

AI-900

Generative AI



AI-900 Agenda



1: AI Overview


2: Computer Vision

3: Natural Language Processing

4: Document Intelligence and Knowledge Mining

5: Generative AI

LP Agenda

- Fundamentals of generative AI 
- Fundamentals of Azure OpenAI Service

API URL


OT

GET
POST

App

Fundamentals of Generative AI



What is generative AI? LLM

AI: imitates human behavior by using machine learning to interact with the environment and execute tasks without explicit directions on what to output.

Generative AI: creates original content, such as generative AI that has been built into chat applications. Generative AI applications take in natural language input, and return appropriate responses in a variety of formats:

VS Code

Github Copilot



Natural language generation



Image generation

DALL-E
Midjourney
...



Code generation

C#
java
PS

Large language models

Generative AI applications are powered by *large language models* (LLMs), which are a specialized type of machine learning model that you can use to perform natural language processing (NLP) tasks, including:

- Determining *sentiment* or otherwise classifying natural language text.
- Summarizing text.
- Comparing multiple text sources for semantic similarity.
- Generating new natural language.

Large language models - transformer model

Transformer model architecture consists of two main components, or blocks:

- An *encoder* block that creates semantic representations of the training vocabulary.
- A *decoder* block that generates new language sequences.

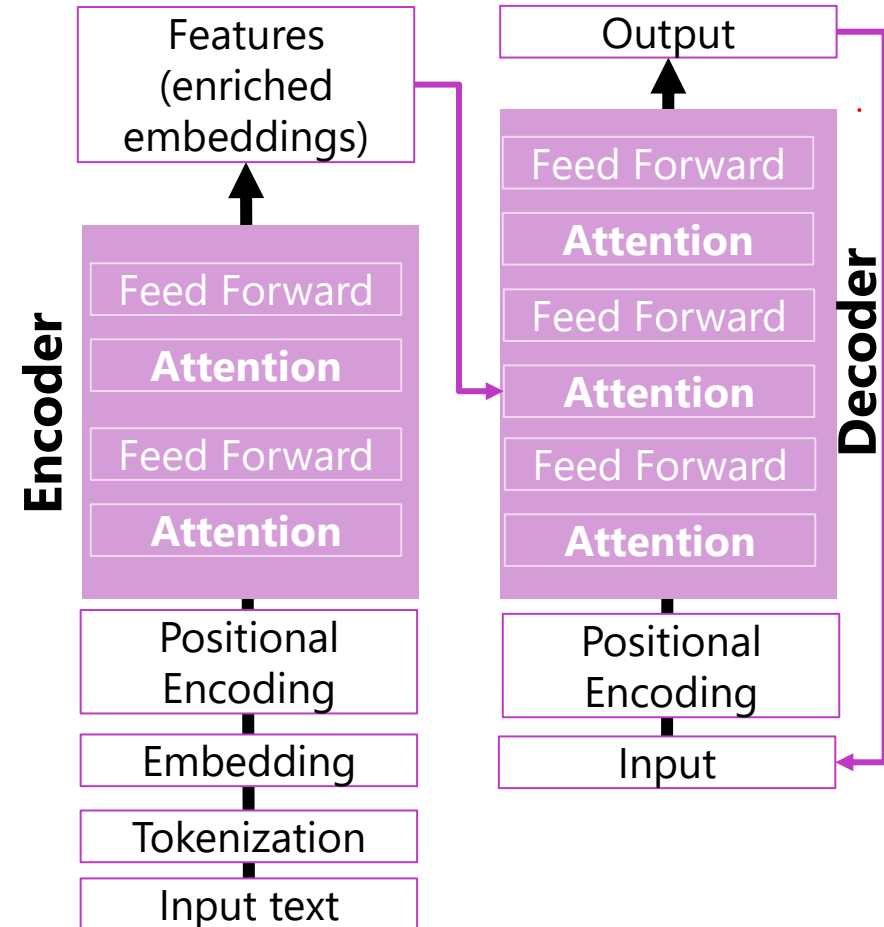
Text is *tokenized* so each word or phrase is represented by a unique numeric token

The tokens are assigned *embeddings* – vector values with multiple dimensions

Attention layers examine each token in turn and determine embedding values that reflect semantic relationships between tokens

In the decoder, these relationships are used to predict the most probable sequence of tokens

3 brown 1 blue
youtube



Large language models – tokenization

Step one: tokenization

The first step in training a transformer model is to decompose the training text into *tokens*.

Example sentence: *I heard a dog bark loudly at a cat.*

"I"=1

"heard"=2

"a"=3

"dog"=4

"bark"=5

"loudly"=6

"at"=7

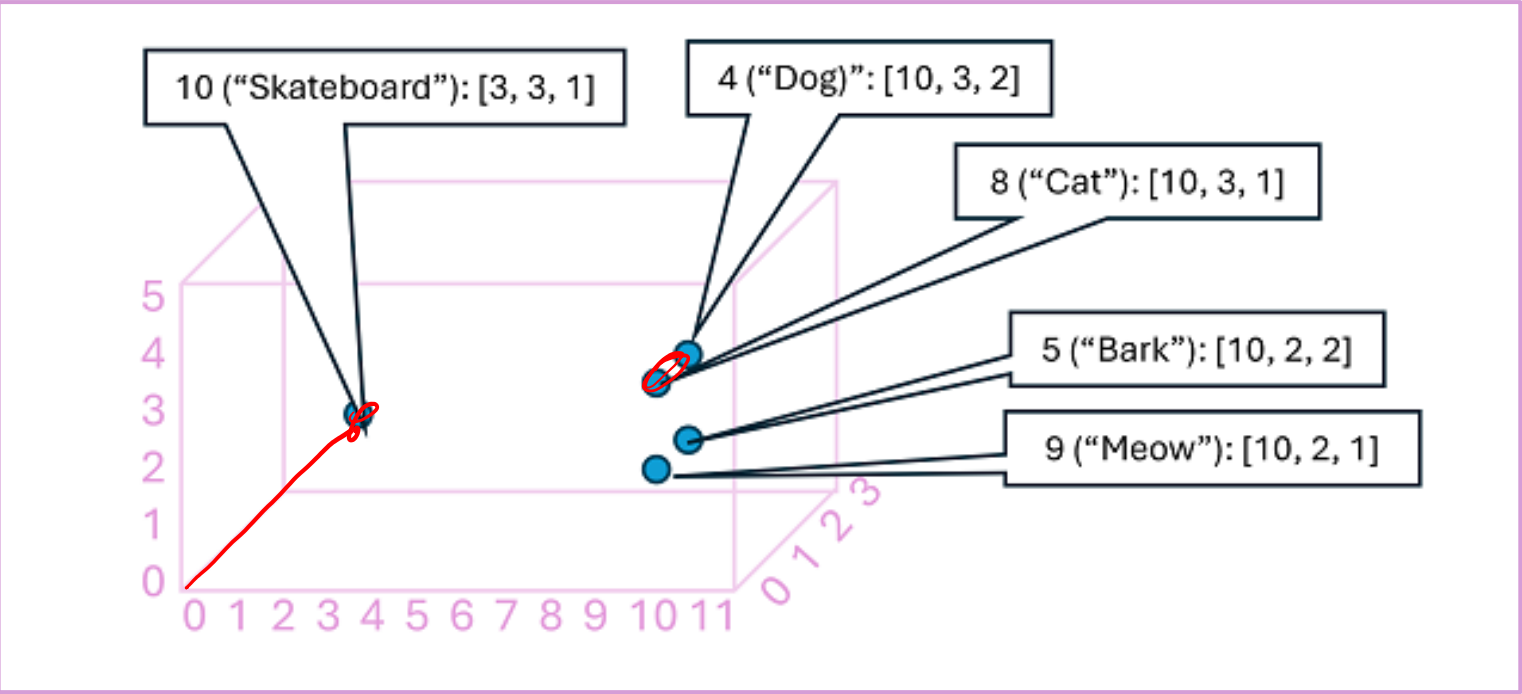
"cat"=8

- The sentence is now represented with the tokens: [1 2 3 4 5 6 7 3 8].
- Note "a" is tokenized as 3 only once.
- Similarly, the sentence "I heard a cat" could be represented with the tokens [1 2 3 8].

Large language models – embeddings

Step two: embeddings

The relationships between tokens are captured as vectors, known as embeddings.



Token	Word	Embedding
10	Skateboard	<u>[3, 3, 1]</u>
4	Dog	[10,3,2]
8	Cat	[10,3,1]
5	Bark	[10,2,2]
9	Meow	[10,2,1]

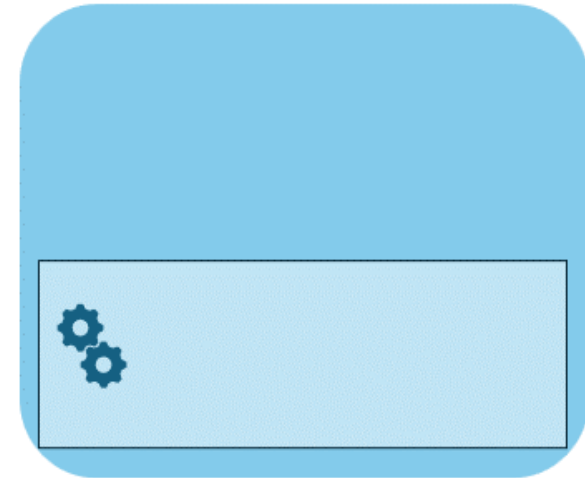
Large language models – attention

Step three: attention

Capture the strength of relationships between tokens using the attention technique.

Example:

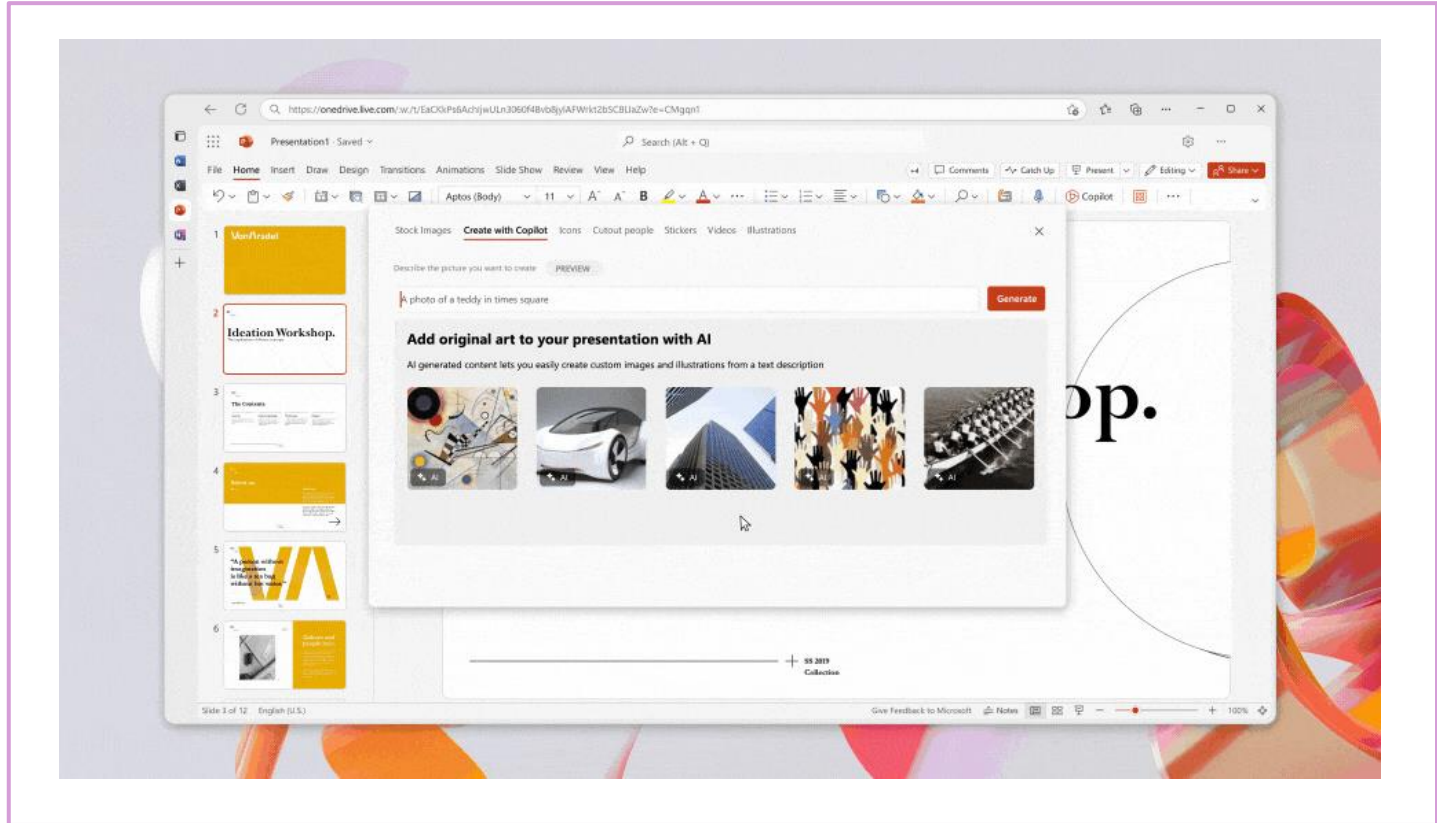
- Goal: predict token after "**dog**."
- Represent "**I heard a dog**" as vectors.
- Assign "**heard**" and "**dog**" more weight.
- Several possible tokens can come after dog.
- The most probable token is added to the sequence, in this case "**bark**."



Copilots

Copilots are often integrated into other applications and provide a way for users to get help with common tasks from a generative AI model.

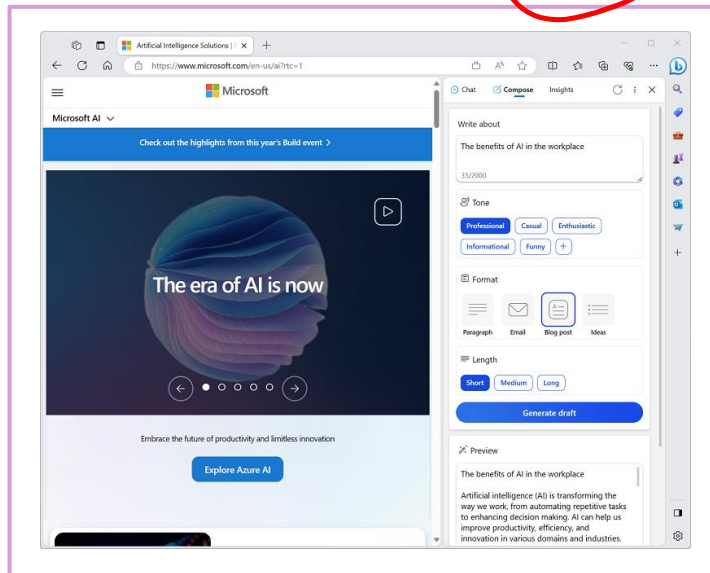
- Developers can build copilots that submit prompts to large language models and generate content for use in applications.
- Business users can use copilots to boost their productivity and creativity with AI-generated content.



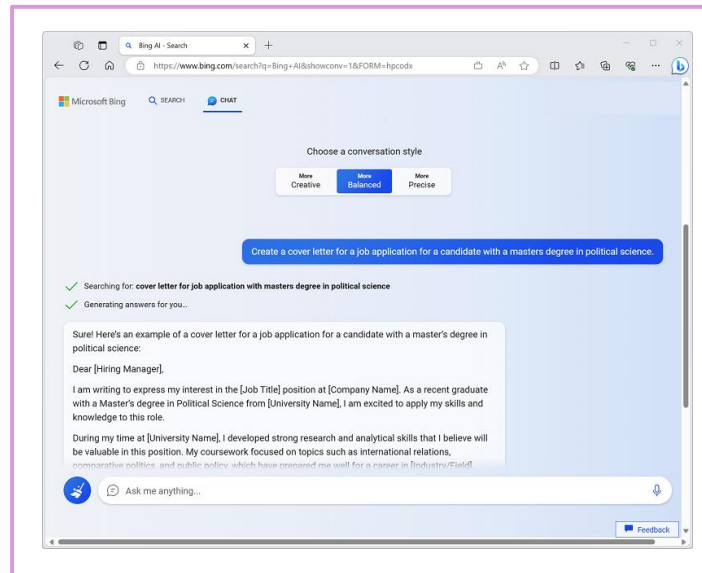
What are copilots?

Other examples of copilots

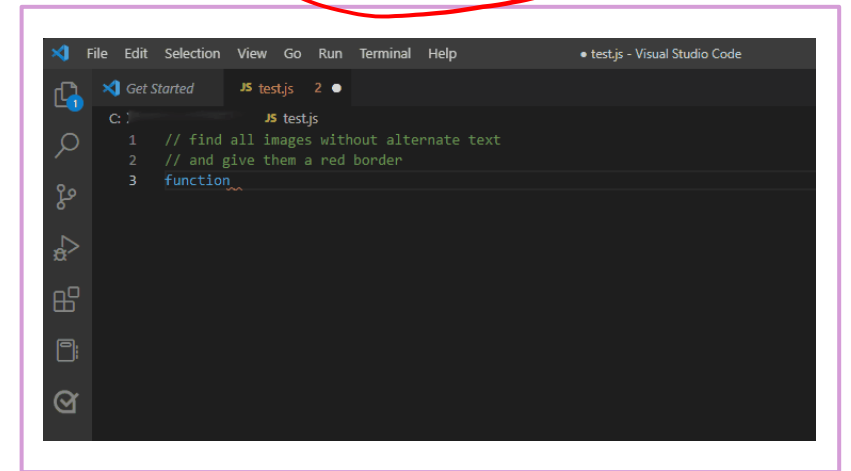
Microsoft Edge browser copilot



Microsoft Bing ~~copilot~~



GitHub copilot



And many more!

M365 Copilot
Security
Azure

Improve generative AI responses with prompt engineering

The term *prompt engineering* describes the process of prompt improvement.

Both developers who design applications and consumers who use applications can improve the quality of responses from generative AI by using direct language, system messages, examples, and/or grounding data.

	Description	Example
Direct language	You can get the most useful completions by being explicit about the kind of response you want.	" Create a list of 10 things to do in Edinburgh during August".
System messages	Describe how the chat should act.	"You're a helpful assistant that responds in a cheerful, friendly manner ".
Providing examples	LLMs generally support <i>zero-shot learning</i> in which responses can be generated without prior examples. However, you can also provide a few example responses, known as <i>few-shot learning</i> .	"Visit the castle in the morning before the crowds arrive".
Grounding data	You can include <i>grounding</i> data to provide context.	Including email text with the prompt "Summarize my email".

RAG

Exercise: Explore generative AI with Bing Chat copilot



In this exercise, you will use Bing Chat copilot to explore generative AI.

1. Use the hosted environment and Azure credentials provided for this exercise.
2. The instructions are also available on Learn: <https://aka.ms/ai900-bing-copilot>

Fundamentals of Azure OpenAI Service



What is Azure OpenAI?

Azure OpenAI service is Microsoft's cloud solution for deploying, customizing, and hosting large language models.

Azure OpenAI service consists of:

- Pre-trained generative AI models.
- Customization capabilities.
- Built-in tools to detect and mitigate harmful use cases so users can implement AI responsibly.
- Enterprise-grade security with role-based access control (RBAC) and private networks.

You can use several methods to develop Azure OpenAI solutions: Azure AI Studio, REST API, supported SDKs, and Azure CLI.

What models does Azure OpenAI support?

Azure OpenAI supports many LLMs:		Description
GPT-4		A set of models that improve on GPT-3.5 and can understand as well as generate natural language and code.
GPT-3.5		A set of models that improve on GPT-3 and can understand as well as generate natural language and code.
Embeddings		A set of models that can convert text into numerical vector form to facilitate text similarity.
DALL-E (preview)		A series of models in preview that can generate original images from natural language.

How to use Azure OpenAI

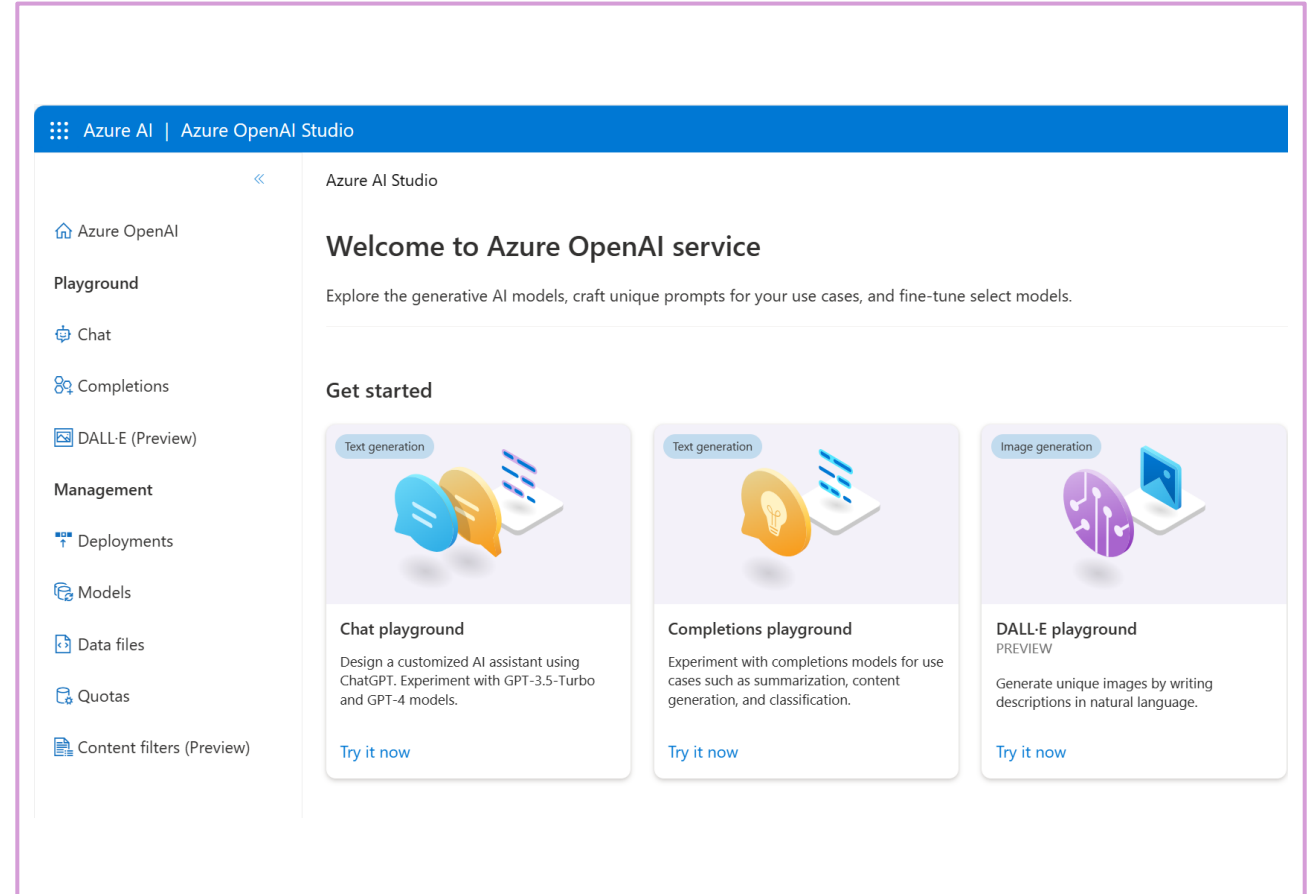
Azure OpenAI Studio:

- Build and deploy AI models for software applications
- Powered by generative AI models optimized for various tasks
- Azure OpenAI models include: **GPT-4, GPT-3.5, Embeddings** and **DALL-E models**



Playgrounds:

- Experiment with Azure OpenAI models without coding
- Use the assistant setup to instruct the model about how it should behave



Azure OpenAI's natural language capabilities

Generative pre-trained transformer (GPT) models are excellent at both understanding and creating natural language.

For example, given a prompt where the user types in text asking for a cooking recipe:

Write recipe instructions for a fruit pie based on these ingredients:

Strawberries
Blueberries
Flour
Eggs
Milk

Here's an example of the Azure OpenAI generated response:

Instructions:

1. Preheat your oven to 350 degrees Fahrenheit.
2. In a medium bowl, mix together 1 cup of flour and 1 egg until it forms a dough.
3. Grease a 9-inch pie dish with butter or non-stick cooking spray.
4. Roll out the dough and place it in the pie dish.
5. In a separate bowl, mix together 1 cup of strawberries, 1 cup of blueberries, and 1 cup of milk.
6. Pour the fruit mixture into the pie dish.
7. Bake

Azure OpenAI's code generation capabilities

GPT models translate natural language or code snippets into code.

Code generation goes beyond just writing code from natural language prompts. Given the following code, it can generate unit tests as shown on the right:

Python

```
# Python 3
def mult_numbers(a, b):
    return a * b

# Unit test
def
```



Python

```
# Python 3
def mult_numbers(a, b):
    return a * b

# Unit test
def test_mult_numbers():
    assert mult_numbers(3, 4) == 12
    assert mult_numbers(0, 10) == 0
    assert mult_numbers(4, 0) == 0

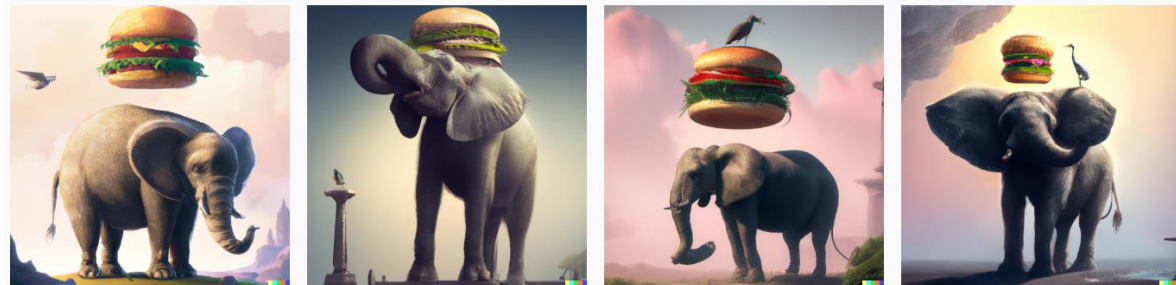
# Unit test
def test_mult_numbers_negative():
    assert mult_numbers(-1, 10) == -10
    assert mult_numbers(10, -1) == -10
```

Azure OpenAI's image generation capabilities

Generative AI models can edit and create images. The model that works with images is called DALL-E, which supports image creation, image editing and image variations creation.

- **Image generation:** With DALL-E, you can even request an image in a particular style. Styles can be used for edits and variations as well.
- **Editing an image:** DALL-E can edit the image as requested by changing its style, adding or removing items, or generating new content to add.
- **Image variations:** Image variations can be created by providing an image and specifying how many variations of the image you would like.

Prompt: Create four variations of an image of an elephant with a hamburger.

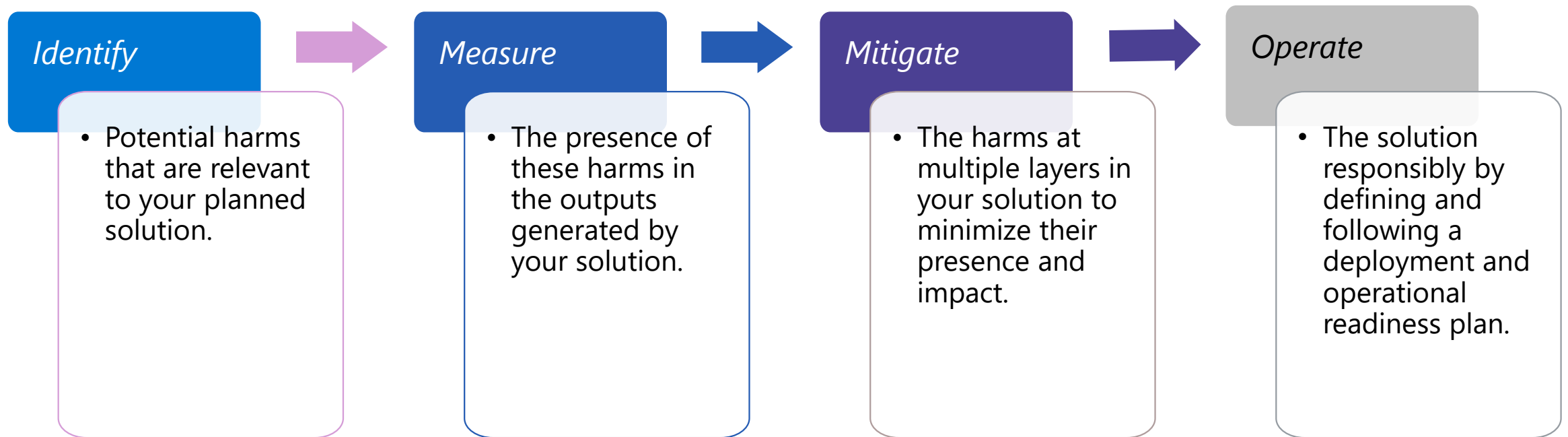


Responsible generative AI



Plan a responsible generative AI solution

Four stage process to develop and implement a plan for responsible AI are:



Knowledge check



Which of the following would you classify as Generative AI?

- ☐ Transcribing speech into text
- ☐ Writing a new report
- ☐ Extracting information from a receipt

How can you demonstrate the generative AI capabilities of Azure OpenAI without writing code?

- ☐ OpenAI Studio
- ☐ Azure portal
- ☐ REST API

You are designing an application using Azure OpenAI services. What steps should you take to ensure responsible use of AI in your application?

- ☐ Nothing is required. Responsible AI is built into the model
- ☐ Create a plan that includes training potential users about responsible AI
- ☐ Create a plan that includes the following steps: Plan, Measure, Mitigate, and Operate

Knowledge check



Which of the following would you classify as Generative AI?

- ☐ Transcribing speech into text
- ☒ Writing a new report
- ☐ Extracting information from a receipt

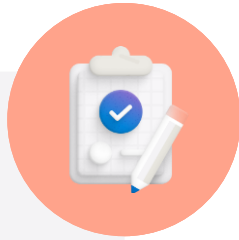
How can you demonstrate the generative AI capabilities of Azure OpenAI without writing code?

- ☒ OpenAI Studio
- ☐ Azure portal
- ☐ REST API

You are designing an application using Azure OpenAI services. What steps should you take to ensure responsible use of AI in your application?

- ☐ Nothing is required. Responsible AI is built into the model
- ☐ Create a plan that includes training potential users about responsible AI
- ☒ Create a plan that includes the following steps: Plan, Measure, Mitigate, and Operate

Summary



Fundamentals of generative AI

- Understand generative AI's place in the development of artificial intelligence.
- Understand large language models and their role in intelligent applications.
- Describe how Azure OpenAI supports intelligent application creation.
- Describe examples of copilots and good prompts.

Fundamentals of Azure OpenAI Service

- Describe Azure OpenAI workloads and access the Azure OpenAI Service
- Understand generative AI models
- Understand Azure OpenAI's language, code, and image capabilities

Explore responsible generative AI

- Understand the four stages for a responsible generative AI solution

