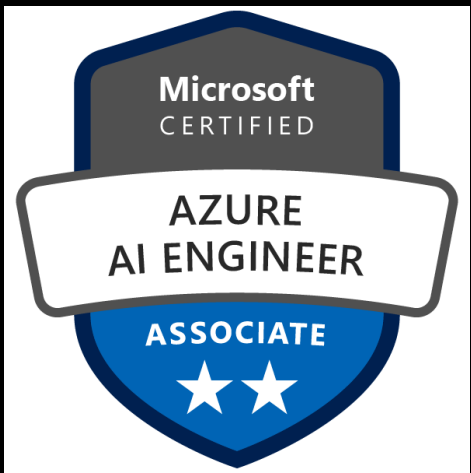
Challenge #4 // **Part 2**

Azure Cognitive Services

Using API's

IN THIS CHALLENGE:

6. Translator API (Cognitive Services and Translator)
7. Run your code using Azure ML Studio
8. Face API
9. Computer Vision
10. LUIS (Language Understanding)



This content is based on the main requirements for AI-102 Certification

Exam AI-102: Designing and Implementing a Microsoft Azure AI Solution

Candidates for Exam AI-102 should have subject matter expertise building, managing, and deploying AI solutions that leverage Azure Cognitive Services, Azure Cognitive Search, and Microsoft Bot Framework.

Candidates for this exam should be proficient in C#, Python, or JavaScript and should be able to use REST-based APIs and SDKs to build computer vision, natural language processing, knowledge mining, and conversational AI solutions on Azure. They should also understand the components that make up the Azure AI portfolio and the available data storage options. Plus, candidates need to understand and be able to apply responsible AI principles.

License agreements of different operating systems

I promise that I will tell every person without asking why Linux is so much better than every other operating system.

Before we start there are some important clarifications points:

(1) **Troubleshooting is REALLY important –** It is important for you to find the bugs in your code, env, IDE, etc.

(2) **The code IS JUST a code –** There are several ways to write a code and different languages. The examples here are just one way to do it.

(3) **This IS NOT a prep course –** The main goal here is to show the practical application of the Azure Resources with a focus on Enterprise AI solutions.

(4) **You won't be graded by the challenges** but, they are a important practical component in your learning experience.

First Steps

(1) Download the source files:

https://github.com/caiogasparine/AIDI1006

(2) Copy the folder IMAGES to your root directory - C:\

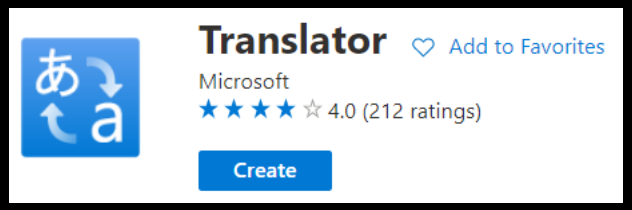
(There are some files used as example in this folder)

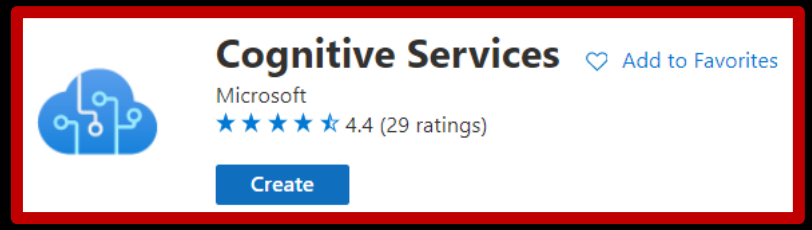
Troubleshooting…

is part of the challenge

Challenge #4.6

Translator API (Cognitive Services)



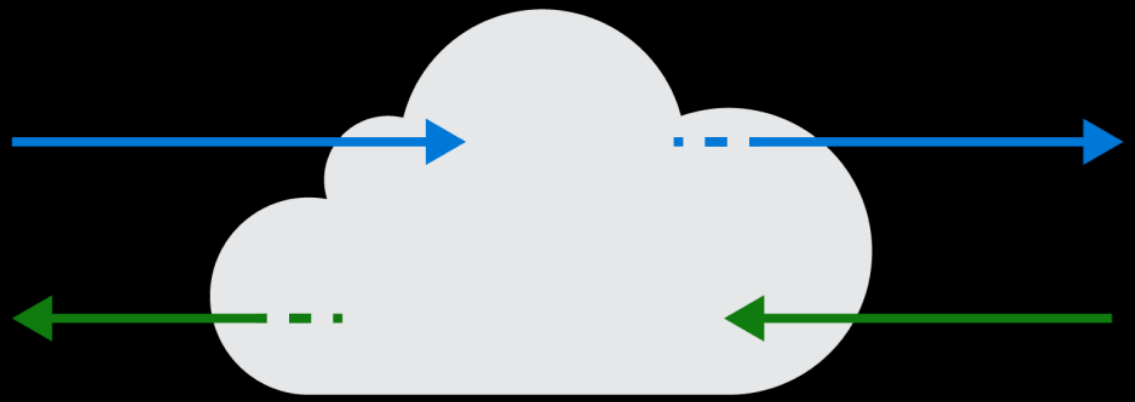


How it works?

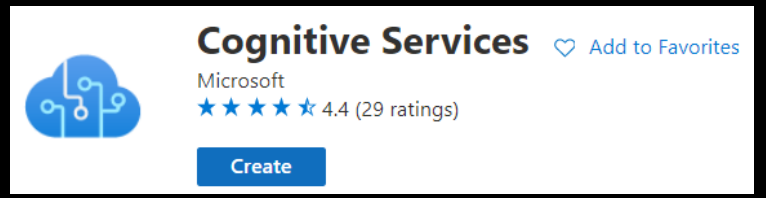
Client app or web page

Cet outil de traduction est un moyen facile d'étendre la portée de vos contenus internes et externes

這個翻譯工具是內容的可以輕鬆地聽到你內部和外部範圍



Web API



Untitled1 - Jupyter Notebook

localhost:8888/notebooks/Untitled1.ipynb?kernel_name=python3

jupyter Untitled1 Last Checkpoint: 11 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

There are several different ways to write your code. This is just one of them.

In [6]:

```
import requests, uuid, json

# Add your subscription key and endpoint
subscription_key = "95[redacted]1beb"
endpoint = "https://api.cognitive.microsofttranslator.com"

# Add your location, also known as region. The default is global.
# This is required if using a Cognitive Services resource.
location = "brazilsouth"

path = '/translate'
constructed_url = endpoint + path

params = {
    'api-version': '3.0',
    'from': 'en',
    'to': ['de', 'it']
}
constructed_url = endpoint + path

headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}

# You can pass more than one object in body.
body = [{
    'text': 'Hello World!'
}]

request = requests.post(constructed_url, params=params, headers=headers, json=body)
```

AIDI1006-text-translator.ipynb

The code // Jupyter Notebook

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-460f-ae...

Microsoft Azure Machine Learning

There are several different ways to write your code. This is just one of them.

Home > Notebooks

Notebooks

Files Samples

Users
caiogasparine
.amlignore
text-translator.ipynb

*text-translator.ipyn

Editors | Compute: machinelearning-pc - Creating | No kernel connected

Editing

```
import requests, uuid, json

# Add your subscription key and endpoint
subscription_key = "ac[redacted]38b"
endpoint = "https://api.cognitive.microsofttranslator.com"

# Add your location, also known as region. The default is global.
# This is required if using a Cognitive Services resource.
location = "eastus"

path = '/translate'
constructed_url = endpoint + path

params = {
    'api-version': '3.0',
    'from': 'en',
    'to': ['de', 'it']
}
constructed_url = endpoint + path

headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}

# You can pass more than one object in body.
```

AIDI1006-text-translator.ipynb

The code // Azure Machine Learning Studio

There are several different ways to write your code. This is just one of them.

```
import requests, uuid, json

# Add your subscription key and endpoint
subscription_key = "adb[redacted]c88b"
endpoint = "https://api.cognitive.microsofttranslator.com"

# Add your location, also known as region. The default is global.
# This is required if using a Cognitive Services resource.
location = "eastus"

path = '/translate'
constructed_url = endpoint + path

params = {
    'api-version': '3.0',
    'from': 'en',
    'to': ['de', 'it']
}
constructed_url = endpoint + path

headers = {
```

```
C:\Users\caiog\PycharmProjects\text-translator\venv\Scripts\python.exe C:/Users/caiog/PycharmProjects/text-translator/main.py
[
    {
        "translations": [
            {
                "text": "Hallo Welt!",
                "to": "de"
            },
            {
                "text": "Salve, mondo!",
```

AIDI1006-text-translator.ipynb

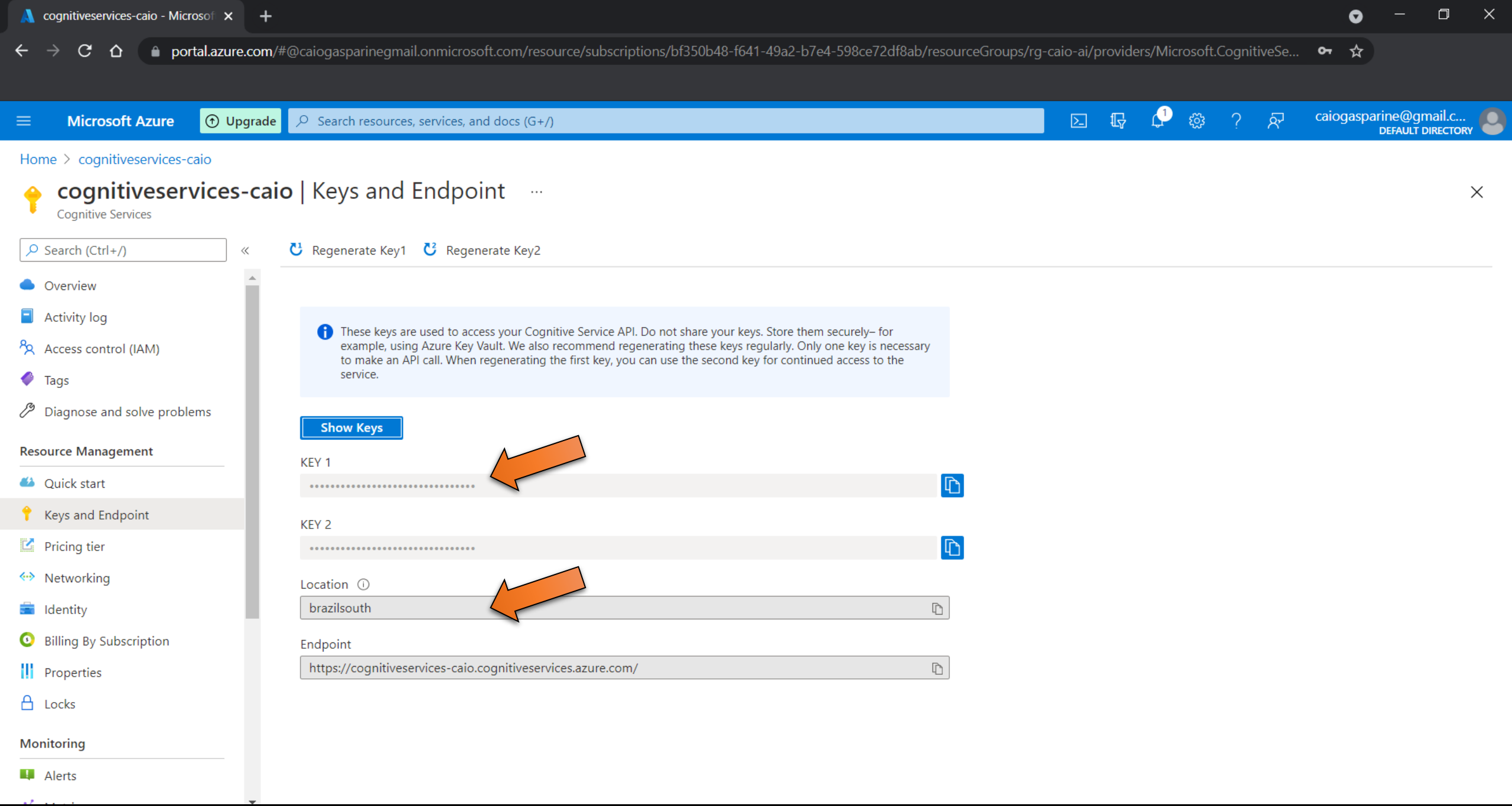
The code // Pycharn

There are several different ways to write your code. This is just one of them.

```
import requests, uuid, json

# Add your subscription key and endpoint
subscription_key = "YOUR SUBSCRIPTION"
endpoint = "https://api.cognitive.microsofttranslator.com"

# Add your location, also known as region. The default is global.
# This is required if using a Cognitive Services r
location = "YOUR SUBSCRIPTION LOCATIO

path = '/translate'
constructed_url = endpoint + path

params = {
    'api-version': '3.0',
    'from': 'en',
    'to': ['de', 'it']
}
constructed_url = endpoint + path

headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}

# You can pass more than one object in body.
body = [{
    'text': 'Hello World!'
}]

request = requests.post(constructed_url, params=params, headers=headers, json=body)
response = request.json()

print(json.dumps(response, sort_keys=True, ensure_ascii=False, indent=4, separators=(',', ': ')))
```

The code

AIDI1006-text-translator.ipynb


cognitiveservices-caio - Microsof
portal.azure.com/#@caiogasparinegmail.onmicrosoft.com/resource/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourceGroups/rg-caio-ai/providers/Microsoft.CognitiveSe...
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
caiogasparine@gmail.c...
DEFAULT DIRECTORY
Home > cognitiveservices-caio
cognitiveservices-caio | Keys and Endpoint
Cognitive Services
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Resource Management
Quick start
Keys and Endpoint
Pricing tier
Networking
Identity
Billing By Subscription
Properties
Locks
Monitoring
Alerts
Regenerate Key1
Regenerate Key2
These keys are used to access your Cognitive Service API. Do not share your keys. Store them securely– for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.
Show Keys
KEY 1
KEY 2
Location
brazilsouth
Endpoint
https://cognitiveservices-caio.cognitiveservices.azure.com/

The code

Untitled1 - Jupyter Notebook

localhost:8888/notebooks/Untitled1.ipynb?kernel_name=python3

jupyter Untitled1 Last Checkpoint: 14 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

There are several different ways to write your code. This is just one of them.

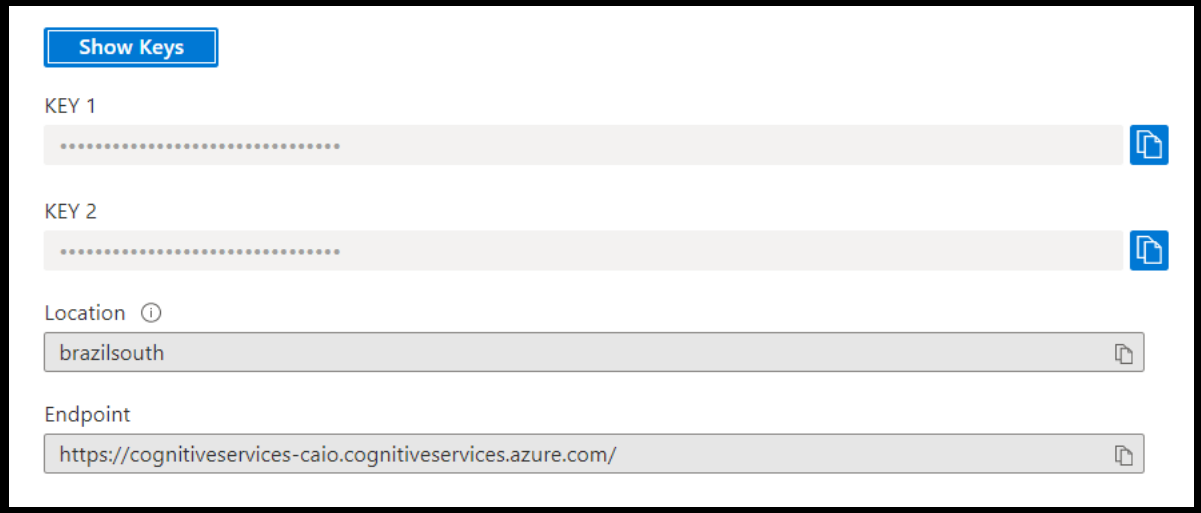
```
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}

# You can pass more than one object in body.
body = [{
    'text': 'Hello World!'
}]

request = requests.post(constructed_url, params=params, headers=headers, json=body)
response = request.json()

print(json.dumps(response, sort_keys=True, ensure_ascii=False, indent=4, separators=(',', ': ')))
```

```
[
    {
        "translations": [
            {
                "text": "Hallo Welt!",
                "to": "de"
            },
            {
                "text": "Salve, mondo!",
                "to": "it"
            }
        ]
    }
]
```

In [ ]:

In [ ]:

The RESULTS



About Keys

- **Key 1 – can be refreshed**
- **Key 2 – can be refreshed**

They are backups in case of endpoint misuse

https://docs.microsoft.com/en-us/azure/search/search-security-api-keys

Challenge #4.6

Translator API (Translator)

Translator - Microsoft Azure

portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryItemDetailsBladeNopdl/product/%7B"displayName"%3A"Translator"%2C"itemDisplayName"%3A"Translator"%2C"id"%3A"...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > Create a resource >

Translator

Microsoft

Translator

Add to Favorites

Microsoft

★★★★☆ 4.0 (212 ratings)

Create

Overview | Plans | Usage Information + Support | Reviews

Easily integrate real-time text translation capabilities into your application's websites, tools, or any solution requiring multi-language support such as website localization, e-commerce, customer support, messaging applications, internal communication, and more.

More offers from Microsoft

See All

Workspace

Microsoft

Virtual Machine

Azure Virtual Desktop resource

Microsoft HPC Pack 2012 R2

Microsoft

Virtual Machine

Enterprise-class HPC solution. Easy to deploy, cost-effective and supports Windows/Linux workloads.

Windows 10 IoT Core Services

Microsoft

Azure Service

Commercialize your project with enterprise-grade security and support

Web App + SQL

Microsoft

Azure Service

Enjoy secure and flexible development, deployment, and scaling options for your web app

Search for TRANSLATOR

Create Translator - Microsoft Azu

portal.azure.com/#create/Microsoft.CognitiveServicesTextTranslation

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > Create a resource > Translator >

Create Translator

manage all your resources.

Subscription * : Free Trial

Resource group * : rg-caio-ai

Create new

Instance details

Please choose the Global region unless your business or application requires a specific region. Applications that do not offer a region selection use the Global region.

Region * : East US

Name * : translator-caio

Pricing tier * : Free F0 (Up to 2M characters translated per month)

View full pricing details

Review + create | < Previous | Next : Identity >

Add all the information for your resource

translator-caio - Microsoft Azure

portal.azure.com/#@caiogasparinegmail.onmicrosoft.com/resource/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourceGroups/rg-caio-ai/providers/Microsoft.CognitiveSe...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > Microsoft.CognitiveServicesTextTranslation-20210611093411 > translator-caio

translator-caio | Quick start

Cognitive Services

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Resource Management

- Quick start
- Keys and Endpoint
- Encryption
- Pricing tier
- Networking
- Identity
- Billing By Subscription
- Properties
- Locks

Monitoring

Congratulations! Your keys are ready.

Now explore the Quickstart guidance to get up and running with Translator.

1

Get the API Key to authenticate your applications and start sending calls to the service.

Every call to Translator API requires the subscription key. The key can be found in the Keys and Endpoint section in the left pane. This key needs to be either passed through a query string parameter or specified in the request header.

2

Make an API call

Get in-depth information about the properties and methods of the API. Test your keys with the built-in testing console without writing a single line of code.

API reference

3

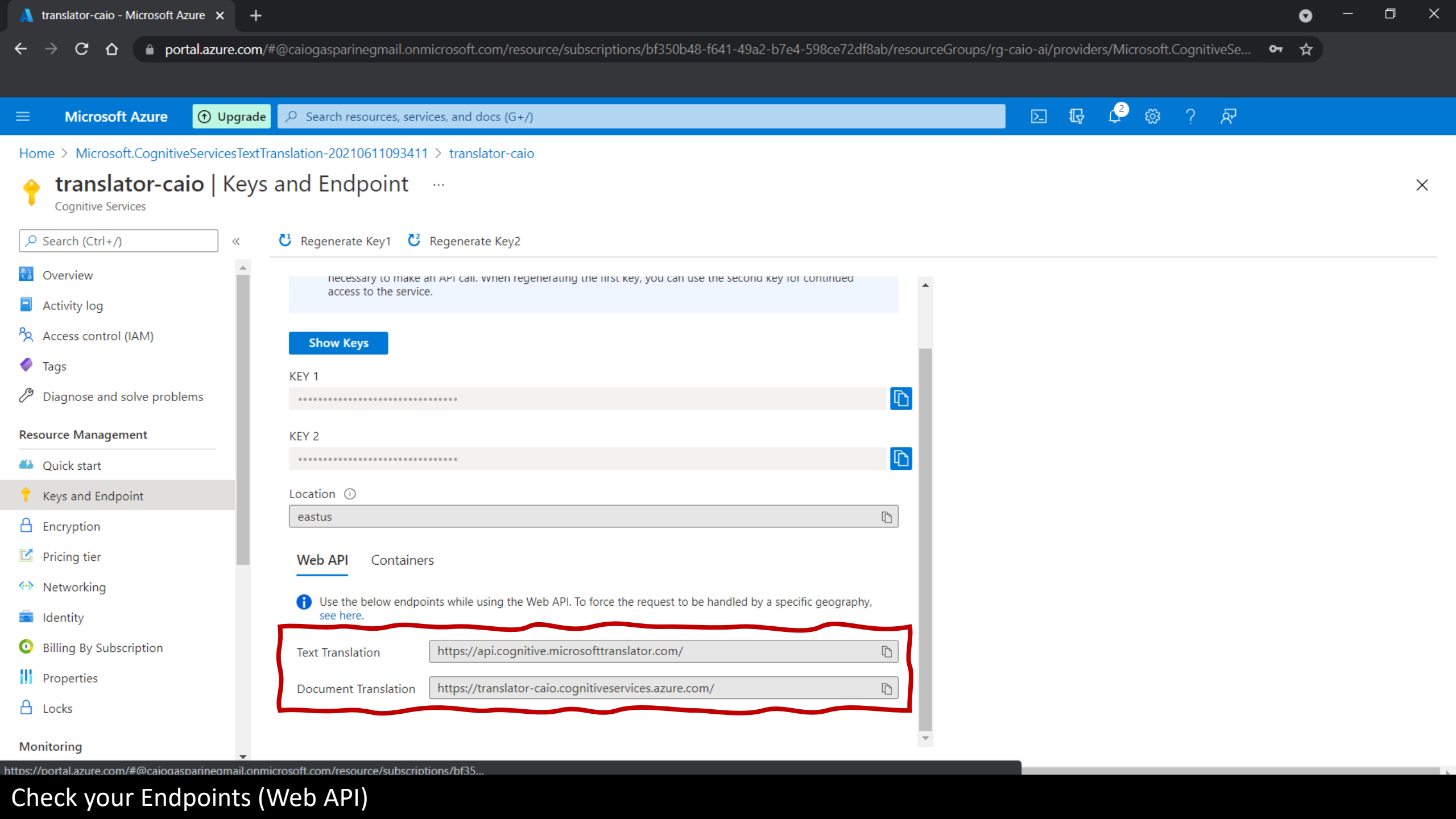
Enjoy coding

Speed up your learning with quick starts, how-to guidance and much more

Documentation

Code samples

Translator up and running!



Check your Endpoints (Web API)

(1) Now try to run your code using the Translator resource instead of Cognitive Services

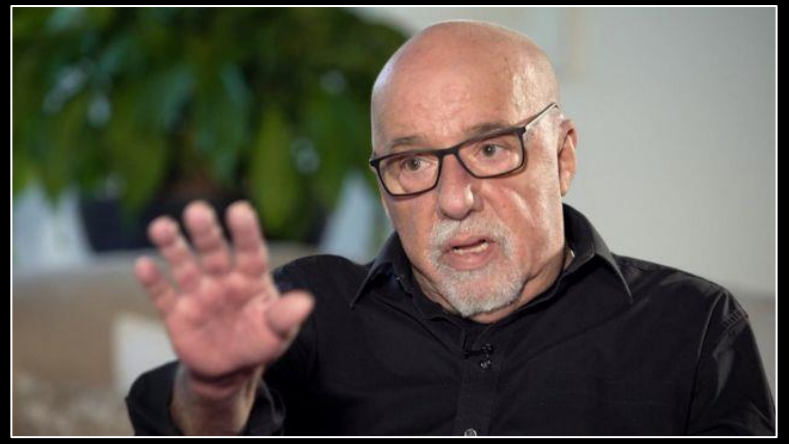
(2) Add translations (text next slide) to Filipino, French (Canada), Punjabi, Turkish, and English.

Text to be translated: (Brazilian Portuguese)

Tudo em torno de Paulo Coelho é superlativo. Do banheiro para visitas decorado com um quadro assinado por Andy Warhol ao elevador de vidro que vai da sala de estar ao enorme terraço sob os Alpes suíços. Dos mais de 325 milhões de livros vendidos e um bilhão de leitores em 150 países, ao recorde de escritor vivo mais traduzido do mundo e quase 50 milhões de seguidores em redes sociais. Da tortura a que foi submetido durante três meses, em 1974, à forma contundente como critica o governo brasileiro, em 2019.

"O compromisso histórico é não ficar calado. Eu tenho que falar. Vou perder leitores? Vou. Tenho perdido? Devo estar perdendo? Não sei. Eu não fico contabilizando", diz, enquanto a esposa Christina Oiticica, que acompanha a entrevista, assente com um leve gesto de aprovação.

Em livros como *Hippie*, o mais recente (2018), ou *O Aleph*, de 2010, Paulo Coelho alerta que o passado pode destruir o presente. Mas, nesta entrevista, ele decide lembrar com detalhes dos meses em que foi espancado, teve os genitais presos a eletrodos e foi trancafiado nu, com um capuz, em uma sala gelada e escura por agentes da ditadura.



Enterview with Paulo Coelho
Brazilian writer on September 25th 2019.

BBC News Brazil
https://www.bbc.com/portuguese/brasil-49665128

Home | AIDI1006-text-translator - Jupyte | localhost:8888/notebooks/AIDI1006-text-translator.ipynb

jupyter AIDI1006-text-translator Last Checkpoint: Last Friday at 9:50 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3

Run | Code

```
response = request.json()

print(json.dumps(response, sort_keys=True, ensure_ascii=False, indent=4, separators=(',', ': ')))
```

```
[
    {
        "translations": [
            {
                "text": "Alles rund um Paulo Coelho ist superlativ. Vom Gästebad mit einem von Andy Warhol signierten Gemälde b
is zum gläsernen Aufzug, der vom Wohnzimmer auf die riesige Terrasse unter den Schweizer Alpen führt. Von den mehr als 325 Mill
ionen verkauften Büchern und einer Milliarde Lesern in 150 Ländern, auf den Rekord für den weltweit meistübersetzten lebenden S
chriftsteller und fast 50 Millionen Follower in sozialen Netzwerken. Von der Folter, der er 1974 drei Monate lang ausgesetzt wa
r, bis hin zu der harten Art und Weise, wie er die brasilianische Regierung 2019 kritisiert. Das historische Engagement besteht
nicht darin, zu schweigen. Ich muss reden. Werde ich Leser verlieren? Ich werde gehen. Habe ich verloren? Sollte ich verlieren?
Ich weiß nicht, ich weiß nicht Ich zähle nicht weiter, sagt er, während seine Frau Christina Oiticica, die das Interview beglei
tet, sich mit einer leichten Geste der Zustimmung niederließ. In Büchern wie Hippie, dem jüngsten (2018) oder The Aleph, 2010,
warnt Paulo Coelho davor, dass die Vergangenheit die Gegenwart zerstören kann. Aber in diesem Interview beschließt er, sich im
Detail an die Monate zu erinnern, in denen er geschlagen wurde, seine Genitalien an Elektroden hängen ließ und nackt, mit Kapuz
e, in einem kalten und dunklen Raum von Agenten der Diktatur eingesperrt wurde.",
                "to": "de"
            },
            {
                "text": "Tutto intorno a Paulo Coelho è superlativo. Dal bagno degli ospiti decorato con un dipinto firmato da
Andy Warhol all'ascensore di vetro che va dal soggiorno all'enorme terrazza sotto le Alpi svizzere. Degli oltre 325 milioni di
libri venduti e un miliardo di lettori in 150 paesi, al record per lo scrittore vivente più tradotto al mondo e quasi 50 milion
i di follower sui social network. Dalla tortura a cui è stato sottoposto per tre mesi, nel 1974, al modo duro in cui critica il
governo brasiliano nel 2019. L'impegno storico non è quello di tacere. Devo parlare. Perderò lettori? Me ne vado. Ho perso? Dov
rei perdere? Non lo so, non lo so Non continuo a contare - dice - mentre sua moglie Christina Oiticica, che accompagna l'interv
ista, si è sistemata con un leggero gesto di approvazione. In libri come Hippie, il più recente (2018), o The Aleph, 2010, Paul
o Coelho avverte che il passato può distruggere il presente. Ma in questa intervista, decide di ricordare nel dettaglio i mesi
in cui è stato picchiato, ha attaccato i genitali agli elettrodi ed è stato chiuso nudo, con un cappuccio, in una stanza fredda
e buia dagli agenti della dittatura.",
                "to": "it"
            },
            {
```

SCREENSHOT!

Expected result...

Challenge #4.7

Azure Cognitive Services

Run your code using Azure ML Studio

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-460f-ae

Microsoft Azure Machine Learning

Home > Notebooks

Notebooks

Files Samples

Users
caiogasparine
.amlignore
text-translator.ipynb

text-translator.ipynb

Editors | Compute: machinelearning-pc - Running

machinelearning-pc · Jupyter kernel idle | Python 3.6.9

```
24        'Content-type': 'application/json',
25        'X-ClientTraceId': str(uuid.uuid4())
26    }
27
28    # You can pass more than one object in body.
29    body = [{
30        'text': 'Hello World!'
31    }]
32
33    request = requests.post(constructed_url, params=params, headers=headers, json=body)
34    response = request.json()
35
36    print(json.dumps(response, sort_keys=True, ensure_ascii=False, indent=4, separators=(',', ': ')))
```

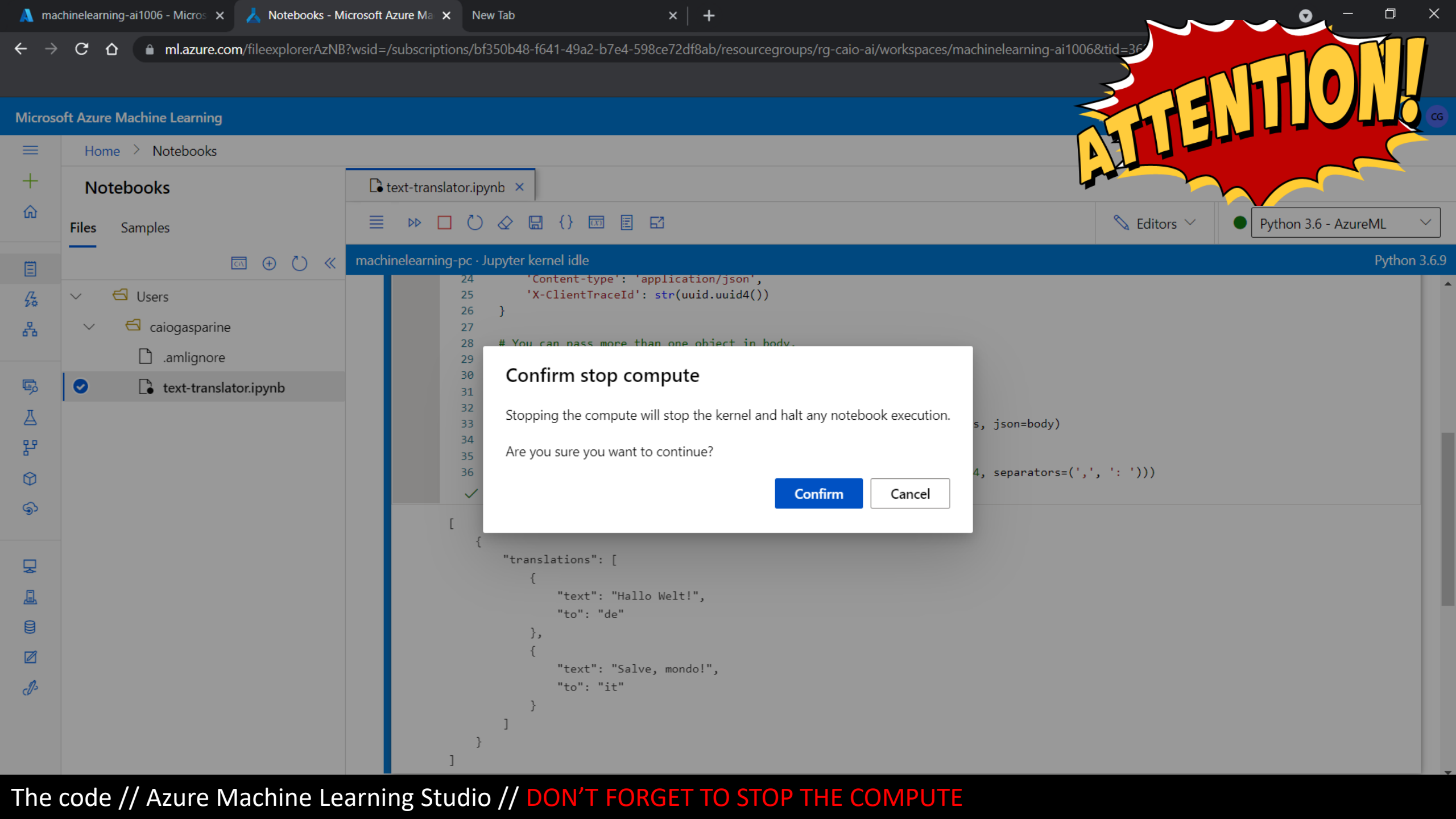
<1 sec

```
[
    {
        "translations": [
            {
                "text": "Hallo Welt!",
                "to": "de"
            },
            {
                "text": "Salve, mondo!",
                "to": "it"
            }
        ]
    }
]
```

There are several different ways to write your code. This is just one of them.

SCREENSHOT!

The code // Azure Machine Learning Studio



The code // Azure Machine Learning Studio // DON'T FORGET TO STOP THE COMPUTE

Challenge #4.8

Azure Cognitive Services

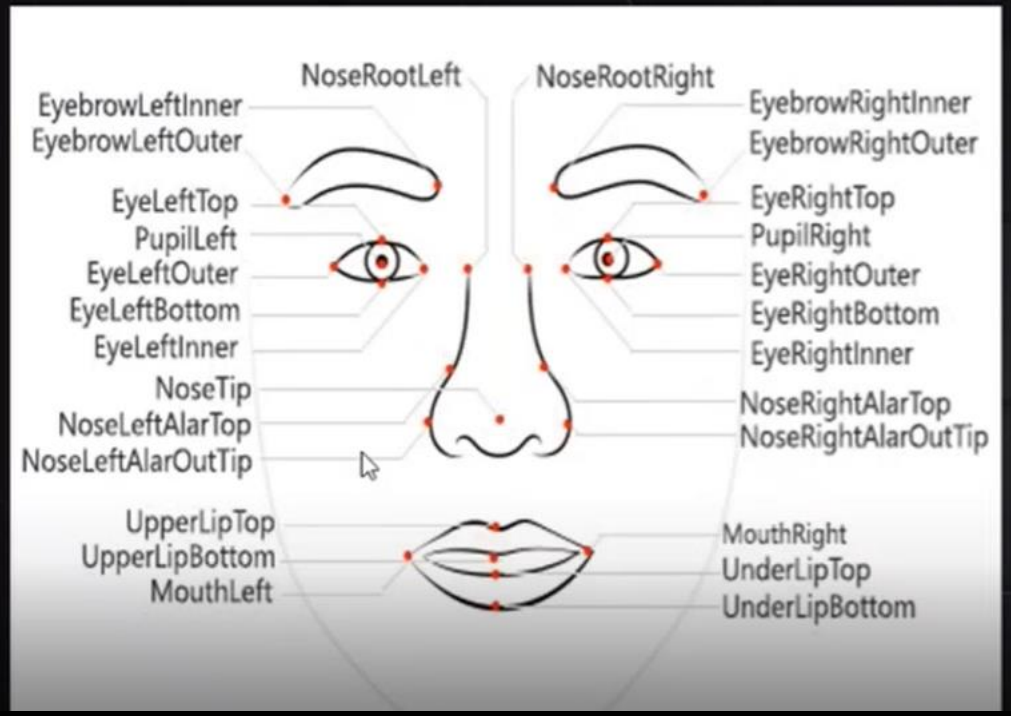
Face API

Face API

The Azure 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 security, natural user interface, image content analysis and management, mobile apps, and robotics.

https://azure.microsoft.com/en-us/services/cognitive-services/face/

Facial analysis



https://docs.microsoft.com/en-us/azure/cognitive-services/face/concepts/face-detection

Attributes

Attributes are a set of features that can optionally be detected by the Face - Detect API. The following attributes can be detected:

- **Accessories**. Whether the given face has accessories. This attribute returns possible accessories including headwear, glasses, and mask, with confidence score between zero and one for each accessory.
- **Age**. The estimated age in years of a particular face.
- **Blur**. The blurriness of the face in the image. This attribute returns a value between zero and one and an informal rating of low, medium, or high.
- **Emotion**. A list of emotions with their detection confidence for the given face. Confidence scores are normalized, and the scores across all emotions add up to one. The emotions returned are happiness, sadness, neutral, anger, contempt, disgust, surprise, and fear.
- **Exposure**. The exposure of the face in the image. This attribute returns a value between zero and one and an informal rating of underExposure, goodExposure, or overExposure.
- **Facial hair**. The estimated facial hair presence and the length for the given face.
- **Gender**. The estimated gender of the given face. Possible values are male, female, and genderless.
- **Glasses**. Whether the given face has eyeglasses. Possible values are NoGlasses, ReadingGlasses, Sunglasses, and Swimming Goggles.
- **Hair**. The hair type of the face. This attribute shows whether the hair is visible, whether baldness is detected, and what hair colors are detected.

https://docs.microsoft.com/en-us/azure/cognitive-services/face/concepts/face-detection

Attributes (2)

• **Head pose**. The face's orientation in 3D space. This attribute is described by the roll, yaw, and pitch angles in degrees, which are defined according to the right-hand rule. The order of three angles is roll-yaw-pitch, and each angle's value range is from -180 degrees to 180 degrees. 3D orientation of the face is estimated by the roll, yaw, and pitch angles in order. See the following diagram for angle mappings:

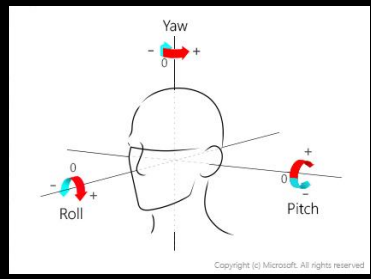
• **Makeup**. Whether the face has makeup. This attribute returns a Boolean value for eyeMakeup and lipMakeup.

• **Mask**. Whether the face is wearing a mask. This attribute returns a possible mask type, and a Boolean value to indicate whether nose and mouth are covered.

• **Noise**. The visual noise detected in the face image. This attribute returns a value between zero and one and an informal rating of low, medium, or high.

• **Occlusion**. Whether there are objects blocking parts of the face. This attribute returns a Boolean value for eyeOccluded, foreheadOccluded, and mouthOccluded.

• **Smile**. The smile expression of the given face. This value is between zero for no smile and one for a clear smile.



https://docs.microsoft.com/en-us/azure/cognitive-services/face/concepts/face-detection

face-api-ai1006 - Microsoft Azure

portal.azure.com/#@caiogasparinegmail.onmicrosoft.com/resource/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourceGroups/rg-caio-ai/providers/Microsoft.CognitiveServices...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > face-api-ai1006

face-api-ai1006 | Quick start

Cognitive Services

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Resource Management

- Quick start
- Keys and Endpoint
- Pricing tier
- Networking
- Identity
- Billing By Subscription
- Properties
- Locks

Monitoring

- Alerts

Explore the Quickstart guidance to get up and running with Face.

1

Get the API Key and endpoint to authenticate your applications and start sending calls to the service.

All Face calls and docker container activations require a key. The key can be found in the Keys and Endpoint section in the left pane. Specify the key either in the request header (Web API), the Face client (SDK), or through the command-line (Docker container).

2

Try the service in the API console - requires API Key and selecting your location.

Use the API Console to try the API without writing code. Be sure to select the correct location for this resource. The API key and the location can be found in the Keys and Endpoint section in the left tab.

API Console

3

Make a web API call - requires your API Key and endpoint

Use the sample code in these quickstarts to begin integrating Face into your applications to detect, recognize and analzye human faces in images

C# Quickstart
Python Quickstart

Learn more about the APIs

https://go.microsoft.com/fwlink/?linkid=2109273

Face API Console

Microsoft
Cognitive Services

APIs Documentation > API Reference

- Face
  - POST Detect
  - POST Find Similar
  - POST Group
  - POST Identify
  - POST Verify
- FaceList
- LargeFaceList
- LargePersonGroup
- LargePersonGroup Person
- PersonGroup
- PersonGroup Person
- Snapshot

Face API - v1.0

API definition

This API is currently available in:

- Australia East - australiaeast.api.cognitive.microsoft.com
- Brazil South - brazilsouth.api.cognitive.microsoft.com
- Canada Central - canadacentral.api.cognitive.microsoft.com
- Central India - centralindia.api.cognitive.microsoft.com
- Central US - centralus.api.cognitive.microsoft.com
- East Asia - eastasia.api.cognitive.microsoft.com
- East US - eastus.api.cognitive.microsoft.com
- East US 2 - eastus2.api.cognitive.microsoft.com
- France Central - francecentral.api.cognitive.microsoft.com
- Japan East - japaneast.api.cognitive.microsoft.com
- Japan West - japanwest.api.cognitive.microsoft.com
- Korea Central - koreacentral.api.cognitive.microsoft.com
- North Central US - northcentralus.api.cognitive.microsoft.com
- North Europe - northeurope.api.cognitive.microsoft.com
- South Africa North - southafricanorth.api.cognitive.microsoft.com
- South Central US - southcentralus.api.cognitive.microsoft.com
- Southeast Asia - southeastasia.api.cognitive.microsoft.com
- UK South - uksouth.api.cognitive.microsoft.com
- West Central US - westcentralus.api.cognitive.microsoft.com
- West Europe - westeurope.api.cognitive.microsoft.com
- West US - westus.api.cognitive.microsoft.com
- West US 2 - westus2.api.cognitive.microsoft.com
- UAE North - uaenorth.api.cognitive.microsoft.com

Face - Detect

Detect human faces in an image, return face rectangles, and optionally with faceIds, landmarks, and attributes.

face-api-ai1006 - Microsoft Azure

portal.azure.com/#@caiogasparinegmail.onmicrosoft.com/resource/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourceGroups/rg-caio-ai/providers/Microsoft.CognitiveSe...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > face-api-ai1006

face-api-ai1006 | Keys and Endpoint

Cognitive Services

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Resource Management

- Quick start
- Keys and Endpoint
- Pricing tier
- Networking
- Identity
- Billing By Subscription
- Properties
- Locks

Monitoring

- Alerts

Regenerate Key1 | Regenerate Key2

These keys are used to access your Cognitive Service API. Do not share your keys. Store them securely– for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.

Show Keys

KEY 1

KEY 2

Location

brazilsouth

Endpoint

https://face-api-ai1006.cognitiveservices.azure.com/

Face API // Keys and Endpoints

face-api-ai1006 - Microsoft Azure

portal.azure.com/#@caiogasparinegmail.onmicrosoft.com/resource/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourceGroups/rg-caio-ai/providers/Microsoft.CognitiveServices...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > face-api-ai1006

face-api-ai1006 | Quick start

Cognitive Services

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Resource Management

- Quick start
- Keys and Endpoint
- Pricing tier
- Networking
- Identity
- Billing By Subscription
- Properties
- Locks

Monitoring

- Alerts

Explore the Quickstart guidance to get up and running with Face.

1

Get the API Key and endpoint to authenticate your applications and start sending calls to the service.

All Face calls and docker container activations require a key. The key can be found in the Keys and Endpoint section in the left pane. Specify the key either in the request header (Web API), the Face client (SDK), or through the command-line (Docker container).

2

Try the service in the API console - requires API Key and selecting your location.

Use the API Console to quickly try the API without writing code. Be sure to select the correct location for this resource. The API key and the location can be found in the Keys and Endpoint section in the left tab.

API Console

3

Make a web API call - requires your API Key and endpoint

Use the sample code in these q[illegible]ts to begin integrating Face into your applications to detect, recognize and analzye human faces in images

C# Quickstart
Python Quickstart

Learn more about the APIs

https://go.microsoft.com/fwlink/?linkid=2109273

Face API QuickStart

There are several different ways to write your code. This is just one of them.

```
import json, os, requests

subscription_key = “YOUR SUBSCRIPTION”
face_api_url = “YOUR ENDPOINT” + '/face/v1.0/detect'

image_url = 'https://raw.githubusercontent.com/Azure-Samples/cognitive-services-sample-data-files/master/ComputerVision/Images/faces.jpg'

headers = {'Ocp-Apim-Subscription-Key': subscription_key}

params = {
    'detectionModel': 'detection_03',
    'returnFaceId': 'true'
}

response = requests.post(face_api_url, params=params,
                         headers=headers, json={"url": image_url})
print(json.dumps(response.json()))
```

AIDI1006-face-detection.ipynb

The code

Home Page - Select or create a n | AIDI1006-face-detection - Jupyte | cognitiveservices-caio - Microsof

localhost:8888/notebooks/AIDI1006-face-detection.ipynb

There are several different ways to write your code. This is just one of them.

jupyter AIDI1006-face-detection Last Checkpoint: 39 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

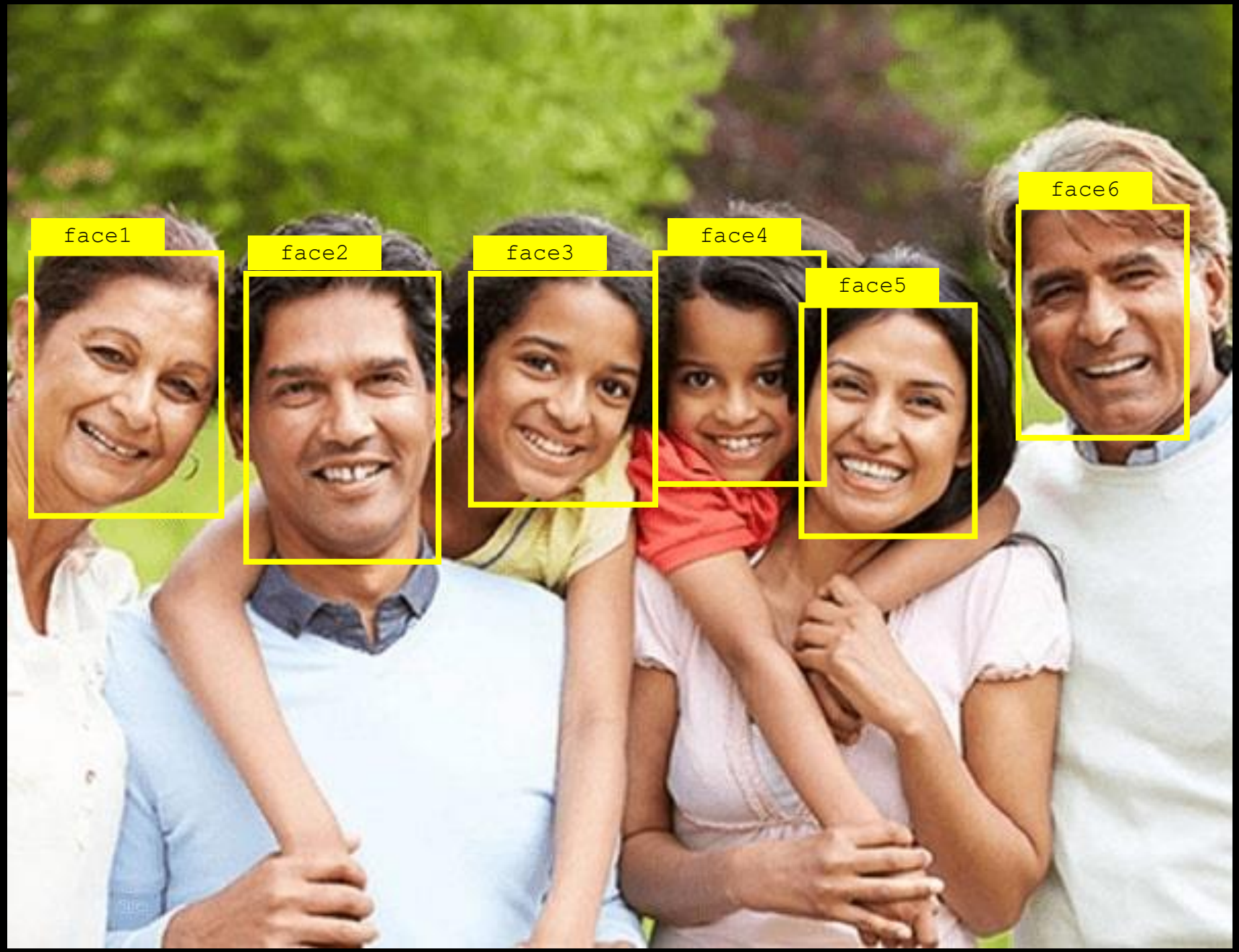
In [2]:
```
import json, os, requests

subscription_key = "95[redacted]1beb"
face_api_url = "https://[redacted]nitiveservices.azure.com" + '/face/v1.0/detect'

image_url = 'https://raw.githubusercontent.com/Azure-Samples/cognitive-services-sample-data-files/master/ComputerVision/Images/fa

headers = {'Ocp-Apim-Subscription-Key': subscription_key}

params = {
    'detectionModel': 'detection_03',
    'returnFaceId': 'true'
}

response = requests.post(face_api_url, params=params,
                         headers=headers, json={"url": image_url})
print(json.dumps(response.json()))
```

```
[{"faceId": "2715ba90-5d64-4c15-966d-69216ef33e36", "faceRectangle": {"top": 139, "left": 118, "width": 89, "height": 126}},
{"faceId": "66ab66d2-ba1a-4224-9b1a-93862f6ecc54", "faceRectangle": {"top": 95, "left": 494, "width": 92, "height": 115}}, {"fa
ceId": "c1134d6c-d283-476b-b449-0c223bd0f3f3", "faceRectangle": {"top": 131, "left": 12, "width": 92, "height": 110}}, {"faceI
d": "5090fad2-fecb-4f2e-9ed8-fea1e56ca615", "faceRectangle": {"top": 149, "left": 389, "width": 82, "height": 107}}, {"faceId":
"7193b0a6-80f0-454c-9ee0-8023e339eef3", "faceRectangle": {"top": 136, "left": 226, "width": 83, "height": 104}}, {"faceId": "7f
fd0c4b-16f6-4740-b36d-c3230c57cd46", "faceRectangle": {"top": 143, "left": 320, "width": 68, "height": 90}}]
```

In [ ]:

AIDI1006-face-detection.ipynb

Face API Response



The result // 6 faces

Public-Domain Test Images for Homeworks and Projects

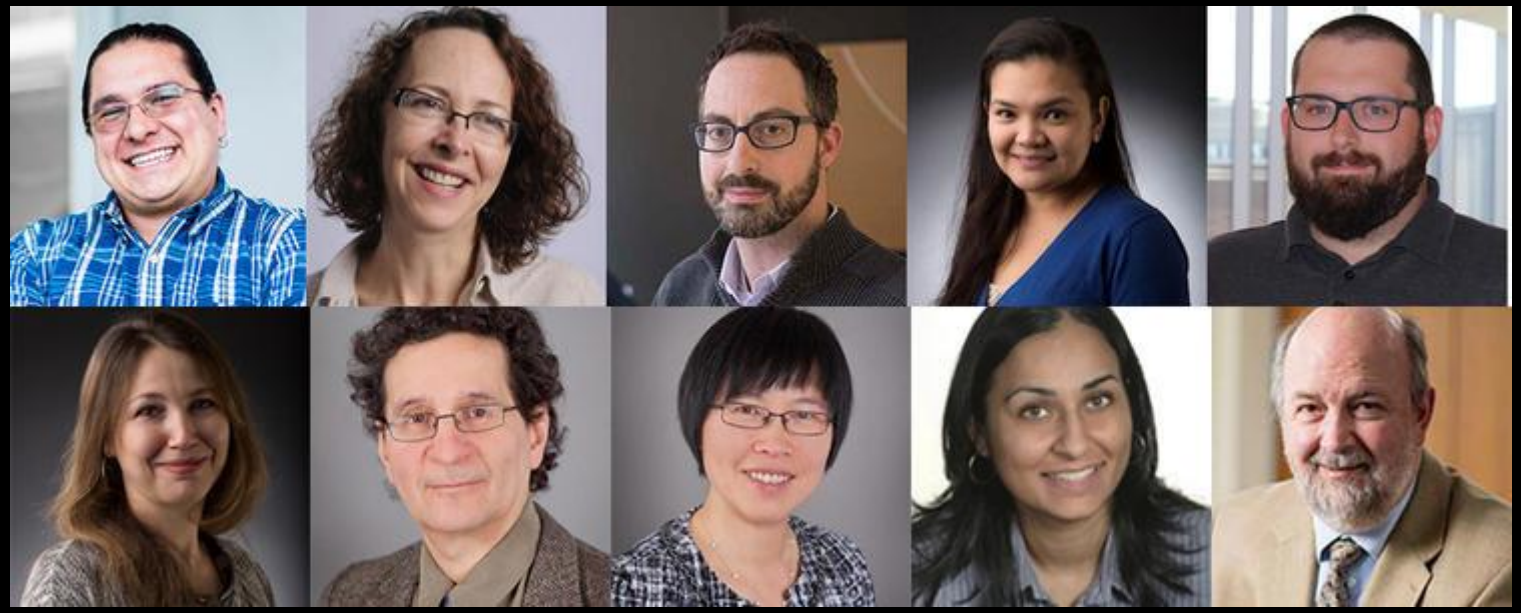
Face recognition ORL database

Photo database provided by Fabien a. p. petitcolas

Other images

- airplane.png
- arctichare.png
- baboon.png
- barbara.bmp, barbara.png
- boat.png
- boy.bmp, boy.ppm
- cameraman.tif
- cat.png
- fprint3.pgm
- fruits.png
- frymire.png
- girl.png
- goldhill.bmp, goldhill.png
- lena.bmp, lenacolor.png, lena.ppm, Lenaclor.ppm
- monarch.png
- mountain.png,mountain.bmp
- p64int.txt
- peppers.png
- pool.png
- sails.bmp, sails.png
- serrano.png
- tulips.png
- us021.pgm
- us092.pgm
- watch.png
- zelda.png

https://homepages.cae.wisc.edu/~ece533/images/ OR search for: “online test images”



SCREENSHOT!

Analyse this file in the following address: https://uwaterloo.ca/public-health-sciences/sites/ca.public-health-sciences/files/resize/uploads/images/sphhs-people-banner-2019-750x300.jpg

AIDI1006-face-detection - Jupyte

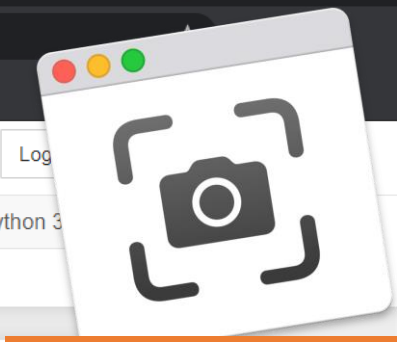
localhost:8888/notebooks/AIDI1006-face-detection.ipynb

jupyter AIDI1006-face-detection Last Checkpoint: 2 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code



SCREENSHOT!

In [6]:
```
import json, os, requests

subscription_key = "95[redacted]1beb"
face_api_url = "https://cognitiveservices-caio.cognitiveservices.azure.com" + '/face/v1.0/detect'

image_url = 'https://uwaterloo.ca/public-health-sciences/sites/ca.public-health-sciences/files/resize/uploads/images/sphhs-people

headers = {'Ocp-Apim-Subscription-Key': subscription_key}

params = {
    'detectionModel': 'detection_03',
    'returnFaceId': 'true'
}

response = requests.post(face_api_url, params=params,
                         headers=headers, json={"url": image_url})
print(json.dumps(response.json()))
```

There are several different ways to write your code. This is just one of them.


```
[{"faceId": "34a4e828-eb14-49fa-abf7-82f83f180f6c", "faceRectangle": {"top": 159, "left": 488, "width": 72, "height": 99}}, {"f
aceId": "e6a80b65-c7db-49f1-a8fd-531dd25a4b59", "faceRectangle": {"top": 165, "left": 185, "width": 73, "height": 96}}, {"faceI
d": "175c32d6-f984-47b3-b36b-85f7b220689d", "faceRectangle": {"top": 11, "left": 184, "width": 68, "height": 98}}, {"faceId":
"24a0a185-81de-4b53-8d25-c17ff3f01bff", "faceRectangle": {"top": 161, "left": 635, "width": 65, "height": 94}}, {"faceId": "9d6
d39e4-bfa4-42f6-8761-a5232999c63f", "faceRectangle": {"top": 14, "left": 638, "width": 66, "height": 88}}, {"faceId": "ad9e1e2d
-95c7-450d-9859-b593bd8f4b3e", "faceRectangle": {"top": 25, "left": 343, "width": 60, "height": 86}}, {"faceId": "e1775398-4499
-4b70-998d-c898bd7411f9", "faceRectangle": {"top": 176, "left": 346, "width": 62, "height": 79}}, {"faceId": "88001126-20c9-4dd
7-bd59-0042d31cfe04", "faceRectangle": {"top": 17, "left": 39, "width": 61, "height": 79}}, {"faceId": "f4eeddd3-5214-445c-8248
-e99569d7b7e1", "faceRectangle": {"top": 170, "left": 53, "width": 56, "height": 78}}, {"faceId": "f03f6cee-f4d6-43d3-ab0b-df8e
b8bae933", "faceRectangle": {"top": 25, "left": 486, "width": 52, "height": 67}}]
```


In [ ]:

The result...

Challenge #4.9

Azure Cognitive Services

Computer Vision

Computer Vision // Describe + Categorize

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. Categorize an image

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. Describe an image

https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-image-analysis

Home Page - Select or create a n | wAIDI1006-describe-categorize

localhost:8888/notebooks/wAIDI1006-describe-categorize.ipynb

There are several different ways to write your code. This is just one of them.

jupyter wAIDI1006-describe-categorize (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
# <snippet_categorize>
'''
Categorize an Image - remote
This example extracts (general) categories from a remote image with a confidence score.
'''
print("===== Categorize an image - remote =====")
# Select the visual feature(s) you want.
remote_image_features = ["categories"]
# Call API with URL and features
categorize_results_remote = computervision_client.analyze_image(remote_image_url , remote_image_features)

# Print results with confidence score
print("Categories from remote image: ")
if (len(categorize_results_remote.categories) == 0):
    print("No categories detected.")
else:
    for category in categorize_results_remote.categories:
        print("'{}' with confidence {:.2f}%".format(category.name, category.score * 100))
# </snippet_categorize>
print()
```

```
===== Describe an Image - local =====
Description of local image:
'a group of people posing for a photo' with confidence 70.30%

===== Describe an image - remote =====
Description of remote image:
'a plane flying over snowy mountains' with confidence 53.50%

===== Categorize an Image - local =====
Categories from local image:
'people_group' with confidence 78.52%

===== Categorize an image - remote =====
Categories from remote image:
'sky_object' with confidence 79.69%
'others_' with confidence 0.78%
'outdoor_' with confidence 0.78%
```

Results // Describe and categorize // REMOTE + LOCAL FILES

Home Page - Select or create a n | wAIDI1006-describe-categorize

localhost:8888/notebooks/wAIDI1006-describe-categorize.ipynb

jupyter wAIDI1006-describe-categorize Last Checkpoint: 2 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

In [3]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from azure.cognitiveservices.vision.computervision.models import OperationStatusCodes
from azure.cognitiveservices.vision.computervision.models import VisualFeatureTypes
from msrest.authentication import CognitiveServicesCredentials

from array import array
import os
from PIL import Image
import sys
import time

subscription_key = "[redacted]a72"
endpoint = "https://[redacted]eservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

# Quickstart variables // These variables are shared by several examples
#images_folder = os.path.join (os.path.dirname(os.path.abspath(__file__)), "images")
images_folder = "C:\images"
remote_image_url = "https://homepages.cae.wisc.edu/~ece533/images/airplane.png"

print("===== Describe an Image - local =====")
# Open local image file
local_image_path = os.path.join (images_folder, "people.jpg")
local_image = open(local_image_path, "rb")

# Call API
description_result = computervision_client.describe_image_in_stream(local_image)

# Get the captions (descriptions) from the response, with confidence level
print("Description of local image: ")
if (len(description_result.captions) == 0):
    print("No description detected.")
else:
    for caption in description_result.captions:
        print("'{}' with confidence {:.2f}%".format(caption.text, caption.confidence * 100))
print()
'''
END - Describe an Image - local
```

AIDI1006-describe-categorize

The code

Home Page - Select or create a n | wAIDI1006-describe-categorize

localhost:8888/notebooks/wAIDI1006-describe-categorize.ipynb

jupyter wAIDI1006-describe-categorize Last Checkpoint: 3 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
print("===== Describe an Image - local =====")
# Open local image file
local_image_path = os.path.join (images_folder, "people.jpg")
local_image = open(local_image_path, "rb")

# Call API
description_result = computervision_client.describe_image_in_stream(local_image)

# Get the captions (descriptions) from the response, with confidence level
print("Description of local image: ")
if (len(description_result.captions) == 0):
    print("No description detected.")
else:
    for caption in description_result.captions:
        print("'{}' with confidence {:.2f}%".format(caption.text, caption.confidence * 100))
print()
'''
END - Describe an Image - local
'''

'''
Describe an Image - remote
This example describes the contents of an image with the confidence score.
'''
print("===== Describe an image - remote =====")
# Call API
description_results = computervision_client.describe_image(remote_image_url )

# Get the captions (descriptions) from the response, with confidence level
print("Description of remote image: ")
if (len(description_results.captions) == 0):
    print("No description detected.")
else:
    for caption in description_results.captions:
        print("'{}' with confidence {:.2f}%".format(caption.text, caption.confidence * 100))
# </snippet_describe>
print()
'''
END - Describe an Image - remote
'''
```

AIDI1006-describe-categorize

The code // Describe Method

Home Page - Select or create a n | wAIDI1006-describe-categorize

localhost:8888/notebooks/wAIDI1006-describe-categorize.ipynb

jupyter wAIDI1006-describe-categorize Last Checkpoint: 4 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

```
'''
Categorize an Image -  local
This example extracts categories from a local image with a confidence score
'''
print("===== Categorize an Image - local =====")
# Open local image file
local_image = open(local_image_path, "rb")
# Select visual feature type(s)
local_image_features = ["categories"]
# Call API
categorize_results_local = computervision_client.analyze_image_in_stream(local_image, local_image_features)

# Print category results with confidence score
print("Categories from local image: ")
if (len(categorize_results_local.categories) == 0):
    print("No categories detected.")
else:
    for category in categorize_results_local.categories:
        print("'{}' with confidence {:.2f}%".format(category.name, category.score * 100))
print()
'''
END - Categorize an Image - local
'''

'''
Categorize an Image - remote
This example extracts (general) categories from a remote image with a confidence score.
'''
print("===== Categorize an image - remote =====")
# Select the visual feature(s) you want.
remote_image_features = ["categories"]
# Call API with URL and features
categorize_results_remote = computervision_client.analyze_image(remote_image_url , remote_image_features)

# Print results with confidence score
print("Categories from remote image: ")
if (len(categorize_results_remote.categories) == 0):
    print("No categories detected.")
else:
    for category in categorize_results_remote.categories:
        print("'{}' with confidence {:.2f}%".format(category.name, category.score * 100))
print()
```

```
===== Describe an Image - local =====
Description of local image:
```

AIDI1006-describe-categorize

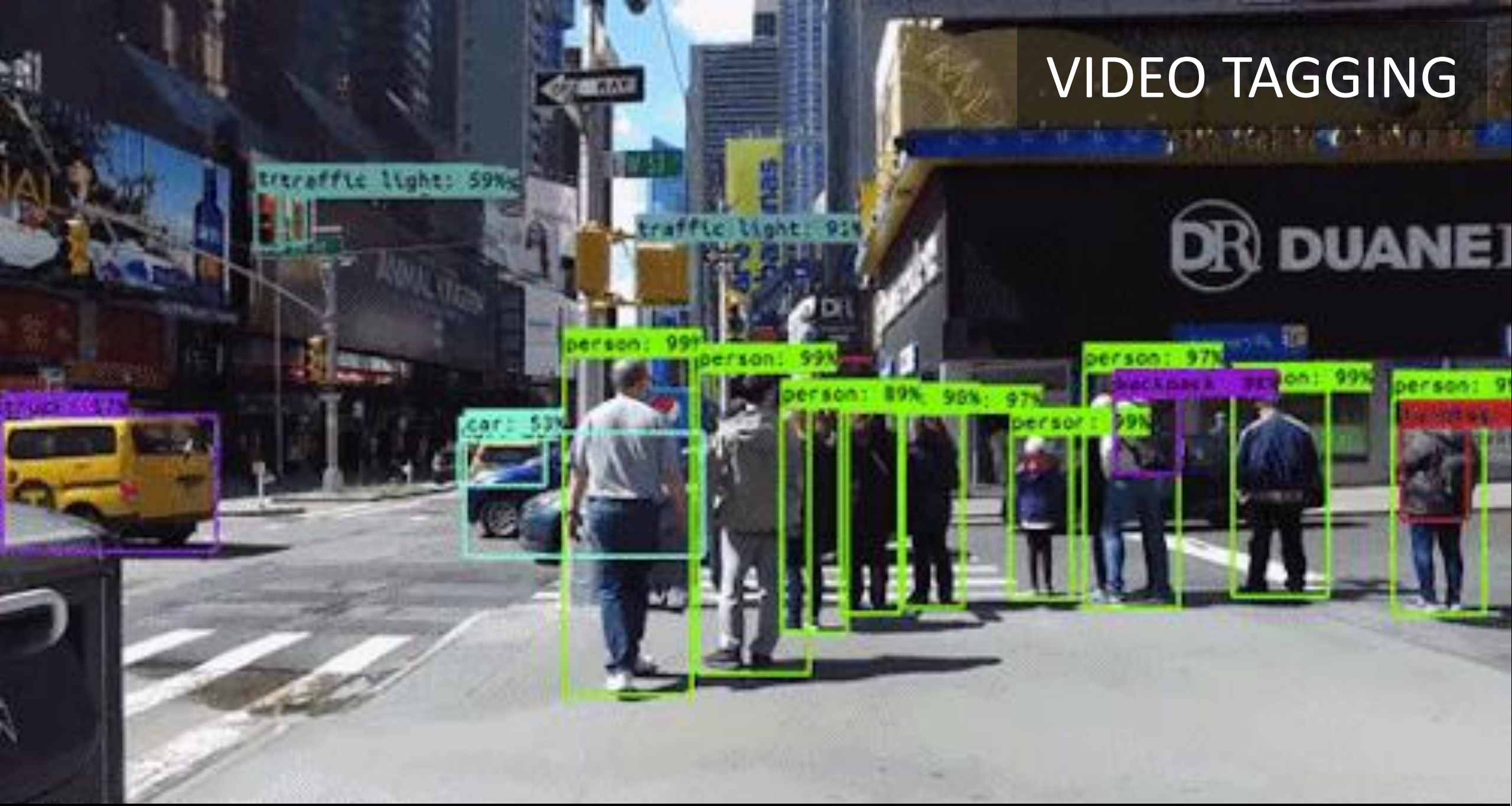
The code // Analyze Method

Your Turn!

Using cognitive services, write your code to analyse your preferred local file (image) and an URL image.

Take a screenshot of your result and add the local image and the URL image.

VIDEO TAGGING



Example

Computer Vision // Image Tagging // REMOTE

Computer vision returns tags based on thousands of recognizable objects, living beings, scenery and actions.

After uploading an image or specifying an image URL, Computer Vision algorithms output tags based on the objects, living beings, and actions identified in the image.

There are several different ways to write your code. This is just one of them.

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from msrest.authentication import CognitiveServicesCredentials
# Authenticate // Authenticates your credentials and creates a client.
subscription_key = "66e[redacted]8a72"
endpoint = "https://[redacted]itiveservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

remote_image_url = "https://health.gov/sites/default/files/styles/max_650x650/public/2020-12/HP2030-LHI.jpg?itok=OrfOq4kM"

# Tag an Image - remote // This example returns a tag (key word) for each thing in the image.
print("===== Tag an image - remote =====")
# Call API with remote image
tags_result_remote = computervision_client.tag_image(remote_image_url)
# Print results with confidence score
print("Tags in the remote image: ")
```

```
C:\Users\caiog\PycharmProjects\image-tagging\venv\Scripts\python.exe C:/Users/caiog/PycharmProjects/image-tagging/main.py
===== Tag an image - remote =====
Tags in the remote image:
'clothing' with confidence 99.95%
'person' with confidence 99.90%
'human face' with confidence 99.74%
'outdoor' with confidence 99.47%
'smile' with confidence 90.81%
'people' with confidence 86.09%
'mammal' with confidence 84.10%
'woman' with confidence 74.32%
'standing' with confidence 72.33%
'street' with confidence 58.61%

Process finished with exit code 0
```

AIDI1006 - image-tagging

https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/quickstarts-sdk/image-analysis-client-library?tabs=visual-studio&pivots=programming-language-python

===== Tag an image - remote =====

Tags in the remote image:

'clothing' with confidence 99.95%

'person' with confidence 99.90%

'human face' with confidence 99.74%

'outdoor' with confidence 99.47%

'smile' with confidence 90.81%

'people' with confidence 86.09%

'mammal' with confidence 84.10%

'woman' with confidence 74.32%

'standing' with confidence 72.33%

'street' with confidence 58.61%

AIDI1006 - image-tagging.ipynb

Result // Output

AIDI1006 - image-tagging - Jupy

localhost:8888/notebooks/AIDI1006%20-%20image-tagging.ipynb

There are several different ways to write your code. This is just one of them.

jupyter AIDI1006 - image-tagging Last Checkpoint: Last Wednesday at 3:39 PM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Code

In [1]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from msrest.authentication import CognitiveServicesCredentials
# Authenticate // Authenticates your credentials and creates a client.
subscription_key = "66[redacted]8a72"
endpoint = "https://[redacted]itiveservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))
remote_image_url = "https://health.gov/sites/default/files/styles/max_650x650/public/2020-12/HP2030-LHI.jpg?itok=OrfQg4kM"
# Tag an Image - remote // This example returns a tag (key word) for each thing in the image.
print("===== Tag an image - remote =====")
# Call API with remote image
tags_result_remote = computervision_client.tag_image(remote_image_url)
# Print results with confidence score
print("Tags in the remote image: ")
if (len(tags_result_remote.tags) == 0):
    print("No tags detected.")
else:
    for tag in tags_result_remote.tags:
        print("'{}' with confidence {:.2f}%".format(tag.name, tag.confidence * 100))
```

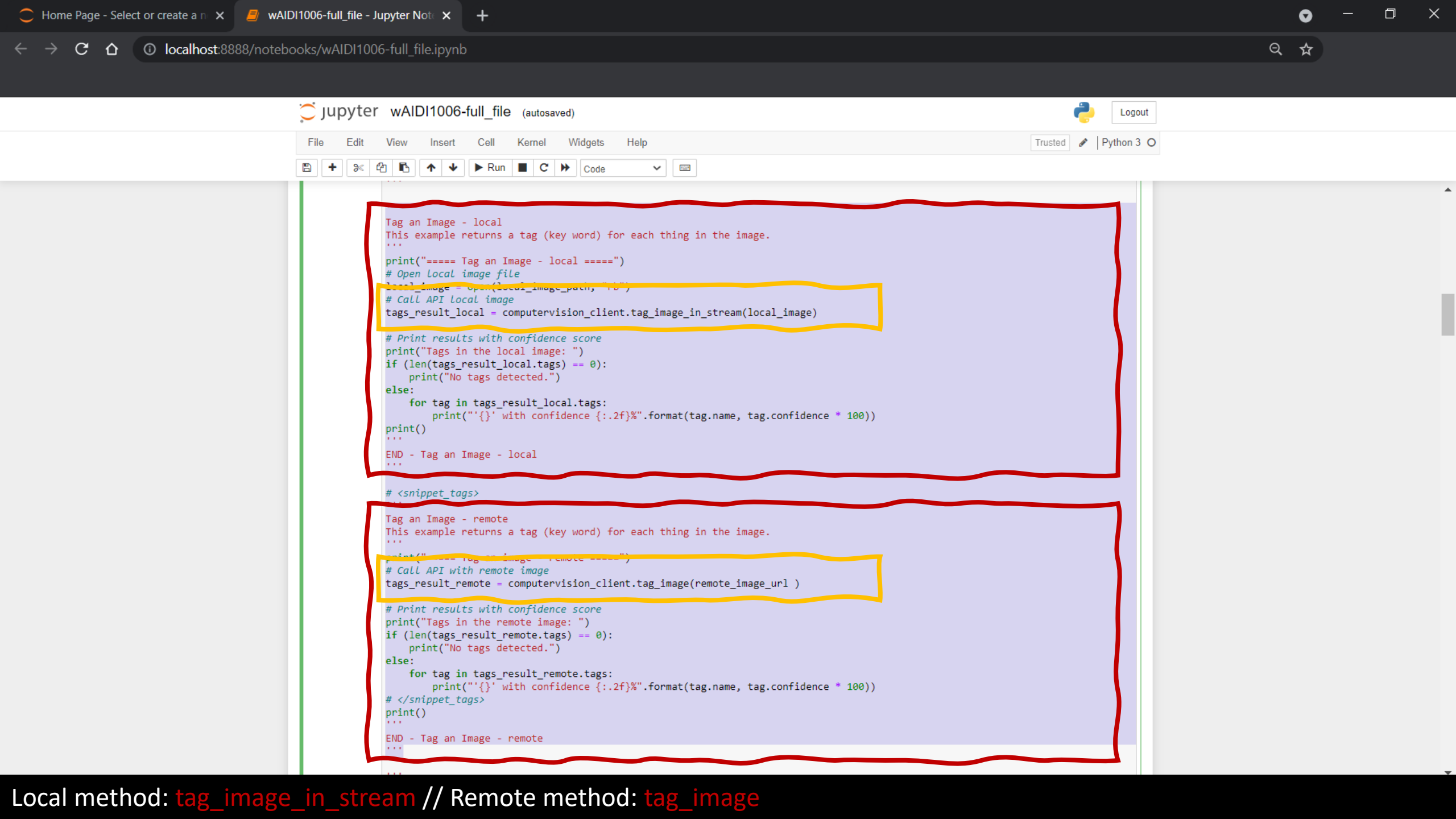
```
===== Tag an image - remote =====
Tags in the remote image:
'clothing' with confidence 99.95%
'person' with confidence 99.90%
'human face' with confidence 99.74%
'outdoor' with confidence 99.47%
'smile' with confidence 90.81%
'people' with confidence 86.09%
'mammal' with confidence 84.10%
'woman' with confidence 74.32%
'standing' with confidence 72.33%
'street' with confidence 58.61%
```

https://health.gov/sites/default/files/styles/max_650x650/public/2020-12/HP2030-LHI.jpg?itok=OrfQg4kM

Computer Vision // Image Tagging // LOCAL

Computer vision returns tags based on thousands of recognizable objects, living beings, scenery and actions.

After uploading an image or specifying an image URL, Computer Vision algorithms output tags based on the objects, living beings, and actions identified in the image.


Home Page - Select or create a n
wAIDI1006-full_file - Jupyter Not
localhost:8888/notebooks/wAIDI1006-full_file.ipynb
jupyter wAIDI1006-full_file (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Tag an Image - local
This example returns a tag (key word) for each thing in the image.
'''
print("===== Tag an Image - local =====")
# Open local image file
# Call API local image
tags_result_local = computervision_client.tag_image_in_stream(local_image)
# Print results with confidence score
print("Tags in the local image: ")
if (len(tags_result_local.tags) == 0):
print("No tags detected.")
else:
for tag in tags_result_local.tags:
print("'{}' with confidence {:.2f}%".format(tag.name, tag.confidence * 100))
print()
'''
END - Tag an Image - local
'''
<snippet_tags>
Tag an Image - remote
This example returns a tag (key word) for each thing in the image.
'''
# Call API with remote image
tags_result_remote = computervision_client.tag_image(remote_image_url)
# Print results with confidence score
print("Tags in the remote image: ")
if (len(tags_result_remote.tags) == 0):
print("No tags detected.")
else:
for tag in tags_result_remote.tags:
print("'{}' with confidence {:.2f}%".format(tag.name, tag.confidence * 100))
# </snippet_tags>
print()
'''
END - Tag an Image - remote
'''
Local method: tag_image_in_stream // Remote method: tag_image

Your Turn!

Using cognitive services, write your code to analyse the following URL and the following local file. URL:

https://wp.en.aleteia.org/wp-content/uploads/sites/2/2020/05/web3-family-large-big-home-brother-sister-mother-father-shutterstock_1197877477.jpg?w=640&crop=1

Local: C:\images\happy-family.jpg

AIDI1006 - image-tagging - Jupy

localhost:8888/notebooks/AIDI1006%20-%20image-tagging.ipynb

There are several different ways to write your code. This is just one of them.

jupyter AIDI1006 - image-tagging Last Checkpoint: Last Wednesday at 3:39 PM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Code

In [1]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from msrest.authentication import CognitiveServicesCredentials
# Authenticate // Authenticates your credentials and creates a client.
subscription_key = "66[redacted]8a72"
endpoint = "https://[redacted]itiveservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))
remote_image_url = "https://health.gov/sites/default/files/styles/max_650x650/public/2020-12/HP2030-LHI.jpg?itok=OrfQg4kM"
# Tag an Image - remote // This example returns a tag (key word) for each thing in the image.
print("===== Tag an image - remote =====")
# Call API with remote image
tags_result_remote = computervision_client.tag_image(remote_image_url)
# Print results with confidence score
print("Tags in the remote image: ")
if (len(tags_result_remote.tags) == 0):
    print("No tags detected.")
else:
    for tag in tags_result_remote.tags:
        print("'{}' with confidence {:.2f}%".format(tag.name, tag.confidence * 100))
```

```
===== Tag an image - remote =====
Tags in the remote image:
'clothing' with confidence 99.95%
'person' with confidence 99.90%
'human face' with confidence 99.74%
'outdoor' with confidence 99.47%
'smile' with confidence 90.81%
'people' with confidence 86.09%
'mammal' with confidence 84.10%
'woman' with confidence 74.32%
'standing' with confidence 72.33%
'street' with confidence 58.61%
```

Results // REMOTE

Computer Vision // Domain Specific Models

Moderate Content

Detect Adult or Racy Content

Home Page - Select or create a n | wAIDI1006-moderate-content - J

localhost:8888/notebooks/wAIDI1006-moderate-content.ipynb

jupyter wAIDI1006-moderate-content Last Checkpoint: Last Thursday at 10:41 AM (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

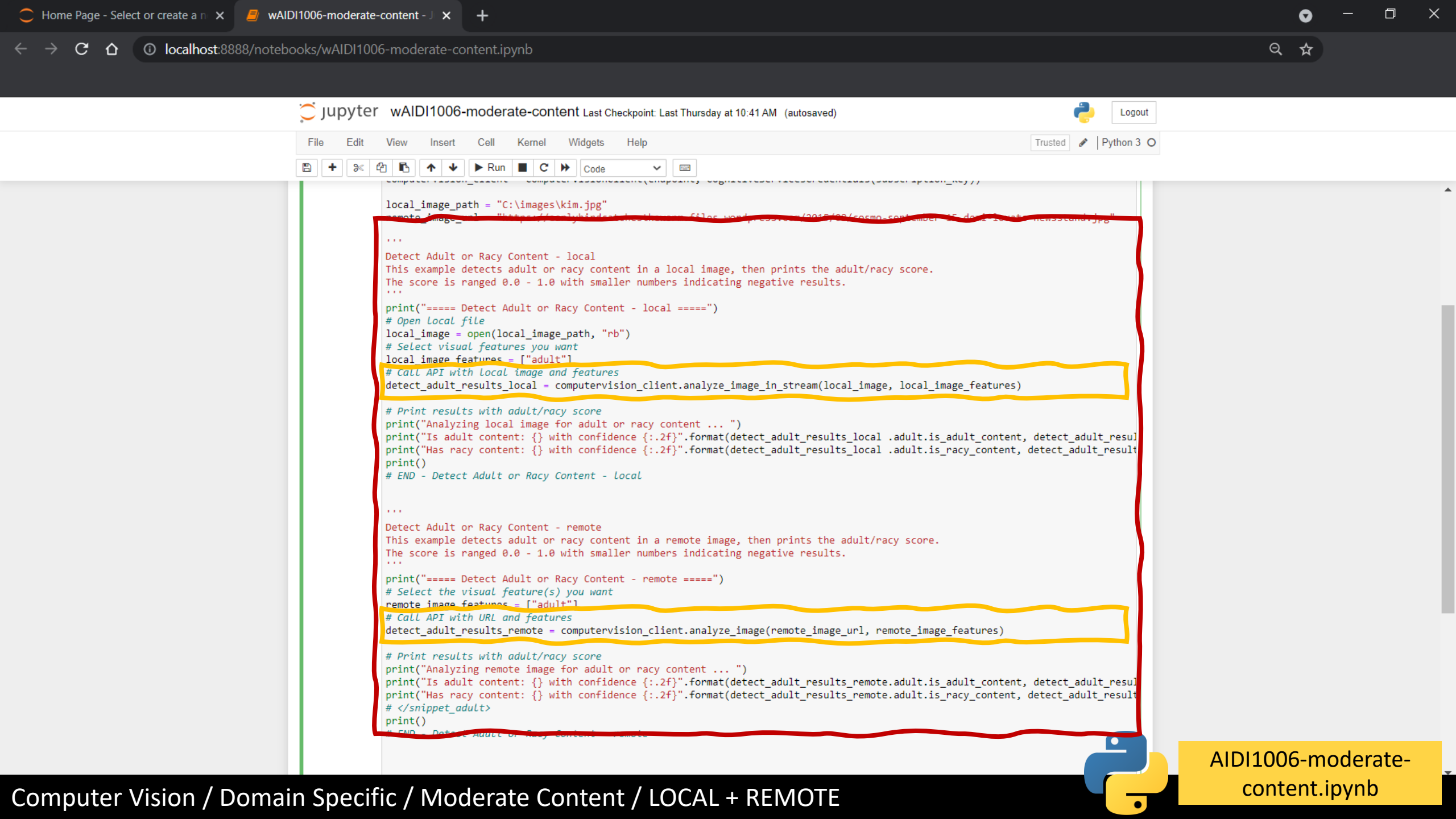
Run | Code

In [1]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from azure.cognitiveservices.vision.computervision.models import OperationStatusCodes
from azure.cognitiveservices.vision.computervision.models import VisualFeatureTypes
from msrest.authentication import CognitiveServicesCredentials

from array import array
import os
from PIL import Image
import sys
import time

subscription_key = "6[redacted]72"
endpoint = "https://c[redacted]services.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

local_image_path = "C:\images\kim.jpg"
remote_image_url = "https://earlybirdcatchestheworm.files.wordpress.com/2015/08/cosmo-september-15-demi-lovato-newsstand.jpg"
'''
Detect Adult or Racy Content - local
This example detects adult or racy content in a local image, then prints the adult/racy score.
The score is ranged 0.0 - 1.0 with smaller numbers indicating negative results.
'''
print("===== Detect Adult or Racy Content - local =====")
# Open local file
local_image = open(local_image_path, "rb")
# Select visual features you want
local_image_features = ["adult"]
# Call API with local image and features
detect_adult_results_local = computervision_client.analyze_image_in_stream(local_image, local_image_features)

# Print results with adult/racy score
print("Analyzing local image for adult or racy content ... ")
print("Is adult content: {} with confidence {:.2f}".format(detect_adult_results_local .adult.is_adult_content, detect_adult_resul
print("Has racy content: {} with confidence {:.2f}".format(detect_adult_results_local .adult.is_racy_content, detect_adult_result
print()
# END - Detect Adult or Racy Content - local


'''
Detect Adult or Racy Content - remote
This example detects adult or racy content in a remote image, then prints the adult/racy score.
The score is ranged 0.0 - 1.0 with smaller numbers indicating negative results.
```

AIDI1006-moderate-content.ipynb

Computer Vision / Domain Specific / Moderate Content / The API


Home Page - Select or create a n
wAIDI1006-moderate-content - J
localhost:8888/notebooks/wAIDI1006-moderate-content.ipynb
jupyter wAIDI1006-moderate-content Last Checkpoint: Last Thursday at 10:41 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
local_image_path = "C:\images\kim.jpg"
'''
Detect Adult or Racy Content - local
This example detects adult or racy content in a local image, then prints the adult/racy score.
The score is ranged 0.0 - 1.0 with smaller numbers indicating negative results.
'''
print("===== Detect Adult or Racy Content - local =====")
# Open local file
local_image = open(local_image_path, "rb")
# Select visual features you want
local_image_features = ["adult"]
# Call API with local image and features
detect_adult_results_local = computervision_client.analyze_image_in_stream(local_image, local_image_features)
# Print results with adult/racy score
print("Analyzing local image for adult or racy content ... ")
print("Is adult content: {} with confidence {:.2f}".format(detect_adult_results_local .adult.is_adult_content, detect_adult_resul
print("Has racy content: {} with confidence {:.2f}".format(detect_adult_results_local .adult.is_racy_content, detect_adult_result
print()
# END - Detect Adult or Racy Content - local
'''
Detect Adult or Racy Content - remote
This example detects adult or racy content in a remote image, then prints the adult/racy score.
The score is ranged 0.0 - 1.0 with smaller numbers indicating negative results.
'''
print("===== Detect Adult or Racy Content - remote =====")
# Select the visual feature(s) you want
remote_image_features = ["adult"]
# Call API with URL and features
detect_adult_results_remote = computervision_client.analyze_image(remote_image_url, remote_image_features)
# Print results with adult/racy score
print("Analyzing remote image for adult or racy content ... ")
print("Is adult content: {} with confidence {:.2f}".format(detect_adult_results_remote.adult.is_adult_content, detect_adult_resul
print("Has racy content: {} with confidence {:.2f}".format(detect_adult_results_remote.adult.is_racy_content, detect_adult_result
# </snippet_adult>
print()
AIDI1006-moderate-content.ipynb
Computer Vision / Domain Specific / Moderate Content / LOCAL + REMOTE

Home Page - Select or create a n | AIDI1006-brands - Jupyter Noteb | Home Page - Select or create a n | AIDI1006-moderate-content - Ju

localhost:8888/notebooks/AIDI1006-moderate-content.ipynb

jupyter AIDI1006-moderate-content Last Checkpoint: a few seconds ago (autosaved)

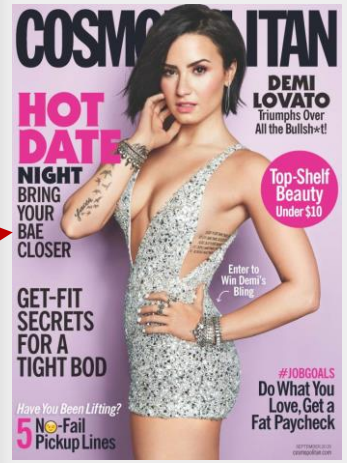
Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
'''
Detect Adult or Racy Content - remote
This example detects adult or racy content in a remote image, then prints the adult/racy score.
The score is ranged 0.0 - 1.0 with smaller numbers indicating negative results.
'''
print("===== Detect Adult or Racy Content - remote =====")
# Select the visual feature(s) you want
remote_image_features = ["adult"]
# Call API with URL and features
detect_adult_results_remote = computervision_client.analyze_image(remote_image_url, remote_image_features)

# Print results with adult/racy score
print("Analyzing remote image for adult or racy content ... ")
print("Is adult content: {} with confidence {:.2f}".format(detect_adult_results_remote.adult.is_adult_content, detect_adult_resul
print("Has racy content: {} with confidence {:.2f}".format(detect_adult_results_remote.adult.is_racy_content, detect_adult_result
# </snippet_adult>
print()
# END - Detect Adult or Racy Content - remote
```




```
===== Detect Adult or Racy Content - local =====
Analyzing local image for adult or racy content ...
Is adult content: False with confidence 9.16
Has racy content: True with confidence 96.98

===== Detect Adult or Racy Content - remote =====
Analyzing remote image for adult or racy content ...
Is adult content: False with confidence 1.74
Has racy content: False with confidence 68.11
```

In [ ]:

AIDI1006-moderate-content.ipynb

Computer Vision / Domain Specific / Moderate Content / Results

Your Turn!

Using cognitive services, write your code to analyse your preferred local file (image) and an URL image.

Take a screenshot of your result and add the local image and the URL image.

Computer Vision // Domain Specific Models

Celebrities and Landmarks

AIDI1006-landmark-celeb.ipynb

Home Page - Select or create a n | wAIDI1006-landmark-celeb - Jup

localhost:8888/notebooks/wAIDI1006-landmark-celeb.ipynb

jupyter wAIDI1006-landmark-celeb Last Checkpoint: Last Thursday at 10:36 AM (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [1]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from azure.cognitiveservices.vision.computervision.models import OperationStatusCodes
from azure.cognitiveservices.vision.computervision.models import VisualFeatureTypes
from msrest.authentication import CognitiveServicesCredentials

from array import array
import os
from PIL import Image
import sys
import time

subscription_key = "66[redacted]8a72"
endpoint = "https://co[redacted]veservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

# Quickstart variables // These variables are shared by several examples
#images_folder = os.path.join (os.path.dirname(os.path.abspath(__file__)), "images")
images_folder = "C:\images"

local_landmark_path = "c:\images\landmark7.jpg"
local_celeb_path = "c:\images\celeb.jpg"
```

```
'''
Detect Domain-specific Content - local
This example detects celebrites and landmarks in local images.
'''
print("===== Detect Domain-specific Content - local =====")
# Open local image file containing a celebtriy
local_image = open(local_celeb_path, "rb")
# Call API with the type of content (celebrities) and local image
detect_domain_results_celebs_local = computervision_client.analyze_image_by_domain_in_stream("celebrities", local_image)

# Print which celebrities (if any) were detected
print("Celebrities in the local image:")
if len(detect_domain_results_celebs_local.result["celebrities"]) == 0:
    print("No celebrities detected.")
else:
    for celeb in detect_domain_results_celebs_local.result["celebrities"]:
```

AIDI1006-landmark-celeb.ipynb

Home Page - Select or create a n | AIDI1006 - FULL FILE - Jupyter N | AIDI1006 - image-tagging - Jupy

localhost:8888/notebooks/AIDI1006%20-%20FULL%20FILE.ipynb

jupyter AIDI1006 - FULL FILE Last Checkpoint: Last Friday at 5:18 PM (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
'''
'''
Detect Domain-specific Content - local
This example detects celebrites and landmarks in local images.
'''
print("===== Detect Domain-specific Content - local =====")
# Open local image file containing a celebtriy
local_image = open(local_image_path, "rb")
# Call API with the type of content (celebrities) and local image
detect_domain_results_celebs_local = computervision_client.analyze_image_by_domain_in_stream("celebrities", local_image)

# Print which celebrities (if any) were detected
print("Celebrities in the local image:")
if len(detect_domain_results_celebs_local.result["celebrities"]) == 0:
    print("No celebrities detected.")
else:
    for celeb in detect_domain_results_celebs_local.result["celebrities"]:
        print(celeb["name"])

# Open local image file containing a landmark
local_image_path_landmark = os.path.join (images_folder, "landmark.jpg")
local_image_landmark = open(local_image_path_landmark, "rb")
# Call API with type of content (landmark) and local image
detect_domain_results_landmark_local = computervision_client.analyze_image_by_domain_in_stream("landmarks", local_image_landmark)
print()

# Print results of landmark detected
print("Landmarks in the local image:")
if len(detect_domain_results_landmark_local.result["landmarks"]) == 0:
    print("No landmarks detected.")
else:
    for landmark in detect_domain_results_landmark_local.result["landmarks"]:
        print(landmark["name"])
print()
'''
END - Detect Domain-specific Content - local
'''

# <snippet_celebs>
'''
```

AIDI1006-landmark-celeb.ipynb

Computer Vision / Domain Specific Models / LOCAL

Home Page - Select or create a n | AIDI1006 - FULL FILE - Jupyter N | AIDI1006 - image-tagging - Jupy

localhost:8888/notebooks/AIDI1006%20-%20FULL%20FILE.ipynb

jupyter AIDI1006 - FULL FILE Last Checkpoint: Last Friday at 5:18 PM (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
'''

# <snippet_celebs>
'''
Detect Domain-specific Content - remote
This example detects celebrites and landmarks in remote images.
'''
print("===== Detect Domain-specific Content - remote =====")
# URL of one or more celebrities
remote_image_url_celebs = "https://raw.githubusercontent.com/Azure-Samples/cognitive-services-sample-data-files/master/ComputerVi
# Call API with content type (celebrities) and URL
detect_domain_results_celebs_remote = computervision_client.analyze_image_by_domain("celebrities", remote_image_url_celebs)

# Print detection results with name
print("Celebrities in the remote image:")
if len(detect_domain_results_celebs_remote.result["celebrities"]) == 0:
    print("No celebrities detected.")
else:
    for celeb in detect_domain_results_celebs_remote.result["celebrities"]:
        print(celeb["name"])
# </snippet_celebs>

# <snippet_landmarks>
# Call API with content type (landmarks) and URL
detect_domain_results_landmarks = computervision_client.analyze_image_by_domain("landmarks", remote_image_url)
print()

print("Landmarks in the remote image:")
if len(detect_domain_results_landmarks.result["landmarks"]) == 0:
    print("No landmarks detected.")
else:
    for landmark in detect_domain_results_landmarks.result["landmarks"]:
        print(landmark["name"])
# </snippet_landmarks>
print()
'''
END - Detect Domain-specific Content - remote
'''

'''
Detect Image Types - local
```

AIDI1006-landmark-celeb.ipynb

Computer Vision / Domain Specific Models / REMOTE

Home Page - Select or create a n | wAIDI1006-landmark-celeb - Jup

localhost:8888/notebooks/wAIDI1006-landmark-celeb.ipynb

jupyter wAIDI1006-landmark-celeb (unsaved changes)

Logout

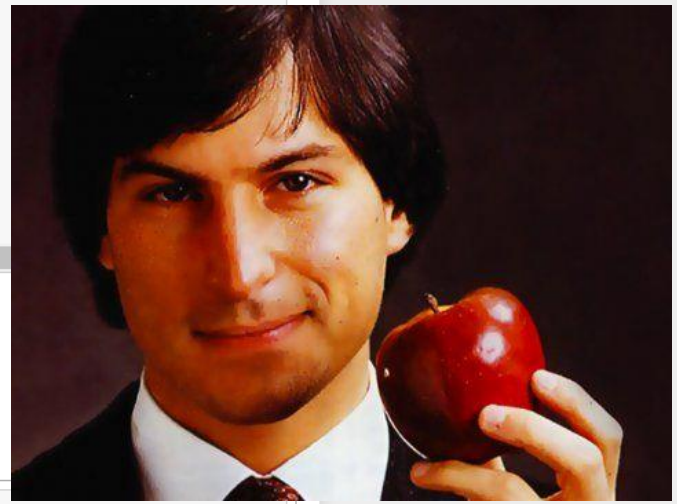
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Code

```
local_image = open(local_celeb_path, "rb")
# Call API with the type of content (celebrities) and local image
detect_domain_results_celebs_local = computervision_client.analyze_image_by_domain_in_stream("celebrities

# Print which celebrities (if any) were detected
print("Celebrities in the local image:")
if len(detect_domain_results_celebs_local.result["celebrities"]) == 0:
    print("No celebrities detected.")
else:
    for celeb in detect_domain_results_celebs_local.result["celebrities"]:
        print(celeb["name"])

# Open local image file containing a landmark
local_image_path_landmark = os.path.join (images_folder, "landmark2.jpeg")
local_image_landmark = open(local_landmark_path, "rb")
# Call API with type of content (landmark) and local image
detect_domain_results_landmark_local = computervision_client.analyze_image_by_domain_in_stream("landmarks", local_image_landmark)
print()

# Print results of landmark detected
print("Landmarks in the local image:")
if len(detect_domain_results_landmark_local.result["landmarks"]) == 0:
    print("No landmarks detected.")
else:
    for landmark in detect_domain_results_landmark_local.result["landmarks"]:
        print(landmark["name"])
print()
```

```
===== Detect Domain-specific Content - local =====
Celebrities in the local image:
Steve Jobs

Landmarks in the local image:
Taj Mahal
```



In [ ]:

AIDI1006-landmark-celeb.ipynb

Computer Vision / Domain Specific Models / REMOTE + LOCAL

Your Turn!

Using cognitive services, write your code to analyse your preferred local file (image) and an URL image.

Take a screenshot of your result and add the local image and the URL image.

Computer Vision // Domain Specific Models

Brands

AIDI1006-brands.ipynb

Home Page - Select or create a n | wAIDI1006-brands - Jupyter Not

localhost:8888/notebooks/wAIDI1006-brands.ipynb

jupyter wAIDI1006-brands (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

In [1]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from azure.cognitiveservices.vision.computervision.models import OperationStatusCodes
from azure.cognitiveservices.vision.computervision.models import VisualFeatureTypes
from msrest.authentication import CognitiveServicesCredentials

from array import array
import os
from PIL import Image
import sys
import time

subscription_key = "66e81584f4914a38a0f627b3ee888a72"
endpoint = "https://computer-vision-caio.cognitiveservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

# Quickstart variables // These variables are shared by several examples
#images_folder = os.path.join (os.path.dirname(os.path.abspath(__file__)), "images")
images_folder = "C:\images"
```

```
'''
Detect Brands - local
This example detects common brands like logos and puts a bounding box around them.
'''
print("===== Detect Brands - local =====")
# Open image file
local_image_path_shirt = os.path.join (images_folder, "gray-shirt-logo.jpg")
local_image_shirt = open(local_image_path_shirt, "rb")
# Select the visual feature(s) you want
local_image_features = ["brands"]
# Call API with image and features
detect_brands_results_local = computervision_client.analyze_image_in_stream(local_image_shirt, local_image_features)

# Print detection results with bounding box and confidence score
print("Detecting brands in local image: ")
if len(detect_brands_results_local.brands) == 0:
    print("No brands detected.")
else:
```

AIDI1006-brands.ipynb

Computer Vision / Domain Specific / Brands / The API

Home Page - Select or create a n | wAIDI1006-brands - Jupyter Not

localhost:8888/notebooks/wAIDI1006-brands.ipynb

jupyter wAIDI1006-brands (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

```
# Quickstart variables // These variables are shared by several examples
#images_folder = os.path.join (os.path.dirname(os.path.abspath(__file__)), "images")
images_folder = "C:\images"


'''
Detect Brands - local
This example detects common brands like logos and puts a bounding box around them.
'''
print("===== Detect Brands - local =====")
# Open image file
local_image_path_shirt = os.path.join (images_folder, "gray-shirt-logo.jpg")
local_image_shirt = open(local_image_path_shirt, "rb")
# Select the visual feature(s) you want
local_image_features = ["brands"]
# Call API with image and features
detect_brands_results_local = computervision_client.analyze_image_in_stream(local_image_shirt, local_image_features)

# Print detection results with bounding box and confidence score
print("Detecting brands in local image: ")
if len(detect_brands_results_local.brands) == 0:
    print("No brands detected.")
else:
    for brand in detect_brands_results_local.brands:
        print("'{}' brand detected with confidence {:.1f}% at location {}, {}, {}, {}".format( \
        brand.name, brand.confidence * 100, brand.rectangle.x, brand.rectangle.x + brand.rectangle.w, \
        brand.rectangle.y, brand.rectangle.y + brand.rectangle.h))
print()
'''
END - Detect brands - local
'''

# <snippet_brands>
'''
Detect Brands - remote
This example detects common brands like logos and puts a bounding box around them.
'''
print("===== Detect Brands - remote =====")
# Get a URL with a brand logo
remote_image_url = "https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/images/gray-shirt-l
```

AIDI1006-brands.ipynb

Computer Vision / Domain Specific / Brands / LOCAL

Home Page - Select or create a n | wAIDI1006-brands - Jupyter Not

localhost:8888/notebooks/wAIDI1006-brands.ipynb

jupyter wAIDI1006-brands (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

```
        brand.rectangle.y, brand.rectangle.y + brand.rectangle.h))
print()
'''
END - Detect brands - local
'''

# <snippet_brands>
'''
Detect Brands - remote
This example detects common brands like logos and puts a bounding box around them.
'''
print("===== Detect Brands - remote =====")
# Get a URL with a brand logo
remote_image_url = "https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/images/gray-shirt-logo.jpg"
# Select the visual feature(s) you want
remote_image_features = ["brands"]
# Call API with URL and features
detect_brands_results_remote = computervision_client.analyze_image(remote_image_url, remote_image_features)

print("Detecting brands in remote image: ")
if len(detect_brands_results_remote.brands) == 0:
    print("No brands detected.")
else:
    for brand in detect_brands_results_remote.brands:
        print("'{}' brand detected with confidence {:.1f}% at location {}, {}, {}, {}".format( \
        brand.name, brand.confidence * 100, brand.rectangle.x, brand.rectangle.x + brand.rectangle.w, \
        brand.rectangle.y, brand.rectangle.y + brand.rectangle.h))
# </snippet_brands>
print()
```

```
===== Detect Brands - local =====
Detecting brands in local image:
'Tommy Hilfiger' brand detected with confidence 69.1% at location 215, 254, 146, 172

===== Detect Brands - remote =====
Detecting brands in remote image:
'Microsoft' brand detected with confidence 62.5% at location 58, 113, 106, 152
'Microsoft' brand detected with confidence 69.8% at location 58, 260, 86, 149
```

AIDI1006-brands.ipynb

Computer Vision / Domain Specific / Brands / REMOTE

Home Page - Select or create a n | AIDI1006-brands - Jupyter Noteb | localhost:8888/notebooks/AIDI1006-brands.ipynb

jupyter AIDI1006-brands (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```
'''
END - Detect brands - local
'''

# <snippet_brands>
'''
Detect Brands - remote
This example detects common brands like logos and puts a bounding box around them.
'''
print("===== Detect Brands - remote =====")
# Get a URL with a brand logo
remote_image_url = "https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/images/gray-shirt-logo.jpg"
# Select the visual feature(s) you want
remote_image_features = ["brands"]
# Call API with URL and features
detect_brands_results_remote = computervision_client.analyze_image(remote_image_url, remote_image_features)

print("Detecting brands in remote image: ")
if len(detect_brands_results_remote.brands) == 0:
    print("No brands detected.")
else:
    for brand in detect_brands_results_remote.brands:
        print("'{}' brand detected with confidence {:.1f}% at location {}, {}, {}, {}".format( \
        brand.name, brand.confidence * 100, brand.rectangle.x, brand.rectangle.x + brand.rectangle.w,
        brand.rectangle.y, brand.rectangle.y + brand.rectangle.h))
# </snippet_brands>
print()
```

```
===== Detect Brands - local =====
Detecting brands in local image:
'Tommy Hilfiger' brand detected with confidence 69.1% at location 215, 254, 146, 172

===== Detect Brands - remote =====
Detecting brands in remote image:
'Microsoft' brand detected with confidence 62.5% at location 58, 113, 106, 152
'Microsoft' brand detected with confidence 69.8% at location 58, 260, 86, 149
```

In [ ]:

AIDI1006-brands.ipynb

Computer Vision / Domain Specific / Brands / LOCAL + REMOTE

Computer Vision

Domain Specific Models // LOCAL

Call API with image and features // LOCAL

```
# Open image file
local_image_path_shirt = os.path.join (images_folder, "gray-shirt-logo.jpg")
local_image_shirt = open(local_image_path_shirt, "rb")

# Select the visual feature(s) you want
local_image_features = ["brands"]

# Call API with image and features
detect_brands_results_local = computervision_client.analyze_image_in_stream(local_image_shirt, local_image_features)
```

AIDI1006-brands.ipynb

Computer Vision

Domain Specific Models // REMOTE

Call API with image and features // REMOTE

```
# Get a URL with a brand logo
remote_image_url = "https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/images/gray-shirt-logo.jpg"

# Select the visual feature(s) you want
remote_image_features = ["brands"]

# Call API with URL and features
detect_brands_results_remote = computervision_client.analyze_image(remote_image_url, remote_image_features)
```

AIDI1006-brands.ipynb

Your Turn!

Using cognitive services, write your code to analyse the following URL and the following local file:

(1) URL:

https://www.incimages.com/uploaded_files/image/1920x1080/GettyImages-71974463_431181.jpg

(2) Local: C:\images\brand2.jpg

EXPECTED RESULTS!

===== Detect Brands - local =====

Detecting brands in local image:

'Pepsi' brand detected with confidence 88.9% at location 773, 867, 284, 443

'Pepsi' brand detected with confidence 69.8% at location 1026, 1165, 461, 518

'Pepsi' brand detected with confidence 88.3% at location 1008, 1177, 268, 441

'Coca-Cola' brand detected with confidence 77.7% at location 210, 501, 302, 507

'Pepsi' brand detected with confidence 88.5% at location 593, 763, 284, 482

===== Detect Brands - remote =====

Detecting brands in remote image:

'Starbucks' brand detected with confidence 78.2% at location 1470, 1818, 3, 278

'Starbucks' brand detected with confidence 94.1% at location 427, 656, 576, 832

Computer Vision

Thumbnails

Computer Vision // Thumbnails

A thumbnail is a miniature representation of a page or image that is used to identify a file by its contents. Thumbnails are an option in file managers, such as Windows Explorer, and they are found in photo editing and graphics program to quickly browse multiple images in a folder.

https://docs.microsoft.com/en-us/rest/api/computervision/3.1/generate-thumbnail

Home Page - Select or create a n | wAIDI1006-thumbnails - Jupyter

localhost:8888/notebooks/wAIDI1006-thumbnails.ipynb

jupyter wAIDI1006-thumbnails Last Checkpoint: Last Thursday at 11:16 AM (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

In [17]:

```
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from azure.cognitiveservices.vision.computervision.models import OperationStatusCodes
from azure.cognitiveservices.vision.computervision.models import VisualFeatureTypes
from msrest.authentication import CognitiveServicesCredentials

from array import array
import os
from PIL import Image
import sys
import time

subscription_key = "66e81584f4914a38a0f627b3ee888a72"
endpoint = "https://computer-vision-caio.cognitiveservices.azure.com/"
computervision_client = ComputerVisionClient(endpoint, CognitiveServicesCredentials(subscription_key))

# Quickstart variables // These variables are shared by several examples
#images_folder = os.path.join (os.path.dirname(os.path.abspath(__file__)), "images")
images_folder = "C:\images"
local_image_path_objects = "C:\images\objects.jpg"
```

```
# Generate Thumbnail - This example creates a thumbnail from both a local and URL image.

print("===== Generate Thumbnail =====")

# Generate a thumbnail from a local image
local_image_path_thumb = os.path.join (images_folder, "objects.jpg")
local_image_thumb = open(local_image_path_objects, "rb")

print("Generating thumbnail from a local image...")
# Call the API with a local image, set the width/height if desired (pixels)
# Returns a Generator object, a thumbnail image binary (list).
thumb_local = computervision_client.generate_thumbnail_in_stream(100, 100, local_image_thumb, True)

# Write the image binary to file
with open("thumb_local.png", "wb") as f:
    for chunk in thumb_local:
        f.write(chunk)
```

AIDI1006-moderate-content.ipynb

Computer Vision / Thumbnails / The API

Home Page - Select or create a n | wAIDI1006-thumbnails - Jupyter

localhost:8888/notebooks/wAIDI1006-thumbnails.ipynb

jupyter wAIDI1006-thumbnails Last Checkpoint: Last Thursday at 11:16 AM (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
print("===== Generate Thumbnail =====")

# Generate a thumbnail from a local image
local_image_path_thumb = os.path.join (images_folder, "objects.jpg")
local_image_thumb = open(local_image_path_objects, "rb")

print("Generating thumbnail from a local image...")
# Call the API with a local image, set the width/height if desired (pixels)
# Returns a Generator object, a thumbnail image binary (list).
thumb_local = computervision_client.generate_thumbnail_in_stream(100, 100, local_image_thumb, True)

# Write the image binary to file
with open("thumb_local.png", "wb") as f:
    for chunk in thumb_local:
        f.write(chunk)

# Uncomment/use this if you are writing many images as thumbnails from a list
for i, image in enumerate(thumb_local, start=0):
    with open('thumb_{0}.jpg'.format(i), 'wb') as f:
        f.write(image)

print("Thumbnail saved to local folder.")
print()

# Generate a thumbnail from a URL image
# URL of faces
remote_image_url_thumb = "https://raw.githubusercontent.com/Azure-Samples/cognitive-services-sample-data-files/master/ComputerVi

print("Generating thumbnail from a URL image...")
# Returns a Generator object, a thumbnail image binary (list).
thumb_remote = computervision_client.generate_thumbnail(
    100, 100, remote_image_url_thumb, True)

# Write the image binary to file
with open("thumb_remote.png", "wb") as f:
    for chunk in thumb_remote:
        f.write(chunk)

print("Thumbnail saved to local folder.")

# Uncomment/use this if you are writing many images as thumbnails from a list
```

AIDI1006-moderate-content.ipynb

Computer Vision / Thumbnails / LOCAL + REMOTE

Home Page - Select or create a n | wAIDI1006-thumbnails - Jupyter

localhost:8888/notebooks/wAIDI1006-thumbnails.ipynb

jupyter wAIDI1006-thumbnails Last Checkpoint: a minute ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
print("Thumbnail saved to local folder.")
print()

# Generate a thumbnail from a URL image
# URL of faces
remote_image_url_thumb = "https://raw.githubusercontent.com/Azure-Samples/cognitive-services-sample-data-files/master/Co

print("Generating thumbnail from a URL image...")
# Returns a Generator object, a thumbnail image binary (list).
thumb_remote = computervision_client.generate_thumbnail(
    100, 100, remote_image_url_thumb, True)

# Write the image binary to file
with open("thumb_remote.png", "wb") as f:
    for chunk in thumb_remote:
        f.write(chunk)

print("Thumbnail saved to local folder.")

# Uncomment/use this if you are writing many images as thumbnails from a list
for i, image in enumerate(thumb_remote, start=0):
    with open('thumb_{0}.jpg'.format(i), 'wb') as f:
        f.write(image)

print()
```

```
===== Generate Thumbnail =====
Generating thumbnail from a local image...
Thumbnail saved to local folder.

Generating thumbnail from a URL image...
Thumbnail saved to local folder.
```

In [ ]:

AIDI1006-moderate-content.ipynb

Computer Vision / Thumbnails / LOCAL + REMOTE

Your Turn!

Using cognitive services, write your code to analyse your preferred local file (image) and an URL image and generate your thumbnails.

Take a screenshot of your result and add the local image and the URL image.

Computer Vision

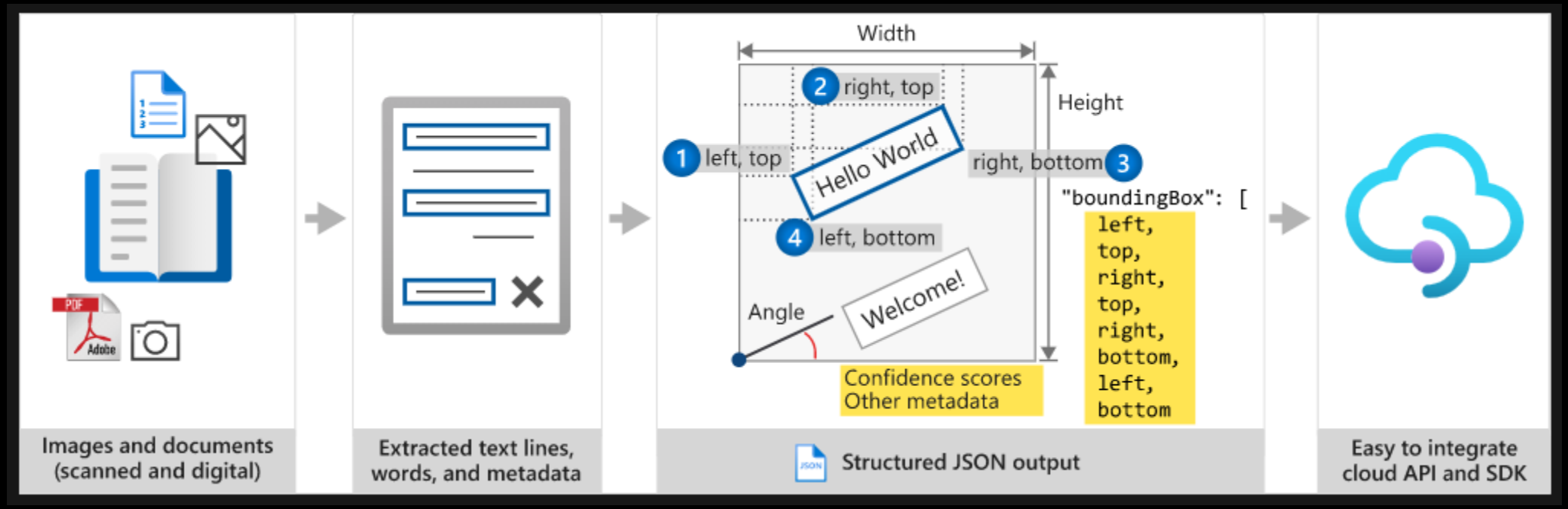
Text Detection

Computer Vision // Text Detection

Optical Character Recognition (OCR) allows you to extract printed or handwritten **text** from images, such as photos of street signs and products, as well as from documents – invoices, bills, financial reports, articles, and more. Microsoft's OCR technologies support extracting **printed text** in several languages.

https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr

Computer Vision // Text Detection



https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr

AIDI1006-ocr - Jupyter Notebook

localhost:8888/notebooks/AIDI1006-ocr.ipynb

jupyter AIDI1006-ocr Last Checkpoint: 4 minutes ago (autosaved)

Logout

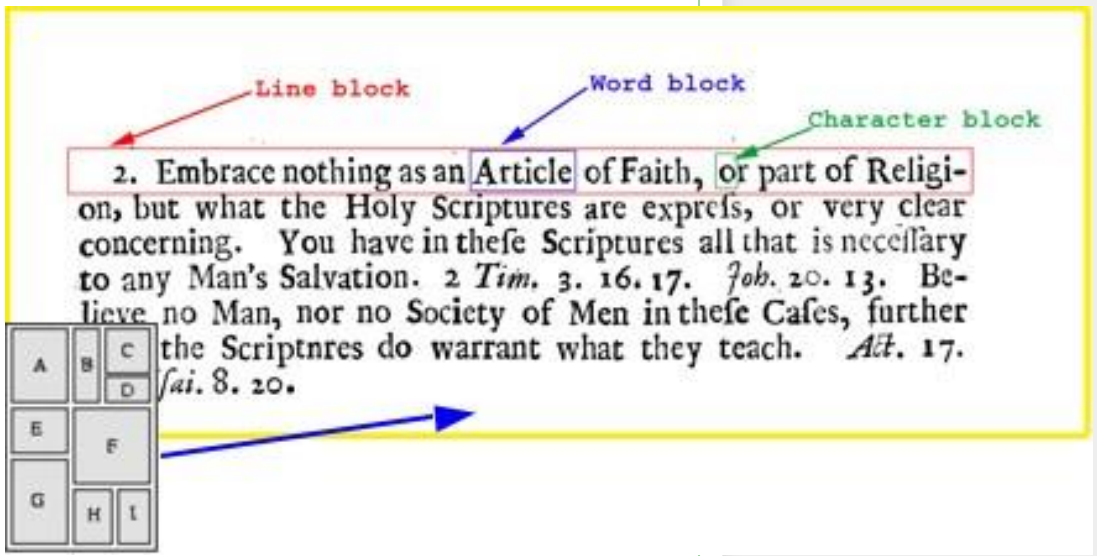
File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

```
if read_result.status == OperationStatusCodes.succeeded:
    for text_result in read_result.analyze_result.read_results:
        for line in text_result.lines:
            print(line.text)
            print(line.bounding_box)
print()
```

```
===== Read File - remote =====
Line block
[99.0, 28.0, 165.0, 27.0, 165.0, 38.0, 99.0, 38.0]
Word block
[235.0, 25.0, 301.0, 25.0, 301.0, 36.0, 235.0, 36.0]
Character block
[323.0, 40.0, 421.0, 40.0, 421.0, 51.0, 323.0, 51.0]
2. Embrace nothing as an Article of Faith, or part of Religi-
[40.0, 59.0, 389.0, 59.0, 389.0, 74.0, 40.0, 75.0]
on, but what the Holy Scriptures are expreis, or very clear
[28.0, 75.0, 390.0, 75.0, 390.0, 89.0, 28.0, 89.0]
concerning. You have in thefe Scriptures all that is neceflary
[27.0, 89.0, 390.0, 88.0, 390.0, 102.0, 27.0, 103.0]
to any Man's Salvation. 2 Tim. 3. 16. 17. foh. 20. 13. Be-
[29.0, 103.0, 390.0, 102.0, 390.0, 116.0, 29.0, 117.0]
lieve no Man, nor no Society of Men in thefe Cafes, further
[28.0, 117.0, 390.0, 116.0, 390.0, 130.0, 28.0, 131.0]
the Scriptures do warrant what they teach. Alt. 17.
[57.0, 131.0, 388.0, 130.0, 388.0, 145.0, 57.0, 146.0]
A
[11.0, 141.0, 18.0, 142.0, 17.0, 149.0, 10.0, 148.0]
c
[45.0, 134.0, 56.0, 134.0, 55.0, 144.0, 44.0, 143.0]
D
[44.0, 151.0, 55.0, 151.0, 55.0, 159.0, 44.0, 159.0]
Vai. 8. 20.
[56.0, 145.0, 118.0, 145.0, 118.0, 159.0, 56.0, 159.0]
E
[9.0, 167.0, 16.0, 167.0, 16.0, 175.0, 9.0, 175.0]
H
[32.0, 202.0, 39.0, 202.0, 38.0, 209.0, 32.0, 209.0]
```



AIDI1006-ocr.ipynb

Computer Vision / OCR

Your Turn!

Using cognitive services, write your code to analyse the following URL:

https://media.wired.com/photos/59327d3b44db296121d6b881/master/w_1600%2Cc_limit/bond_0011_Layer-5.jpg

EXPECTED RESULTS!

===== Read File - remote =====

BEN MILLER

[416.0, 42.0, 583.0, 43.0, 582.0, 68.0, 416.0, 68.0]

Dear sam.

[87.0, 164.0, 291.0, 170.0, 290.0, 217.0, 86.0, 211.0]

Welcome onboard. I'm very excited to have you

[122.0, 253.0, 897.0, 253.0, 897.0, 301.0, 122.0, 302.0]

on our team for this project. your skills will

[120.0, 315.0, 819.0, 310.0, 819.0, 357.0, 120.0, 364.0]

certainly be a valued asset and I'm looking

[119.0, 372.0, 810.0, 370.0, 810.0, 415.0, 119.0, 416.0]

forward to seeing what you come up with

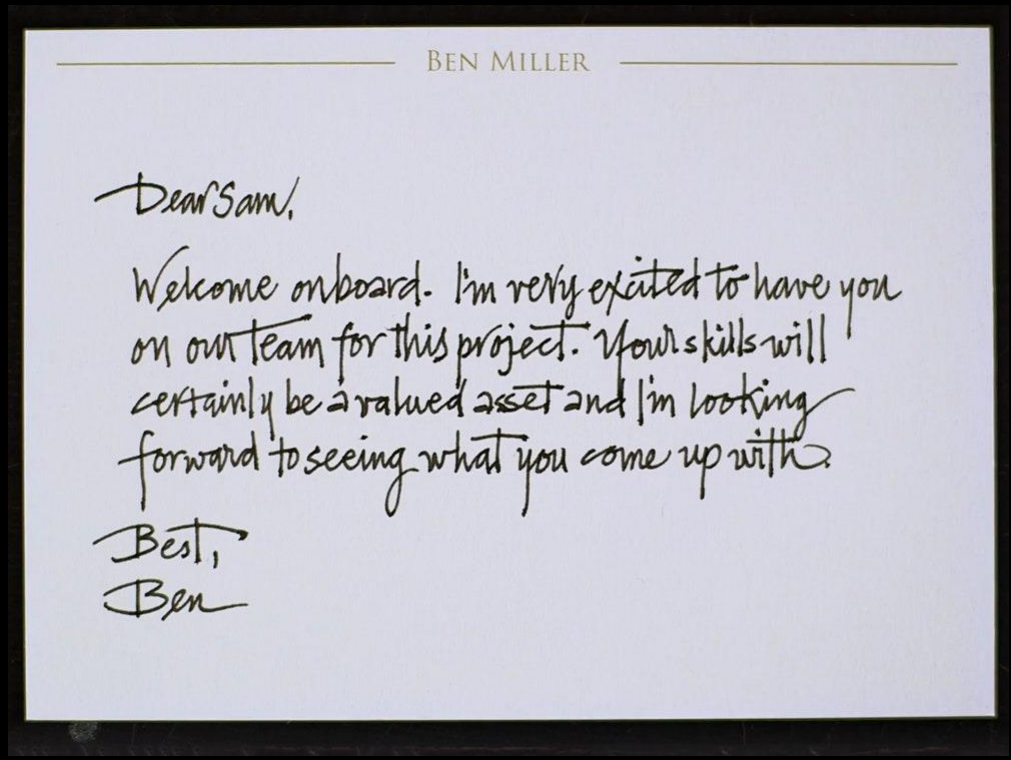
[111.0, 429.0, 825.0, 427.0, 825.0, 472.0, 111.0, 477.0]

Best,

[88.0, 516.0, 229.0, 516.0, 229.0, 559.0, 88.0, 558.0]

Ben

[102.0, 570.0, 235.0, 574.0, 234.0, 615.0, 100.0, 614.0]



Computer Vision // Face Detection

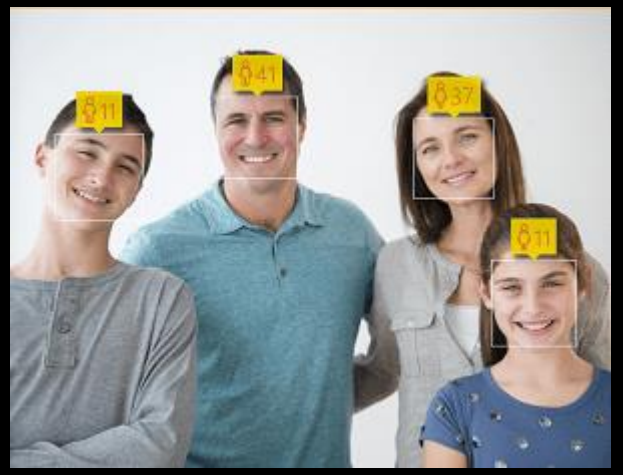
Computer Vision can detect human faces within an image and generate the age, gender, and rectangle for each detected face.

https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/concept-detecting-faces

Computer Vision // Face Detection

JSON


```
{
    "faces": [
        {
            "age": 23,
            "gender": "Female",
            "faceRectangle": {
                "top": 45,
                "left": 194,
                "width": 44,
                "height": 44
            }
        }
    ],
    "requestId": "8439ba87-de65-441b-a0f1-c85913157ecd",
    "metadata": {
        "height": 200,
        "width": 300,
        "format": "Png"
    }
}
```

JSON


```
{
    "faces": [
        {
            "age": 11,
            "gender": "Male",
            "faceRectangle": {
                "top": 62,
                "left": 22,
                "width": 45,
                "height": 45
            }
        },
        {
            "age": 11,
            "gender": "Female",
            "faceRectangle": {
                "top": 127,
                "left": 240,
                "width": 42,
                "height": 42
            }
        },
        {
            "age": 37,
            "gender": "Female",
            "faceRectangle": {
                "top": 55,
                "left": 200,
                "width": 41,
                "height": 41
            }
        },
        {
            "age": 41,
            "gender": "Male",
```


https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/concept-detecting-faces

File Edit View Navigate Code Refactor Run Tools VCS Window Help AIDI1006-faceatributes - main.py

AIDI1006-faceatributes > main.py

main

Project

AIDI1006-faceatributes C:\Users\caiog\PycharmProjects\AIDI

venv library root

main.py

External Libraries

Scratches and Consoles

main.py

```
# Create an authenticated FaceClient.
face_client = FaceClient(ENDPOINT, CognitiveServicesCredentials(KEY))

'''
Detect face(s) with attributes in a URL image
'''
# Image of face(s)
face1_url = 'https://www.cheapflights.co.uk/news/wp-content/uploads/sites/138/2016/03/11-people-you-ll-find-in-london-01-620x414.jpg'
face1_name = os.path.basename(face1_url)
face2_url = 'https://assets.londonist.com/uploads/2018/11/i875/img-20171028-wa0004.jpg'
face2_name = os.path.basename(face2_url)

# List of url images
url_images = [face1_url, face2_url]

# Attributes you want returned with the API call, a list of FaceAttributeType enum (string format)
face_attributes = ['age', 'gender', 'headPose', 'smile', 'facialHair', 'glasses', 'emotion']

# Detect a face with attributes, returns a list[DetectedFace]
for image in url_images:
    detected_faces = face_client.face.detect_with_url(url=image, return_face_attributes=face_attributes)
    if not detected_faces:
        raise Exception(
            'No face detected from image {}'.format(os.path.basename(image)))
```

if __name__ == '__main__'

Run: main

```
C:\Users\caiog\PycharmProjects\AIDI1006-faceatributes\venv\Scripts\python.exe C:/Users/caiog/PycharmProjects/AIDI1006-faceatributes/main.py

Detected face ID from 11-people-you-ll-find-in-london-01-620x414.jpg :
5713f667-f36d-4fec-9b75-1a8d29aa6aad

Facial attributes detected:
```

4: Run TODO 6: Problems Terminal Python Console

AIDI1006-faceatributes.py

Computer Vision / Face Detection / Code

Detected face ID from 11-people-you-ll-find-in-london-01-620x414.jpg : 805e497a-b6cb-43ab-b846-540814f956a1

Facial attributes detected:
Age: 28.0
Gender: Gender.female
Head pose: {'additional_properties': {}, 'roll': 9.7, 'yaw': -4.0, 'pitch': -6.3}
Smile: 1.0
Facial hair: {'additional_properties': {}, 'moustache': 0.0, 'beard': 0.0, 'sideburns': 0.0}
Glasses: GlassesType.no_glasses
Emotion:

- Anger: 0.0
- Contempt: 0.0
- Disgust: 0.0
- Fear: 0.0
- Happiness: 1.0
- Neutral: 0.0
- Sadness: 0.0
- Surprise: 0.0

Drawing rectangle around face... see popup for results.

AIDI1006-faceatributes.py

Detected face ID from img-20171028-wa0004.jpg : d507a419-8426-447b-bd57-b5400d3f3ede
Facial attributes detected:
Age: 27.0
Gender: Gender.male
Head pose: {'additional_properties': {}, 'roll': 0.1, 'yaw': 4.9, 'pitch': -0.1}
Smile: 1.0
Facial hair: {'additional_properties': {}, 'moustache': 0.1, 'beard': 0.1, 'sideburns': 0.1}
Glasses: GlassesType.no_glasses
Emotion:
Anger: 0.0
Contempt: 0.0
Disgust: 0.0
Fear: 0.0
Happiness: 1.0
Neutral: 0.0
Sadness: 0.0
Surprise: 0.0

Detected face ID from img-20171028-wa0004.jpg : 9d79f1dc-237b-4a33-864a-07945943cc1e
Facial attributes detected:
Age: 29.0
Gender: Gender.female
Head pose: {'additional_properties': {}, 'roll': 4.9, 'yaw': -20.7, 'pitch': -0.0}
Smile: 0.992
Facial hair: {'additional_properties': {}, 'moustache': 0.0, 'beard': 0.0, 'sideburns': 0.0}
Glasses: GlassesType.no_glasses
Emotion:
Anger: 0.0
Contempt: 0.0
Disgust: 0.0
Fear: 0.0
Happiness: 0.992
Neutral: 0.008
Sadness: 0.0
Surprise: 0.0

AIDI1006-faceatributes.py

Computer Vision // Available API's returns (1)

Describe an Image

- This example describes the contents of an image with the confidence score.

Categorize an Image

- This example extracts (general) categories from a remote image with a confidence score.

Tag an Image

- This example returns a tag (key word) for each thing in the image.

Detect Faces

- This example detects faces in a local image, gets their gender and age, and marks them with a bounding box.

Detect Adult or Racy Content

- This example detects adult or racy content in a local image, then prints the adult/racy score.

Detect Color

- This example detects the different aspects of its color scheme in a local image.

Computer Vision // Available API’s returns (2)

Detect Domain-specific Content

- This example detects celebrities and landmarks in local images.

Detect Image Types

- This example detects an image's type (clip art/line drawing).

Detect Objects

- This example detects different kinds of objects with bounding boxes in a local image.

Detect Brands

- This example detects common brands like logos and puts a bounding box around them.

Generate Thumbnail

- This example creates a thumbnail from both a local and URL image.

Challenge #4.10

Azure Cognitive Services

LUIS

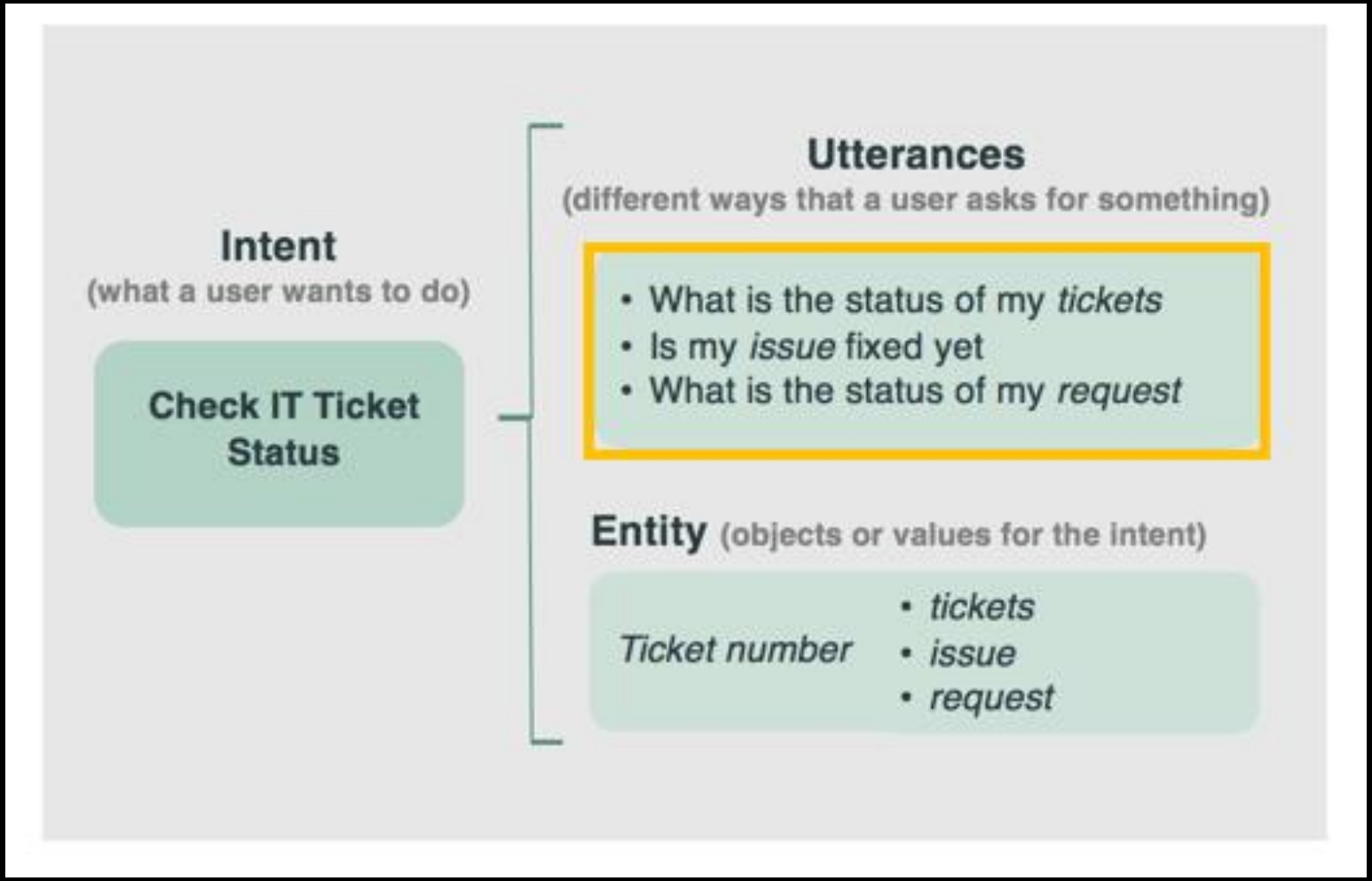
Language Understanding (LUIS)

Is a cloud-based conversational AI service that applies custom machine-learning intelligence to a users' conversational, natural language text to predict overall meaning, and pull out relevant, detailed information.

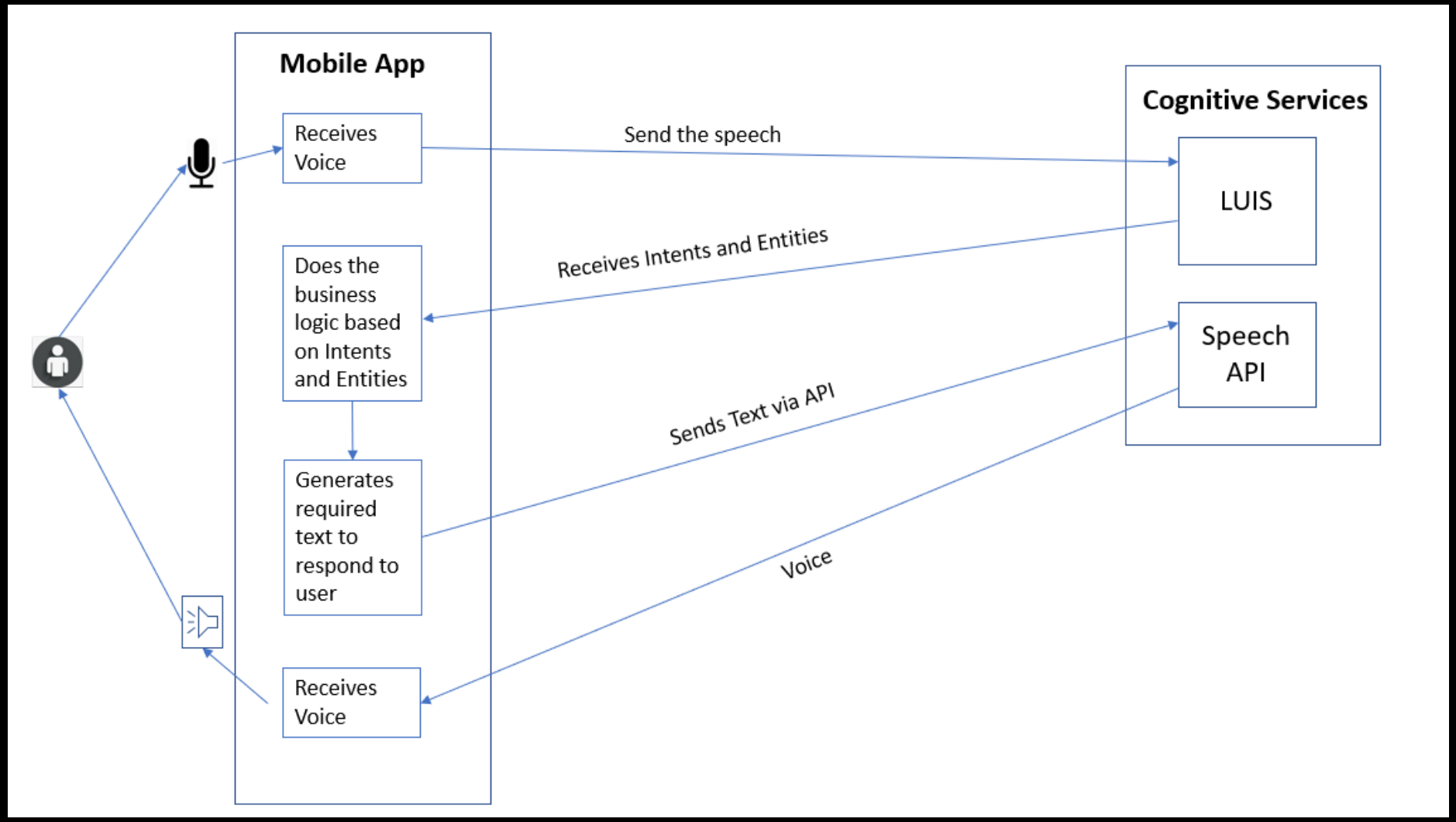
What is NLP?

- Text analysis and entity recognition
- Sentiment analysis
- Speech recognition and synthesis
- Machine translation
- Semantic language modeling

Utterance – Entity – Intent



Possible Project



LUIS (Language Understanding)

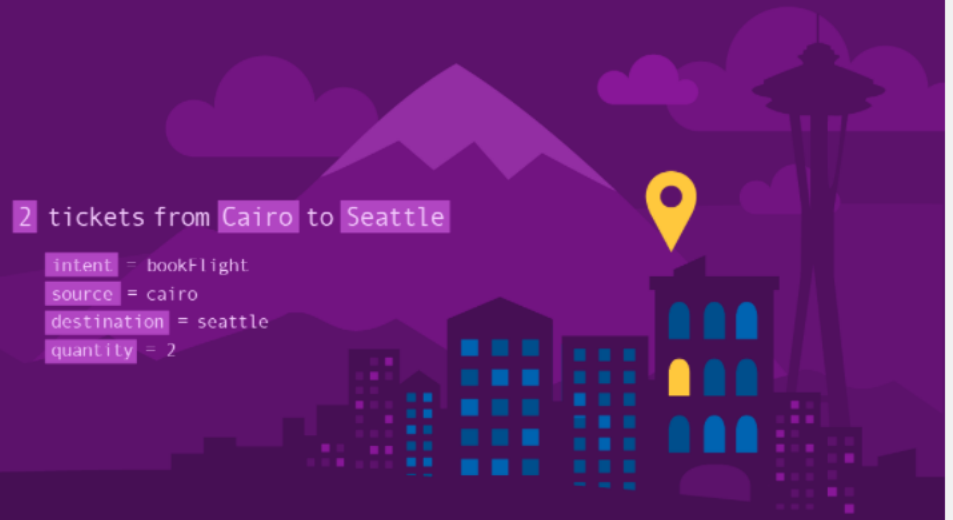
luis.ai

Microsoft Azure Cognitive Services Language Understanding Support Resources Sign in

Language Understanding (LUIS)

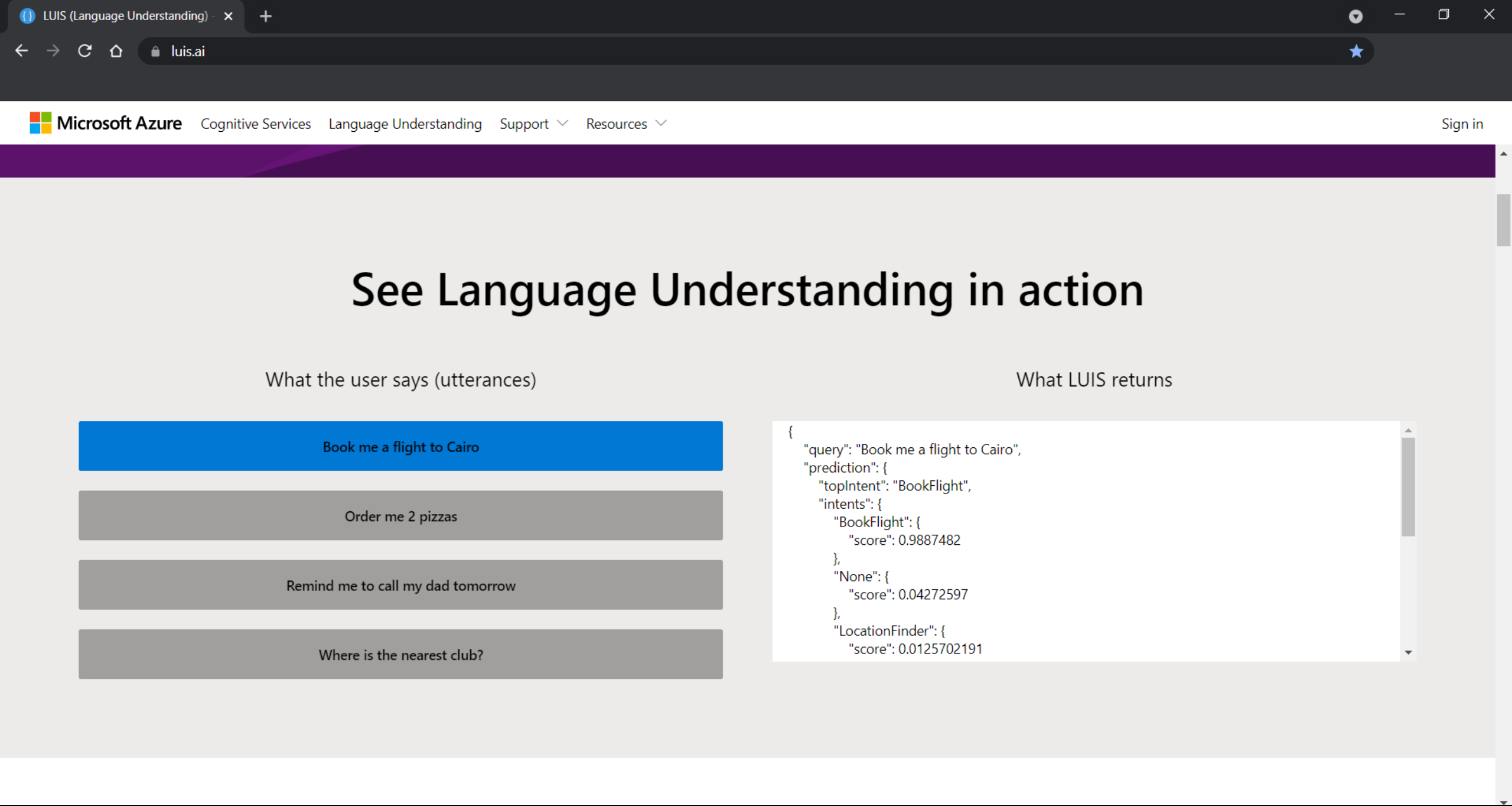
A machine learning-based service to build natural language into apps, bots, and IoT devices.
Quickly create enterprise-ready, custom models that continuously improve.

Login / Sign up



See Language Understanding in action

https://www.luis.ai/



https://www.luis.ai/

Language Understanding - Micro

portal.azure.com/#blade/Microsoft_Azure_Marketplace/GalleryItemDetailsBladeNopdl/product/%7B"displayName"%3A"Language%20Understanding"%2C"itemDisplayName"%3A"Language...

Microsoft Azure

Search resources, services, and docs (G+/)

- Create a resource
- Home
- Dashboard
- All services
- FAVORITES
- All resources
- Resource groups
- App Services
- Function App
- SQL databases
- Azure Cosmos DB
- Virtual machines
- Load balancers
- Storage accounts
- Virtual networks
- Azure Active Directory
- Monitor
- Advisor
- Security Center
- Cost Management + Bill...

Home > Create a resource >

Language Understanding

Microsoft

Language Understanding

Add to Favorites

Microsoft

★★★☆ 3.8 (8 ratings)

Create

Overview | Plans | Usage Information + Support | Reviews

Language Understanding (LUIS) is a natural language processing service that enables you to understand human language in your own application, website, chatbot, IoT device, and more. After you configure and publish your LUIS model, your application can easily receive user input in natural language and take action. You don't need to understand machine learning to solve the problem of extracting meaning from input. Instead you get to focus on your own application logic and let LUIS do the heavy lifting on your behalf. After your LUIS model is built and deployed, it exports a simple HTTP endpoint that is called by your application.

More offers from Microsoft

See All

Workspace
Microsoft
Virtual Machine
Azure Virtual Desktop resource

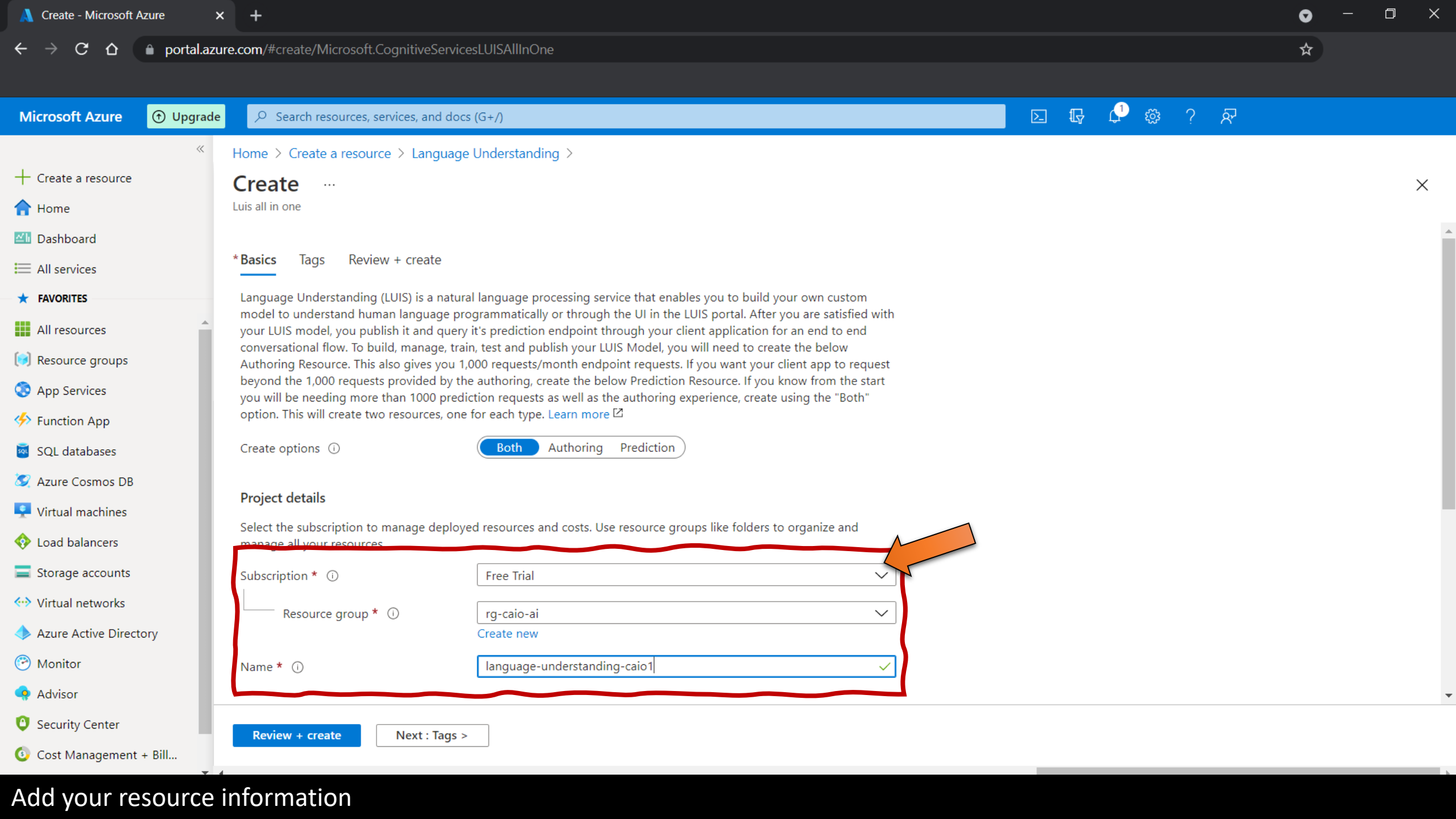
Microsoft HPC Pack 2012 R2
Microsoft
Virtual Machine
Enterprise-class HPC solution. Easy to deploy, cost-effective and supports Windows/Linux workloads

Windows 10 IoT Core Services
Microsoft
Azure Service
Commercialize your project with enterprise-grade security and support

Web App + SQL
Microsoft
Azure Service
Enjoy secure and flexible development, deployment, and scaling options for your web app

https://portal.azure.com/#

Search for Language Understanding



Add your resource information

Create - Microsoft Azure

portal.azure.com/#create/Microsoft.CognitiveServicesLUISAllInOne

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Bill...

Home > Create a resource > Language Understanding >

Create

Luis all in one

manage all your resources.

Subscription * : Free Trial

Resource group * : rg-caio-ai

Create new

Name * : language-understanding-caio1

Authoring Resource

Select pricing and location for Authoring Resource

Authoring location * : (US) West US

Authoring pricing tier (Learn More) * : Free F0 (5 Calls per second, 1M Calls per month)

Prediction Resource

Select pricing and location for Prediction Resource

Prediction location * : (US) East US

Prediction pricing tier (Learn More) * : Standard S0 (50 Calls per second)

Review + create | Next : Tags >

Add your resource information

Create - Microsoft Azure

portal.azure.com/#create/Microsoft.CognitiveServicesLUISAllInOne

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

- Create a resource
- Home
- Dashboard
- All services
- FAVORITES
- All resources
- Resource groups
- App Services
- Function App
- SQL databases
- Azure Cosmos DB
- Virtual machines
- Load balancers
- Storage accounts
- Virtual networks
- Azure Active Directory
- Monitor
- Advisor
- Security Center
- Cost Management + Bill...

Home > Create a resource > Language Understanding >

Create

Luis all in one

manage all your resources.

Subscription * : Free Trial

Resource group * : rg-caio-ai

Create new

Name * : language-understanding-caio1

Authoring Resource

Select pricing and location for Authoring Resource

Authoring location * : (US) West US

Authoring pricing tier (Learn More) * : Free F0 (5 Calls per second, 1M Calls per month)

Prediction Resource

Select pricing and location for Prediction Resource

Prediction location * : (US) East US

Prediction pricing tier (Learn More) * : Standard S0 (50 Calls per second)

Remember to always use the Free Tier to save money in your subscription

Review + create | Next : Tags >

Select REVIEW+CREATE

LUIS (Language Understanding)

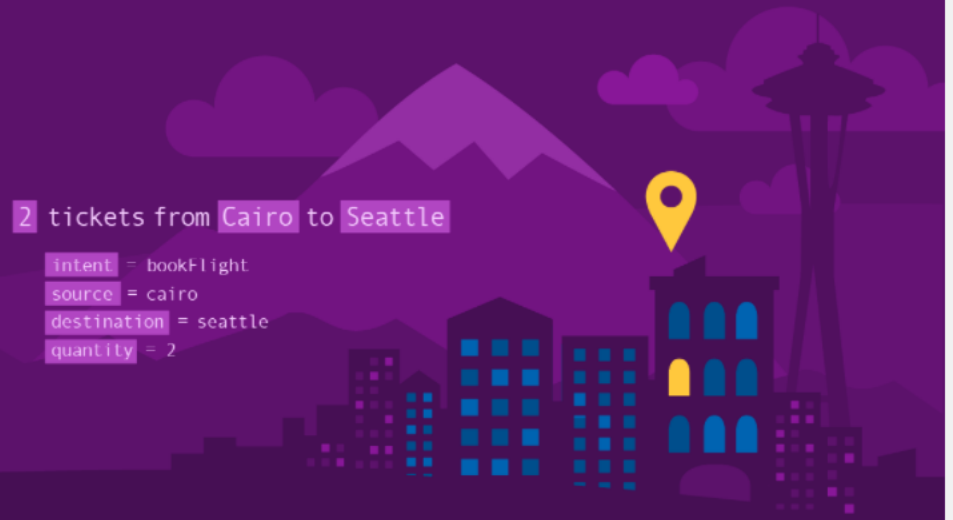
luis.ai

Microsoft Azure Cognitive Services Language Understanding Support Resources Sign in

Language Understanding (LUIS)

A machine learning-based service to build natural language into apps, bots, and IoT devices.
Quickly create enterprise-ready, custom models that continuously improve.

Login / Sign up



See Language Understanding in action

https://www.luis.ai/

Conversation apps

luis.ai/applications

Cognitive Services | Language Understanding

Conversation apps

Azure subscription: No subscription selected / Authoring resource: No authoring resource selected Choose a different authoring resource.

New app | Rename | Export | Imp

Name | Endpoint hits

Welcome to the Language Understanding Intelligent Service (LUIS)!

To access LUIS, you need an authoring resource. You can use an existing resource in your Azure subscription or create a new one. Learn more about creating an Azure resource.

Current Azure directory

- Name: Default Directory
- Id: 362067a6-2936-460f-ae05-bf263cebe940

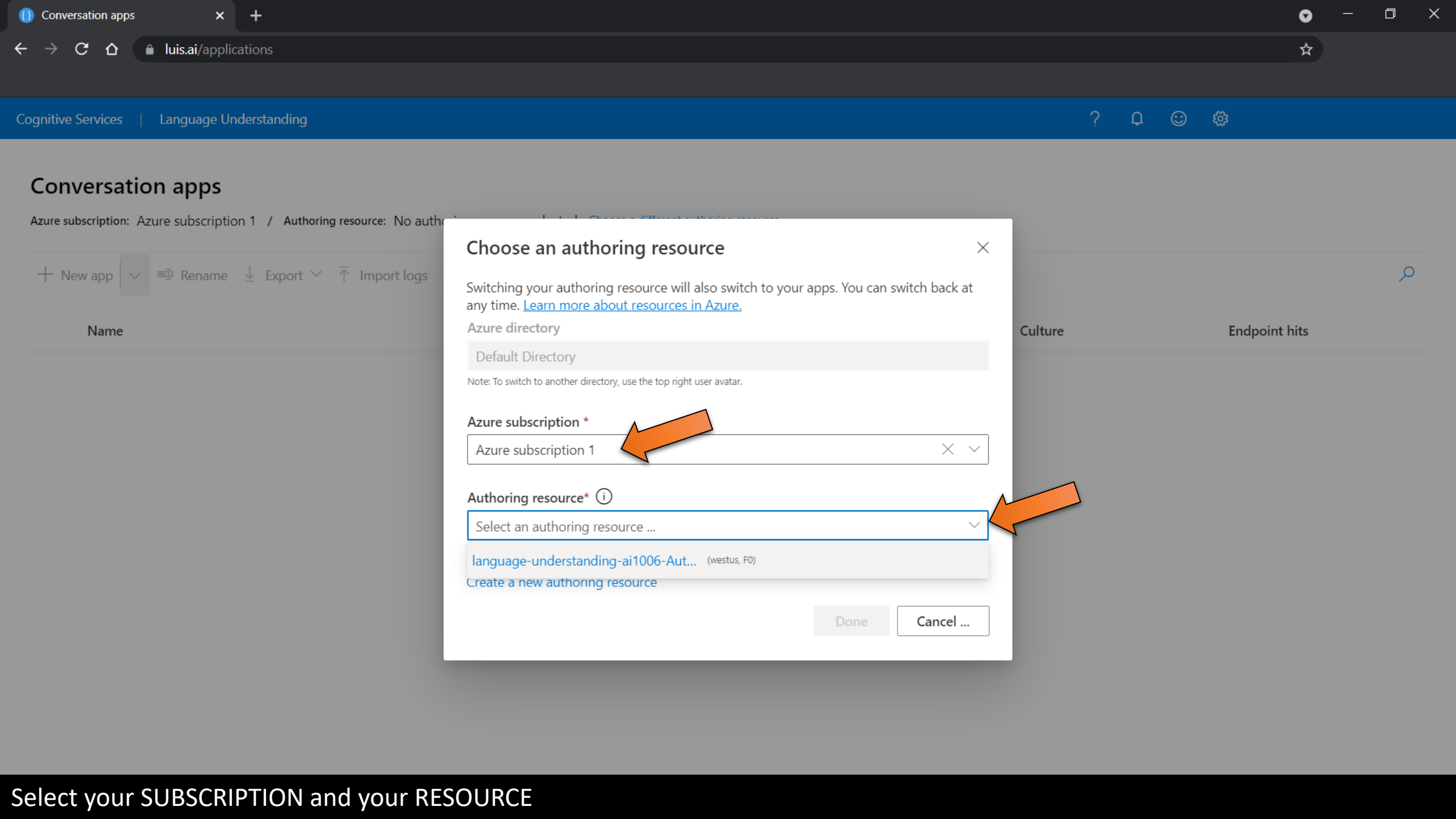
Switch to a different Azure directory

Subscription

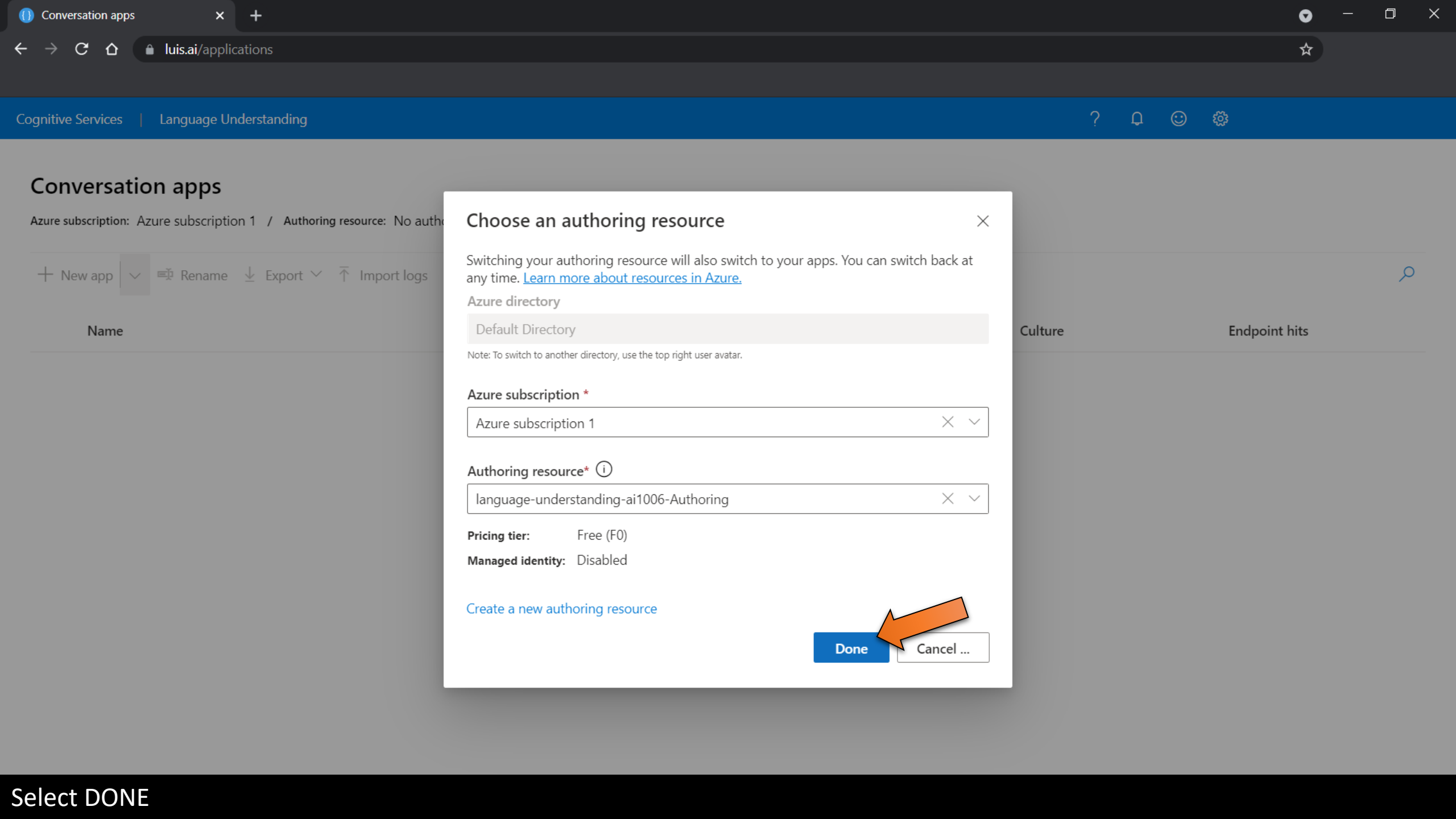
Azure subscription 1

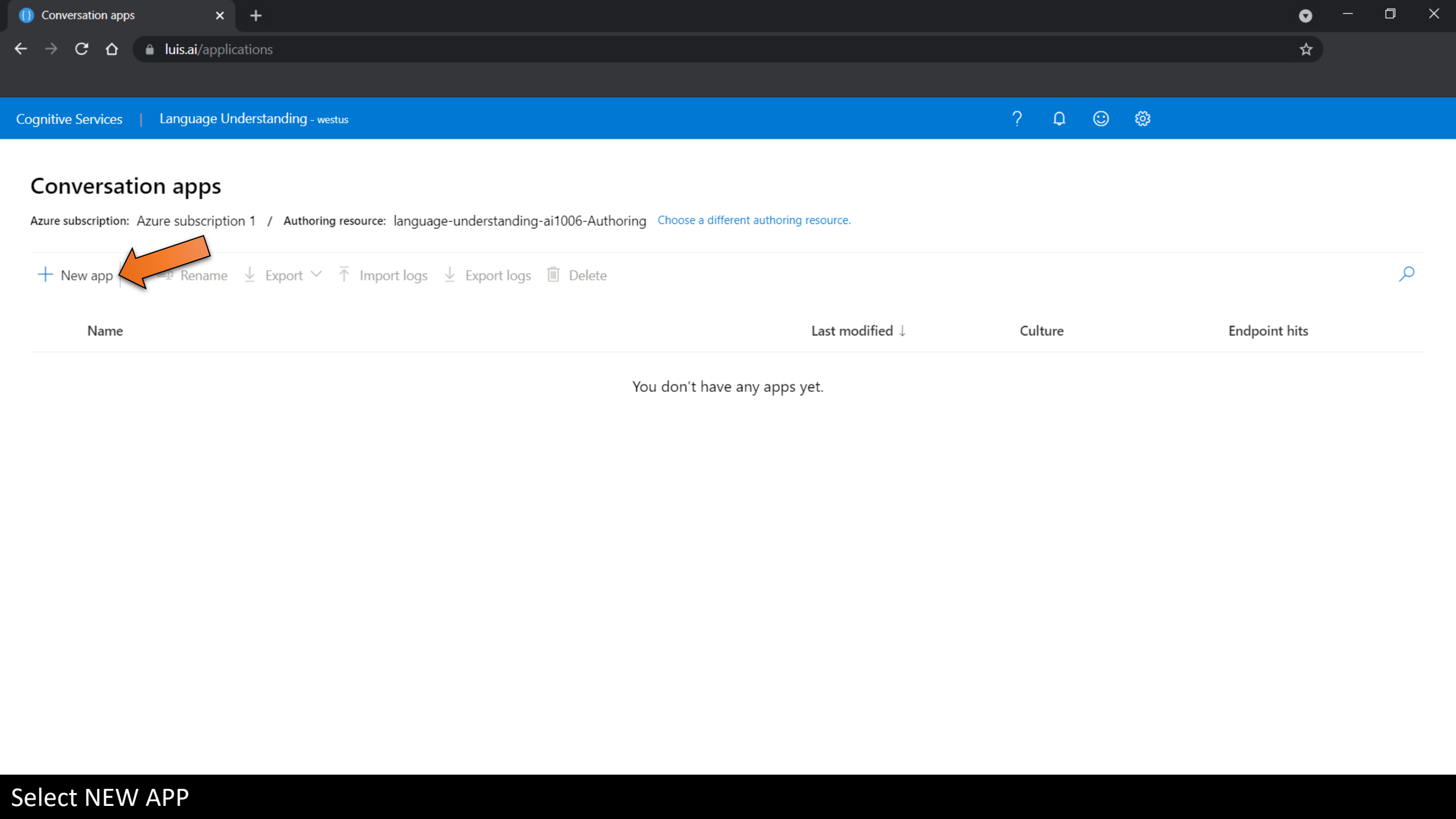
Select or create an authoring resource

https://docs.microsoft.com/en-us/azure/cognitive-services/luis/luis-get-started-create-app

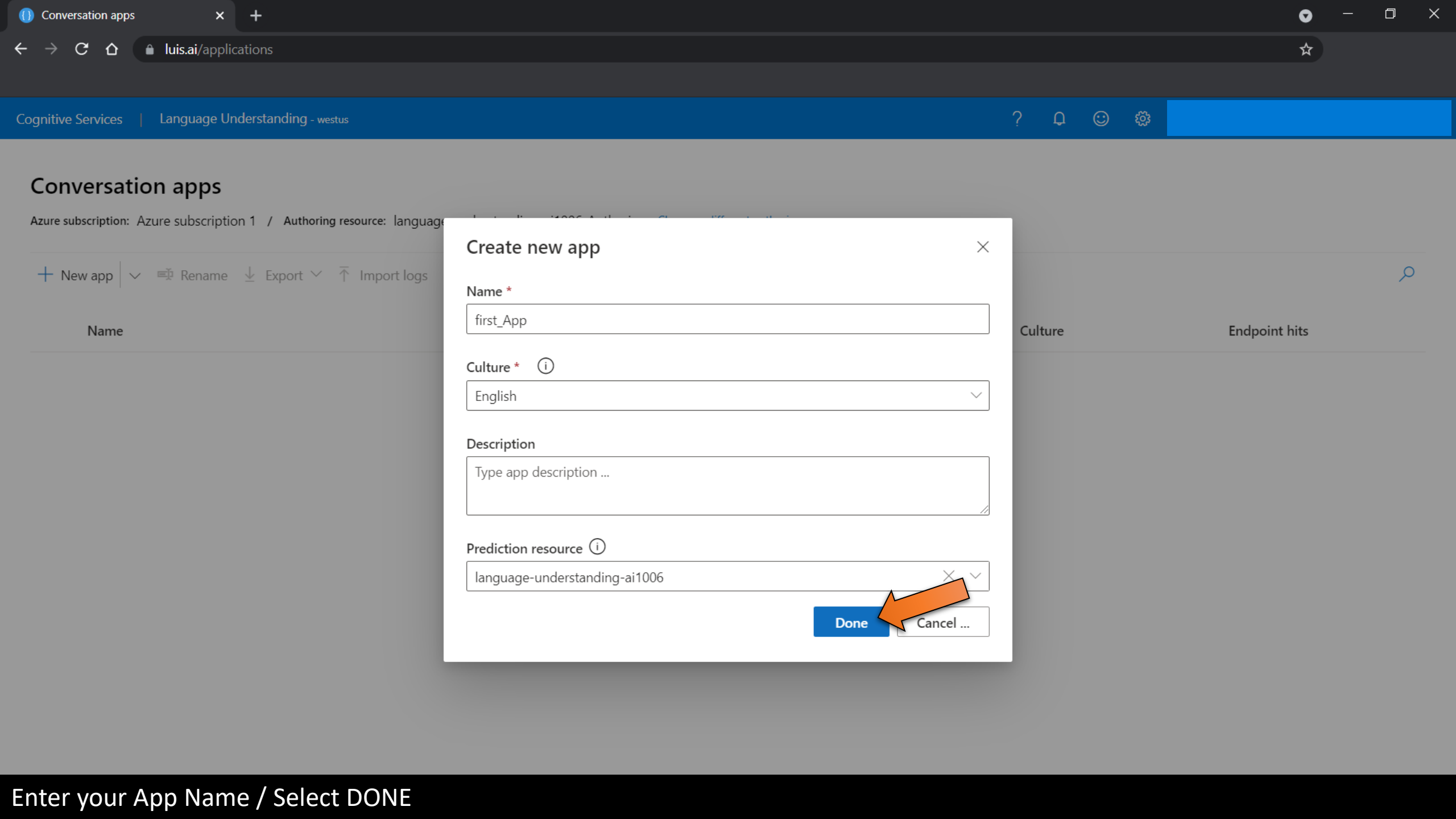


Select your SUBSCRIPTION and your RESOURCE


Conversation apps
luis.ai/applications
Cognitive Services | Language Understanding
Conversation apps
Azure subscription: Azure subscription 1 / Authoring resource: No auth
New app
Rename
Export
Import logs
Name
Culture
Endpoint hits
Choose an authoring resource
Switching your authoring resource will also switch to your apps. You can switch back at any time. Learn more about resources in Azure.
Azure directory
Default Directory
Note: To switch to another directory, use the top right user avatar.
Azure subscription *
Azure subscription 1
Authoring resource*
language-understanding-ai1006-Authoring
Pricing tier: Free (F0)
Managed identity: Disabled
Create a new authoring resource
Done
Cancel ...
Select DONE



Select NEW APP



Enter your App Name / Select DONE

Intents

luis.ai/applications/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/versions/0.1/build/intents

Cognitive Services | Language Understanding - westus

My LUIS / first_App V0.1

Train Test Publish

App Assets

Intents

Entities

Prebuilt Domains

Improve app performance

Review endpoint utterances

Features

Patterns

Intents

Create

Search for an intent by name ...

How to create an effective LUIS app

1. Design your schema
2. Build your model
3. Improve your app

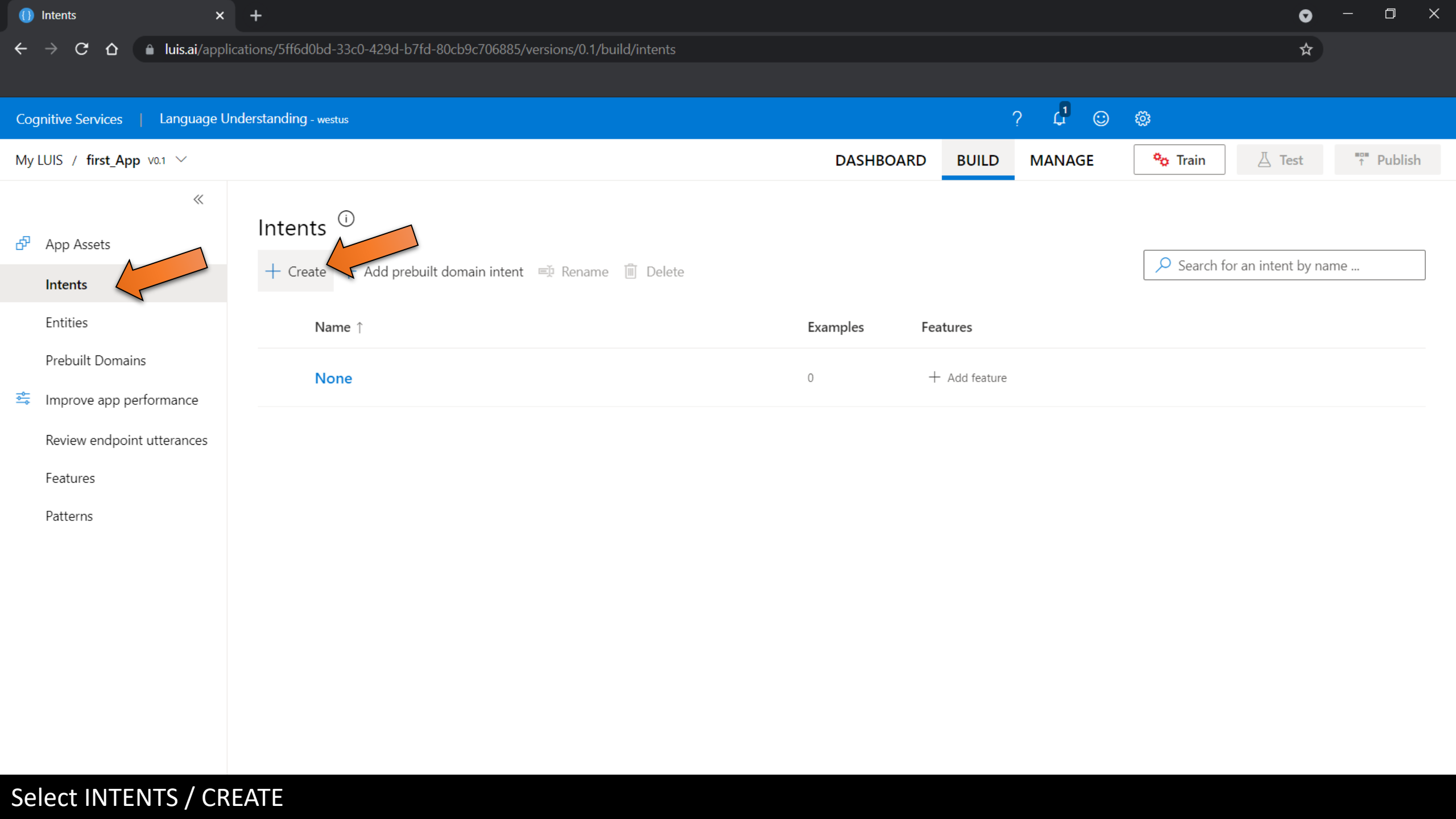
Create a schema that matches your real-world scenario.

Create an intent for each action your bot can perform.
Use entities to collect data needed to complete that action.

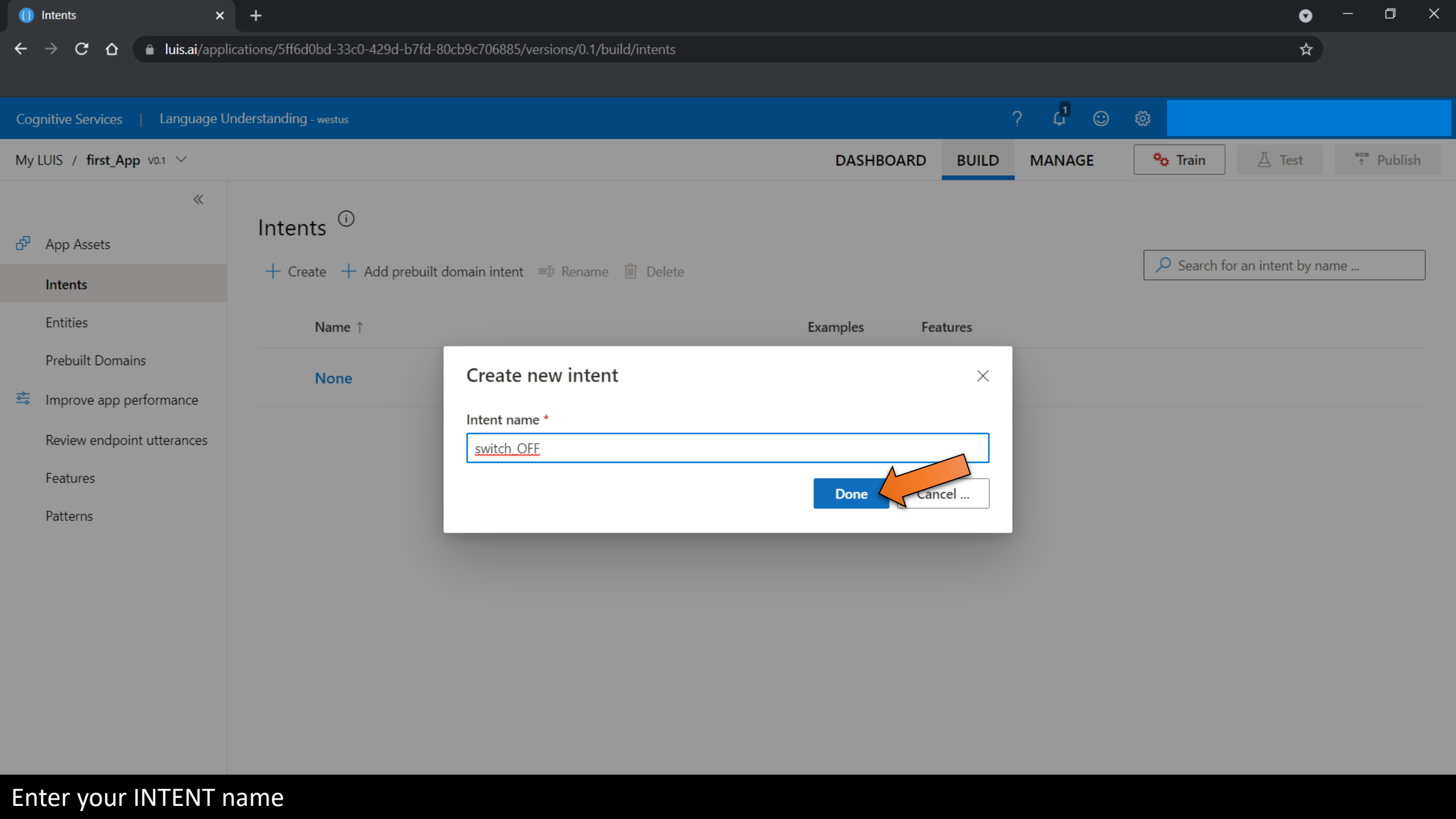
Example schema — Determine your bot's capabilities; create corresponding intents; add example utterances, and create entities to collect data. Create additional entities for each type of information collected.

Bot action:	Purchase airline tickets	Reserve rental cars	Make hotel reservations
Intent:	bookFlight	rentCar	reserveHotel
Example utterances:	"Book a flight to **Seattle**" "Buy 3 tickets to **New York**" "book a flight to **Rio** on **Contoso Air**" "get me a flight to **Rio** next week"	"Book a car on July 1 from **Seattle**" "reserve a car in **New York**"	"find hotels in **Seattle** from May 1-5" "reserve a hotel room in **New York**"
Entities:	city, **airline**	city	city

Go



Select INTENTS / CREATE



Enter your INTENT name

LUIS: Intent Page: switch_OFF

luis.ai/applications/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/versions/0.1/build/intents/938631e7-e618-4313-ba02-12580573dd40

Cognitive Services | Language Understanding - westus

My LUIS / first_App V0.1

DASHBOARD BUILD MANAGE Train Test Publish

- App Assets
- Intents
- Entities
- Prebuilt Domains
- Improve app performance
- Review endpoint utterances
- Features
- Patterns

switch_OFF

Machine learning features

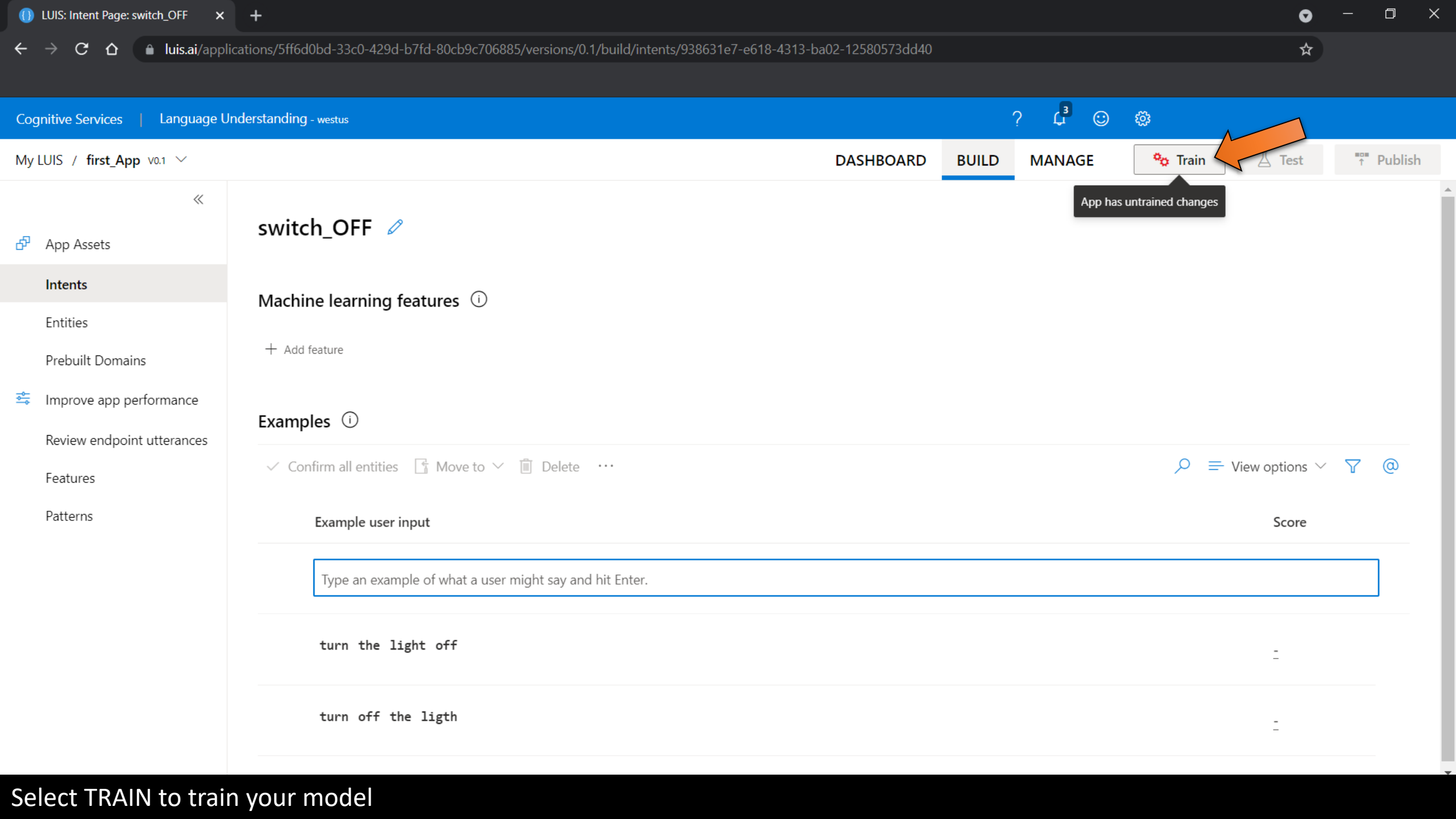
Add feature

Examples

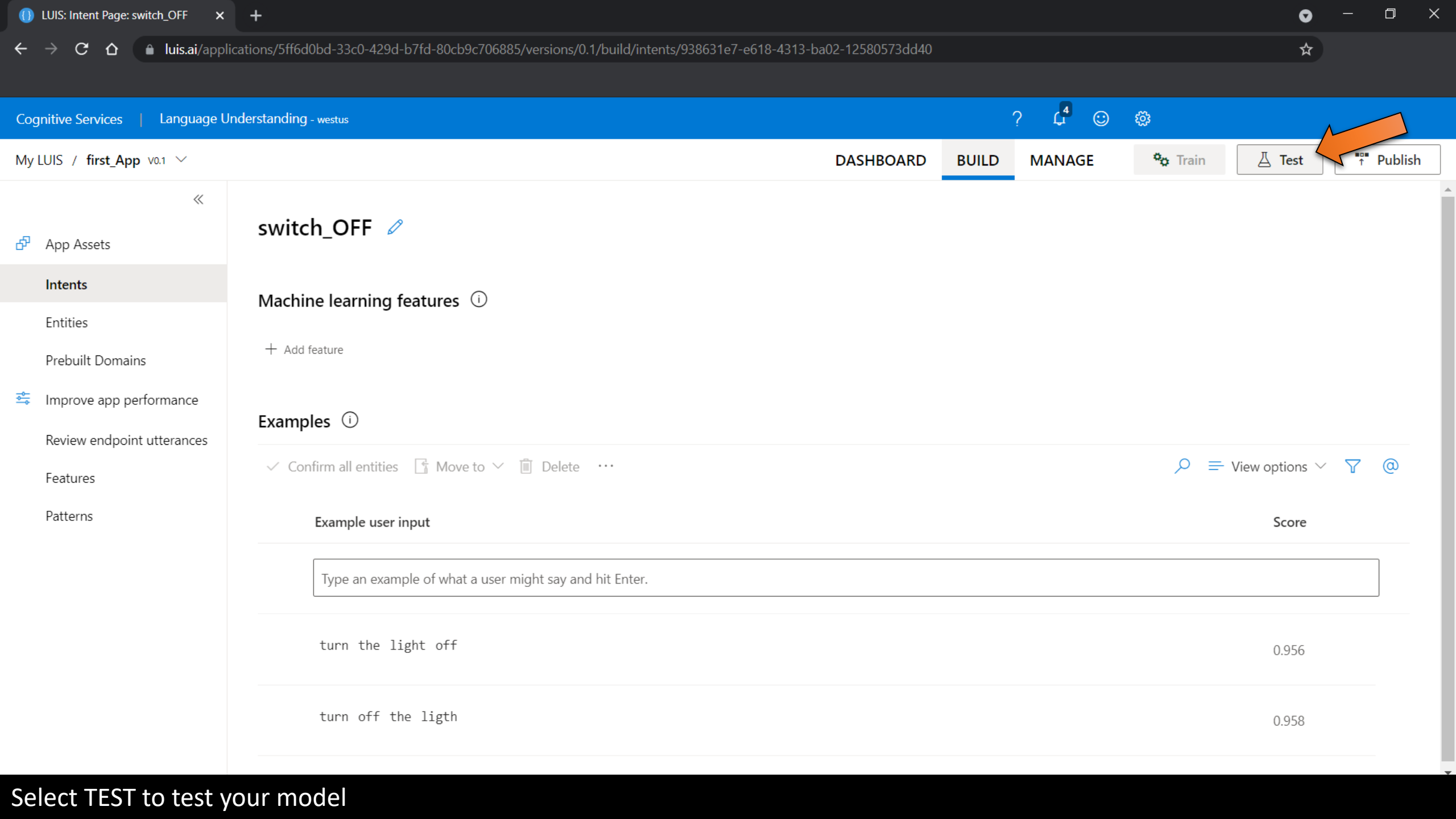
Confirm all entities | Move to | Delete | ... | View options

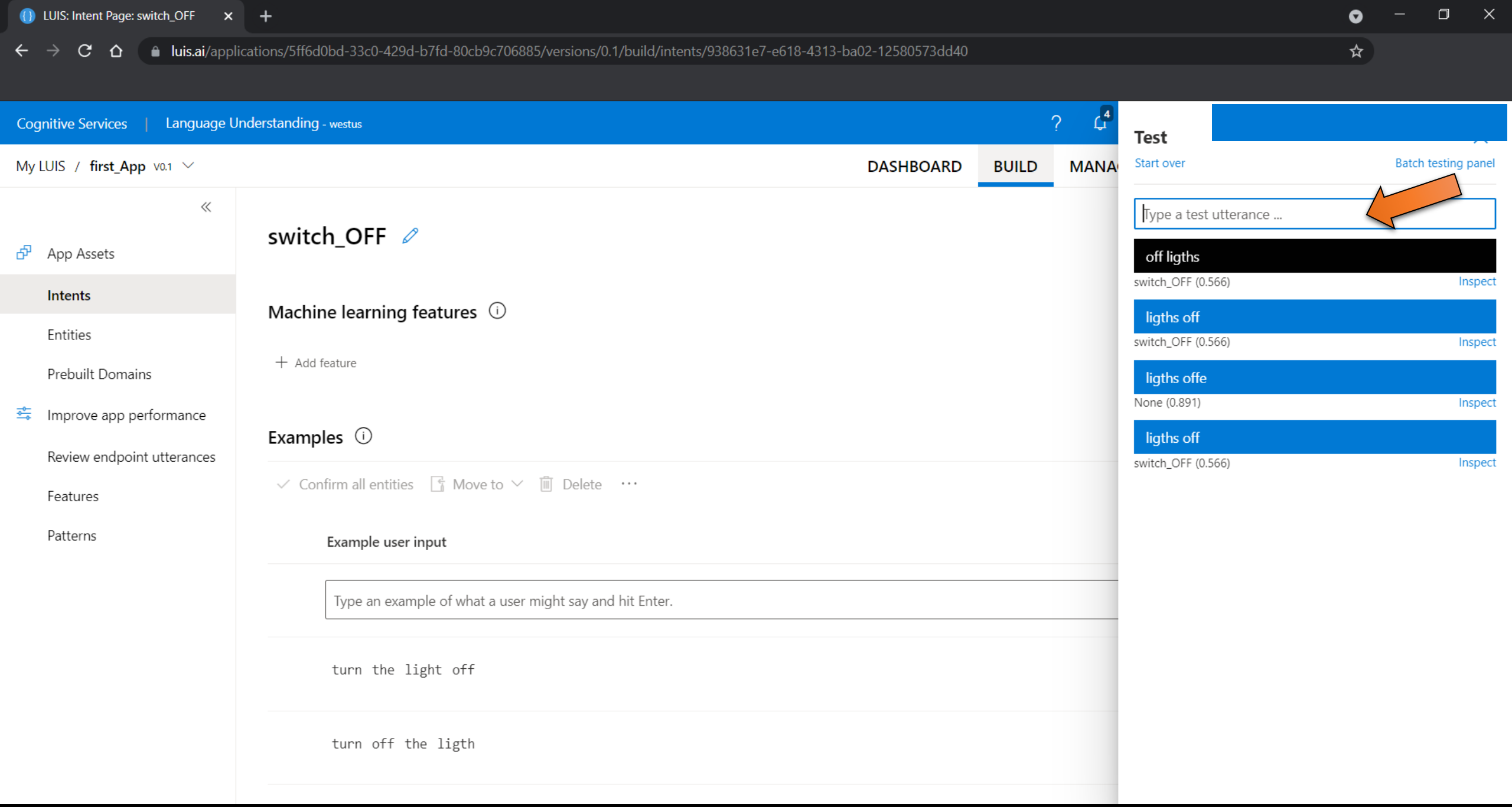
Example user input	Score
Type an example of what a user might say and hit Enter.	
turn the light off	-
turn off the ligth	-

Add a couple of INTENTS

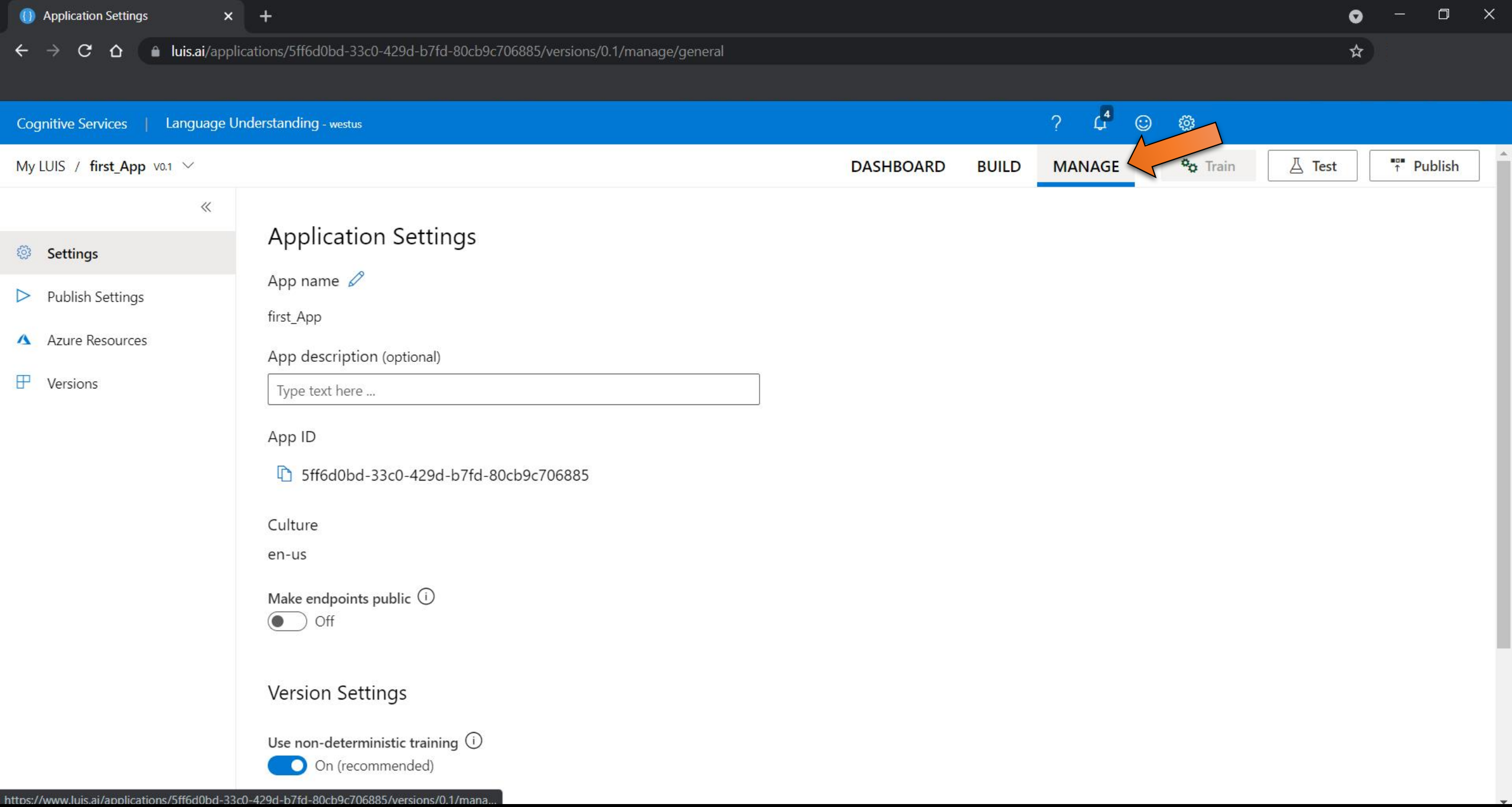


Select TRAIN to train your model

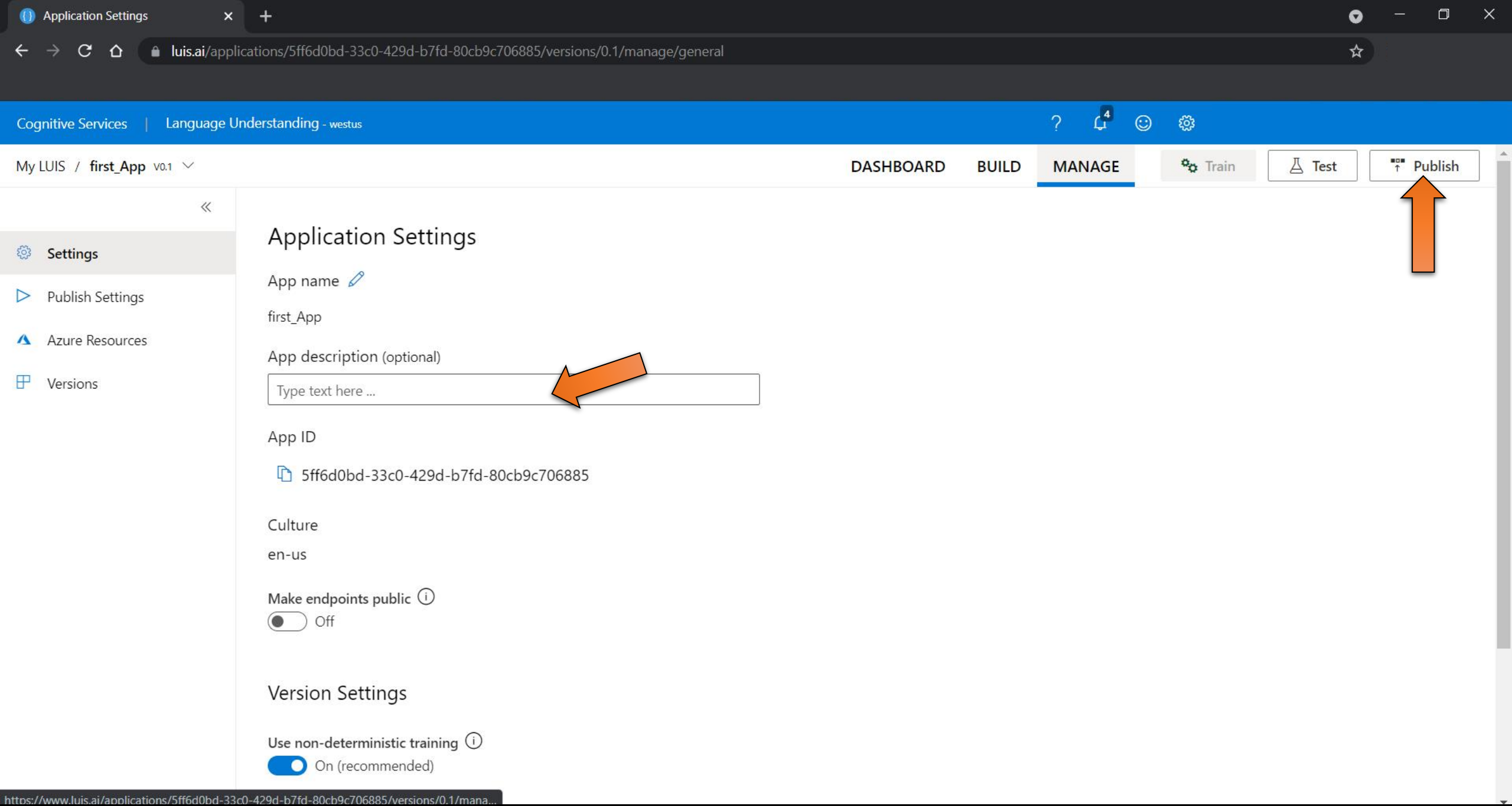

LUIS: Intent Page: switch_OFF
luis.ai/applications/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/versions/0.1/build/intents/938631e7-e618-4313-ba02-12580573dd40
Cognitive Services | Language Understanding - westus
My LUIS / first_App V0.1
DASHBOARD
BUILD
MANAGE
Train
Test
Publish
App Assets
Intents
Entities
Prebuilt Domains
Improve app performance
Review endpoint utterances
Features
Patterns
switch_OFF
Machine learning features
Add feature
Examples
Confirm all entities
Move to
Delete
View options
Example user input
Score
Type an example of what a user might say and hit Enter.
turn the light off
0.956
turn off the ligth
0.958
Select TEST to test your model



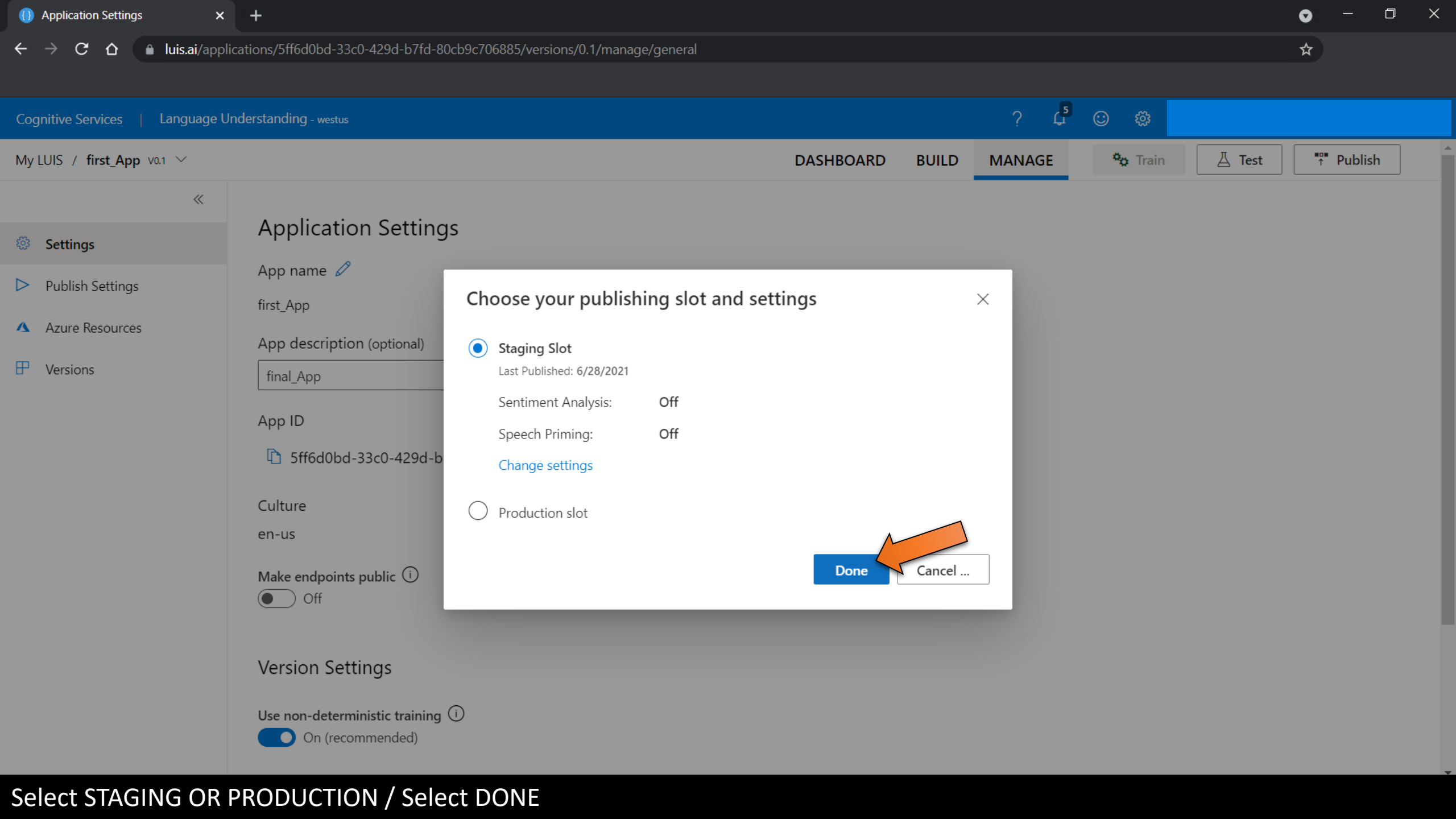
Enter some test utterances



Select MANAGE



Define your App description and select PUBLISH


Application Settings
luis.ai/applications/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/versions/0.1/manage/general
Cognitive Services | Language Understanding - westus
My LUIS / first_App V0.1
DASHBOARD
BUILD
MANAGE
Train
Test
Publish
Settings
Publish Settings
Azure Resources
Versions
Application Settings
App name
first_App
App description (optional)
final_App
App ID
5ff6d0bd-33c0-429d-b
Culture
en-us
Make endpoints public
Off
Version Settings
Use non-deterministic training
On (recommended)
Choose your publishing slot and settings
Staging Slot
Last Published: 6/28/2021
Sentiment Analysis: Off
Speech Priming: Off
Change settings
Production slot
Done
Cancel ...
Select STAGING OR PRODUCTION / Select DONE

Application Settings

luis.ai/applications/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/versions/0.1/manage/general

Cognitive Services | Language Understanding - westus

My LUIS / first_App V0.1

DASHBOARD BUILD MANAGE Train Test Publish

- Settings
- Publish Settings
- Azure Resources
- Versions

Application Settings

App name

first_App

App description (optional)

final_App

App ID

5ff6d0bd-33c0-429d-b7fd-80cb9c706885

Culture

en-us

Make endpoints public

Off

Version Settings

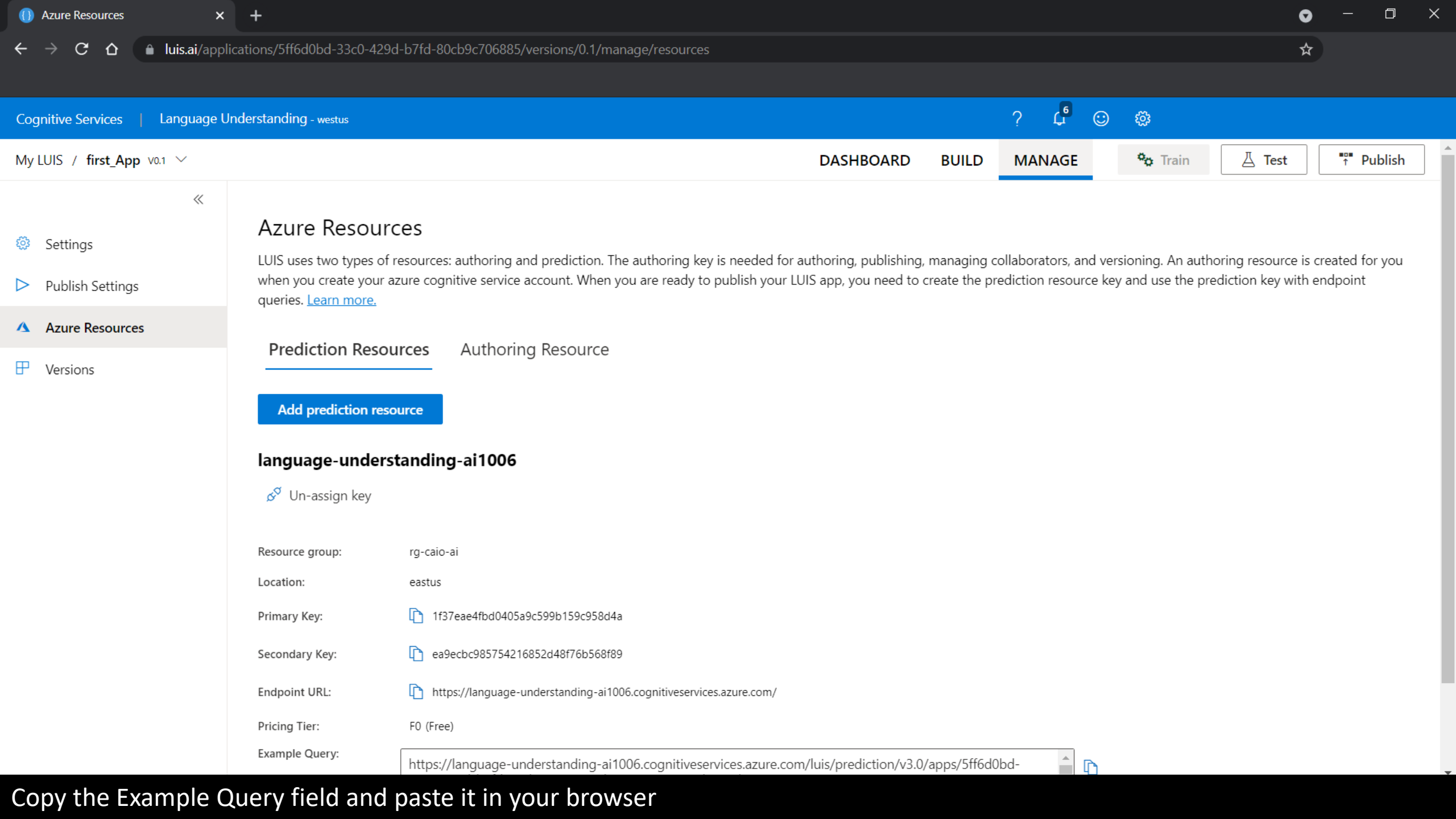
Use non-deterministic training

On (recommended)

Publish app 'first_App' completed 4:18 PM

Access your endpoint Urls

Select to access your app endpoints



Copy the Example Query field and paste it in your browser

Azure Resources

https://language-understanding-

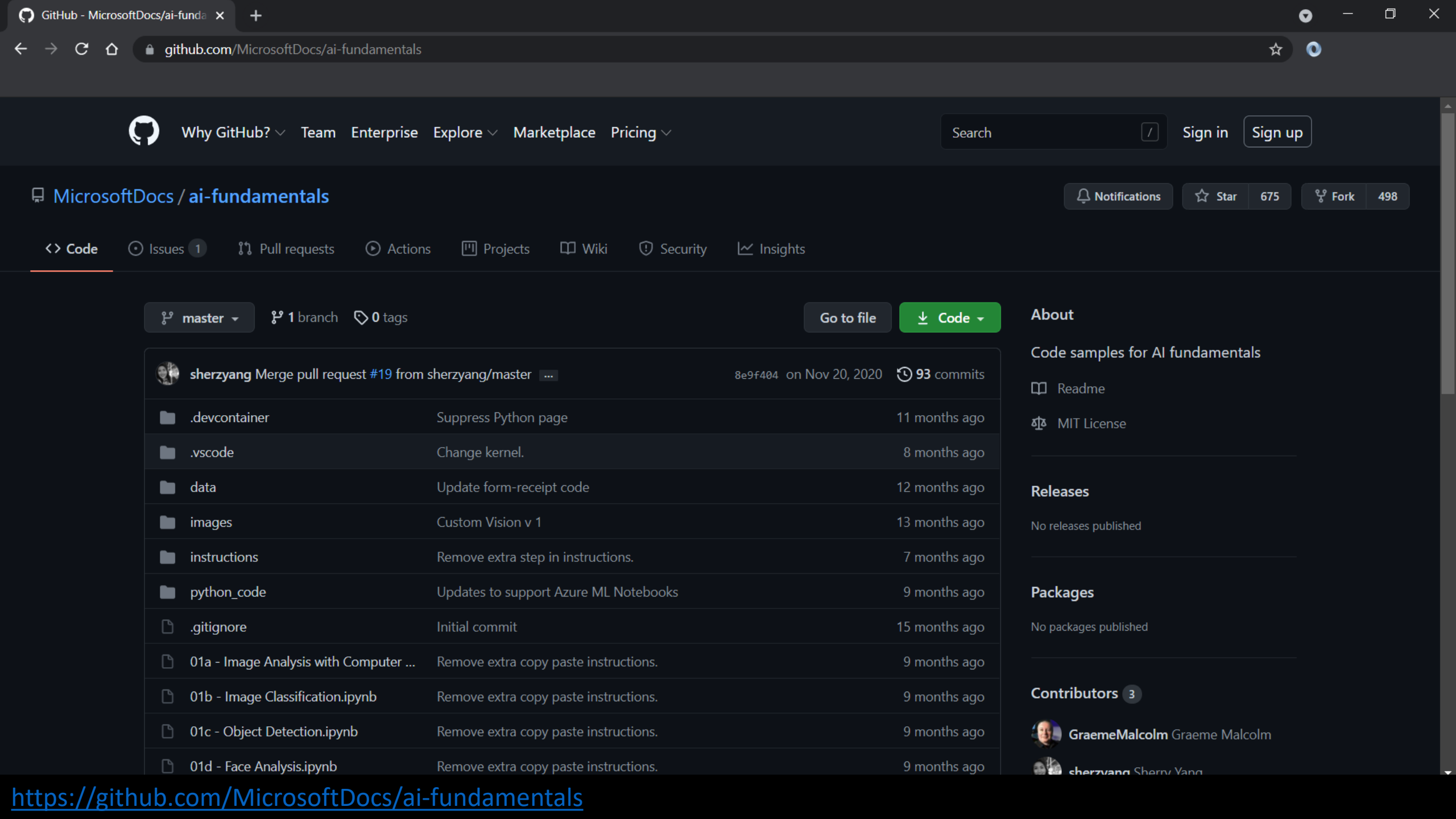
language-understanding-ai1006.cognitiveservices.azure.com/luis/prediction/v3.0/apps/5ff6d0bd-33c0-429d-b7fd-80cb9c706885/slots/staging/predict?subscription-key=1f37eae4fbd0405a9...

View source


```
{
    query: "YOUR_QUERY_HERE",
  - prediction: {
        topIntent: "switch_OFF",
      - intents: {
          - switch_OFF: {
                score: 0.3454776
            },
          - None: {
                score: 0.29123414
            }
        },
        entities: { }
    }
}
```


Results

Download the source code (ai-fundamentals)



https://github.com/MicrosoftDocs/ai-fundamentals

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

Microsoft Azure Machine Learning

Home > Notebooks

- New
- Home

Author
- Notebooks
- Automated ML
- Designer

Assets
- Datasets
- Experiments
- Pipelines
- Models
- Endpoints

Manage
- Compute
- Environments (preview)
- Datastores
- Data Labeling
- Linked Services

Notebooks

Files | Samples

- Users
  - caiogasparine
    - regression-automl-nyc-taxi-data
    - .amlignore
    - text-translator.ipynb

Create new file
Create new folder
Upload files
Upload folder
Copy folder path
Open terminal

Notebooks allow users to work with files, folders and Jupyter Notebooks directly in the workspace.

Browse your files and shared files with easy collaboration tools. You can also start with a Jupyter Notebook in the workspace with easy access to all workspace assets including experiment details, datasets, models and more. Learn more

Create | Terminal

View Azure Machine Learning tutorials

Go to your Notebook on Azure Machine Learning Studio

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

Microsoft Azure Machine Learning

Home > Notebooks

Success: Successfully uploaded all files

- New
- Home

Author
- Notebooks
- Automated ML
- Designer

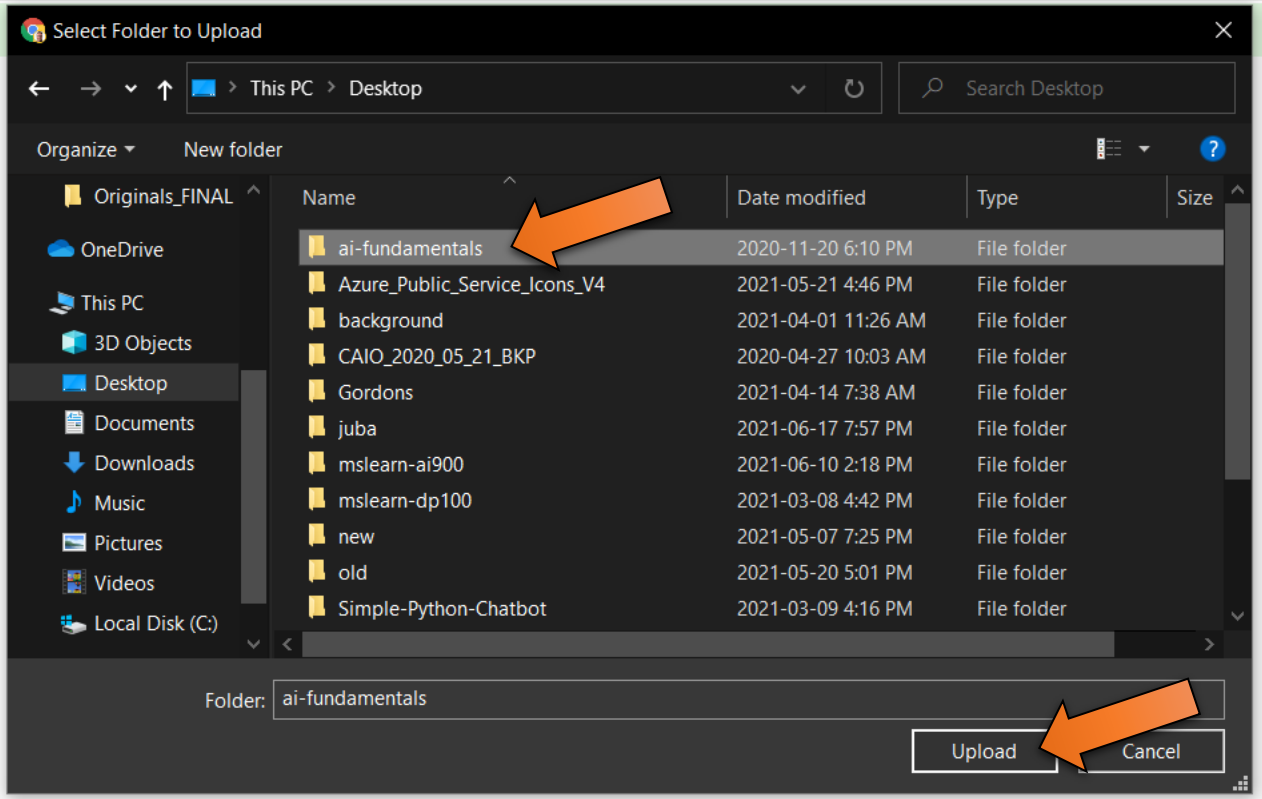
Assets
- Datasets
- Experiments
- Pipelines
- Models
- Endpoints

Manage
- Compute
- Environments (preview)
- Datastores
- Data Labeling
- Linked Services

Notebooks

Files | Samples

- Users
  - caiogasparine
    - ai-fundamentals
    - regression-automl-nyc-taxi-data
    - .amlignore
    - text-translator.ipynb



Notebooks directly in the workspace.

Browse your files and shared files with easy collaboration tools. You can also start with a Jupyter Notebook in the workspace with easy access to all workspace assets including experiment details, datasets, models and more. Learn more

ai-fundamentals-m....zip

Show all

And upload all the files (python code)

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

Microsoft Azure Machine Learning

Home > Notebooks

Success: Successfully uploaded all files

- New
- Home
- Author
- Notebooks
- Automated ML
- Designer
- Assets
- Datasets
- Experiments
- Pipelines
- Models
- Endpoints
- Manage
- Compute
- Environments (preview)
- Datastores
- Data Labeling
- Linked Services

Notebooks

Files | Samples

- Users
  - caiogasparine
    - ai-fundamentals
      - .devcontainer
      - data
      - images
      - instructions
      - python_code
      - .gitignore
      - 01a - Image Analysis with Computer ...
      - 01b - Image Classification.ipynb
      - 01c - Object Detection.ipynb
      - 01d - Face Analysis.ipynb
      - 01e - Optical Character Recognition.i...

ai-fundamentals

Notebooks allow users to work with files, folders and Jupyter Notebooks directly in the workspace.

Browse your files and shared files with easy collaboration tools. You can also start with a Jupyter Notebook in the workspace with easy access to all workspace assets including experiment details, datasets, models and more. Learn more

ai-fundamentals-m....zip

Show all

Wait for the upload and check the imported folder

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

Microsoft Azure Machine Learning

6

- New
- Home

Author

- Notebooks
- Automated ML
- Designer

Assets

- Datasets
- Experiments
- Pipelines
- Models
- Endpoints

Manage

- Compute
- Environments (preview)
- Datastores
- Data Labeling
- Linked Services

Home > Notebooks

Success: Successfully uploaded all files

Notebooks

Files | Samples

- 01a - Image Analysis with Computer Vision.ipynb
- 01b - Image Classification.ipynb
- 01c - Object Detection.ipynb
- 01d - Face Analysis.ipynb
- 01e - Optical Character Recognition.ipynb
- 01f - Receipts with Form Recognizer.ipynb
- 02a - Text Analytics.ipynb
- 02b - Speech.ipynb
- 02c - Translation.ipynb
- 02d - Language Understanding.ipynb
- 03a - QnA Bot.ipynb
- CODE_OF_CONDUCT.md
- LICENSE
- README.md
- SECURITY.md

Notebooks allow users to work with files, folders and Jupyter Notebooks directly in the workspace.

Browse your files and shared files with easy collaboration tools. You can also start with a Jupyter Notebook in the workspace with easy access to all workspace assets including experiment details, datasets, models and more. Learn more

Create | Terminal

There are several examples using python

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

Microsoft Azure Machine Learning

Home > Notebooks

Success: Successfully uploaded all files

New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Environments (preview)
Datastores
Data Labeling
Linked Services

Notebooks

Files Samples

01a - Image Analysis with Computer Vision.ipynb
01b - Image Classification.ipynb
01c - Object Detection.ipynb
01d - Face Analysis.ipynb
01e - Optical Character Recognition.ipynb
01f - Receipts with Form Recognizer.ipynb
02a - Text Analytics.ipynb
02b - Speech.ipynb
02c - Translation.ipynb
02d - Language Understanding.ipynb
03a - QnA Bot.ipynb
CODE_OF_CONDUCT.md
LICENSE
README.md
SECURITY.md

02d - Language Unc

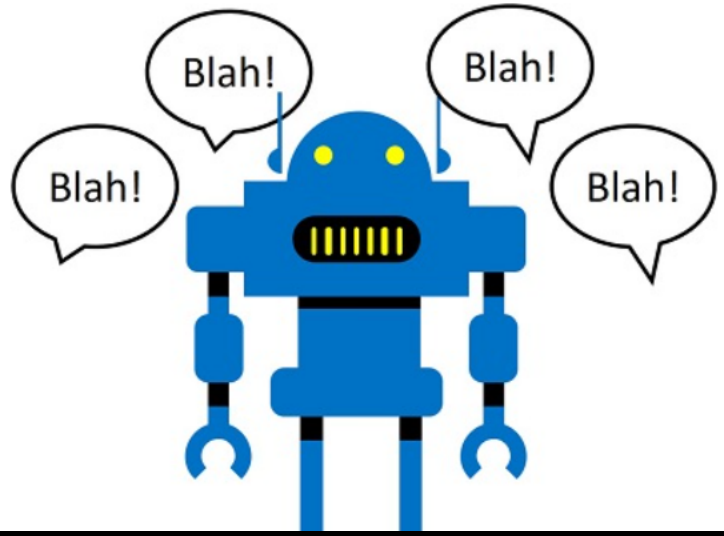
Editors | Compute: ml-c... | No ...

Your document is currently not connected to a compute. To connect, you can run a cell, switch to a running compute or create a new compute.

Viewing

Language Understanding

Increasingly, we expect computers to be able to use AI in order to understand spoken or typed commands in natural language. For example, you might want to implement a home automation system that enables you to control devices in your home by using voice commands such as "switch on the light" or "put the fan on", and have an AI-powered device understand the command and take appropriate action.



https://www.luis.ai/

machinelearning-ai1006 - Micros | Notebooks - Microsoft Azure Ma

ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...

There are several different ways to write your code. This is just one of them.

Microsoft Azure Machine Learning

Home > Notebooks

Success: Successfully uploaded all files

New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Environments (preview)
Datastores
Data Labeling
Linked Services

Notebooks

Files Samples

01a - Image Analysis with Computer Vision.ipynb
01b - Image Classification.ipynb
01c - Object Detection.ipynb
01d - Face Analysis.ipynb
01e - Optical Character Recognition.ipynb
01f - Receipts with Form Recognizer.ipynb
02a - Text Analytics.ipynb
02b - Speech.ipynb
02c - Translation.ipynb
02d - Language Understanding.ipynb
03a - QnA Bot.ipynb
CODE_OF_CONDUCT.md
LICENSE
README.md
SECURITY.md

02d - Language Unc

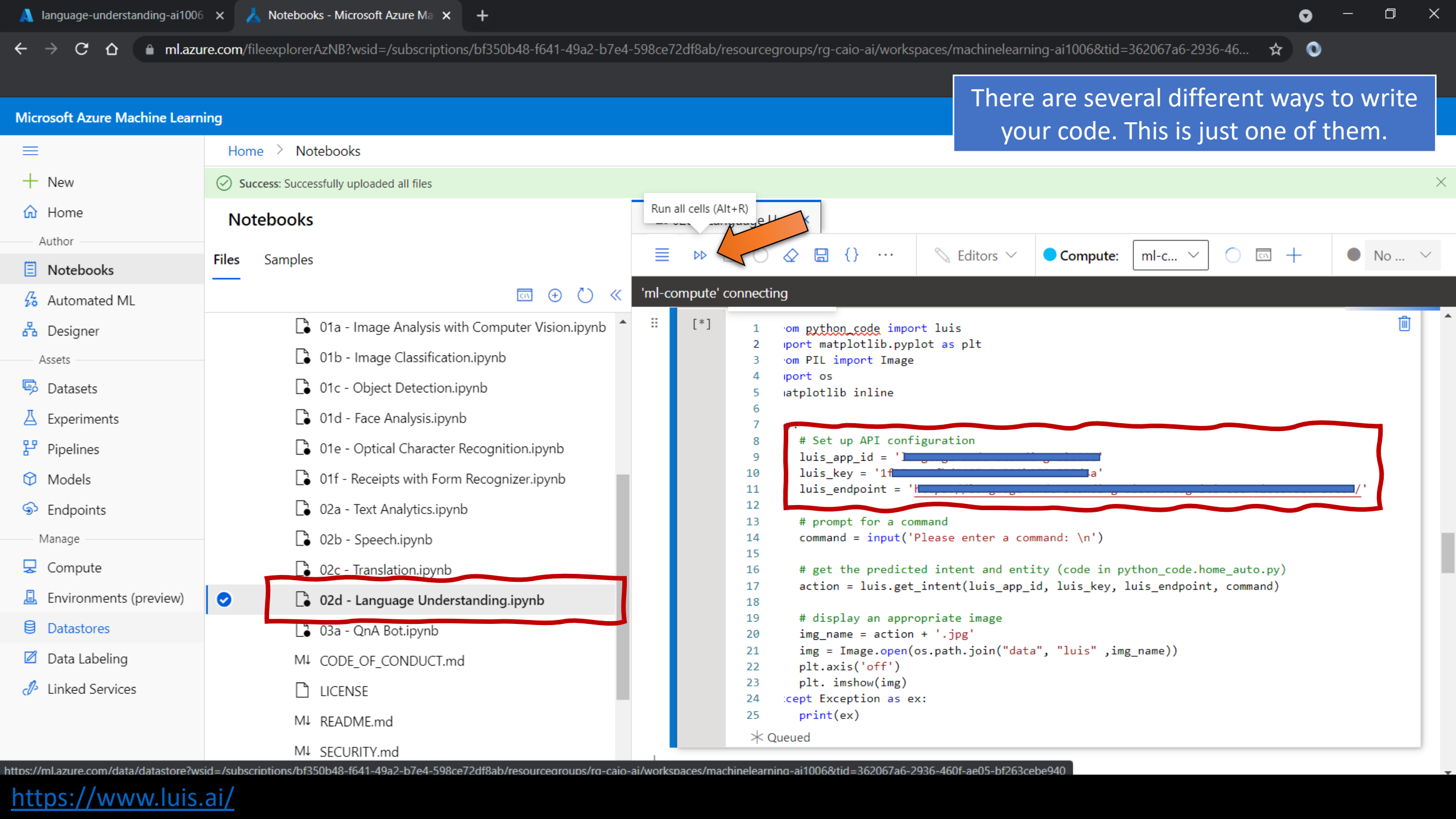
Editors | Compute: ml-c... | No ...

Your document is currently not connected to a compute. To connect, you can run a cell, switch to a running compute or create a new compute.

Editing

```
1  from python_code import luis
2  import matplotlib.pyplot as plt
3  from PIL import Image
4  import os
5  %matplotlib inline
6
7  try:
8      # Set up API configuration
9      luis_app_id = 'YOUR_LU_APP_ID'
10     luis_key = 'YOUR_LU_KEY'
11     luis_endpoint = 'YOUR_LU_ENDPOINT'
12
13     # prompt for a command
14     command = input('Please enter a command: \n')
15
16     # get the predicted intent and entity (code in python_code.home_auto.py)
17     action = luis.get_intent(luis_app_id, luis_key, luis_endpoint, command)
18
19     # display an appropriate image
20     img_name = action + '.jpg'
21     img = Image.open(os.path.join("data", "luis" ,img_name))
22     plt.axis('off')
23     plt. imshow(img)
24 except Exception as ex:
25     print(ex)
```

Add your APP-ID, KEY and ENDPOINT to the code


language-understanding-ai1006
Notebooks - Microsoft Azure Ma
ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-46...
There are several different ways to write your code. This is just one of them.
Microsoft Azure Machine Learning
Home > Notebooks
Success: Successfully uploaded all files
Notebooks
Files
Samples
New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Environments (preview)
Datastores
Data Labeling
Linked Services
01a - Image Analysis with Computer Vision.ipynb
01b - Image Classification.ipynb
01c - Object Detection.ipynb
01d - Face Analysis.ipynb
01e - Optical Character Recognition.ipynb
01f - Receipts with Form Recognizer.ipynb
02a - Text Analytics.ipynb
02b - Speech.ipynb
02c - Translation.ipynb
02d - Language Understanding.ipynb
03a - QnA Bot.ipynb
CODE_OF_CONDUCT.md
LICENSE
README.md
SECURITY.md
Run all cells (Alt+R)
Editors
Compute:
ml-c...
No ...
'ml-compute' connecting
[*]
om python_code import luis
port matplotlib.pyplot as plt
om PIL import Image
port os
atplotlib inline
# Set up API configuration
luis_app_id = '
luis_key = '1f
a'
luis_endpoint = '
/'
# prompt for a command
command = input('Please enter a command: \n')
# get the predicted intent and entity (code in python_code.home_auto.py)
action = luis.get_intent(luis_app_id, luis_key, luis_endpoint, command)
# display an appropriate image
img_name = action + '.jpg'
img = Image.open(os.path.join("data", "luis" ,img_name))
plt.axis('off')
plt. imshow(img)
cept Exception as ex:
print(ex)
Queued
https://ml.azure.com/data/datastore?wsid=/subscriptions/bf350b48-f641-49a2-b7e4-598ce72df8ab/resourcegroups/rg-caio-ai/workspaces/machinelearning-ai1006&tid=362067a6-2936-460f-ae05-bf263cebe940
https://www.luis.ai/

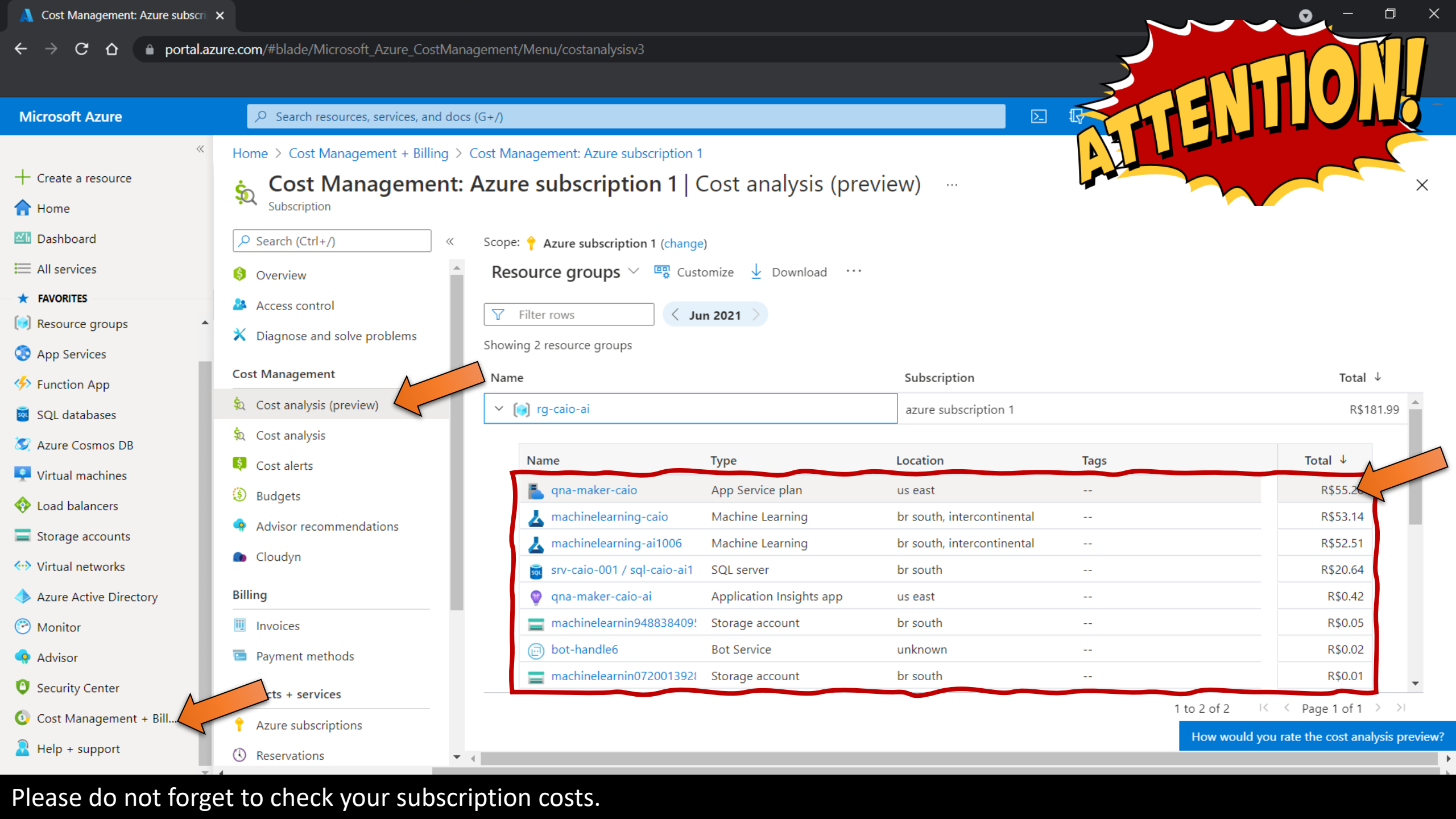
MAIN STEPS

- Create Language Understanding resource - Azure Portal
- Login to LUIS.AI (www.luis.ai)
- Create a **new project**
- Enter App + intent + entities
- **Train** the model + **Publish** the model
- Copy **App ID**
- Copy **Keys + Endpoints** to the code
- Execute the code

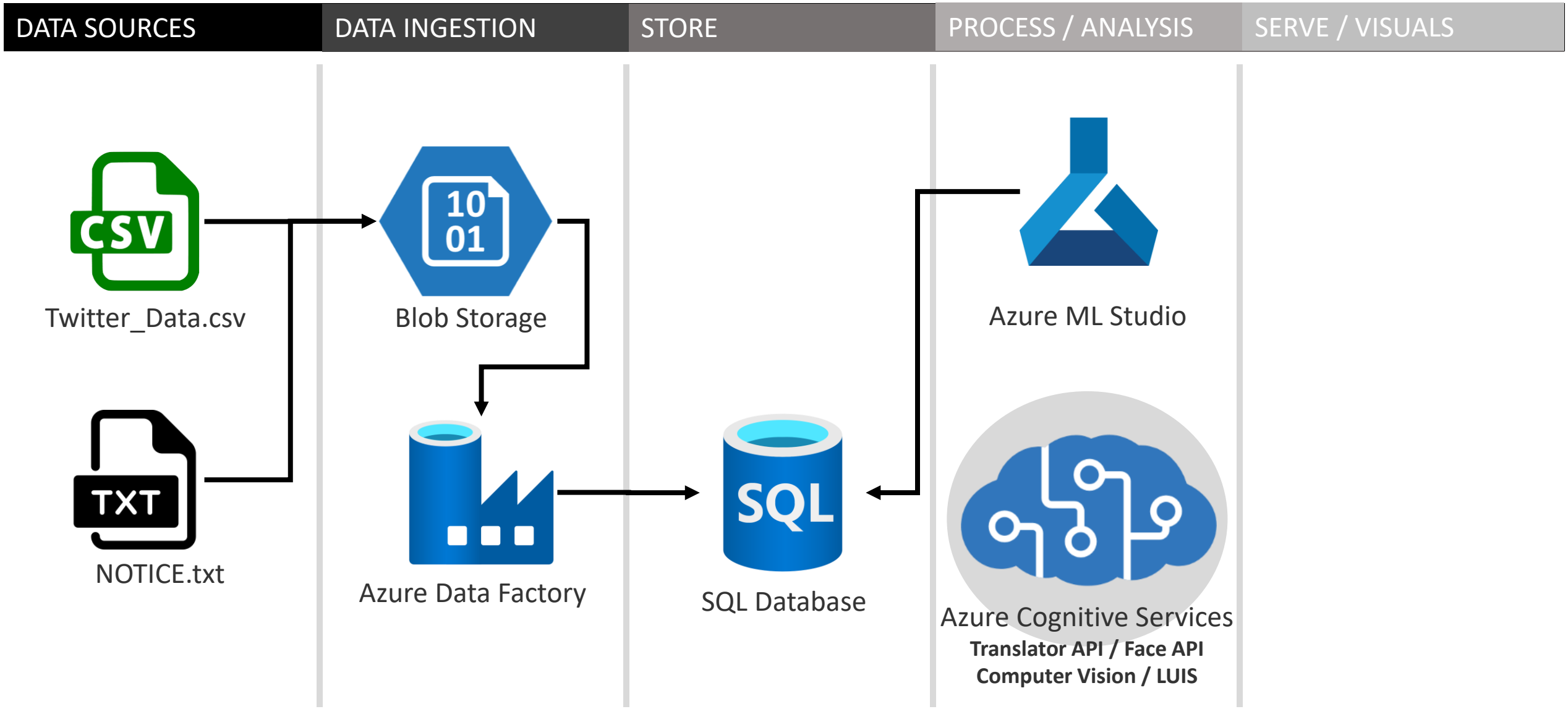
ASSIGNMENT

Prepare your own code. Follow the instructions in the link below and prepare you first LUIS App.

https://docs.microsoft.com/en-us/azure/cognitive-services/luis/client-libraries-rest-api?tabs=windows&pivots=programming-language-python


Cost Management: Azure subscri
portal.azure.com/#blade/Microsoft_Azure_CostManagement/Menu/costanalysisv3
ATTENTION!
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Cost Management + Billing > Cost Management: Azure subscription 1
Cost Management: Azure subscription 1 | Cost analysis (preview)
Subscription
Create a resource
Home
Dashboard
All services
FAVORITES
Resource groups
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Azure Active Directory
Monitor
Advisor
Security Center
Cost Management + Bill...
Help + support
Search (Ctrl+/)
Overview
Access control
Diagnose and solve problems
Cost Management
Cost analysis (preview)
Cost analysis
Cost alerts
Budgets
Advisor recommendations
Cloudyn
Billing
Invoices
Payment methods
Azure subscriptions
Reservations
Scope: Azure subscription 1 (change)
Resource groups
Customize
Download
Filter rows
Jun 2021
Showing 2 resource groups
Name
Subscription
Total
rg-caio-ai
azure subscription 1
R$181.99
Name
Type
Location
Tags
Total
qna-maker-caio
App Service plan
us east
--
R$55.2
machinelearning-caio
Machine Learning
br south, intercontinental
--
R$53.14
machinelearning-ai1006
Machine Learning
br south, intercontinental
--
R$52.51
srv-caio-001 / sql-caio-ai1
SQL server
br south
--
R$20.64
qna-maker-caio-ai
Application Insights app
us east
--
R$0.42
machinelearnin948838409
Storage account
br south
--
R$0.05
bot-handle6
Bot Service
unknown
--
R$0.02
machinelearnin072001392
Storage account
br south
--
R$0.01
1 to 2 of 2
Page 1 of 1
How would you rate the cost analysis preview?
Please do not forget to check your subscription costs.

Data Architecture so far…



References

Official Prep Training Exam **AI-102: Designing and Implementing a Microsoft Azure AI Solution**

Azure Portal, https://portal.azure.com/#home

Azure **Cognitive Services**, https://azure.microsoft.com/en-ca/services/cognitive-services/

Azure **Machine Learning**, https://azure.microsoft.com/en-ca/services/machine-learning/

Azure **Machine Learning Studio**, https://studio.azureml.net/

Microsoft Learn, https://docs.microsoft.com/en-us/learn/

Microsoft Learn, Course AI-102T00: Designing and Implementing a Microsoft Azure AI Solution, https://docs.microsoft.com/en-us/learn/certifications/courses/ai-102t00

Microsoft Learn, AI-102, https://docs.microsoft.com/en-us/learn/certifications/azure-ai-engineer/

Microsoft Learn, Code Samples, https://docs.microsoft.com/en-us/samples/browse/

Microsoft Official **Git Hub, AI-Basic examples for all APIs**, https://github.com/MicrosoftDocs/ai-fundamentals

Microsoft Official **Git Hub, Microsoft learn AI-900**, https://github.com/MicrosoftLearning/mslearn-ai900

Microsoft Official **Git Hub, Microsoft learn AI-102**, https://github.com/MicrosoftLearning/AI-102-AIEngineer

Thank you! ;-)

Please e-mail me the screenshots of your final steps for each component / service to validate your bonus points.