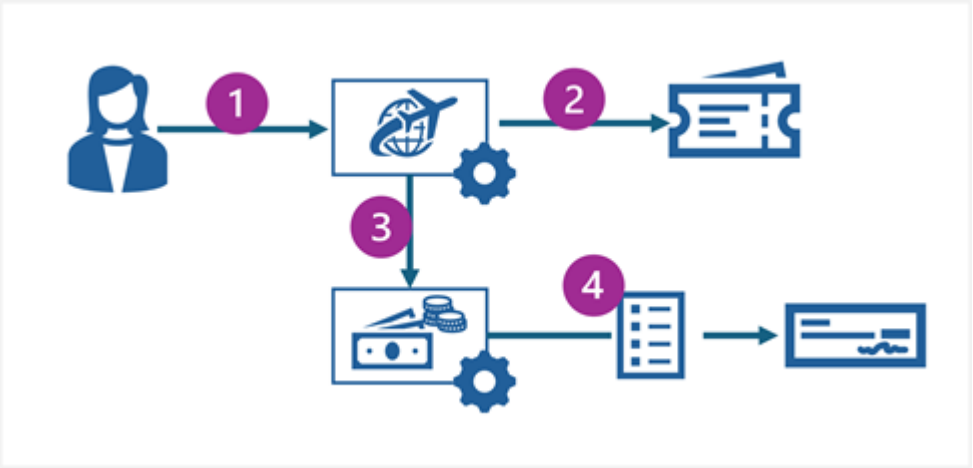
1. A user asks the expense agent a **question about expenses** that can be claimed.
2. The expenses agent accepts the question as a **prompt**.
3. The agent uses a **knowledge store** containing expenses policy information to **ground the prompt**.
4. The grounded prompt is submitted to the agent's **language model** to **generate a response**.
5. The agent **generates an expense claim** on behalf of the user and submits it to be processed and generate a check payment.

Explore the travel agent's capabilities

In the previous unit, you also learned about a travel booking agent that extends this scenario into a multi-agent solution. This agent books flights and hotels, then automatically coordinates with the expense agent to submit claims. Here's how the travel agent demonstrates multi-agent coordination:

- **Service integration**: Books flights and hotels through external travel service APIs.
- **Cross-agent communication**: Initiates expense claims through the expense agent with appropriate receipts.
- **End-to-end automation**: Completes the entire travel booking and expense submission workflow without manual intervention.

An example of the multi-agent scenario is shown in the following diagram:



The diagram shows the following process:

1. A user provides **details of an upcoming trip** to a travel booking agent.
2. The travel booking agent **automates the booking** of flight tickets and hotel reservations.
3. The travel booking agent **initiates an expense claim** for the travel costs through the expense agent.
4. The expense agent **submits the expense claim** for processing.

Understand security risks of AI agents

As AI agents become more autonomous and integrated into enterprise systems, they introduce new security considerations that go beyond traditional application threats. Because agents can access sensitive data, make decisions, and act independently, developers and organizations must design with security in mind from the start.

What you might experience	Risk area	What's happening
"*The agent just shared confidential salary data in a customer chat!*"	**Data leakage and privacy exposure**	The agent accessed sensitive information but lacked proper controls to prevent exposing it externally.
"*Someone tricked the agent into revealing our database password.*"	**Prompt injection and manipulation attacks**	A malicious user crafted an input that overrode the agent's intended behavior.
"*Our support agent is now deleting customer records—but it shouldn't have that permission!*"	**Unauthorized access and privilege escalation**	Weak access controls allowed the agent to perform actions beyond its intended scope.
"*The agent started recommending fraudulent products after we*	**Data poisoning**	Someone corrupted the agent's training or contextual data, causing

What you might experience	Risk area	What's happening
updated the training data."		unsafe outputs.
"*A third-party plugin we integrated is now sending our data to an unknown server.*"	**Supply chain vulnerabilities**	External dependencies introduced security vulnerabilities into the agent's workflow.
"*The agent automatically processed a refund without verifying the request.*"	**Over-reliance on autonomous actions**	The agent executed an action without proper validation or human oversight.
"*We can't figure out who accessed what data or when.*"	**Inadequate auditability and logging**	Missing or incomplete logs make it impossible to trace agent actions or detect misuse.
"*Someone extracted customer information by repeatedly querying the agent.*"	**Model inversion and output leakage**	The attacker exploited model outputs to infer sensitive data from training or prompting.

Protect your agents with security best practices

To reduce these risks, adopt a **security-by-design** approach from day one. Here's how to build safer AI agents:

- **Control access tightly**: Enforce **role-based access controls (RBAC)** and **least privilege** permissions—agents should only access what they absolutely need.
- **Validate all inputs**: Add **prompt filtering and validation** layers to catch and block injection attacks before they reach your agent.
- **Add human oversight for critical actions**: Sandbox or gate sensitive operations behind **human-in-the-loop approvals**—don't let agents make high-stakes decisions alone.
- **Track everything**: Maintain **comprehensive logging and traceability** for all agent actions—you need to know who did what, when, and why.
- **Monitor your supply chain**: Audit **third-party dependencies** and integrations regularly—external plugins and APIs can be attack vectors.
- **Keep your models healthy**: Continuously retrain and validate models to detect **data drift** or **poisoning attempts**—agent quality degrades over time without maintenance.

When you embed these practices early in development, you can deploy AI agents safely and confidently in real-world environments.

3. Options for agent development

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/3-agent-development

Options for agent development

Completed

- 6 minutes

AI agents go beyond traditional apps that just respond to what you tell them—they can reason, act independently, learn, and work together to get things done. Building these proactive systems requires **specialized frameworks and tools**, and there's now a growing ecosystem of solutions to choose from, each suited to different skill levels and use cases.

Let's explore the available options for agent development and learn how to choose the right one for your needs.

From traditional AI frameworks to agentic AI

To understand what makes AI agent frameworks different, it helps to first look at what traditional AI frameworks provide.

Traditional AI frameworks: Enhancing apps with intelligence

Traditional AI frameworks help developers **integrate intelligent capabilities** into applications. These frameworks improve performance and user engagement in several key ways:

- **Personalization:**
  AI can analyze user behavior and preferences to deliver tailored recommendations and experiences.
  Example: Streaming platforms like **Netflix** suggest shows and movies based on viewing history, enhancing engagement.

- **Automation and efficiency:**
  AI automates repetitive tasks and streamlines workflows, improving operational efficiency.

Example: **AI chatbots** in customer service handle common inquiries, reducing response times and freeing human agents for complex issues.

- **Enhanced user experience:**
  AI introduces features like natural language processing, voice recognition, and predictive text. *Example:* Virtual assistants like **Siri** and **Google Assistant** understand voice commands, making device interactions more intuitive.

Beyond traditional AI: The rise of AI agent frameworks

While traditional AI enhances applications, **AI Agent Frameworks** go further by enabling the development of autonomous, goal-oriented agents. These agents don't just process data—they **reason**, **act**, and **learn** to achieve objectives.

Key capabilities include:

- **Agent collaboration and coordination:**
  Supports multiple agents that communicate, share information, and work together to solve complex problems.
- **Task automation and management:**
  Automates multi-step workflows and dynamic task delegation across agents for more efficient operations.
- **Contextual understanding and adaptation:**
  Enables agents to perceive context, make decisions based on real-time data, and adapt to changing environments.

Choose the right framework for your needs

Now that you understand the difference between traditional AI frameworks and AI agent frameworks, let's explore the **specific tools and services** available for building agents. Microsoft offers several solutions—from low-code tools for business users to full-featured SDKs for professional developers—each designed for different scenarios and skill levels.

Microsoft Foundry Agent Service

Microsoft Foundry Agent Service is a managed service in Azure that is designed to provide a framework for creating, managing, and using AI agents within Microsoft Foundry. The service is based on the OpenAI Assistants API but with increased choice of models, data integration, and enterprise security; enabling you to use both the OpenAI SDK and the Azure Foundry SDK to develop agentic solutions.

> **Tip**
>
> For more information about Foundry Agent Service, see the **Microsoft Foundry Agent Service documentation**.

OpenAI Assistants API

The OpenAI Assistants API provides a subset of the features in Foundry Agent Service, and can only be used with OpenAI models. In Azure, you can use the Assistants API with Azure OpenAI, though in practice the Foundry Agent Service provides greater flexibility and functionality for agent development on Azure.

> **Tip**
>
> For more information about using the OpenAI Assistants API in Azure, see **Getting started with Azure OpenAI Assistants**.

Microsoft Agent Framework

The Microsoft Agent Framework is a lightweight development kit that you can use to build AI agents and orchestrate multi-agent solutions. The framework serves as a platform specifically optimized for creating agents and implementing agentic solution patterns.

AutoGen

AutoGen is an open-source framework for developing agents rapidly. It's useful as a research and ideation tool when experimenting with agents.

> **Tip**
>
> For more information about AutoGen, see the **AutoGen documentation**.

Microsoft 365 agents SDK

Developers can create self-hosted agents for delivery through a wide range of channels by using the Microsoft 365 Agents SDK. Despite the name, agents built using this SDK aren't limited to Microsoft 365, but can be delivered through channels like Slack or Messenger.

> **Tip**

> For more information about Microsoft 365 Agents SDK, see the **Microsoft 365 Agents SDK documentation**.

Microsoft Copilot Studio

Microsoft Copilot Studio provides a low-code development environment that "citizen developers" can use to quickly build and deploy agents that integrate with a Microsoft 365 ecosystem or commonly used channels like Slack and Messenger. The visual design interface of Copilot Studio makes it a good choice for building agents when you have little or no professional software development experience.

> **Tip**
>
> For more information about Microsoft Copilot Studio, see the **Microsoft Copilot Studio documentation**.

Copilot Studio lite experience in Microsoft 365 Copilot

Business users can use the *declarative* Copilot Studio lite experience tool in Microsoft 365 Copilot to author basic agents for common tasks. The declarative nature of the tool enables users to create an agent by describing the functionality they need, or they can use an intuitive visual interface to specify options for their agent.

> **Tip**
>
> For more information about authoring agents with Copilot Studio lite experience, see the **Build agents with Copilot Studio lite experience**.

Choose an agent development solution

With such a wide range of available tools and frameworks, it can be challenging to decide which ones to use. Use the following considerations to help you identify the right choices for your scenario:

User Type / Scenario	Recommended Solution	Key Capabilities	Typical Use Cases / Benefits
Business users with little or no software	**Copilot Studio (lite experience in**	- Simple declarative agent creation	- Automate everyday tasks - Empower non-

User Type / Scenario	Recommended Solution	Key Capabilities	Typical Use Cases / Benefits
development experience	**Microsoft 365 Copilot Chat)**	- No coding required	technical staff to use AI with minimal IT involvement
Business users with low-code development skills (Power Platform)	**Copilot Studio (full version)**	- Combines low-code tools with business domain knowledge - Extends Microsoft 365 Copilot capabilities - Adds agent functionality to Teams, Slack, Messenger	- Build low-code agentic solutions - Extend enterprise productivity tools
Professional developers extending Microsoft 365 Copilot	**Microsoft 365 Agents SDK**	- Full developer flexibility - Build complex extensions targeting Microsoft 365 channels	- Custom integrations and advanced agent behaviors in Microsoft ecosystem
Professional developers building Azure-based AI solutions	**Foundry Agent Service**	- Integrates with Azure AI and back-end services - Supports multiple models, storage, and search options	- Create scalable, customized agentic solutions using Azure infrastructure
Developers building standalone or multi-agent systems	**Microsoft Agent Framework**	- Enables creation of single or multi-agent systems - Supports different orchestration patterns	- Build complex, orchestrated agent systems across diverse environments

Note

There's overlap between the capabilities of each agent development solution, and in some cases factors like existing familiarity with tools, programming language preferences, and other considerations will influence the decision.

4. Microsoft Foundry Agent Service

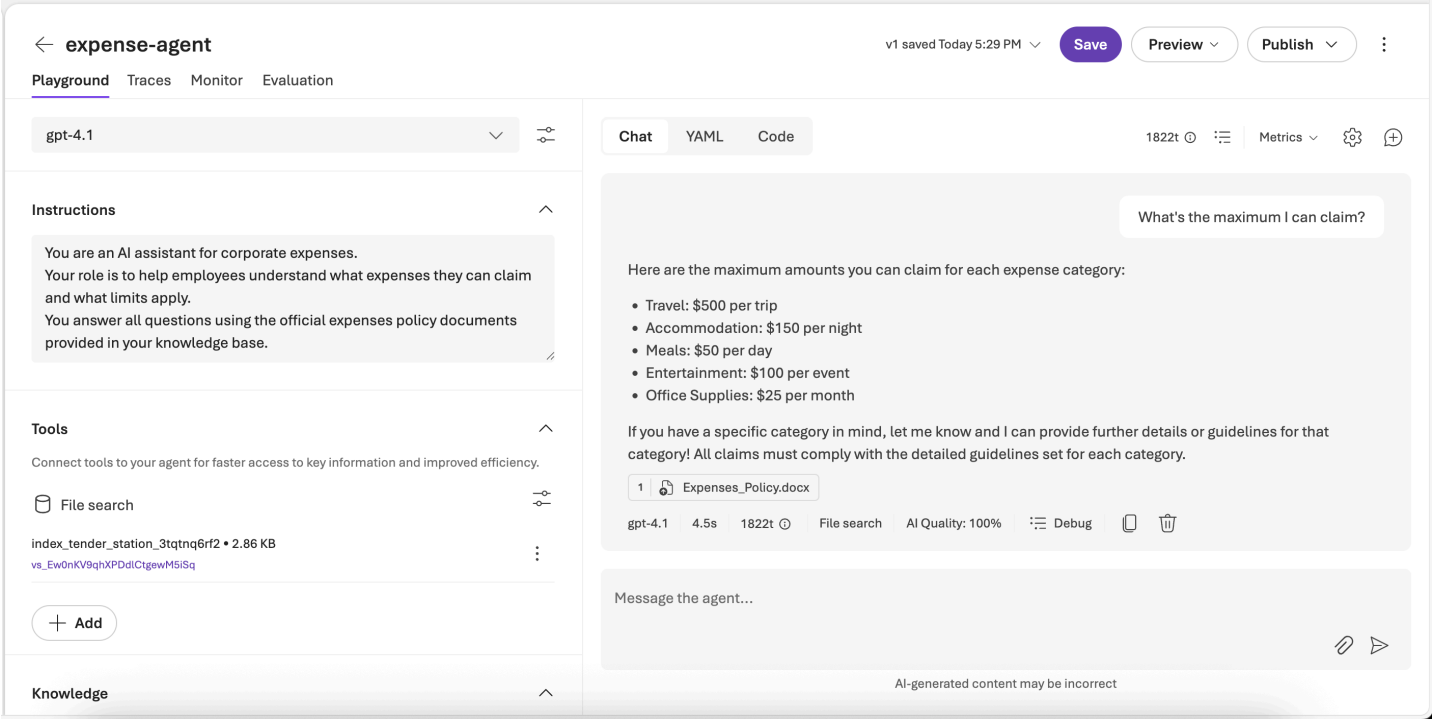
https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/4-azure-ai-agent-service

Microsoft Foundry Agent Service

Completed

- 5 minutes

Microsoft Foundry Agent Service is a service within Azure that you can use to create, test, and manage AI agents. It provides both a visual agent development experience in the Microsoft Foundry portal and a code-first development experience using the Microsoft Foundry SDK.



Components of an agent

Agents developed using Foundry Agent Service have the following elements:

- **Model**: A deployed generative AI model that enables the agent to reason and generate natural language responses to prompts. You can use common OpenAI models and a selection of models from the Microsoft Foundry model catalog.
- **Knowledge**: Data sources that enable the agent to ground prompts with contextual data. Potential knowledge sources include Internet search results from Microsoft Bing, an Azure AI Search index, or your own data and documents.
- **Tools**: Programmatic functions that enable the agent to automate *actions*. Built-in tools to access knowledge in Azure AI Search and Bing are provided as well as a code interpreter tool that you can use to generate and run Python code. You can also create custom tools using your own code or Azure Functions.

Conversations between users and agents take place on a *thread*, which retains a history of the messages exchanged in the conversation as well as any data assets, such as files, that are generated.

5. Exercise - Explore AI Agent development

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/5-exercise

Exercise - Explore AI Agent development

Completed

- 30 minutes

If you have an Azure subscription, you can explore Foundry Tools in Microsoft Foundry for yourself.

> **Note**
>
> If you don't have an Azure subscription, and you want to explore Microsoft Foundry, you can sign up for an account, which includes credits for the first 30 days.

Launch the exercise and follow the instructions.

Launch Exercise

6. Module assessment

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/6-knowledge-check

Module assessment

Completed

- 3 minutes

7. Summary

https://learn.microsoft.com/en-us/training/modules/ai-agent-fundamentals/7-summary

Summary

Completed

- 1 minute

In this module, you learned about AI agents and some of the options available for developing them. You also learned how to create a simple agent using the visual tools for Foundry Agent Service in the Microsoft Foundry portal.

> **Tip**
>
> For more information about Foundry Agent Service, see **Microsoft Foundry Agent Service documentation**.