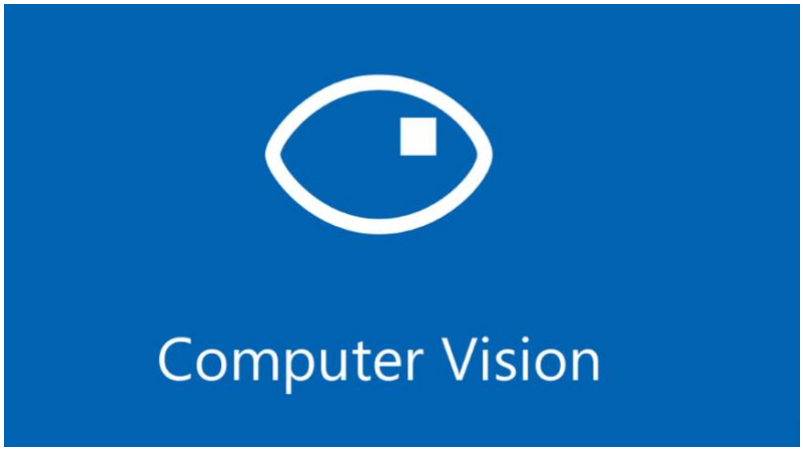
ITIS 6177 - System Integration
Final Project - Computer Vision API

By Mounika Gottumukkala
801276287



Computer Vision API documentation

Introduction to Computer Vision using Azure

What is Computer Vision?

Computer Vision is an AI Service part of the Azure Cognitive Services that analyzes content in images and video. It extracts rich information from images to categorize and process visual data and protect your users from unwanted content with this Azure Cognitive Service.

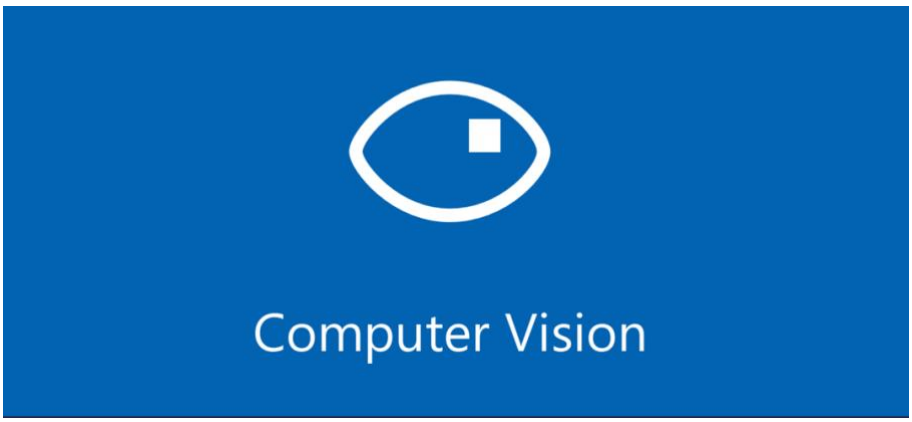
Computer vision is one of the core areas of artificial intelligence (AI) and focuses on creating solutions that enable AI applications to "see" the world and make sense of it.

Of course, computers don't have biological eyes that work the way ours do, but they are capable of processing images; either from a live camera feed or from digital photographs or videos. This ability to process images is the key to creating software that can emulate human visual perception.

Some potential uses for computer vision include:

- **Content Organization**: Identify people or objects in photos and organize them based on that identification. Photo recognition applications like this are commonly used in photo storage and social media applications.
- **Text Extraction**: Analyze images and PDF documents that contain text and extract the text into a structured format.
- **Spatial Analysis**: Identify people or objects, such as cars, in a space and map their movement within that space.

The cloud-based Computer Vision API provides developers with access to advanced algorithms for processing images and returning information. By uploading an image or specifying an image URL, Microsoft Computer Vision algorithms can analyze visual content in different ways based on inputs and user choices. Learn how to analyze visual content in different ways with QuickStart’s, tutorials, and samples.



Computer Vision for digital asset management

Computer Vision can power many digital asset management (DAM) scenarios. DAM is the business process of organizing, storing, and retrieving rich media assets and managing digital rights and permissions. For example, a company may want to group and identify images based on visible logos, faces, objects, colors, and so on. Or, you might want to automatically generate captions for images and attach keywords so they're searchable. For an all-in-one DAM solution using Cognitive Services, Azure Cognitive Search, and intelligent reporting.

There are some services for processing images and returning information.

Optical Character Recognition (OCR)

The Optical Character Recognition (OCR) service extracts text from images. You can use the new Read API to extract printed and handwritten text from photos and documents. It uses deep-learning-based models and works with text on a variety of surfaces and backgrounds. These include business documents, invoices, receipts, posters, business cards, letters, and whiteboards. The OCR APIs support extracting printed text in several languages.

Image Analysis

The Image Analysis service extracts many visual features from images, such as objects, faces, adult content, and auto-generated text descriptions.

Face

The Face service provides AI algorithms that detect, recognize, and analyze human faces in images. Facial recognition software is important in many different scenarios, such as identity verification, touchless access control, and face blurring for privacy.

Spatial Analysis

The Spatial Analysis service analyzes the presence and movement of people on a video feed and produces events that other systems can respond to. Install the Spatial Analysis container to get started.

Image requirements

Computer Vision can analyze images that meet the following requirements:

1. The image must be presented in JPEG, PNG, GIF, or BMP format
2. The file size of the image must be less than 4 megabytes (MB)
3. The dimensions of the image must be greater than 50 x 50 pixels
4. For the Read API, the dimensions of the image must be between 50 x 50 and 10000 x 10000 pixels.

Data privacy and security

As with all the Cognitive Services, developers using the Computer Vision service should be aware of Microsoft's policies on customer data.

Let’s see on to get started building Computer Vision into your app.

Computer vision comes under artificial intelligence and gives the power to the system to recognize and see the world to make some logical sense of it. To any particular system, an image or video is just an array of pixels with some values that define their colors. These numeric values can be used as features by the machine learning models to process, analyze them to give some meaningful relations, and logic, and make predictions about the image and the content it contains. This helps in many several cases listed as cases:

1. **Organization of content**: This helps in identifying people, and objects in the photos or videos and organizing them or giving relevant information. This is used in many social media websites that help tag a particular person.
2. **Extracting Text**: This helps in analyzing the text in different formats (handwritten, PDF) and extracting the relevant text in a structured form. This is used in the form, of bill processing.
3. **Analyzing given space**: This helps in analyzing a particular space and gives information about the people, objects in that space, and their movements with time.

Microsoft Azure – Computer Vision Service

Microsoft Azure Computer Vision Service is a cognitive service that helps analyze images and get detailed information about them without much effort because of the pre-trained computer vision capabilities that it offers. This helps create models that can analyze images and give relevant information in seconds.

Resources Required to Use the Service

To use this service, we need to create some resources in Microsoft Azure. We can use either of the following:

A. Computer Vision: It is a separate resource available in Microsoft Azure and specifically only for computer vision-related tasks. We should use this resource only when we don’t wish to use any other cognitive services provided by Azure with our current service in our product. We should also use this service whenever we wish to track the cost and utilization of our computer vision tasks separately.

B. Cognitive Service: It is a general service that includes all the cognitive services provided by Microsoft Azure to help us carry out our tasks. It includes computer vision, translator text, text analytics, form processing, and many others. Whenever we create a resource (any one of the above mentioned) it will come with two pieces of information that will allow us to use the resource we created.

1. **Key**: It will be used to authenticate the client applications and requests.
2. **Endpoint**: It will provide us with an HTTP address using which we can access our resources

What is Image Analysis

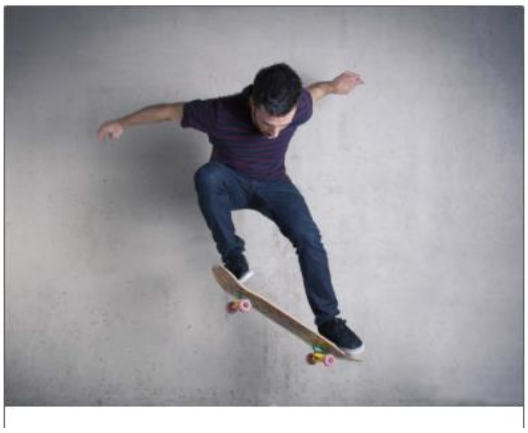
The Computer Vision Image Analysis service can extract a wide variety of visual features from your images. For example, it can determine whether an image contains adult content, find specific brands or objects, or find human faces.
The latest version of Image Analysis, 4.0, which is now in public preview, has new features like synchronous OCR and people detection. We recommend you use this version going forward.
You can use Image Analysis through a client library SDK or by calling the REST API directly.

Image Analysis features
You can analyze images to provide insights about their visual features and characteristics. All of the features in the list below are provided by the Analyze Image API.

Extract text from images (preview)
Version 4.0 preview of Image Analysis offers the ability to extract text from images. Compared with the async Computer Vision 3.2 GA Read, the new version offers the familiar Read OCR engine in a unified performance-enhanced synchronous API that makes it easy to get all image insights including OCR in a single API operation.

Detect people in images (preview)
Version 4.0 preview of Image Analysis offers the ability to detect people appearing in images. The bounding box coordinates of each detected person are returned, along with a confidence score.

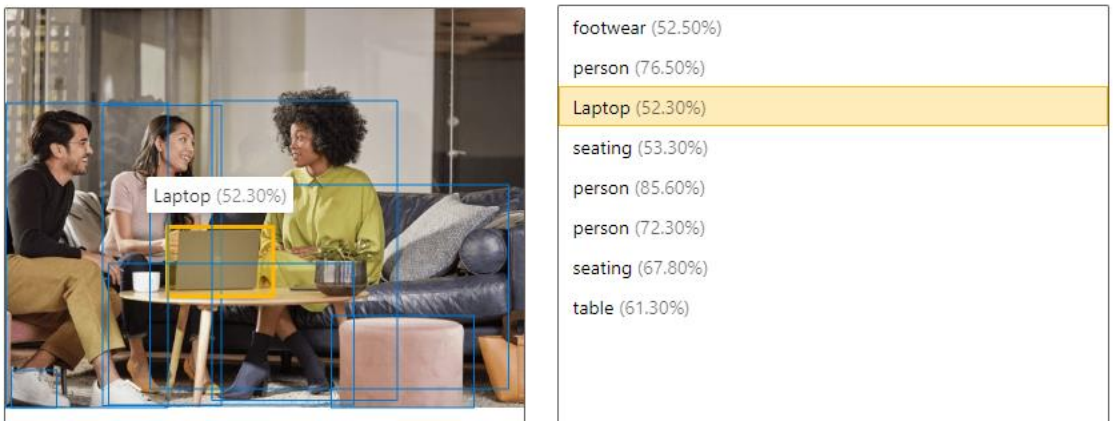
Tag visual features
Identify and tag visual features in an image, from a set of thousands of recognizable objects, living things, scenery, and actions. When the tags are ambiguous or not common knowledge, the API response provides hints to clarify the context of the tag. Tagging isn't limited to the main subject, such as a person in the foreground, but also includes the setting (indoor or outdoor), furniture, tools, plants, animals, accessories, gadgets, and so on



sport (99.60%)
person (99.56%)
footwear (98.05%)
skating (96.27%)
boardsport (95.58%)
skateboarding equipment (94.43%)
clothing (94.02%)
wall (93.81%)
skateboarding (93.78%)
skateboarder (93.25%)
individual sports (92.80%)
street stunts (90.81%)
balance (90.81%)

Detect objects
Object detection is like tagging, but the API returns the bounding box coordinates for each tag applied. For example, if an image contains a dog, cat, and person, the Detect operation will list those objects together with their coordinates in the image. You can use this functionality to process

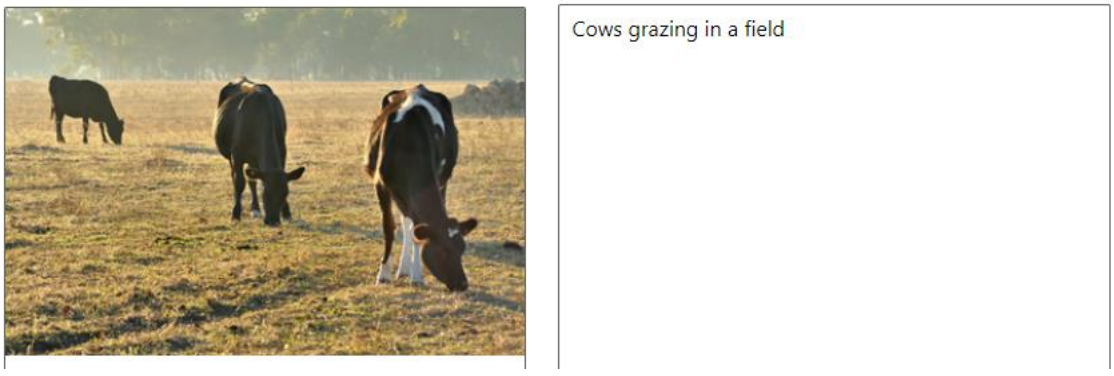
further relationships between the objects in an image. It also lets you know when there are multiple instances of the same tag in an image.



Detect brands
Identify commercial brands in images or videos from a database of thousands of global logos. You can use this feature, for example, to discover which brands are most popular on social media or most prevalent in media product placement.

Categorize an image
Identify and categorize an entire image, using a category taxonomy with parent/child hereditary hierarchies. Categories can be used alone, or with our new tagging models.Currently, English is the only supported language for tagging and categorizing images.

Describe an image
Generate a description of an entire image in human-readable language, using complete sentences. Computer Vision's algorithms generate various descriptions based on the objects identified in the image. The descriptions are each evaluated, and a confidence score generated. A list is then returned ordered from highest confidence score to lowest.



Detect faces
Detect faces in an image and provide information about each detected face. Computer Vision returns the coordinates, rectangle, gender, and age for each detected face.

You can also use the dedicated Face API for these purposes. It provides more detailed analysis, such as facial identification and pose detection.

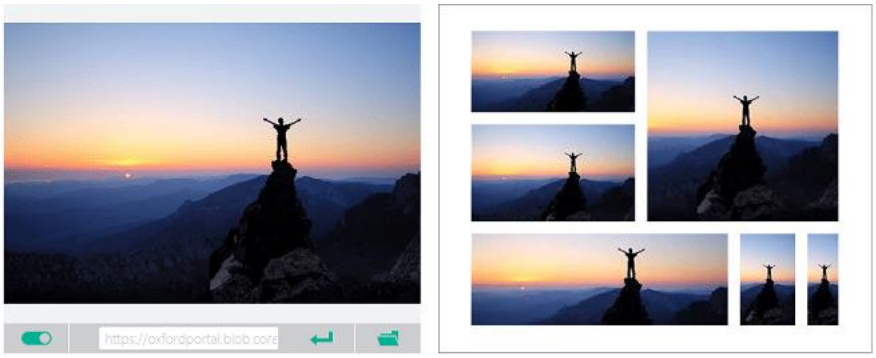
Detect image types
Detect characteristics about an image, such as whether an image is a line drawing or the likelihood of whether an image is clip art.

Detect domain-specific content
Use domain models to detect and identify domain-specific content in an image, such as celebrities and landmarks. For example, if an image contains people, Computer Vision can use a domain model for celebrities to determine if the people detected in the image are known celebrities.

Detect the color scheme
Analyze color usage within an image. Computer Vision can determine whether an image is black & white or color and, for color images, identify the dominant and accent colors.

Get the area of interest / smart crop

Analyze the contents of an image to return the coordinates of the area of interest that matches a specified aspect ratio. Computer Vision returns the bounding box coordinates of the region, so the calling application can modify the original image as desired.



Moderate content in images
You can use Computer Vision to detect adult content in an image and return confidence scores for different classifications. The threshold for flagging content can be set on a sliding scale to accommodate your preferences.

QuickStart: Image Analysis

Get started with Image Analysis by following the steps in your preferred development language. I chose 'JAVA' language.

The Analyze Image service provides you with AI algorithms for processing images and returning information on their visual features. Use the Image Analysis client library to analyze a remote image for tags, text description, faces, adult content, and more. Sample code has been added to the Github.

Prerequisites

- An Azure subscription - Create one for free
- The current version of the Java Development Kit (JDK)
- The Gradle build tool, or another dependency manager.
- Once you have your Azure subscription, create a Computer Vision resource in the Azure portal to get your key and endpoint. After it deploys, click **Go to resource**.
- You will need the key and endpoint from the resource you create to connect your application to the Computer Vision service. You'll paste your key and endpoint into the code below later in the quickstart.
- You can use the free pricing tier (F0) to try the service, and upgrade later to a paid tier for production.

Analyze image: Follow these steps and check attached screenshots

1. Create a new Gradle project.
2. In a console window (such as cmd, PowerShell, or Bash), create a new directory for your app, and navigate to it.

```
mkdir myapp
cd myapp
```

3. Run the gradle init command from your working directory. This command will create essential build files for Gradle, including build.gradle.kts, which is used at runtime to create and configure your application.

```
gradle init --type basic
```

4. When prompted to choose a **DSL**, select **Kotlin**.
5. Install the client library.
6. This quickstart uses the Gradle dependency manager. You can find the client library and information for other dependency managers on the Maven Central Repository.

7. Locate build.gradle.kts and open it with your preferred IDE or text editor.
8. Then copy in the following build configuration.
9. This configuration defines the project as a Java application whose entry point is the class **ImageAnalysisQuickstart**. It imports the Computer Vision library.

```
plugins {
    java
    application
}
application {
    mainClass.set("ImageAnalysisQuickstart")
}
repositories {
    mavenCentral()
}
dependencies {
    implementation(group = "com.microsoft.azure.cognitiveservices", name =
"azure-cognitiveservices-computervision", version = "1.0.9-beta")
}
```

- Then, Create a Java file.
- From your working directory, run the following command to create a project source folder:

mkdir -p src/main/java

- Navigate to the new folder and create a file called ImageAnalysisQuickstart.java.
- Find the key and endpoint.
- Go to the Azure portal. If the Computer Vision resource you created in the **Prerequisites** section deployed successfully, click the **Go to Resource** button under **Next Steps**. You can find your key and endpoint in the resource's **key and endpoint** page, under **resource management**.

Open ImageAnalysisQuickstart.java in your preferred editor or IDE and paste in the following code.

```
import com.microsoft.azure.cognitiveservices.vision.computervision.*;
import
com.microsoft.azure.cognitiveservices.vision.computervision.implementation.Com
puterVisionImpl;
import com.microsoft.azure.cognitiveservices.vision.computervision.models.*;

import java.io.*;
import java.nio.file.Files;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

public class ImageAnalysisQuickstart {

   static String subscriptionKey =
"PASTE_YOUR_COMPUTER_VISION_SUBSCRIPTION_KEY_HERE";
   static String endpoint =
"PASTE_YOUR_COMPUTER_VISION_ENDPOINT_HERE";

   public static void main(String[] args) {

      System.out.println("\nAzure Cognitive Services Computer Vision - Java
Quickstart Sample");

      // Create an authenticated Computer Vision client.
      ComputerVisionClient compVisClient = Authenticate(subscriptionKey,
endpoint);

      // Analyze local and remote images
      AnalyzeRemoteImage(compVisClient);

   }

   public static ComputerVisionClient Authenticate(String subscriptionKey, String
endpoint){
      return
ComputerVisionManager.authenticate(subscriptionKey).withEndpoint(endpoint);
   }


   public static void AnalyzeRemoteImage(ComputerVisionClient compVisClient)
{
```

```
        /*
         * Analyze an image from a URL:
         *
         * Set a string variable equal to the path of a remote image.
         */
        String pathToRemoteImage = "https://github.com/Azure-Samples/cognitive-services-sample-data-files/raw/master/ComputerVision/Images/faces.jpg";

        // This list defines the features to be extracted from the image.
        List<VisualFeatureTypes> featuresToExtractFromRemoteImage = new ArrayList<>();
        featuresToExtractFromRemoteImage.add(VisualFeatureTypes.TAGS);

        System.out.println("\n\nAnalyzing an image from a URL ...");

        try {
            // Call the Computer Vision service and tell it to analyze the loaded image.
            ImageAnalysis analysis = compVisClient.computerVision().analyzeImage().withUrl(pathToRemoteImage)
                    .withVisualFeatures(featuresToExtractFromRemoteImage).execute();


            // Display image tags and confidence values.
            System.out.println("\nTags: ");
            for (ImageTag tag : analysis.tags()) {
                System.out.printf("\'%s\' with confidence %f\n", tag.name(), tag.confidence());
            }
        }

        catch (Exception e) {
            System.out.println(e.getMessage());
            e.printStackTrace();
        }
    }
    // END - Analyze an image from a URL.

}
```

Note: Paste your key and endpoint into the code where indicated. Your Computer Vision endpoint has the form https://<your_computer_vision_resource_name>.cognitiveservices.azure.com/.

Important Note: Remember to remove the key from your code when you're done, and never post it publicly. For production, use a secure way of storing and accessing your credentials like **Azure Key Vault**.

Navigate back to the project root folder, and build the app with:

gradle build

Then, run it with the gradle run command:

gradle run

Output: Analyzing an image from a URL ...

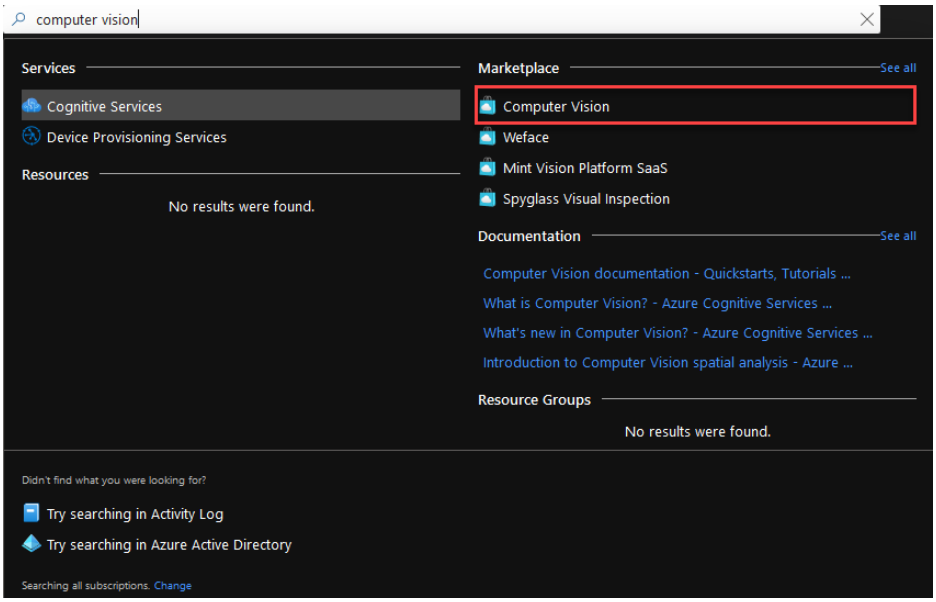
Tags:
'person' with confidence 0.998895
'Human face' with confidence 0.997437
'smile' with confidence 0.991973
'outdoor' with confidence 0.985962
'happy' with confidence 0.969785
'clothing' with confidence 0.961570
'friendship' with confidence 0.946441
'tree' with confidence 0.917331
'female person' with confidence 0.890976
'girl' with confidence 0.888741
'social group' with confidence 0.872044
'posing' with confidence 0.865493
'adolescent' with confidence 0.857371
'love' with confidence 0.852553
'laugh' with confidence 0.850097
'people' with confidence 0.849922
'lady' with confidence 0.844540
'woman' with confidence 0.818172
'group' with confidence 0.792975
'wedding' with confidence 0.615252
'dress' with confidence 0.517169

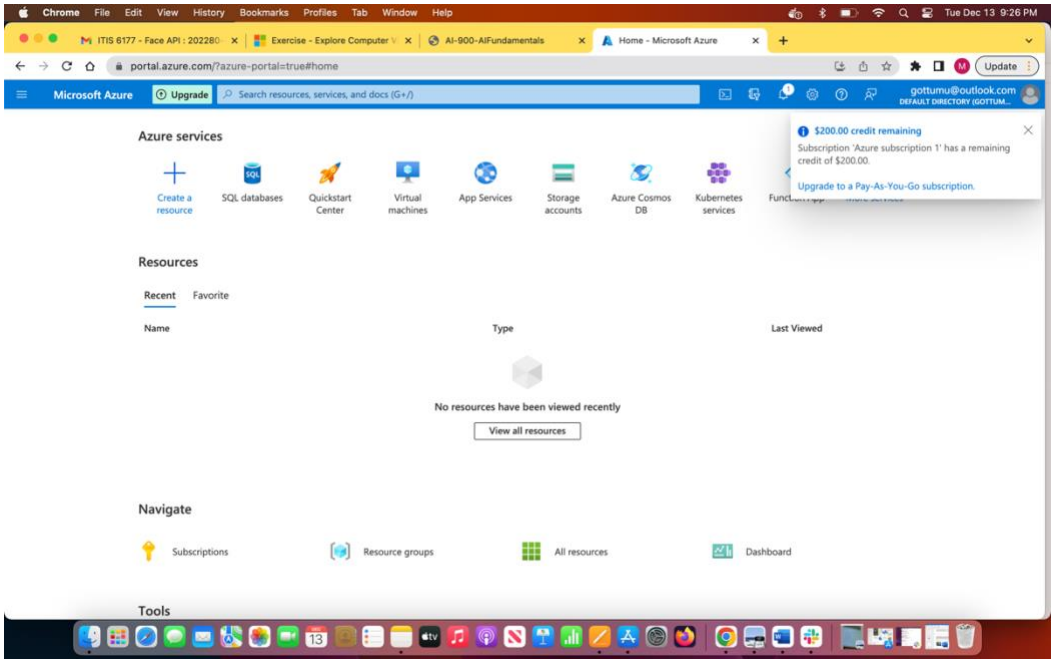
Analyzing images with the Service
The computer vision service has many pre-trained machine learning models to help us analyze images and gain information from them effectively.

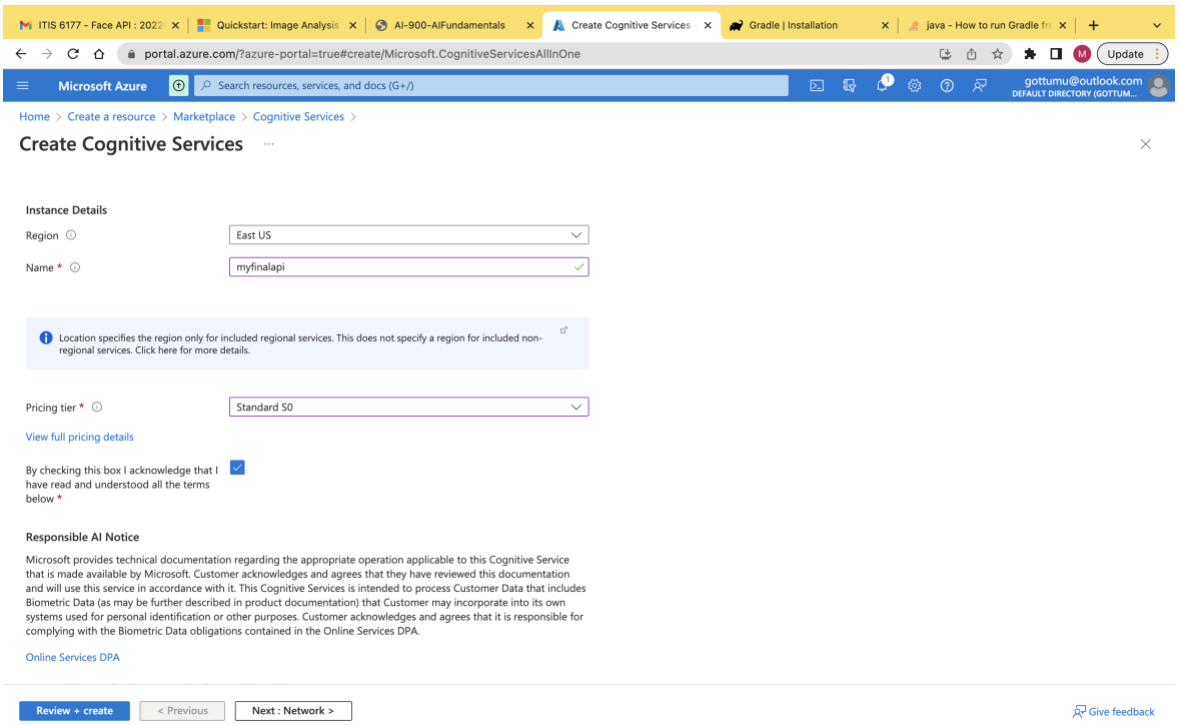
Prerequisites:
You need a Azure subscription for which you can avail free 12-month subscription.

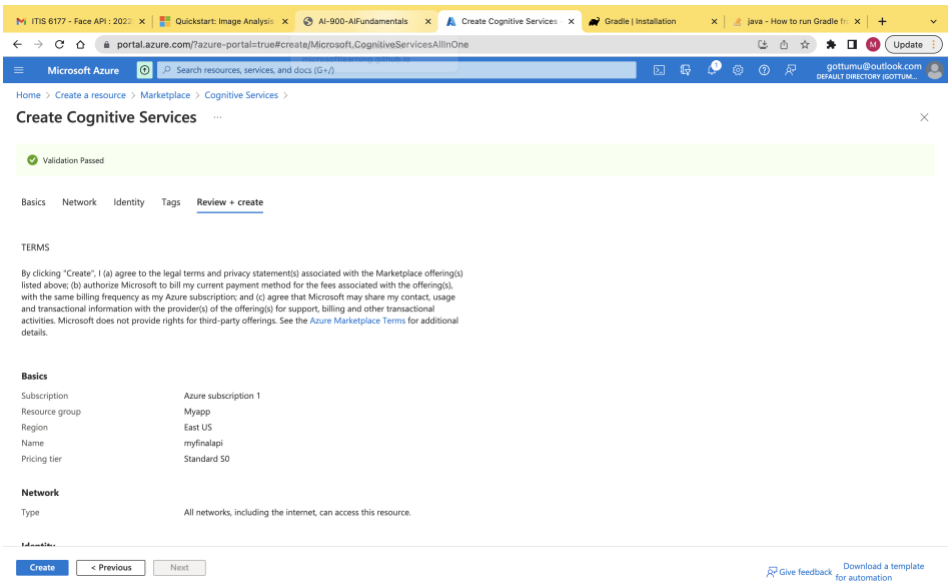
Create

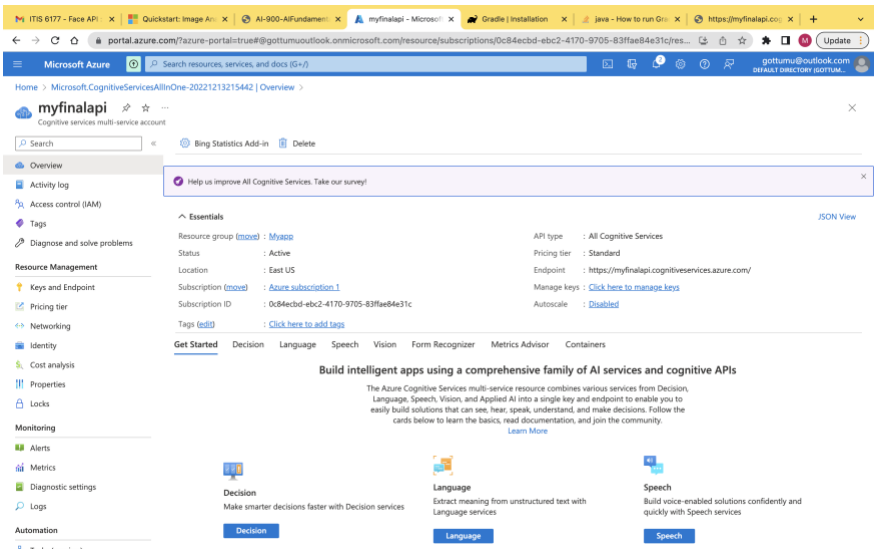
- Go to Azure Portal and search for Computer Vision. Select Computer Vision from the Marketplace.
- Fill the form. The fields with the '*' are mandatory.
- Subscription should already be filled in with your default subscription.
- For Resource Group, you can use an existing one, or create a new one.
- You can leave Region with the pre-selected region.
- Name is the name of your new Computer Vision resource.
- Any Pricing tier will do for this demo.






ITIS 6177 - Face API : 2022
Quickstart: Image Analysis
AI-900-AIFundamentals
Create Cognitive Services
Gradle | Installation
java - How to run Gradle fr
portal.azure.com/?azure-portal=true#create/Microsoft.CognitiveServicesAllInOne
Update
Microsoft Azure
Search resources, services, and docs (G+/)
gottumu@outlook.com
DEFAULT DIRECTORY (GOTTUM...
Home > Create a resource > Marketplace > Cognitive Services >
Create Cognitive Services
Instance Details
Region
East US
Name *
myfinalapi
Location specifies the region only for included regional services. This does not specify a region for included non-regional services. Click here for more details.
Pricing tier *
Standard S0
View full pricing details
By checking this box I acknowledge that I have read and understood all the terms below *
Responsible AI Notice
Microsoft provides technical documentation regarding the appropriate operation applicable to this Cognitive Service that is made available by Microsoft. Customer acknowledges and agrees that they have reviewed this documentation and will use this service in accordance with it. This Cognitive Services is intended to process Customer Data that includes Biometric Data (as may be further described in product documentation) that Customer may incorporate into its own systems used for personal identification or other purposes. Customer acknowledges and agrees that it is responsible for complying with the Biometric Data obligations contained in the Online Services DPA.
Online Services DPA
Review + create
< Previous
Next : Network >
Give feedback


ITIS 6177 - Face API : 2022
Quickstart: Image Analysis
AI-900-AIFundamentals
Create Cognitive Services
Gradle | Installation
java - How to run Gradle fr
portal.azure.com/?azure-portal=true#create/Microsoft.CognitiveServicesAllInOne
Update
Microsoft Azure
Search resources, services, and docs (G+/)
gottumu@outlook.com
DEFAULT DIRECTORY (GOTTUM...
Home > Create a resource > Marketplace > Cognitive Services >
Create Cognitive Services
Validation Passed
Basics
Network
Identity
Tags
Review + create
TERMS
By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.
Basics
Subscription
Azure subscription 1
Resource group
Myapp
Region
East US
Name
myfinalapi
Pricing tier
Standard S0
Network
Type
All networks, including the internet, can access this resource.
Create
< Previous
Next
Give feedback
Download a template for automation




```
2 actionable tasks: 2 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % mkdir -p src/main/java
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % cd src
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd main
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd java
mounikavarmagottumukkala@Mounikas-MacBook-Air java % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle build

BUILD SUCCESSFUL in 16s
5 actionable tasks: 5 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle run

> Task :run

Azure Cognitive Services Computer Vision - Java Quickstart Sample


Analyzing an image from a URL ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Tags:
'person' with confidence 0.998895
'human face' with confidence 0.997437
'smile' with confidence 0.991973
'outdoor' with confidence 0.985962
'happy' with confidence 0.969785
'clothing' with confidence 0.961570
'friendship' with confidence 0.946441
'tree' with confidence 0.917331
'female person' with confidence 0.890976
'girl' with confidence 0.888741
'social group' with confidence 0.872044
'posing' with confidence 0.865493
'adolescent' with confidence 0.857371
'love' with confidence 0.852553
'laugh' with confidence 0.850097
'people' with confidence 0.849922
'lady' with confidence 0.844540
'woman' with confidence 0.818172
'group' with confidence 0.792975
'wedding' with confidence 0.615252
'dress' with confidence 0.517169

BUILD SUCCESSFUL in 1s
2 actionable tasks: 1 executed, 1 up-to-date
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP %
```

```
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/gettext... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/openssl@1.1... (64B)
Pruned 0 symbolic links and 4 directories from /opt/homebrew
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle init --type basic


Welcome to Gradle 7.6!

Here are the highlights of this release:
 - Added support for Java 19
 - Introduced `--rerun` flag for individual task rerun.
 - Improved dependency block for test suites to be strongly typed.
 - Added a pluggable system for Java toolchains provisioning.

For more details see https://docs.gradle.org/7.6/release-notes.html

Starting a Gradle Daemon (subsequent builds will be faster)

Select build script DSL:
  1: Groovy
  2: Kotlin
Enter selection (default: Groovy) [1..2] 2

Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no] yes
Project name (default: MYAPP):

> Task :init
Get more help with your project: Learn more about Gradle by exploring our samples at https://docs.gradle.org/7.6/samples

BUILD SUCCESSFUL in 39s
2 actionable tasks: 2 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % mkdir -p src/main/java
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % cd src
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd main
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd java
mounikavarmagottumukkala@Mounikas-MacBook-Air java % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle build

BUILD SUCCESSFUL in 16s
5 actionable tasks: 5 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP %
```

```
BUILD SUCCESSFUL in 39s
2 actionable tasks: 2 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % mkdir -p src/main/java
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % cd src
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd main
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd java
mounikavarmagottumukkala@Mounikas-MacBook-Air java % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd ..
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle build

BUILD SUCCESSFUL in 16s
5 actionable tasks: 5 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle run

> Task :run

Azure Cognitive Services Computer Vision - Java Quickstart Sample


Analyzing an image from a URL ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Tags:
'person' with confidence 0.998895
'human face' with confidence 0.997437
'smile' with confidence 0.991973
'outdoor' with confidence 0.985962
'happy' with confidence 0.969785
'clothing' with confidence 0.961570
'friendship' with confidence 0.946441
'tree' with confidence 0.917331
'female person' with confidence 0.890976
'girl' with confidence 0.888741
'social group' with confidence 0.872044
'posing' with confidence 0.865493
'adolescent' with confidence 0.857371
'love' with confidence 0.852553
'laugh' with confidence 0.850897
'people' with confidence 0.849322
'lady' with confidence 0.844540
'woman' with confidence 0.818172
'group' with confidence 0.792975
'wedding' with confidence 0.615252
'dress' with confidence 0.517169

BUILD SUCCESSFUL in 1s
2 actionable tasks: 1 executed, 1 up-to-date
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP %
```

```
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/mongodb-database-tools--100.5.2.zip... (57.3MB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/openssl@1.1--1.1.1w... (3MB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/wget--1.21.3... (3.5MB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/openssl@1.1_bottle_manifest--1.1.1w... (7.6KB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/portable-ruby-2.6.8.arm64_big_sur.bottle.tar.gz... (8.7MB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/ca-certificates_bottle_manifest--2022-07-19... (1.8KB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/libidn2_bottle_manifest--2.3.2... (8.7KB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/wget_bottle_manifest--1.21.3... (9.1KB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/gettext_bottle_manifest--0.21... (10.5KB)
Removing: /Users/mounikavarmagottumukkala/Library/Caches/Homebrew/libunistring_bottle_manifest--1.0... (8.2KB)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/wget... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/libidn2... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/mongodb-database-tools... (131B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/libunistring... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/mongodb-community@4.4... (130B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/ca-certificates... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/gettext... (64B)
Removing: /Users/mounikavarmagottumukkala/Library/Logs/Homebrew/openssl@1.1... (64B)
Pruned 0 symbolic links and 4 directories from /opt/homebrew
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % gradle init --type basic


Welcome to Gradle 7.6!

Here are the highlights of this release:
 - Added support for Java 19
 - Introduced `--rerun` flag for individual task rerun.
 - Improved dependency block for test suites to be strongly typed.
 - Added a pluggable system for Java toolchains provisioning.

For more details see https://docs.gradle.org/7.6/release-notes.html

Starting a Gradle Daemon (subsequent builds will be faster)

Select build script DSL:
  1: Groovy
  2: Kotlin
Enter selection (default: Groovy) [1..2] 2

Generate build using new APIs and behavior (some features may change in the next minor release)? (default: no) [yes, no] yes
Project name (default: MYAPP):

> Task :init
Get more help with your project: Learn more about Gradle by exploring our samples at https://docs.gradle.org/7.6/samples

BUILD SUCCESSFUL in 39s
2 actionable tasks: 2 executed
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % mkdir -p src/main/java
mounikavarmagottumukkala@Mounikas-MacBook-Air MYAPP % cd src
mounikavarmagottumukkala@Mounikas-MacBook-Air src % cd main
mounikavarmagottumukkala@Mounikas-MacBook-Air main % cd java
mounikavarmagottumukkala@Mounikas-MacBook-Air java %
```

Explore Computer Vision

1. Create a Cognitive Services resource
2. You can use the Computer Vision service by creating either a Computer Vision resource or a Cognitive Services resource. If you haven't already done so, create a Cognitive Services resource in your Azure subscription.
3. In another browser tab, open the Azure portal at https://portal.azure.com, signing in with your Microsoft account.
4. Click the +Create a resource button, search for Cognitive Services, and create a Cognitive Services resource with the following settings:

    - **Subscription**: Your Azure subscription.
    - **Resource group**: Select or create a resource group with a unique name.
    - **Region**: Choose any available region.
    - **Name**: Enter a unique name.
    - **Pricing tier**: Standard S0
    - By checking this box, I acknowledge that I have read and understood all the terms below: Selected.
2. Review and create the resource and wait for deployment to complete. Then go to the deployed resource.
3. View the **Keys and Endpoint** page for your Cognitive Services resource. You will need the endpoint and keys to connect from client applications.
4. Run Cloud Shell

To test the capabilities of the Computer Vision service, we'll use a simple command-line application that runs in the Cloud Shell on Azure.

1. In the Azure portal, select the **[>_]** (Cloud Shell) button at the top of the page to the right of the search box. This opens a Cloud Shell pane at the bottom of the portal.
2. The first time you open the Cloud Shell, you may be prompted to choose the type of shell you want to use (Bash or PowerShell). Select **PowerShell**. If you do not see this option, skip the step.
3. If you are prompted to create storage for your Cloud Shell, ensure your subscription is specified and select **Create storage**. Then wait a minute or so for the storage to be created.
4. Make sure the the type of shell indicated on the top left of the Cloud Shell pane is switched to PowerShell. If it is Bash, switch to PowerShell by using the drop-down menu.
5. Wait for PowerShell to start. You should see the following screen in the Azure portal:
6. Configure and run a client application. Now that you have a Cloud Shell environment, you can run a simple application that uses the Computer Vision service to analyze an image.
7. In the command shell, enter the following command to download the sample application and save it to a folder called ai-900.

```
git clone https://github.com/MicrosoftLearning/AI-900-AIFundamentals ai-900
```

- The files are downloaded to a folder named **ai-900**. Now we want to see all of the files in your Cloud Shell storage and work with them. Type the following command into the shell:

```
code .
```

- Notice how this opens up an editor like the one in the image below:
- In the Files pane on the left, expand ai-900 and select analyze-image.ps1. This file contains some code that uses the Computer Vision service to analyze an image, as shown here
- Don't worry too much about the code, the important thing is that it needs the endpoint URL and either of the keys for your Cognitive Services resource. Copy these from the Keys and Endpoints page for your resource from the Azure portal and paste them into the code editor, replacing the YOUR_KEY and YOUR_ENDPOINT placeholder values respectively.
- After pasting the key and endpoint values, the first two lines of code should look similar to this:

```
- $key="1a2b3c4d5e6f7g8h9i0j...."
- $endpoint="https..."
```

- At the top right of the editor pane, use the **…** button to open the menu and select **Save** to save your changes.
- In the PowerShell pane, enter the following commands to run the code:

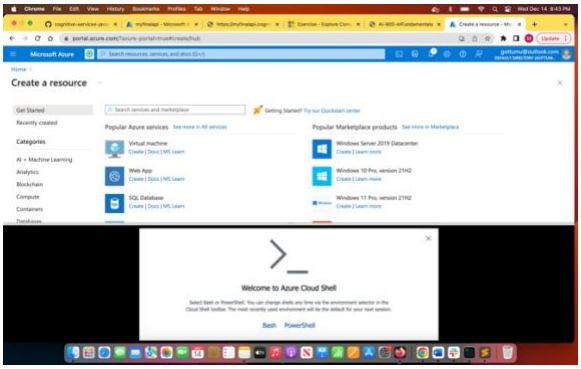
```
cd ai-900
./analyze-image.ps1 store-camera-1.jpg
```

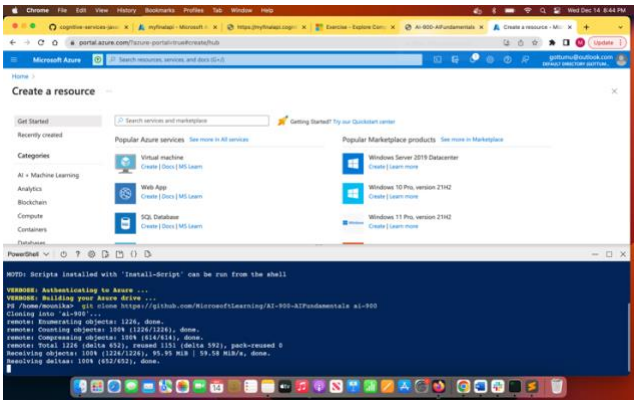
- Review the results of the image analysis, which include:
- A suggested caption that describes the image.
- A list of objects identified in the image.
- A list of "tags" that are relevant to the image.
- Now let's try another image:
- To analyze the second image, enter the following command:

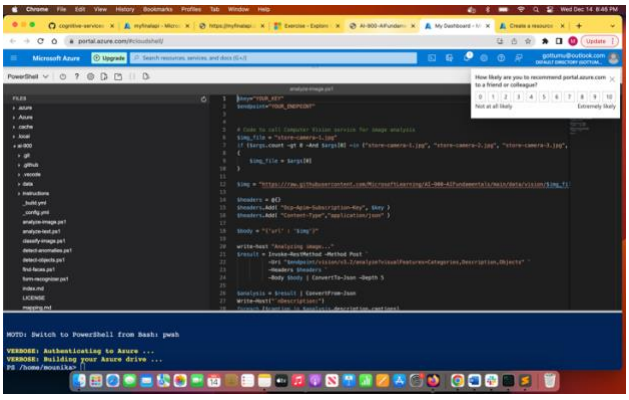
```
./analyze-image.ps1 store-camera-2.jpg
```

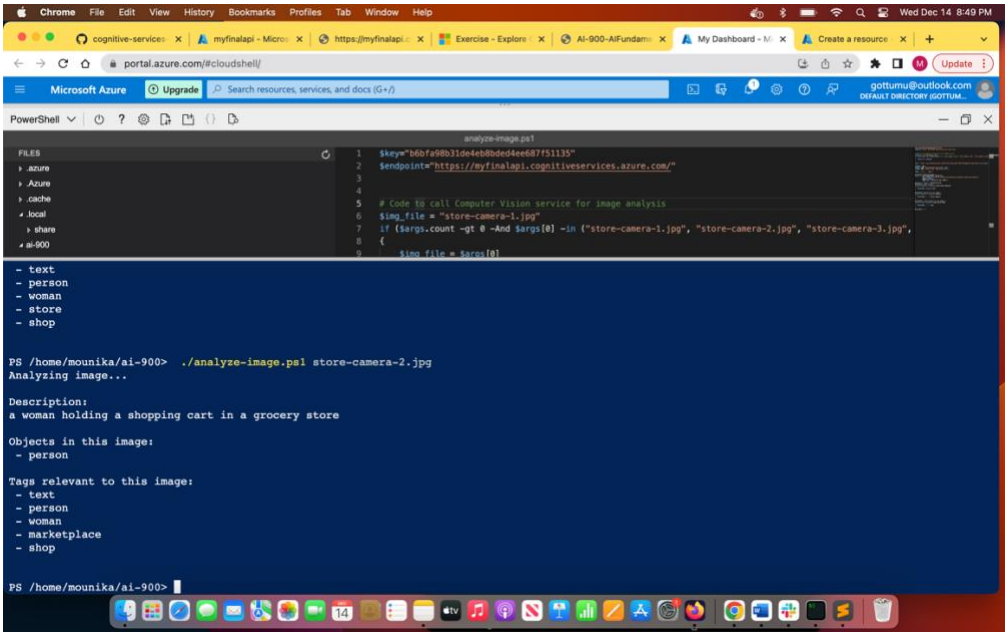
- Review the results of the image analysis for the second image.
- To analyze the third image, enter the following command:

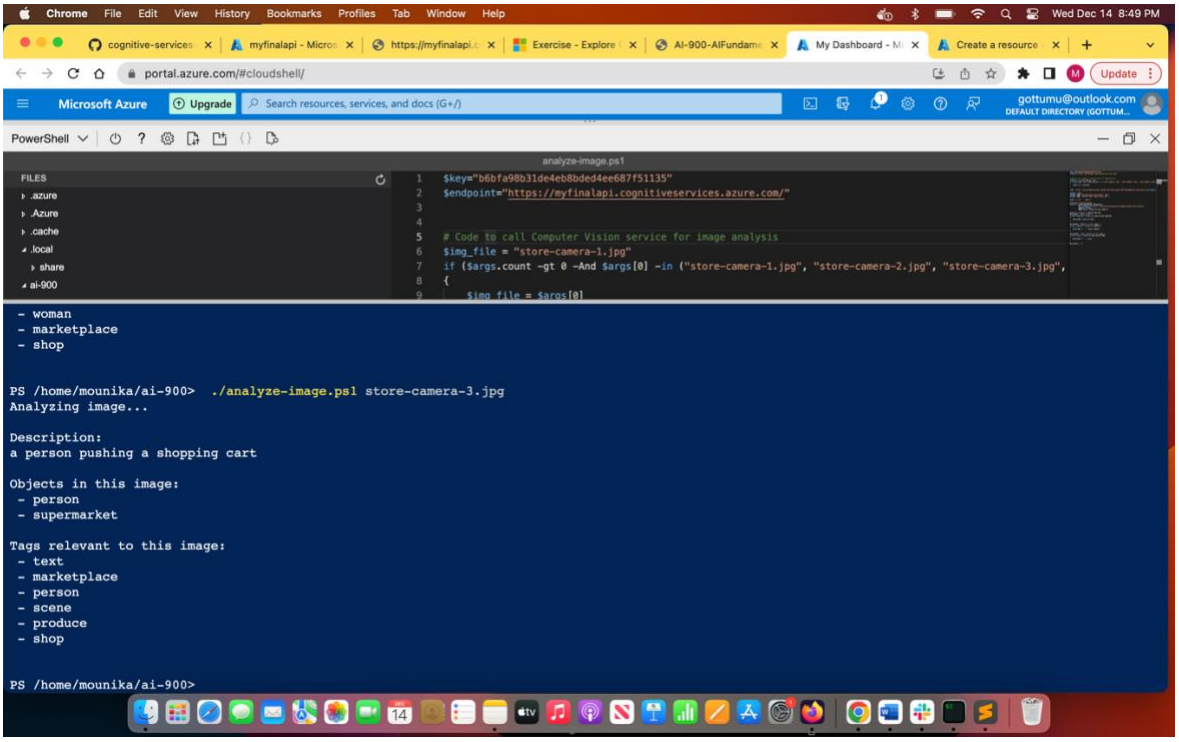
```
./analyze-image.ps1 store-camera-3.jpg
```


Create a resource
Welcome to Azure Cloud Shell
Bash
PowerShell


Create a resource
MOTD: Scripts installed with 'Install-Script' can be run from the shell
VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/mounika> git clone https://github.com/MicrosoftLearning/AI-900-AIFundamentals ai-900
Cloning into 'ai-900'...
remote: Enumerating objects: 1226, done.
remote: Counting objects: 100% (1226/1226), done.
remote: Compressing objects: 100% (614/614), done.
remote: Total 1226 (delta 652), reused 1151 (delta 592), pack-reused 0
Receiving objects: 100% (1226/1226), 95.95 MiB | 59.58 MiB/s, done.
Resolving deltas: 100% (652/652), done.


MOTD: Switch to PowerShell from Bash: pwsh
VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/mounika>


- text
- person
- woman
- store
- shop
PS /home/mounika/ai-900> ./analyze-image.ps1 store-camera-2.jpg
Analyzing image...
Description:
a woman holding a shopping cart in a grocery store
Objects in this image:
- person
Tags relevant to this image:
- text
- person
- woman
- marketplace
- shop
PS /home/mounika/ai-900>


- woman
- marketplace
- shop
PS /home/mounika/ai-900> ./analyze-image.ps1 store-camera-3.jpg
Analyzing image...
Description:
a person pushing a shopping cart
Objects in this image:
- person
- supermarket
Tags relevant to this image:
- text
- marketplace
- person
- scene
- produce
- shop
PS /home/mounika/ai-900>

Call the Image Analysis API

Submit data to the service

In your main class, save a reference to the URL of the image you want to analyze(Java)

String pathToRemoteImage = "https://github.com/Azure-Samples/cognitive-services-sample-data-files/raw/master/ComputerVision/Images/faces.jpg";

Determine how to process the data

Select visual features : The Analyze API gives you access to all of the service's image analysis features. Choose which operations to do based on your own use case.

```
// This list defines the features to be extracted from the image.
List<VisualFeatureTypes> featuresToExtractFromRemoteImage = new ArrayList<>();
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.DESCRIPTION);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.CATEGORIES);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.TAGS);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.FACES);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.ADULT);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.COLOR);
featuresToExtractFromRemoteImage.add(VisualFeatureTypes.IMAGE_TYPE);
```

Use the AnalyzeImageOptionalParameter input in your Analyze call to specify a language.
A method call that specifies a language might look like the following.

```
ImageAnalysis analysis =
compVisClient.computerVision().analyzeImage().withUrl(pathToRemoteImage)
    .withVisualFeatures(featuresToExtractFromLocalImage)
    .language("en")
    .execute();
```

Get results from the service

This section shows you how to parse the results of the API call. It includes the API call itself.The following code calls the Image Analysis API and prints the results to the console.

```
// Call the Computer Vision service and tell it to analyze the loaded image.
ImageAnalysis analysis =
compVisClient.computerVision().analyzeImage().withUrl(pathToRemoteImage)
        .withVisualFeatures(featuresToExtractFromRemoteImage).execute();

// Display image captions and confidence values.
System.out.println("\nCaptions: ");
for (ImageCaption caption : analysis.description().captions()) {
    System.out.printf("\'%s\' with confidence %f\n", caption.text(), caption.confidence());
```

```
}

// Display image category names and confidence values.
System.out.println("\nCategories: ");
for (Category category : analysis.categories()) {
   System.out.printf("\'%s\' with confidence %f\n", category.name(), category.score());
}

// Display image tags and confidence values.
System.out.println("\nTags: ");
for (ImageTag tag : analysis.tags()) {
   System.out.printf("\'%s\' with confidence %f\n", tag.name(), tag.confidence());
}

// Display any faces found in the image and their location.
System.out.println("\nFaces: ");
for (FaceDescription face : analysis.faces()) {
   System.out.printf("\'%s\' of age %d at location (%d, %d), (%d, %d)\n", face.gender(),
face.age(),
         face.faceRectangle().left(), face.faceRectangle().top(),
         face.faceRectangle().left() + face.faceRectangle().width(),
         face.faceRectangle().top() + face.faceRectangle().height());
}

// Display whether any adult or racy content was detected and the confidence
// values.
System.out.println("\nAdult: ");
System.out.printf("Is adult content: %b with confidence %f\n", analysis.adult().isAdultContent(),
      analysis.adult().adultScore());
System.out.printf("Has racy content: %b with confidence %f\n",
analysis.adult().isRacyContent(),
      analysis.adult().racyScore());

// Display the image color scheme.
System.out.println("\nColor scheme: ");
System.out.println("Is black and white: " + analysis.color().isBWImg());
System.out.println("Accent color: " + analysis.color().accentColor());
System.out.println("Dominant background color: " +
analysis.color().dominantColorBackground());
System.out.println("Dominant foreground color: " +
analysis.color().dominantColorForeground());
System.out.println("Dominant colors: " + String.join(", ", analysis.color().dominantColors()));

// Display any celebrities detected in the image and their locations.
System.out.println("\nCelebrities: ");
for (Category category : analysis.categories()) {
```

```
    if (category.detail() != null && category.detail().celebrities() != null) {
        for (CelebritiesModel celeb : category.detail().celebrities()) {
            System.out.printf("\'%s\' with confidence %f at location (%d, %d), (%d, %d)\n",
celeb.name(),
                    celeb.confidence(), celeb.faceRectangle().left(), celeb.faceRectangle().top(),
                    celeb.faceRectangle().left() + celeb.faceRectangle().width(),
                    celeb.faceRectangle().top() + celeb.faceRectangle().height());
        }
    }
}

// Display any landmarks detected in the image and their locations.
System.out.println("\nLandmarks: ");
for (Category category : analysis.categories()) {
    if (category.detail() != null && category.detail().landmarks() != null) {
        for (LandmarksModel landmark : category.detail().landmarks()) {
            System.out.printf("\'%s\' with confidence %f\n", landmark.name(),
landmark.confidence());
        }
    }
}

// Display what type of clip art or line drawing the image is.
System.out.println("\nImage type:");
System.out.println("Clip art type: " + analysis.imageType().clipArtType());
System.out.println("Line drawing type: " + analysis.imageType().lineDrawingType());
```

For Live API: create a droplet in digital ocean and run the commands.

Conclusion:

As you can see the Computer Vision is, as the name suggests, good at recognizing objects. But it's not a plant image classification tool. Instead, its main function is to pick objects out of a photo and classify them in general. That has lots of applicable use cases, like letting police quickly scanning photos to find, for example, a criminal suspect walking down an otherwise empty alley. API makes it simpler to set up the API as the programmer can set up multiple target vendor APIs from one web site. Plus, developers can write their APIs and upload them.

References

https://learn.microsoft.com/en-us/azure/cognitive-services/computer-vision/
https://binarygrounds.com/2021/02/02/computer-vision-introduction.html