Analyze images

With the Azure AI Vision service, you can use **pre-trained models** to **analyze images** and **extract insights and information** from them.

Learning objectives

After completing this module, you'll be able to:

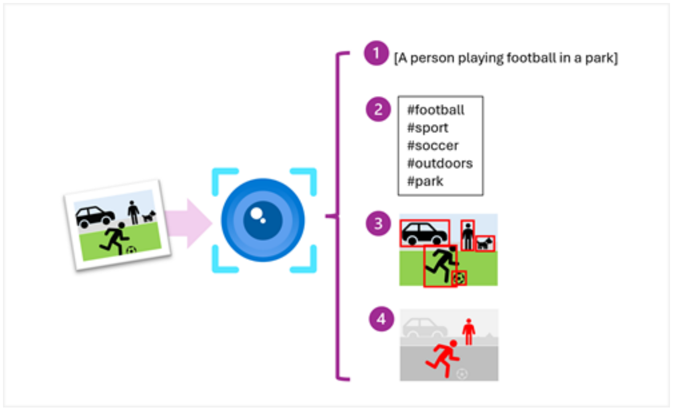
- Provision an Azure AI Vision resource.
- Use the Azure AI Vision SDK to connect to your resource.
- Write code to analyze an image.

Introduction

Computer Vision is a branch of artificial intelligence (AI) in which software **interprets visual input, often from images or video feeds**. In Microsoft Azure, you can use the Azure AI Vision service to implement multiple computer vision scenarios, including:

- *Image analysis*
- *Optical character recognition (OCR)*
- *Face detection and analysis*
- *Video analysis*

In this module, we'll focus on **image analysis**, and explore how to build applications that use the Azure AI Vision service to **analyze and extract and infer insights from images**.

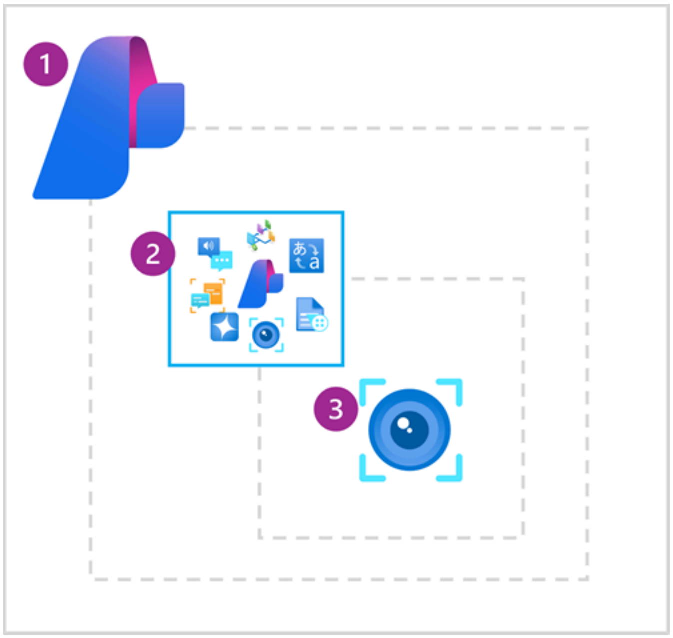


As shown in this conceptual diagram, the Azure AI Vision service provides services that you can use to analyze images and:

- Generate a **caption** for an image based on its contents.
- Suggest appropriate **tags** to associate with an image.
- **Detect and locate common objects** in an image.
- **Detect and locate people** in an image.

Provision an Azure AI Vision resource

To use Azure AI Vision image analysis services, you need to provision an **Azure AI Vision resource** in your Azure subscription. You can choose from multiple provisioning options:



1. Create an **Azure AI Foundry** project and an associated **hub**. By default, an **Azure AI Foundry hub** includes an **Azure AI services** multi-service resource, which includes Azure AI Vision. Azure AI Foundry projects are recommended for development of AI solutions on Azure that combine generative AI, agents, and pre-built Azure AI services, or which involve collaborative development by a team of software engineers and service operators.
2. If you don't need all of the functionality in an Azure AI Foundry hub, you can create an **Azure AI services multi-service resource** in your Azure subscription. You can then use this resource to access Azure AI Vision services and other AI services through a **single endpoint** and **key**.
3. If you only need to use Azure AI Vision functionality, or you're just experimenting with the service, you can create a standalone **Computer Vision** resource in your Azure subscription. One benefit of this approach is that the standalone service provides a free tier that you can use to explore the service at no cost.

> Tip: If you're unfamiliar with Azure AI Foundry and Azure AI services, consider completing the Plan and prepare to develop AI solutions on Azure module.

Connecting to your resource

After you've deployed your resource, you can use the Azure AI Vision REST API or a **language-specific SDK** (such as the Python SDK or Microsoft .NET SDK) to connect to it from a client application.

Every Azure AI Vision resource provides an **endpoint** to which client applications must connect. You can find the endpoint for your resource in the Azure portal, or if you're working in an Azure AI Foundry project, in the Azure AI Foundry portal. The endpoint is in the form of a URL, and typically looks something like this:

```
https://<resource_name>.cognitiveservices.azure.com/
```

To connect to the endpoint, client applications must be *authenticated*. Options for **authentication** include:

- **Key-based authentication**: Client applications are authenticated by passing an **authorization key** (which you can find and manage in the portal).
- **Microsoft Entra ID authentication**: Client applications are authenticated by using a **Microsoft Entra ID token** for credentials that have permission to access the Azure AI Vision resource in Azure.

When developing and testing an application, it's common to use key-based authentication or Microsoft Entra ID authentication based on your own Azure credentials. **In production, consider using Microsoft Entra ID authentication based on a managed identity for your Azure application or use Azure Key Vault to store authorization keys securely**.

> Note: When using an Azure AI services resource in an Azure AI Foundry project, you can use the Azure AI Foundry SDK to connect to the project using Microsoft Entra ID authentication, and then retrieve the connection information for your Azure AI services resource, including the authorization key, from the project.

Analyze an image

After connecting to your Azure AI Vision resource endpoint, your client application can use the service to perform image analysis tasks.

Note the following requirements for image analysis:

- The image must be presented in **JPEG, PNG, GIF, or BMP** format.
- The file size of the image must be **less than 4 megabytes (MB)**.
- The dimensions of the image must be **greater than 50 x 50 pixels**.

Submitting an image for analysis

To analyze an image, you can use the Analyze Image REST method or the equivalent method in the SDK for your preferred programming language, specifying the visual features you want to include in the analysis.

```
from azure.ai.vision.imageanalysis import ImageAnalysisClient
from azure.ai.vision.imageanalysis.models import VisualFeatures
from azure.core.credentials import AzureKeyCredential

client = ImageAnalysisClient(     endpoint="<YOUR_RESOURCE_ENDPOINT>",     credential=AzureKeyCredential("<YOUR_AUTHORIZATION_KEY>") )

result = client.analyze(     image_data=<IMAGE_DATA_BYTES>, # Binary data from your image file     visual_features=[VisualFeatures.CAPTION, VisualF
```

> Note: In this code example, the client app uses **key-based authentication**. To use Microsoft Entra ID authentication, you can use a **TokenCredential** instead of an **AzureKeyCredential**. The code example submits the image data as a binary object (which would typically be read from an image file). You can also analyze an image based on a URL by using the `analyze_from_url method` .

Available visual features are contained in the `VisualFeatures` enumeration:

- `VisualFeatures.Tags` : Identifies **tags** about the image, including objects, scenery, setting, and actions
- `VisualFeatures.Objects` : Returns the **bounding box** for each detected object
- `VisualFeatures.Caption` : Generates a **caption** of the image in natural language
- `VisualFeatures.DenseCaptions` : Generates **more detailed captions** for the objects detected
- `VisualFeatures.People` : Returns the **bounding box for detected people**
- `VisualFeatures.SmartCrops` : Returns the **bounding box of the specified aspect ratio** for the area of interest
- `VisualFeatures.Read` : Extracts **readable text**

Specifying the visual features you want analyzed in the image determines what information the response will contain. Most responses will contain a **bounding box** (if a location in the image is reasonable) or a **confidence score** (for features such as **tags or captions**).

Processing the response

This method returns a JSON document containing the requested information. The JSON response for image analysis looks similar to this example, depending on your requested features:

```
 {    "apim-request-id": "abcde-1234-5678-9012-f1g2h3i4j5k6",    "modelVersion": "<version>",    "denseCaptionsResult": {     "values": [       {
}
```

Exercise - Analyze images

In this exercise, you use the Azure AI Vision SDK to develop a client application that analyzes images.

Analyze images

Azure AI Vision is an artificial intelligence capability that enables software systems to interpret visual input by analyzing images. In Microsoft Azure, the **Vision** Azure AI service provides pre-built models for common computer vision tasks, including analysis of images to suggest captions and tags, detection of common objects and people. You can also use the Azure AI Vision service to remove the background or create a foreground matting of images.

Module assessment

1. What is the purpose of Azure AI Vision Image Analysis? **To extract information about visual features in images**.
2. You want to use Azure AI Vision Image Analysis to generate suggested text descriptions for an image. Which visual feature should you specify? **DenseCaptions**

Summary

In this module, you learned how to provision an Azure AI Vision resource and use it from a client application to analyze images.

You can use Azure AI Vision's image analysis capabilities in scenarios that require information extraction or inference from images. A common use case is **digital asset management (DAM)**, in which you need to *tag, catalog, and index image-based data*.

To learn more about image analysis with the Azure AI Vision service, see the Azure AI Vision documentation.

Read text in images with OCR

The **Azure AI Vision Image Analysis** service uses algorithms to process images and return information. This module teaches you how to use the Image Analysis API for **optical character recognition (OCR)**.

Learning objectives

In this module, you'll learn how to:

- Describe the OCR capabilities of **Azure AI Vision's Image Analysis API**.
- Use the Azure AI Vision service Image Analysis API to **extract text from images**.

Introduction

We live in a digital world, in which data is increasingly captured as images. Often, those images contain text, which you need to be able to extract from their pixelated format in the image for processing, indexing, and other tasks. Everyday examples include:

- Meeting a new business associate and taking a photograph of their **business card** to store their contact details digitally.
- **Scanning a document or ID card** to include in an application for a government or commercial service.
- Taking a photo of a **menu or recipe** to store it in a digital notebook.
- Photographing **street signs or store fronts** so you can submit the text they contain to a translation app.
- Digitizing **handwritten notes** using a cellphone camera.

In this module, we'll explore the **optical character recognition (OCR)** capabilities of the Azure AI Vision Image Analysis API, which makes these scenarios, and more, possible.

Explore Azure AI options for reading text

There are multiple Azure AI services that read text from documents and images, each optimized for results depending on the input and the specific requirements of your application.

Azure AI Vision

Azure AI Vision includes an *image analysis* capability that supports **optical character recognition (OCR)**. Consider using Azure AI Vision in the following scenarios:

- **Text location and extraction from scanned documents**: Azure AI Vision is a great solution for general, unstructured documents that have been scanned as images. For example, **reading text in labels, menus, or business cards**.
- **Finding and reading text in photographs**: Examples include photo's that include **street signs and store names**.
- **Digital asset management (DAM)**: Azure AI Vision includes functionality for analyzing images beyond extracting text; including **object detection**, **describing or categorizing an image**, **generating smart-cropped thumbnails** and more. These capabilities make it a useful service when you need to catalog, index, or analyze large volumes of digital image-based content.

Azure AI Document Intelligence

Azure AI Document Intelligence is a service that you can use to **extract information from complex digital documents**. Azure AI Document Intelligence is designed for **extracting text, key-value pairs, tables, and structures** from documents automatically and accurately. Key considerations for choosing Azure AI Document Intelligence include:

- **Form processing**: Azure AI Document Intelligence is specifically designed to **extract data from forms, invoices, receipts, and other structured documents**.
- **Prebuilt models**: Azure AI Document Intelligence provides prebuilt models for common document types to reduce complexity and integrate into workflows or applications.

- **Custom models**: Creating custom models tailored to your specific documents, makes Azure AI Document Intelligence a flexible solution that can be used in many business scenarios.

Azure AI Content Understanding

Azure AI Content Understanding is a service that you can use to *analyze and extract information from multiple kinds of content*; including **documents, images, audio streams, and video**. It is suitable for:

- **Multimodal content extraction**: **Extracting content and structured fields** from documents, forms, audio, video, and images.
- **Custom content analysis scenarios**: Support for customizable analyzers enables you to extract specific content or fields tailored to business needs.

> Note: In the rest of this module, we'll focus on the OCR image analysis feature in Azure AI Vision. To learn more about Azure AI Document Intelligence and Azure AI Content understanding, consider completing the following training modules:
> Plan an Azure AI Document Intelligence solution
> Analyze content with Azure AI Content Understanding

Read text with Azure AI Vision Image Analysis

To use Azure AI Vision for image analysis, including optical character recognition, you must provision an **Azure AI Vision resource** in an Azure subscription. The resource can be:

- An **Azure AI Services** multi-service resource (either deployed as part of an Azure AI Foundry hub and project, or as a standalone resource).
- A **Computer Vision** resource.

To use your deployed resource in an application, you must connect to its **endpoint** using either **key-based authentication** or **Microsoft Entra ID authentication**. You can find the endpoint for your resource in the Azure portal, or if you're working in an Azure AI Foundry project, in the Azure AI Foundry portal. **The endpoint is in the form of a URL**, and typically looks something like this:

```
https://<resource_name>.cognitiveservices.azure.com/
```

After establishing a connection, you can use the OCR feature by calling the **ImageAnalysis** function (via the REST API or with an equivalent SDK method), passing the image URL or binary data, and optionally specifying the language the text is written in (with a default value of en for English).

```
https://<endpoint>/computervision/imageanalysis:analyze?features=read&...
```

To use the Azure AI Vision Python SDK to extract text from an image, install the **azure-ai-vision-imageanalysis** package. Then, in your code, use either *key-based authentication* or *Microsoft Entra ID authentication* to connect an **ImageAnalysisClient** object to an Azure AI Vision resource. To find and read text in an image, call the **analyze** (or analyze_from_url) method, specifying the **VisualFeatures.READ** enumeration.

```
from azure.ai.vision.imageanalysis import ImageAnalysisClient
from azure.ai.vision.imageanalysis.models import VisualFeatures
from azure.core.credentials import AzureKeyCredential

client = ImageAnalysisClient(     endpoint="<YOUR_RESOURCE_ENDPOINT>",     credential=AzureKeyCredential("<YOUR_AUTHORIZATION_KEY>") )

result = client.analyze(     image_data=<IMAGE_DATA_BYTES>, # Binary data from your image file     visual_features=[VisualFeatures.READ],     langu
```

The results of the Read OCR function are returned **synchronously**, either as **JSON** or the **language-specific object** of a similar structure. These results are broken down in **blocks** (with the current service only using one block), then **lines**, and then **words**. Additionally, the text values are included at both the **line** and **word** levels, making it easier to read entire lines of text if you don't need to extract text at the individual word level.

Exercise - Read text in images

Now it's your turn to try using the OCR capabilities of Azure AI Vision.

In this exercise, you use the Azure AI Vision Image Analysis SDK to develop a client application that extracts text from images.

Read text in images

Optical character recognition (OCR) is a subset of computer vision that deals with reading text in images and documents. The Azure AI Vision Image Analysis service provides an API for reading text, which you'll explore in this exercise.

Module assessment

1. Which service should you use to locate and read text in signs within a photograph of a street? **Azure AI Vision Image Analysis**
2. Which visual feature enumeration should you use to return OCR results from an image analysis call? **VisualFeatures.Read**
3. Text location information in an image is returned at which levels by Azure AI Vision Image Analysis? **A block containing the location of lines of text as well as individual words**.

Summary

In this module, you learned how to provision an Azure AI Vision resource and use it from a client application to extract text from images.

To learn more about using Azure AI Vision for OCR, see the OCR - Optical Character Recognition in the Azure AI Vision documentation.

Detect, analyze, and recognize faces

The ability for applications to **detect human faces, analyze facial features and emotions, and identify individuals** is a key artificial intelligence capability.

Learning objectives

After completing this module, you'll be able to:

- Describe the capabilities of the **Azure AI Vision Face service**.
- Write code to **detect and analyze faces** in an image.
- Describe **facial recognition** support in Azure AI Vision Face.
- Describe **responsible AI** considerations when developing facial solutions.

Introduction

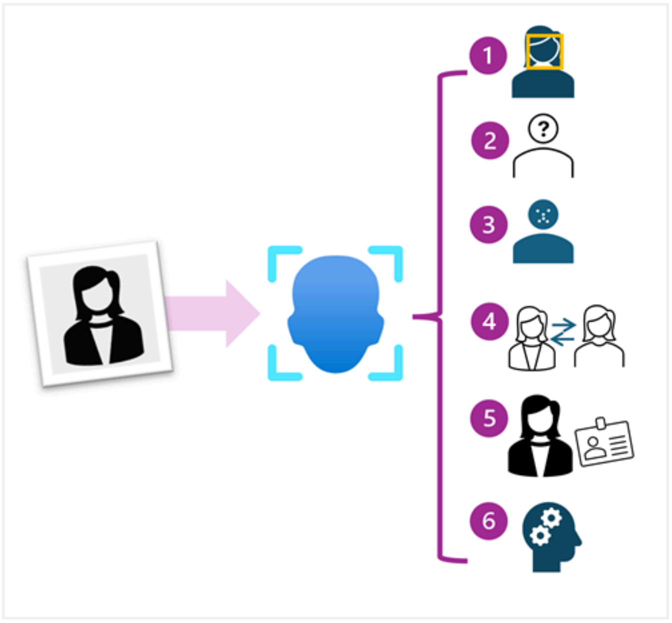
Face detection, analysis, and recognition are all common computer vision challenges for AI systems. The ability to detect when a person is **present**, analyze a person's **facial features**, or **recognize an individual** based on their face is a key way in which AI systems can exhibit human-like behavior and build empathy with users.

In this module, you'll explore how the **Azure AI Vision Face API** enables you to build solutions that analyze faces in images.

> Note: Access to the full capabilities of the Face API is restricted in accordance with Microsoft's responsible AI policies. For details, see Limited Access to Face API. This module describes some capabilities that require explicit access. The practical exercise in the module is based on unrestricted features of the service.

Plan a face detection, analysis, or recognition solution

The **Face** service provides comprehensive **facial detection, analysis, and recognition** capabilities.



Face Features

The Face service provides functionality that you can use for:

1. **Face detection** - for each detected face, the results include an ID that identifies the **face and the bounding box coordinates** indicating its location in the image.
2. **Face attribute analysis** - you can return a wide range of facial attributes, including:
   - **Head pose** (pitch, roll, and yaw orientation in 3D space)
   - **Glasses** (No glasses, Reading glasses, Sunglasses, or Swimming Goggles)
   - **Mask** (the presence of a face mask)
   - **Blur** (low, medium, or high)
   - **Exposure** (under exposure, good exposure, or over exposure)
   - **Noise** (visual noise in the image)
   - **Occlusion** (objects obscuring the face)
   - **Accessories** (glasses, headwear, mask)
   - **QualityForRecognition** (low, medium, or high)
3. **Facial landmark location (spatial points)** - coordinates for key landmarks in relation to facial features (for example, **eye corners, pupils, tip of nose**, and so on)
4. **Face comparison** - you can compare faces across multiple images for similarity (to find individuals with similar facial features) and verification (to determine that a face in one image is the same person as a face in another image)
5. **Facial recognition** - you can train a model with a collection of faces belonging to specific individuals, and use the model to identify those people in new images.
6. **Facial liveness** - liveness can be used to determine if the input video is a real stream or a fake to prevent bad-intentioned individuals from spoofing a facial recognition system.

Face detection and recognition models

The Azure AI Vision Face API is built on face detection and recognition models that have been pre-trained. Multiple versions of these models are available, each with specific strengths and capabilities. For example, newer models exhibit greater accuracy when working with small images; but may not provide the same breadth of facial analysis capabilities. When you use the service in an application, you must select the model you want to use based on your requirements.

> Tip: For guidance about selecting a detection model, see Specify a face detection model. For guidance about how to select a recognition model, see Specify a face recognition model.

Detect and analyze faces

To use the **Azure AI Vision Face API**, you must provision a resource for the service in an Azure subscription. You can provision Face as a single-service resource, or you can use the Face API in a multi-service Azure AI Services resource; which can be provisioned as a standalone resource or as part of an Azure AI Foundry hub.

To use your resource from a client application you must connect to its **endpoint** using either **key-based authentication** or **Microsoft Entra AI authentication**. When using the REST interface you can provide the authentication key or token in the request header. When using a language-specific SDK (for example, the Python `azure-ai-vision-face` package or the Microsoft .NET `Azure.AI.Vision.Face` package), you use a **FaceClient** object to connect to the service.

```
from azure.ai.vision.face import FaceClient
from azure.ai.vision.face.models import *
from azure.core.credentials import AzureKeyCredential

face_client = FaceClient(     endpoint="<YOUR_RESOURCE_ENDPOINT>",     credential=AzureKeyCredential("<YOUR_RESOURCE_KEY>"))
```

To detect and analyze faces in an image, you must specify the model-specific features you want the service to return, and then use the client to call the **Detect** method.

```
# Specify facial features to be retrieved
features = [FaceAttributeTypeDetection01.HEAD_POSE,             FaceAttributeTypeDetection01.OCCLUSION,             FaceAttributeTypeDetection01.AC

# Use client to detect faces in an image
with open("<IMAGE_FILE_PATH>", mode="rb") as image_data:     detected_faces = face_client.detect(         image_content=image_data.read(),
```

The response from the service depends on:

- The model-specific features requested.
- The number of faces detected in the image.

A response for an image containing a single face might look similar to the following example:

Verify and identify faces

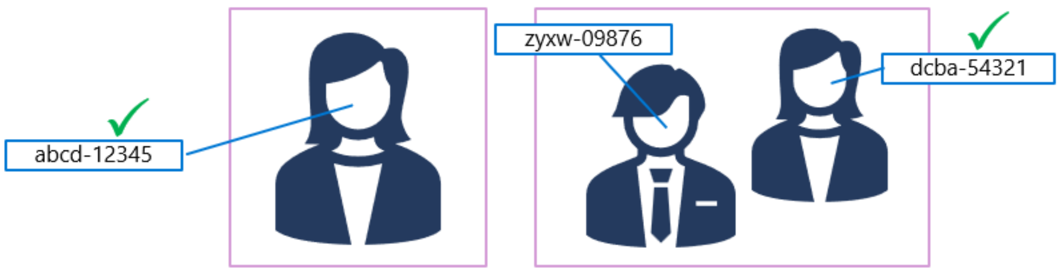
In addition to detecting and analyzing faces, you can use the **Azure AI Vision Face service** to *compare and recognize* faces.

> Important: Usage of facial recognition, comparison, and verification requires approval through a Limited Access policy.

Verifying faces

When a face is detected by the Face service, **a unique ID** is assigned to it and **retained in the service resource for 24 hours**. The ID is a GUID, with no indication of the individual's identity other than their facial features.

While the detected face ID is cached, subsequent images can be used to compare the new faces to the cached identity and determine if they're *similar* (in other words, they share similar facial features) or to *verify* that the same person appears in two images.



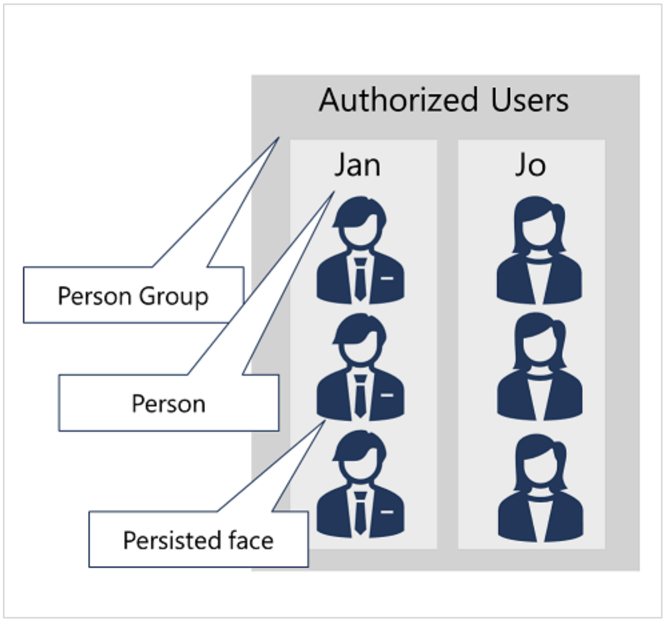
This ability to compare faces anonymously can be useful in systems where it's important to **confirm that the same person is present on two occasions**, without the need to know the actual identity of the person. For example, by taking images of people as they enter and leave a secured space to verify that everyone who entered leaves.

Identifying faces

For scenarios where you need to positively identify individuals, you can train a facial recognition model using face images.

To train a facial recognition model with the Face service:

1. Create a **Person Group** that defines the set of individuals you want to identify (for example, employees).
2. Add a **Person** to the **Person Group** for each individual you want to identify.
3. Add detected faces from multiple images to each **person**, preferably in various poses. The IDs of these faces will no longer expire after 24 hours (so they're now referred to as *persisted* faces).
4. Train the model.



The trained model is stored in your Face (or Azure AI Services) resource, and can be used by client applications to:

1. **Identify** individuals in images.
2. **Verify** the identity of a detected face.
3. Analyze new images to find faces that are **similar** to a known, persisted face.

> Tip: To learn more about using face verification and identification to implement a facial recognition solution, see Face recognition in the Azure AI Vision Face documentation.

Responsible AI considerations for face-based solutions

While all applications of artificial intelligence require considerations for responsible, system that rely on facial or other biometric data can be particularly problematic.

When building a solution that uses facial data, considerations include (but aren't limited to):

- **Data privacy and security**. Facial data is personally identifiable, and should be considered sensitive and private. You should ensure that you have implemented adequate protection for facial data used for model training and inferencing.
- **Transparency**. Ensure that **users are informed** about **how their facial data is used**, and **who will have access** to it.
- **Fairness and inclusiveness**. Ensure that your face-based system can't be used in a manner that is prejudicial to individuals based on their appearance, or to unfairly target individuals.

> Note: You can find details of the responsible AI measures taken in the implementation of the Face API in Data and privacy for Face.

Exercise - Detect and analyze faces

Now it's your turn to try using the Azure AI Vision Face service.

In this exercise, you use the Azure AI Vision Face API to develop a client application that detects and analyzes faces in an image.

Detect and analyze faces

The ability to detect and analyze human faces is a core AI capability. In this exercise, you'll explore two **Azure AI Services** that you can use to work with faces in images: the Azure AI Vision service, and the **Face** service.

Module assessment

1. You need to create a facial recognition solution to identify named employees. Which service should you use? **Azure AI Vision Face**
2. What should you return when analyzing an image to see the spatial points related to facial features? **Landmarks**

Summary

In this module, you have learned how to **detect, analyze, and recognize faces**. Facial analysis and recognition solutions can help you address business requirements for security and authorization, digital asset management, and other scenarios. However, you should be mindful of the need to use biometric data such as facial imaging responsibly.

> Tip: To find out more about the Azure AI Vision Face service, see the Face documentation.

Classify images

Image classification is used to **determine the main subject of an image**. You can use the Azure **AI Custom Vision services** to train a model that classifies images based on your own categorizations.

Learning objectives

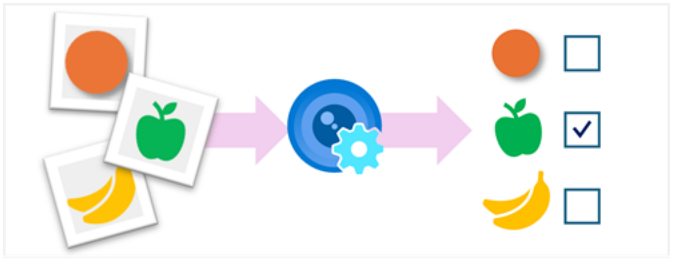
After completing this module, you'll be able to:

- Provision Azure resources for **Azure AI Custom Vision**.
- Train an **image classification model**.
- Use the Azure AI Custom Vision SDK to create an image classification client application.

Introduction

Image classification is a common computer vision problem that requires software to **analyze an image and categorize (or classify) it**.

For example, an unattended checkout system in a grocery store might use a camera to scan each item a customer adds to their cart, and use image classification to identify the product in the image.



In this module, you'll learn how the **Azure AI Custom Vision service** enables you to build your own computer vision models for image classification.

Azure AI Custom Vision

The Azure AI Custom Vision service enables you to build your own computer vision models for **image classification** or **object detection**.

To use the Custom Vision service to create a solution, you need **two Custom Vision resources** in your Azure subscription:

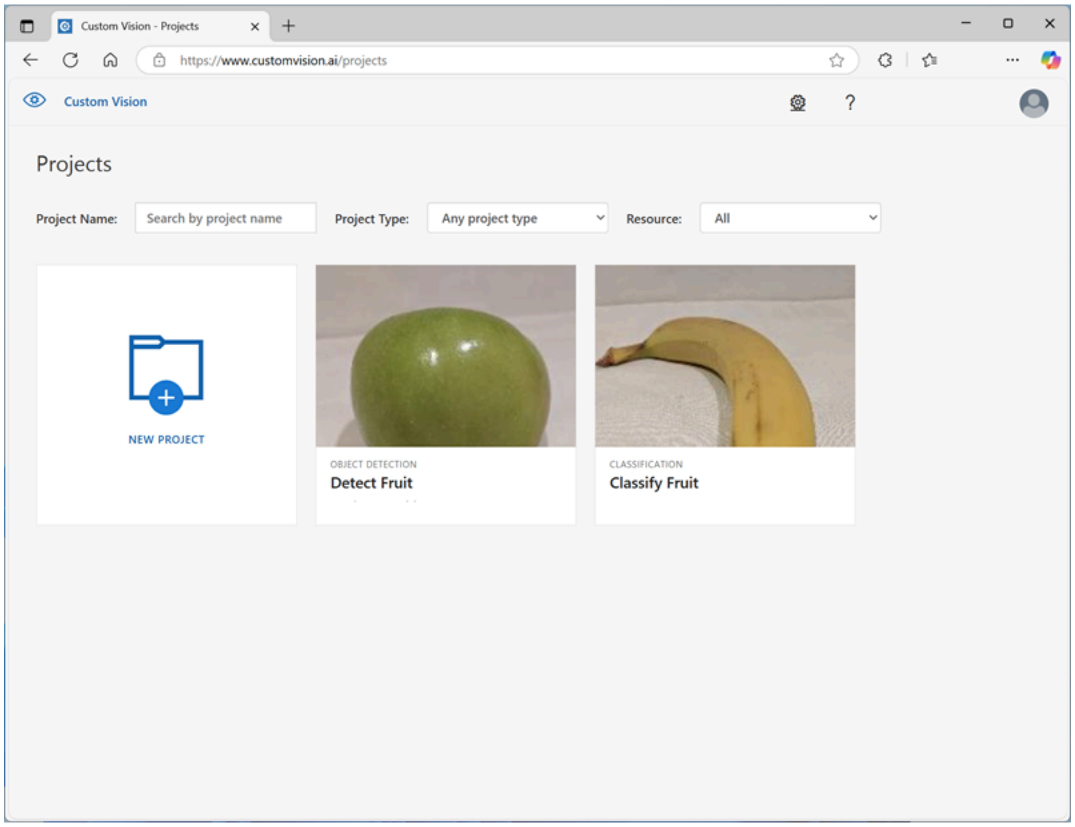
- An **Azure AI Custom Vision training** resource - used to **train a custom model** based on your own training images.
- An **Azure AI Custom Vision prediction** resource - used to **generate predictions** from new images based on your trained model.

When you provision the Azure AI Custom Vision service in an Azure subscription, you can choose to create one or both of these resources. This separation of training and prediction provides flexibility. For example, you can use a training resource in one region to train your model using your own image data; and then deploy one or more prediction resources in other regions to support computer vision applications that need to use your model.

Each resource has its own **unique endpoint and authentication keys**; which are used by client applications to connect and authenticate to the service.

The Custom Vision portal

Azure AI Custom Vision provides a web-based portal, in which you can **train, publish, and test custom vision models**.



You can sign into the Custom Vision portal using your Azure credentials and use it to create image classification or object detection projects that use Azure AI Custom Vision resources in your Azure subscription.

Each project has a unique project ID; which is used by client applications to perform training or prediction tasks using code.

Custom Vision SDKs

You can write code to train and consume custom models by using the Azure AI Custom Vision language-specific SDKs.

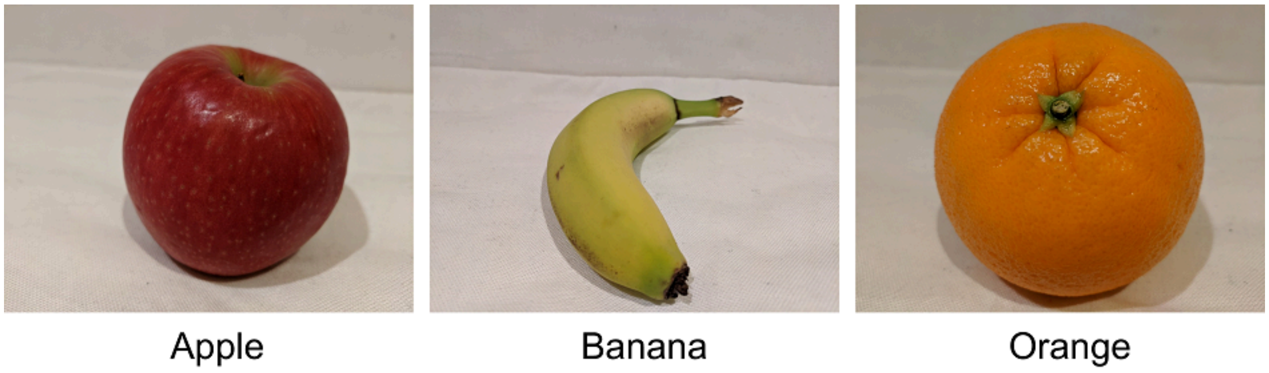
For example, Microsoft C# developers can use the Microsoft.Azure.CognitiveServices.Vision.CustomVision.Training and Microsoft.Azure.CognitiveServices.Vision.CustomVision.Prediction Microsoft .NET packages for training and prediction respectively.

Python developers can perform both training and prediction tasks by using the azure-cognitiveservices-vision-customvision package.

Train an image classification model

Image classification is a computer vision technique in which a model is trained to predict a class label for an image based on its contents. Usually, the class label relates to the main subject of the image.

For example, the following images have been classified based on the type of fruit they contain.

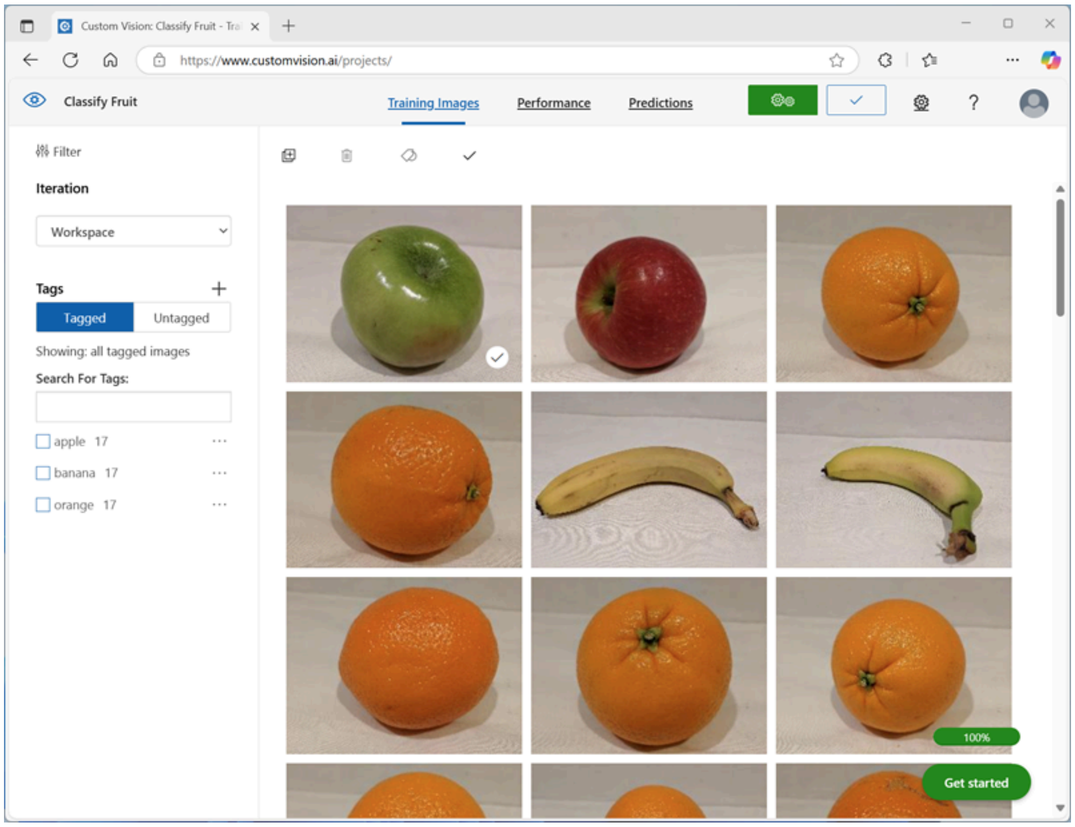


Models can be trained for **multiclass classification** (in other words, there are multiple classes, but each image can belong to only one class) or **multilabel classification** (in other words, an image might be associated with multiple labels).

Training an image classification model

To train an image classification model with the **Azure AI Custom Vision** service, you can use the **Azure AI Custom Vision portal**, the **Azure AI Custom Vision REST API or SDK**, or a combination of both approaches.

In most cases, you'll typically use the Azure AI Custom Vision portal to train your model.



The portal provides a graphical interface that you can use to:

1. **Create** an image classification project for your model and associate it with a training resource.
2. **Upload** images, **assigning** class label tags to them.
3. **Review** and edit tagged images.
4. **Train and evaluate** a classification model.
5. **Test** a trained model.
6. **Publish** a trained model to a prediction resource.

The REST API and SDKs enable you to perform the same tasks by writing code, which is useful if you need to automate model training and publishing as part of a DevOps process.

Create an image classification client application

After you've trained an image classification model, you can use the Azure AI Custom Vision SDK to develop a client application that submits new images to be classified.

```
from msrest.authentication import ApiKeyCredentials
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredictionClient

# Authenticate a client for the prediction API
credentials = ApiKeyCredentials(in_headers={"Prediction-key": "<YOUR_PREDICTION_RESOURCE_KEY>"})
prediction_client = CustomVisionPredictionClient(endpoint="<YOUR_PREDICTION_RESOURCE_ENDPOINT>", credentials=credentials)
```

```
# Get classification predictions for an image
image_data = open("<PATH_TO_IMAGE_FILE>"), "rb").read()
results = prediction_client.classify_image("<YOUR_PROJECT_ID>",                                             "<YOUR_PUBLISHED_MODEL_NAME>",

# Process predictions
for prediction in results.predictions:
if prediction.probability > 0.5:
print(image, ': {} ({:.0%})'.format(prediction.tag_name, prediction.probability))
```

Exercise - Classify images

Now it's your turn to try using the Azure AI Custom Vision service.

In this exercise, you train and publish a custom image classification model, and use the Azure AI Custom Vision SDK to test it.

Classify images

The **Azure AI Custom Vision** service enables you to create computer vision models that are trained on your own images. You can use it to train image classification and object detection models; which you can then publish and consume from applications.

In this exercise, you will use the *Custom Vision* service to train an image classification model that can identify three classes of fruit (apple, banana, and orange).

In the code file, update the configuration values it contains to reflect the **Endpoint** and an **authentication Key** for your Custom Vision training resource, and the **Project ID** for the custom vision project you created previously.

```
PredictionEndpoint=YOUR_PREDICTION_ENDPOINT
PredictionKey=YOUR_PREDICTION_KEY
ProjectID=YOUR_PROJECT_ID
ModelName=fruit-classifier
```

Module assessment

1. What Azure AI Custom Vision resources should you create in Azure to create a custom vision solution? **A Custom Vision resource for training, and another for prediction**.
2. You want to train a model that can categorize an image as "cat" or "dog" based on its subject. What kind of Azure AI Custom Vision project should you create? **Image classification (multiclass)**
3. What information does a client application need to connect to Azure AI Custom Vision and classify an image? **The endpoint and key for the Custom Vision prediction resource, the project ID for your image classification project, and the name of your deployed model**.

Summary

In this module, you learned how to use the Azure AI Custom Vision service to build your own custom vision models for image classification.

Detect objects in images

Object detection is used to **locate and identify objects in images**. You can use **Azure AI Custom Vision** to train a model to detect specific classes of object in images.

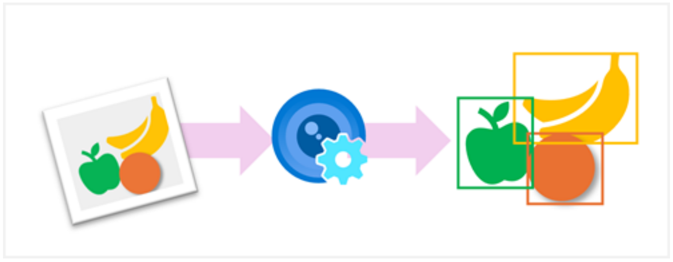
Learning objectives

After completing this module, you'll be able to:

- Provision Azure resources for **Azure AI Custom Vision**
- Understand **object detection**
- Train an **object detector**
- Use the Azure AI Custom Vision SDK to create an **object detection client application**

Introduction

Object detection is a common computer vision problem that requires software to **identify the location of specific classes of object in an image**.



For example, an automated checkout system in a grocery store might use a camera to monitor a checkout conveyer belt on which there might be multiple different items at any one time. The system could use object detection to identify which items are on the belt, and where in the image they appear.

In this module, you'll learn how to use the Azure AI Custom Vision service to create object detection models.

Use Azure AI Custom Vision for object detection

To use the Custom Vision service to create an object detection solution, you need two Custom Vision resources in your Azure subscription:

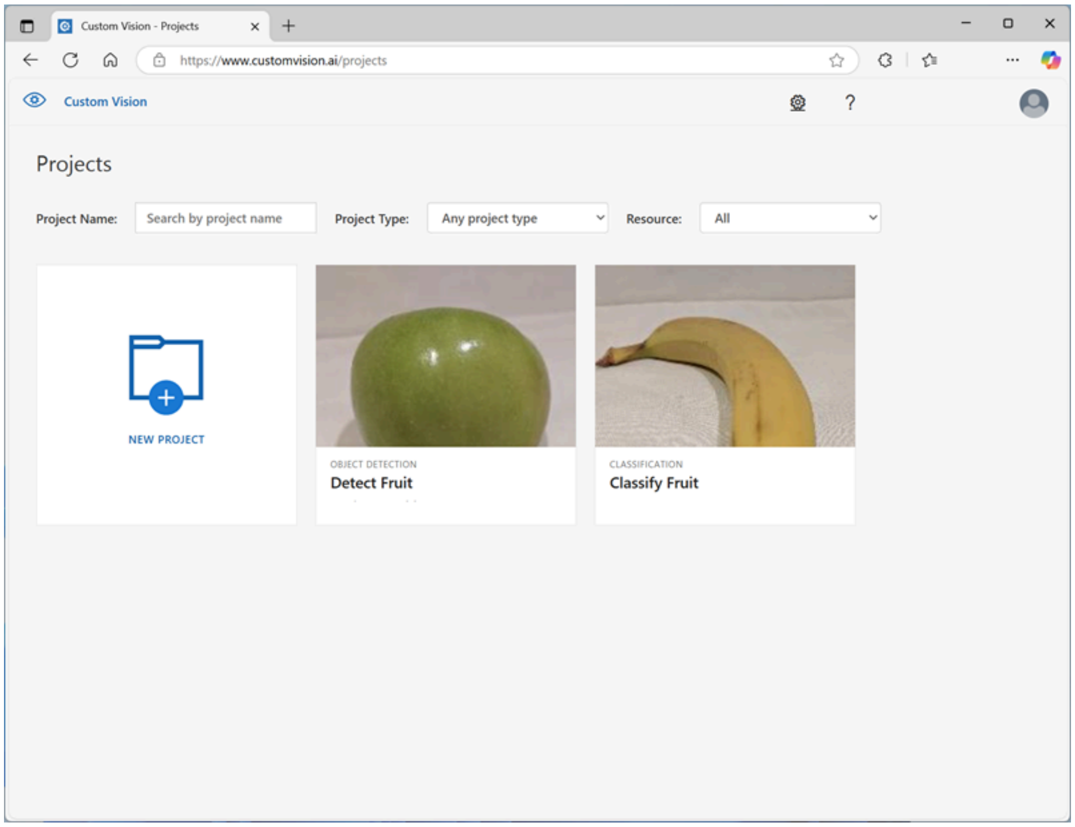
- An **Azure AI Custom Vision *training*** resource - used to train a custom model based on your own training images.
- An **Azure AI Custom Vision *prediction*** resource - used to generate predictions from new images based on your trained model.

When you provision the Azure AI Custom Vision service in an Azure subscription, you can choose to create one or both of these resources. This separation of training and prediction provides flexibility. For example, you can use a training resource in one region to train your model using your own image data; and then deploy one or more prediction resources in other regions to support computer vision applications that need to use your model.

Each resource has its own unique **endpoint and authentication keys**; which are used by client applications to connect and authenticate to the service.

The Custom Vision portal

Azure AI Custom Vision provides a **web-based portal**, in which you can train, publish, and test custom vision models.



You can sign into the Custom Vision portal using your Azure credentials and use it to create image classification or object detection projects that use Azure AI Custom Vision resources in your Azure subscription.

Each project has a unique **project ID**; which is *used by client applications to perform training or prediction tasks using code*.

Custom Vision SDKs

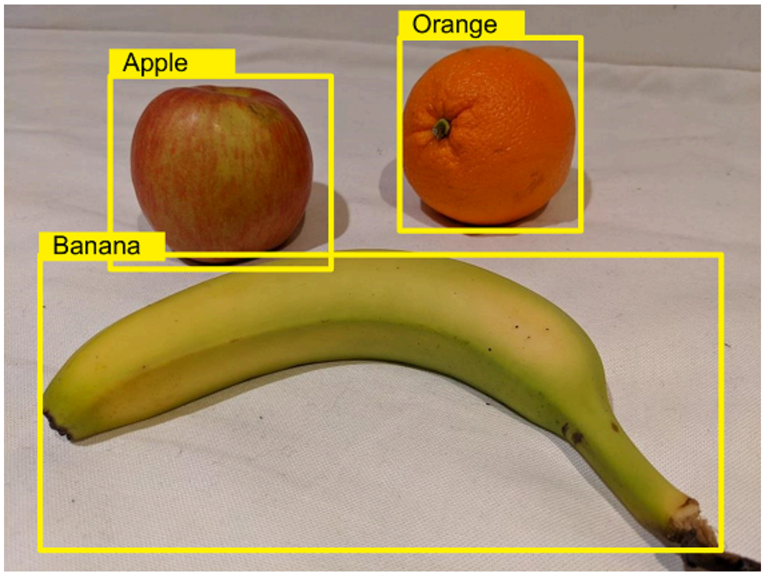
You can write code to train and consume custom models by using the Azure AI Custom Vision language-specific SDKs.

For example, Microsoft C# developers can use the Microsoft.Azure.CognitiveServices.Vision.CustomVision.Training and Microsoft.Azure.CognitiveServices.Vision.CustomVision.Prediction Microsoft .NET packages for training and prediction respectively.

Python developers can perform both training and prediction tasks by using the azure-cognitiveservices-vision-customvision package.

Train an object detector

Object detection is a form of computer vision in which a model is trained to **detect the presence and location of one or more classes of object in an image**.



There are two components to an object detection prediction:

- The **class label** of each object detected in the image. For example, you might ascertain that an image contains an apple, an orange, and a banana.
- The **location** of each object within the image, indicated as coordinates of a bounding box that encloses the object.

To train an object detection model, you can use the **Azure AI Custom Vision portal** to upload and label images before training, evaluating, testing, and publishing the model; or you can use the **REST API or a language-specific SDK** to write code that performs the training tasks.

Image labeling

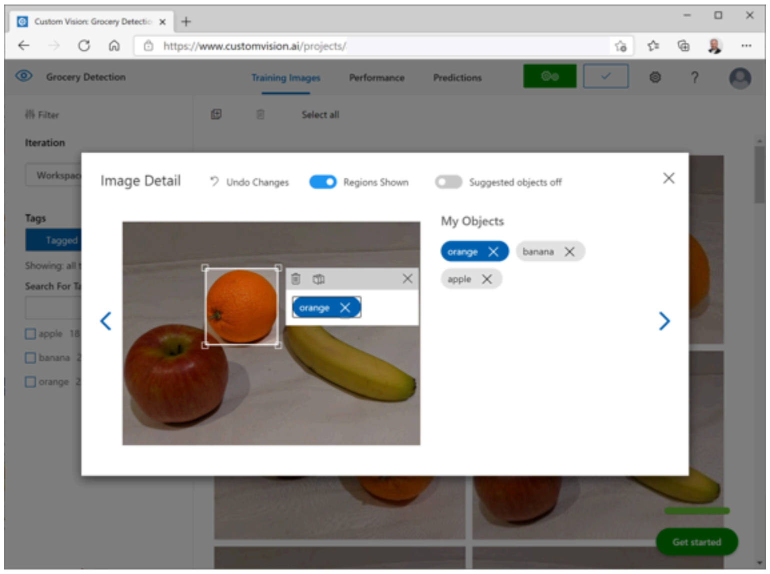
You can use Azure AI Custom Vision to create projects for *image classification* or *object detection*.

The most significant **difference between training an image classification model and training an object detection model** is the **labeling of the images with tags**. While **image classification** requires **one or more tags** that apply to the whole image, **object detection** requires that each label consists of ***a tag and a region that defines the bounding box*** for each object in an image.

Functions	Requirements
Image classification	One or more tags
Object detection	A tag and a region that defines the bounding box.

Labeling images in the Azure AI Custom Vision portal

The Azure AI Custom Vision portal provides a graphical interface that you can use to label your training images.



The easiest option for labeling images for object detection is to use the **interactive interface in the Azure AI Custom Vision portal**. This interface automatically suggests regions that contain objects, to which you can assign tags or adjust by dragging the bounding box to enclose the object you want to label.

Additionally, after tagging an initial batch of images, you can train the model. Subsequent labeling of new images can benefit from the **smart labeler tool** in the portal, which can suggest not only the regions, but the classes of object they contain.

Alternative labeling approaches

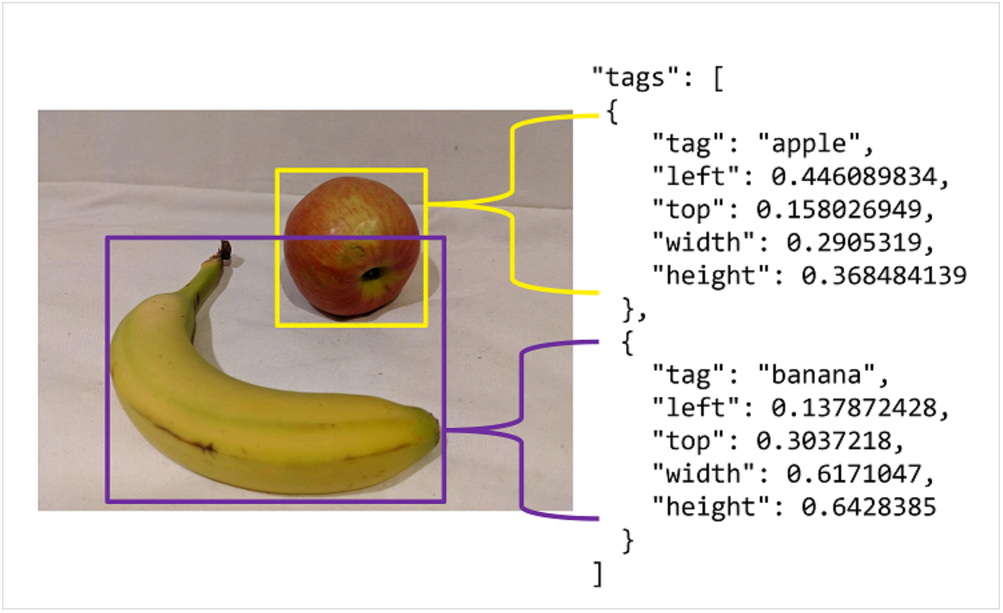
Alternatively, you can use **a custom or third-party labeling tool**, or choose to **label images manually**, to take advantage of other features, such as assigning image labeling tasks to multiple team members.

If you choose to use a labeling tool other than the Azure AI Custom Vision portal, you may need to adjust the output to match the measurement units expected by the Azure AI Custom Vision API. **Bounding boxes** are defined by **four values that represent the left (X) and top (Y) coordinates of the top-left corner of the bounding box**, and **the width and height of the bounding box**. These values are expressed as **proportional values relative to the source image size**. For example, consider this bounding box:

- Left: 0.1
- Top: 0.5
- Width: 0.5
- Height: 0.25

This defines a box in which the left is located 0.1 (one tenth) from the left edge of the image, and the top is 0.5 (half the image height) from the top. The box is half the width and a quarter of the height of the overall image.

The following image shows labeling information in JSON format for objects in an image.



```
"tags": [
 {
    "tag": "apple",
    "left": 0.446089834,
    "top": 0.158026949,
    "width": 0.2905319,
    "height": 0.368484139
  },
  {
    "tag": "banana",
    "left": 0.137872428,
    "top": 0.3037218,
    "width": 0.6171047,
    "height": 0.6428385
  }
]
```

Develop an object detection client application

After you've trained an object detection model, you can use the **Azure AI Custom Vision SDK to develop a client application** that submits new images to be analyzed.

```
from msrest.authentication import ApiKeyCredentials
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredictionClient

# Authenticate a client for the prediction API
credentials = ApiKeyCredentials(in_headers={"Prediction-key": "<YOUR_PREDICTION_RESOURCE_KEY>"}) prediction_client = CustomVisionPredictionClient(e

# Get classification predictions for an image
image_data = open("<PATH_TO_IMAGE_FILE>"), "rb").read() results = prediction_client.detect_image("<YOUR_PROJECT_ID>",
# Process predictions
for prediction in results.predictions:
if prediction.probability > 0.5:
left = prediction.bounding_box.left
top = prediction.bounding_box.top
height = prediction.bounding_box.height
width =  prediction.bounding_box.width
print(f"{prediction.tag_name} ({prediction.probability})")
print(f"  Left:{left}, Top:{top}, Height:{height}, Width:{width}")
```

Exercise - Detect objects in images

If you have an Azure subscription, you can use Azure AI Custom Vision to create a custom object detection model for yourself.

In this exercise, you'll train an object detection model and test it from a client application.

Detect objects in images

The **Azure AI Custom Vision** service enables you to create computer vision models that are trained on your own images. You can use it to train image classification and object detection models; which you can then publish and consume from applications.

In this exercise, you will use the Custom Vision service to train an object detection model that can detect and locate three classes of fruit (apple, banana, and orange) in an image.

Module assessment

1. What does an object detection model predict? **The location and class of specific classes of object in an image**.
2. What must you do before taking advantage of the smart labeler tool when creating an object detection model? **Tag some images with objects of each class and train an initial object detection model**.

Summary

In this module, you have learned how to use the Azure AI Custom Vision service to create object detection models.

> Tip: To find out more about the Azure AI Custom Vision service, see the Azure AI Custom Vision documentation.

Analyze video

Azure Video Indexer is a service to extract insights from video, including **face identification, text recognition, object labels, scene segmentations**, and more.

Learning objectives

After completing this module, you'll be able to:

- Describe Azure Video Indexer capabilities
- Extract custom insights
- Use Azure Video Indexer widgets and APIs

Introduction

It's increasingly common for organizations and individuals to generate content in video format. For example, you might use a cellphone to capture a live event, or you might record a teleconference that combines webcam footage and presentation of slides or documents. As a result, a great deal of information is encapsulated in video files, and you may need to extract this information for analysis or to support indexing for searchability.

In this module, you will learn how to use the **Azure Video Indexer service** to analyze videos.

After completing this module, you'll be able to:

- Describe Azure Video Indexer capabilities.
- Extract custom insights.
- Use Azure Video Indexer widgets and APIs.

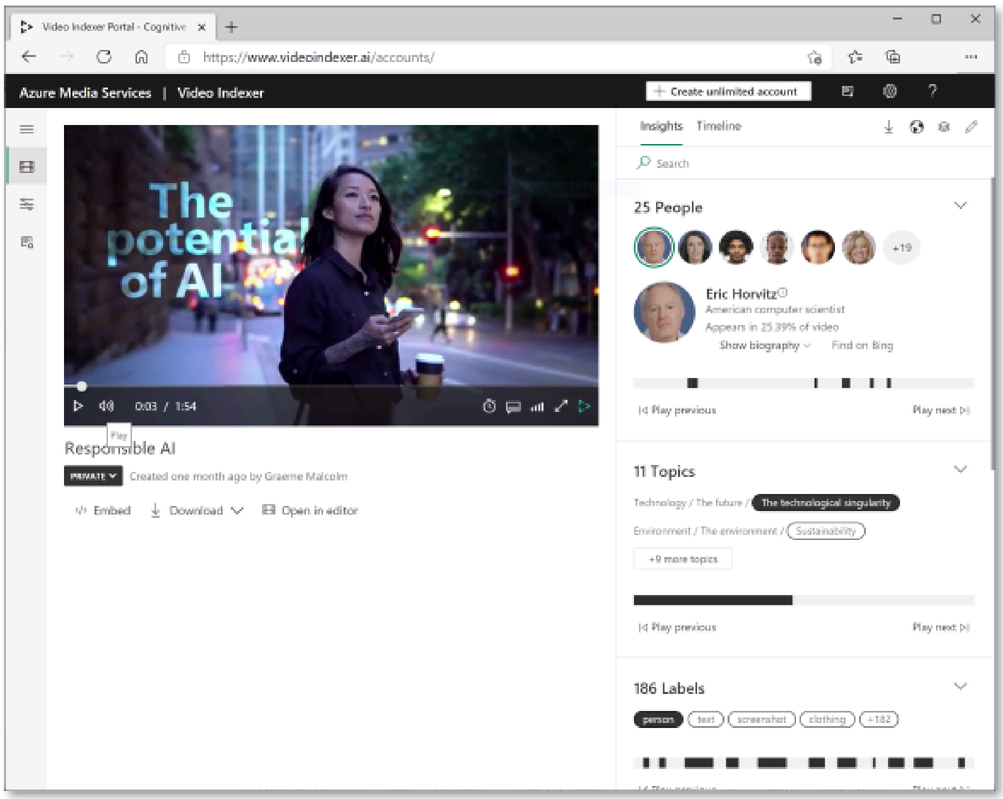
Understand Azure Video Indexer capabilities

The **Azure Video Indexer service** is designed to help you extract information from videos. It provides functionality that you can use for:

- **Facial recognition** - detecting the presence of individual people in the image. This requires Limited Access approval.
- **Optical character recognition** - reading text in the video.
- **Speech transcription** - creating a text transcript of spoken dialog in the video.
- **Topics** - identification of key topics discussed in the video.
- **Sentiment** - analysis of how positive or negative segments within the video are.
- **Labels** - label tags that identify key objects or themes throughout the video.

- **Content moderation** - detection of adult or violent themes in the video.
- **Scene segmentation** - a breakdown of the video into its constituent scenes.

The Video Analyzer service provides a portal website that you can use to upload, view, and analyze videos interactively.


Video Indexer Portal - Cognitive
https://www.videoindexer.ai/accounts/
Azure Media Services | Video Indexer
Create unlimited account
Insights
Timeline
Search
25 People
Eric Horvitz
American computer scientist
Appears in 25.39% of video
Show biography
Find on Bing
Play previous
Play next
11 Topics
Technology / The future / The technological singularity
Environment / The environment / Sustainability
+9 more topics
186 Labels
person
text
screenshot
clothing
+182
The potential of AI
0:03 / 1:54
Responsible AI
PRIVATE
Created one month ago by Graeme Malcolm
Embed
Download
Open in editor

Extract custom insights

Azure Video Indexer includes predefined models that can **recognize well-known celebrities, do OCR, and transcribe spoken phrases into text**. You can extend the recognition capabilities of Video Analyzer by creating custom models for:

- **People**. Add images of the faces of people you want to recognize in videos, and train a model. Video Indexer will then recognize these people in all of your videos.

> Note: This only works after Limited Access approval, adhering to our Responsible AI standard.

- **Language**. If your organization uses specific terminology that may not be in common usage, you can train a custom model to detect and transcribe it.
- **Brands**. You can train a model to recognize specific names as brands, for example to identify products, projects, or companies that are relevant to your business.

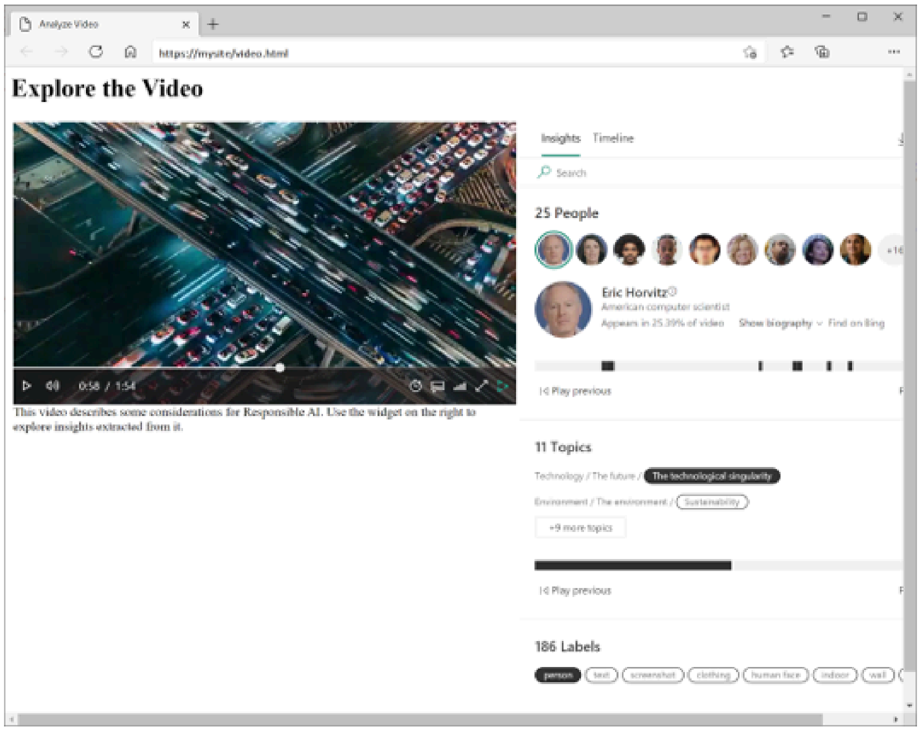
> Note: People only works after Limited Access approval, adhering to our Responsible AI standard.

Use Video Analyzer widgets and APIs

While you can perform all video analysis tasks in the Azure Video Indexer portal, you may want to incorporate the service into custom applications. There are two ways you can accomplish this.

Azure Video Indexer widgets

The **widgets** used in the **Azure Video Indexer portal** to play, analyze, and edit videos can be embedded in your own custom HTML interfaces. You can use this technique to share insights from specific videos with others without giving them full access to your account in the Azure Video Indexer portal.



Explore the Video

This video describes some considerations for Responsible AI. Use the widget on the right to explore insights extracted from it.

Azure Video Indexer API

Azure Video Indexer provides a **REST API** that you can use to obtain information about your account, including an **access token**.

```
https://api.videoindexer.ai/Auth/<location>/Accounts/<accountId>/AccessToken
```

You can then use your token to consume the REST API and automate video indexing tasks, creating projects, retrieving insights, and creating or deleting custom models.

For example, a GET call to `https://api.videoindexer.ai/<location>/Accounts/<accountId>/Customization/CustomLogos/Logos/<logoId>?<accessToken>` REST endpoint returns the specified logo.

In another example, you can send a GET request to `https://api.videoindexer.ai/<location>/Accounts/<accountId>/Videos?<accessToken>` which returns details of videos in your account, similar to the following JSON example:

Deploy with ARM template

Azure Resource Manager (ARM) templates are available to create the Azure AI Video Indexer resource in your subscription, based on the parameters specified in the template file.

For a full list of available APIs, see the Video Indexer Developer Portal.

Exercise - Analyze video

Now it's your turn to try using the Azure AI Video Indexer service.

In this exercise, you analyze a video using the Azure AI Video Indexer portal and its API.

Analyze video

A large proportion of the data created and consumed today is in the format of video. **Azure AI Video Indexer** is an AI-powered service that you can use to index videos and extract insights from them.

Module assessment

1. You want Azure Video Indexer to analyze a video. What must you do first? **Upload the video to Azure Video Indexer and index it**.
2. You want Azure Video Indexer to recognize brands in videos recorded from conference calls. What should you do? **Edit the Brands model to show brands suggested by Bing, and add any new brands you want to detect**.

Summary

In this module, you learned how to use the **Azure Video Indexer** service to analyze videos.

Now that you've completed this module, you can:

- Describe Azure Video Indexer capabilities.
- Extract custom insights.
- Use **Azure Video Indexer widgets** and **APIs**.

To find out more about the Azure Video Indexer service, see the Azure Video Indexer documentation.

Develop a vision-enabled generative AI application

A picture says a thousand words, and **multimodal generative AI models** can interpret images to respond to visual prompts. Learn how to build **vision-enabled chat apps**.

Learning objectives

After completing this module, you'll be able to:

- Deploy a **vision-enabled generative AI model** in Azure AI Foundry.
- Test an **image-based prompt** in the chat playground.
- Create a chat app that submits **image-based prompts**.

Introduction

Generative AI models enable you to develop chat-based applications that reason over and respond to input. Often this input takes the form of a text-based prompt, but increasingly **multimodal models that can respond to visual input** are becoming available.

In this module, we'll discuss **vision-enabled generative AI** and explore how you can use Azure AI Foundry to create generative AI solutions that respond to prompts that include **a mix of text and image data**.

Deploy a multimodal model

To handle prompts that include images, you need to deploy a **multimodal generative AI model** - in other words, a model that supports **not only text-based input, but image-based (and in some cases, audio-based) input** as well. Multimodal models available in Azure AI Foundry include (among others):

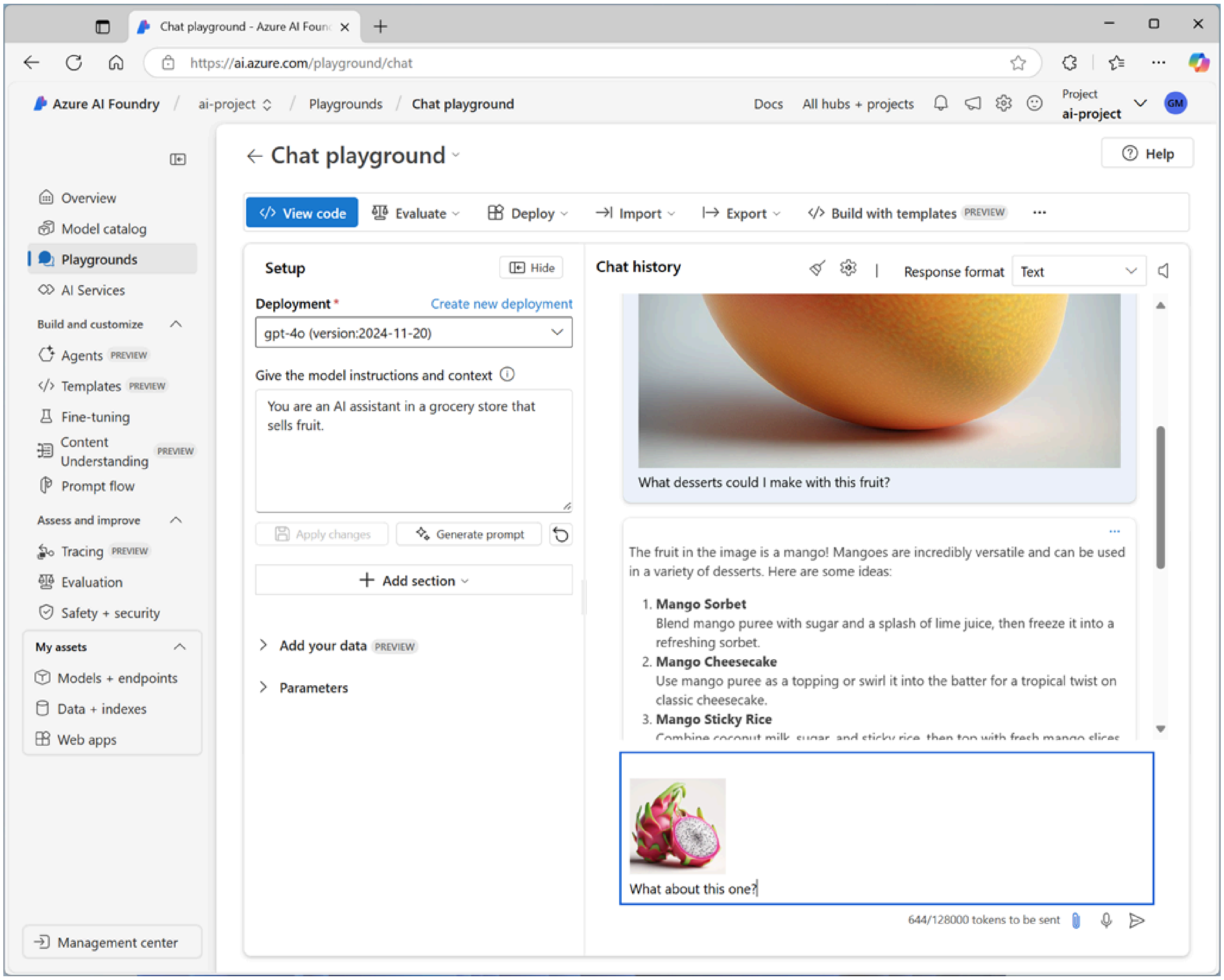
- Microsoft Phi-4-multimodal-instruct

- OpenAI gpt-4o
- OpenAI gpt-4o-mini

> Tip: To learn more about available models in Azure AI Foundry, see the Model catalog and collections in Azure AI Foundry portal article in the Azure AI Foundry documentation.

Testing multimodal models with image-based prompts

After deploying a multimodal model, you can test it in the chat playground in Azure AI Foundry portal.



Deployment *

gpt-4o (version:2024-11-20)

Give the model instructions and context

You are an AI assistant in a grocery store that sells fruit.

Response format Text

What desserts could I make with this fruit?

The fruit in the image is a mango! Mangoes are incredibly versatile and can be used in a variety of desserts. Here are some ideas:

1. **Mango Sorbet**
   Blend mango puree with sugar and a splash of lime juice, then freeze it into a refreshing sorbet.
2. **Mango Cheesecake**
   Use mango puree as a topping or swirl it into the batter for a tropical twist on classic cheesecake.
3. **Mango Sticky Rice**
   Combine coconut milk, sugar, and sticky rice, then top with fresh mango slices

What about this one?

In the chat playground, you can upload an image from a local file and add text to the message to elicit a response from a multimodal model.

Develop a vision-based chat app

To develop a client app that engages in **vision-based chats with a multimodal model**, you can use the same basic techniques used for text-based chats. You require a connection to the endpoint where the model is deployed, and you use that **endpoint** to submit prompts that consists of messages to the model and process the responses.

The **key difference** is that prompts for *a vision-based chat include multi-part user messages that contain both **a text (or audio where supported) content item** and **an image content item***.

The JSON representation of a prompt that includes a multi-part user message looks something like this:

```
{
    "messages": [
        { "role": "system", "content": "You are a helpful assistant." },
        { "role": "user", "content": [
            {
                "type": "text",
                "text": "Describe this picture:"
            },
            {
                "type": "image_url",
                "image_url": {
                    "url": "https://....."
                }
            }
        ] }
    ]
}
```

The image content item can be:

- A URL to an image file in a web site.
- Binary image data

When using binary data to submit a local image file, the **image_url content** takes the form of a **base64 encoded value** in a data URL format:

```
{
    "type": "image_url",
    "image_url": {
       "url": "data:image/jpeg;base64,<binary_image_data>"
    }
}
```

Depending on the model type, and where you deployed it, you can use **Microsoft Azure AI Model Inference** or **OpenAI APIs** to submit **vision-based prompts**. These libraries also provide language-specific SDKs that abstract the underlying REST APIs.

In the exercise that follows in this module, you can use the Python or .NET SDK for the **Azure AI Model Inference API** and the **OpenAI API** to develop a vision-enabled chat application.

Exercise - Develop a vision-enabled chat app

Develop a vision-enabled chat app

In this exercise, you use the **Phi-4-multimodal-instruct** generative AI model to generate responses to prompts that include images. You'll develop an app that provides AI assistance with fresh produce in a grocery store by using Azure AI Foundry and the Azure AI Model Inference service.

Module assessment

1. Which kind of model can you use to respond to visual input? **Multimodal models**.
2. How can you submit a prompt that asks a model to analyze an image? **Submit a prompt that contains a multi-part user message, containing both text content and image content**.
3. How can you include an image in a message? **As a URL or as binary data**.

Summary

In this module, you learned about **vision-enabled generative AI models** and how to implement chat solutions that include image-based input.

Vision-enabled models let you create AI solutions that can **understand images and respond to related questions or instructions**. Beyond just identifying objects in pictures, some models can also use reasoning based on what they see. For instance, they can interpret a chart or assess if an object is damaged.

> Tip: For more information about working with multimodal models in Azure AI Foundry, see How to use image and audio in chat completions with Azure AI model inference and Quickstart: Use images in your AI chats.

Generate images with AI

In **Azure AI Foundry**, you can use image generation models to create original images based on natural language prompts.

Learning objectives

After completing this module, you'll be able to:

- Describe the capabilities of image generation models
- Use the Images playground in Azure AI Foundry portal
- Integrate image generation models into your apps

Introduction

With Azure AI Foundry, you can use language models to generate content based on natural language prompts. Often the generated content is in the form of natural language text, but increasingly, models can generate other kinds of content.

For example, the OpenAI DALL-E image generation model can create original graphical content based on a description of a desired image.

The ability to **use AI to generate graphics has many applications**; including

- the creation of illustrations or photorealistic images for articles or marketing collateral
- generation of unique product or company logos, or
- any scenario where a desired image can be described.

In this module, you'll learn how to develop an application that uses generative AI to generate original images.

What are image-generation models?

Azure AI Foundry supports multiple models that are capable of generating images, including (but not limited to):

- DALL-E 3
- GPT-Image 1

> Tip: For the latest information about model availability in Azure AI Foundry, view the model catalog. See Model catalog and collections in Azure AI Foundry portal for details.

Image generation models are generative AI model that can create graphical data from natural language input. Put more simply, you can provide the model with a description and it can generate an appropriate image.

For example, you might submit the following natural language prompt to an image generation model:

```
A squirrel on a motorcycle
```

This prompt could result in the generation of graphical output such as the following image.

The images generated are original; they aren't retrieved from a curated image catalog. In other words, the model isn't a search system for finding appropriate images - it is an artificial intelligence (AI) model that generates new images based on the data on which it was trained.

Explore image-generation models in Azure AI Foundry portal

To experiment with image generation models, you can create an Azure AI Foundry project and use the **Images playground** in Azure AI Foundry portal to submit prompts and view the resulting generated images.

When using the playground, you can adjust the **settings** to control the output. For example, when using a DALL-E model you can specify:

- The **resolution (size)** of the generated images. Available sizes are **1024x1024 (which is the default value), 1792x1024, or 1024x1792**.

- The **image style** to be generated (such as vivid or natural).
- The **image quality** (choose from standard or HD).

Create a client application that uses an image generation model

You can use a **REST API** to consume DALL-E models from applications. Alternatively, you can use **a language-specific SDK** (for example, the OpenAI Python SDK or the Azure OpenAI .NET SDK) to abstract the REST methods.

You initiate the image generation process by **submitting a request to the service endpoint with the authorization key in the header**. **The request contains parameters describing the image-generation requirements**. For example, parameters for a DALL-E model include:

- **prompt**: The description of the image to be generated.
- **n**: The number of images to be generated. DALL-E 3 only supports n=1.
- **size**: The resolution of the image(s) to be generated (1024x1024, 1792x1024, or 1024x1792 for DALL-E 3)
- **quality** Optional: The quality of the image (standard or hd). Defaults to standard.
- **style** Optional: The visual style of the image (natural or vivid). Defaults to vivid.

For example, the following JSON could be submitted via the REST API to a DALL-E model, prompting it to generate an 1024 x 1024 image of a badger wearing a tuxedo:

```
{     "prompt": "A badger wearing a tuxedo",     "n": 1,     "size": "1024x1024",     "quality": "hd",     "style": "vivid" }
```

With DALL-E 3, the result from the request is **processed synchronously** with the response containing the URL for the generated image. The response is similar to the following JSON:

```
{     "created": 1686780744,     "data": [         {             "url": "<URL of generated image>",             "revised_prompt": "<prompt that was
```

The **data** element includes the url value, which references **a PNG image file** generated from the prompt that you can then view or download. The response also contains a revised prompt that was used to generate the image, which was updated by the system to achieve the most desirable results. In this example, the image might look similar to the following image.

Exercise - Generate images with AI

Now it's your chance to use generative AI to create images. In this exercise, you'll provision an Azure AI Foundry project and deploy a **DALL-E model**. Then, you'll explore image generation in the Azure AI Foundry portal. Finally, you'll use the Python or .NET SDK to consume the DALL-E model from a custom application.

In this exercise, you use the the OpenAI DALL-E generative AI model to generate images. You'll develop your app by using Azure AI Foundry and the Azure OpenAI service.

Generate images with AI

In this exercise, you use the **OpenAI DALL-E** generative AI model to generate images. You also use the **OpenAI Python SDK** to create a simple app to generate images based on your prompts.

Module assessment

1. You want to use a model in Azure AI Foundry to generate images. Which model should you use? **DALL-E**
2. Which playground in Azure AI Foundry portal should you use to test an image-generation model? **Images**
3. In a REST request to generate images, what does the n parameter indicate? **The number of images to be generated**.

Summary

This module described **image generation models**, and how you can use them in Azure AI Foundry to generate images based on natural language prompts. You can explore image generation models using the **Images playground** in Azure AI Foundry portal, and you can use **REST APIs or SDKs** to build applications that generate new images.

> Tip: To learn more about using DALL-E in the Azure OpenAI service, see Quickstart: Generate images with Azure OpenAI Service in the Azure OpenAI service documentation.