TUGAS ASINKRON 1 MINGGU 2 KELAS DAI-002
Explore Computer Vision in Microsoft Azure

Oleh :

KELOMPOK 4

- Dhea Amalia Ariantoputri
- Dinda Ega Fajarwati
- Fadhil Rausyanfikr
- Dwira Kurnia Larasati
- Renaka Agusta
- Eka Aperinda Putri
- Dinna Nurfadlillah

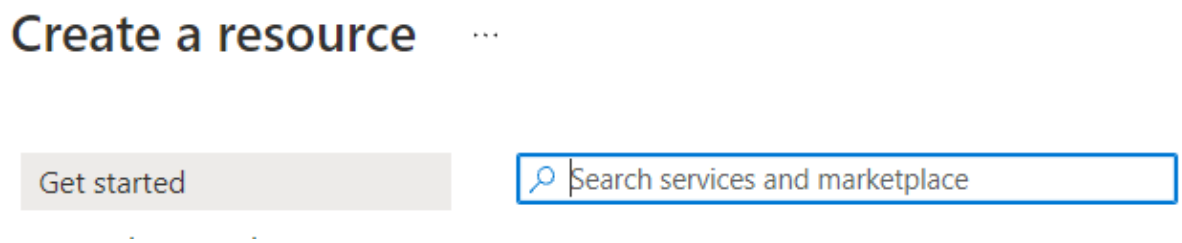
Mentor :
Noviyanti Tri Maretta Sagala, S.TI., M.Sc

Asisten Mentor :
Ainul Zakiy

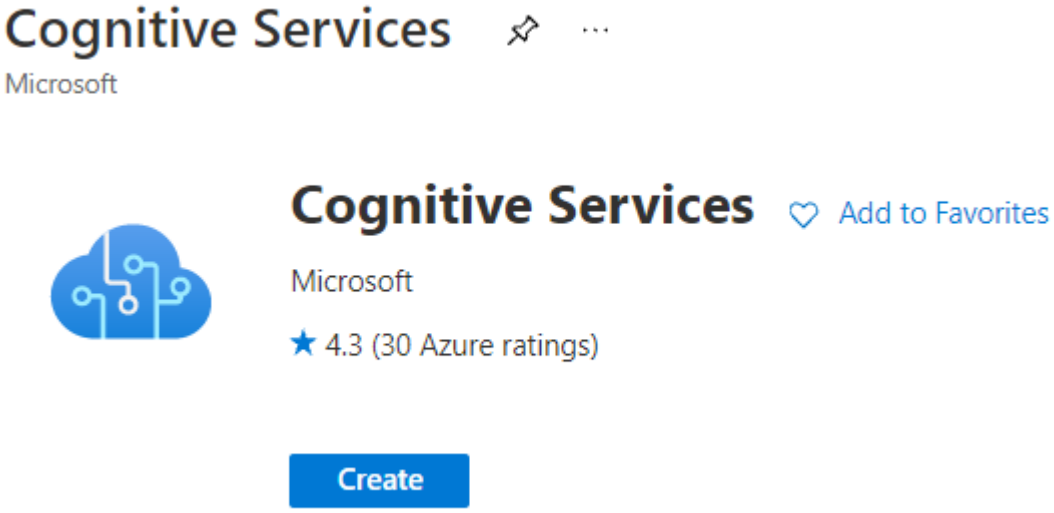
STUDI INDEPENDEN PT. MICROSOFT INDONESIA
OKTOBER 2021

PERCOBAAN 1

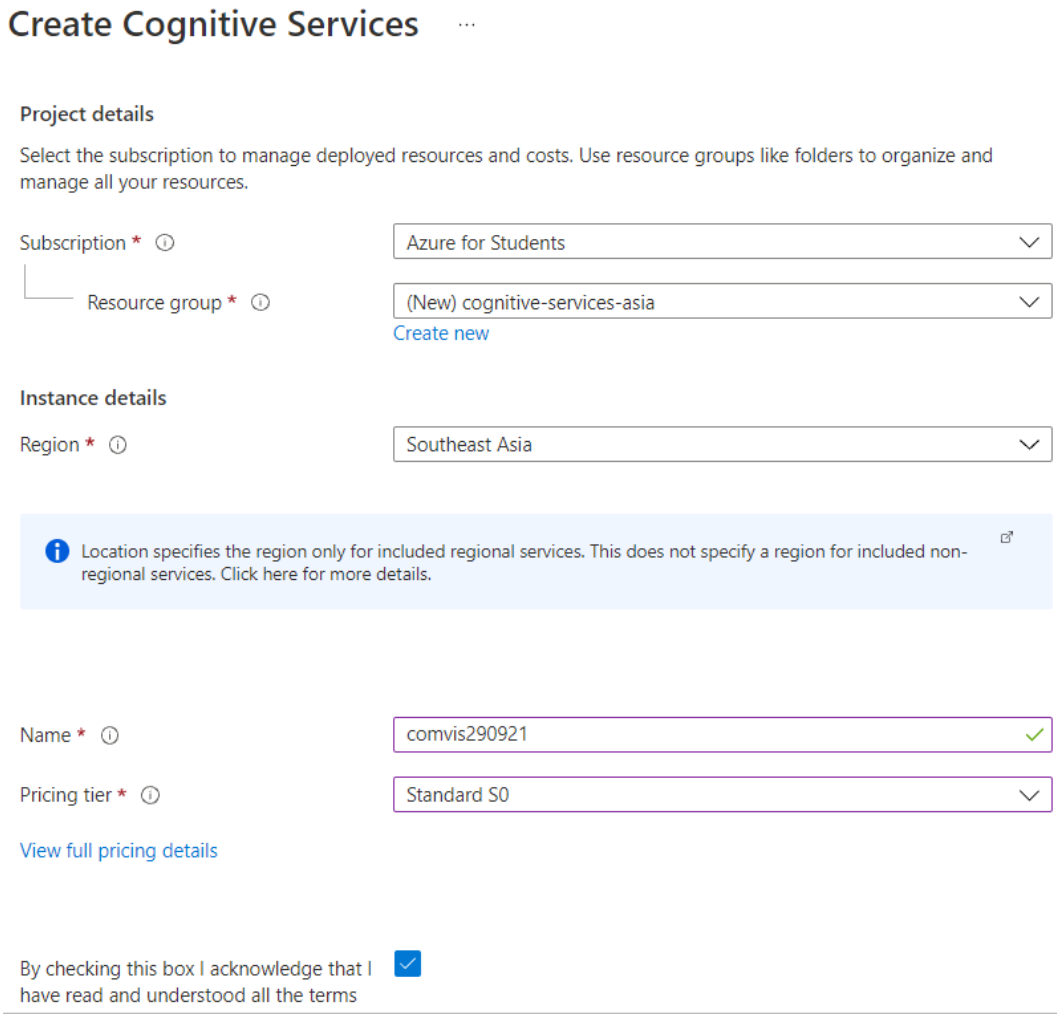
Analyze images with the Computer Vision service



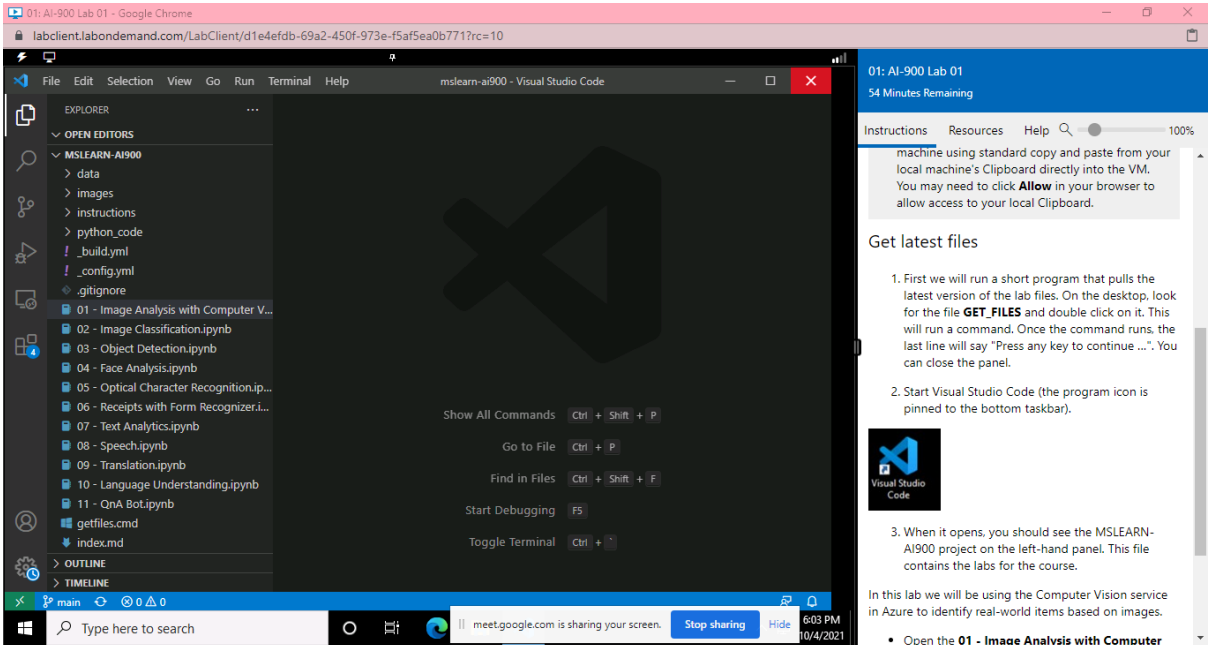
1. Langkah pertama adalah membuat resource group pada portal Azure.



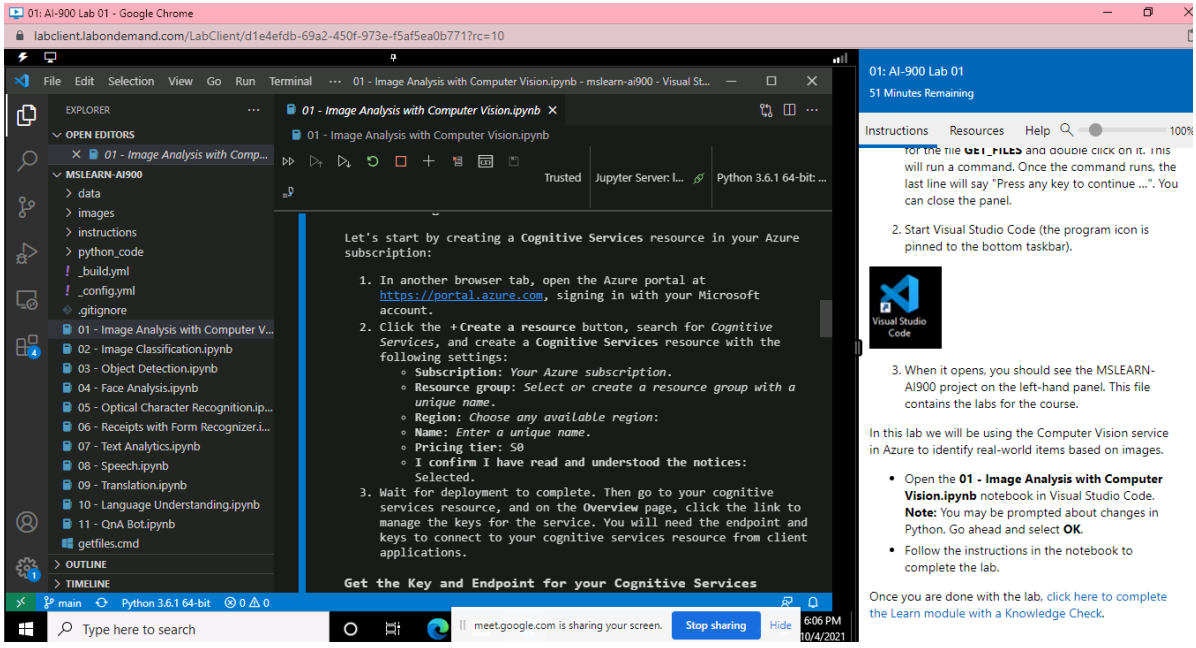
2. Selanjutnya, pilih “Cognitive Services” dan klik “Create”. Resource ini dipilih karena meliputi fitur-fitur kognitif seperti Text Analytics, Translator Text, dll. Resource type ini nantinya kemudian akan menginformasikan dua hal penting yaitu “key” dan “endpoint”.



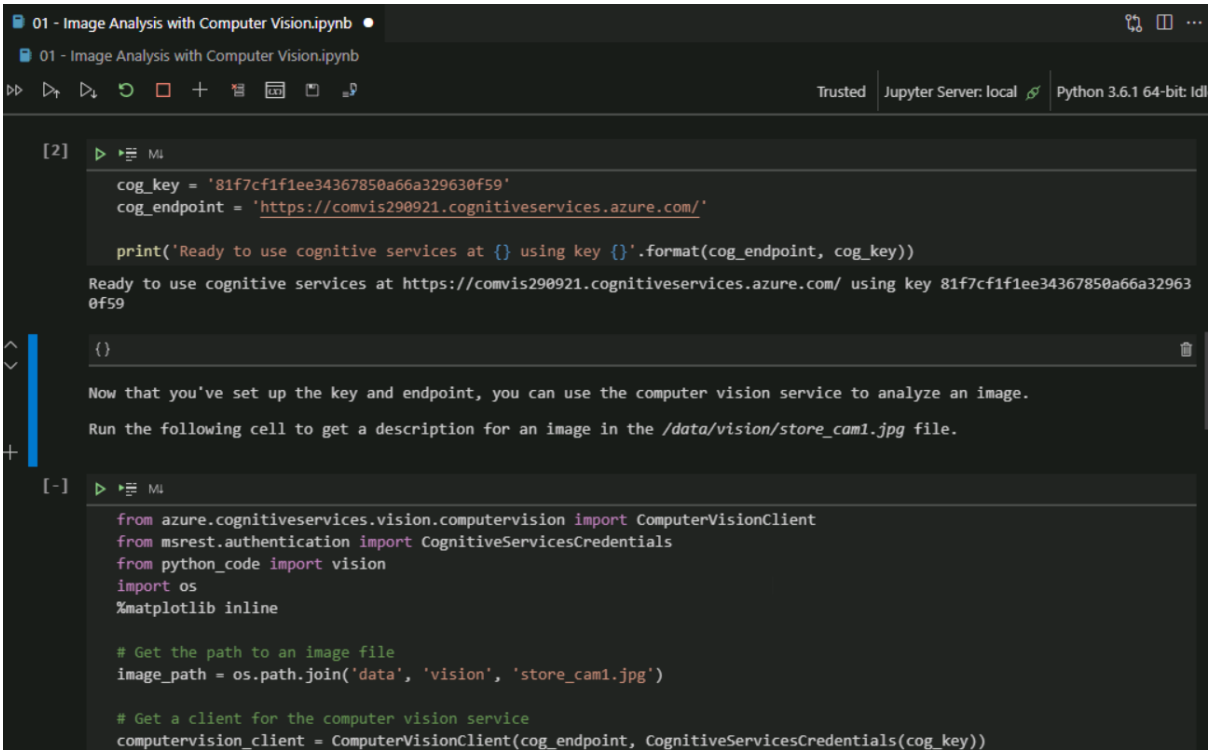
3. Ketika sudah mengklik cognitive services, buat cognitive service. Masukkan detail informasi yang diminta.



4. Pada awal launch lab pada microsoft learn, virtual machine akan muncul dan ketika sudah masukan password, kemudian klik GET_FILES dan tunggu hingga selesai running. Jika sudah, maka klik Visual Studio Code dan tertampil seperti gambar diatas.



5. Klik Image Analysis with Computer Vision.ipynb pada panel kiri, dan muncul seperti gambar diatas, ikuti langkah-langkahnya. Ketentuan untuk membuat cognitive services pada gambar nomor 3



6. Ketika pembuatan resource telah selesai, klik go to resource dan klik key and endpoints. Lalu copy paste kan key dan endpoint nya ke notebook Visual Studio Code seperti gambar diatas, kemudian klik RUN.

```python
from azure.cognitiveservices.vision.computervision import ComputerVisionClient
from msrest.authentication import CognitiveServicesCredentials
from python_code import vision
import os
%matplotlib inline

# Get the path to an image file
image_path = os.path.join('data', 'vision', 'store_cam1.jpg')

# Get a client for the computer vision service
computervision_client = ComputerVisionClient(cog_endpoint, CognitiveServicesCredentials(cog_key))

# Get a description from the computer vision service
image_stream = open(image_path, "rb")
description = computervision_client.describe_image_in_stream(image_stream)

# Display image and caption (code in helper_scripts/vision.py)
vision.show_image_caption(image_path, description)
```

7. Code pada IDE python seperti gambar diatas dijalankan (klik tombol RUN).

8. Hasil dari code nomor 7 menghasilkan gambar seperti diatas

```
# Get the path to an image file
image_path = os.path.join('data', 'vision', 'store_cam1.jpg')

# Specify the features we want to analyze
features = ['Description', 'Tags', 'Adult', 'Objects', 'Faces']

# Get an analysis from the computer vision service
image_stream = open(image_path, "rb")
analysis = computervision_client.analyze_image_in_stream(image_stream, visual_features=features)

# Show the results of analysis (code in helper_scripts/vision.py)
vision.show_image_analysis(image_path, analysis)
```

9. Kemudian jalankan code diatas, klik RUN

```
vision.show_image_analysis(image_path, analysis)
```

Ratings:
- Adult: False
- Racy: False
- Gore: False

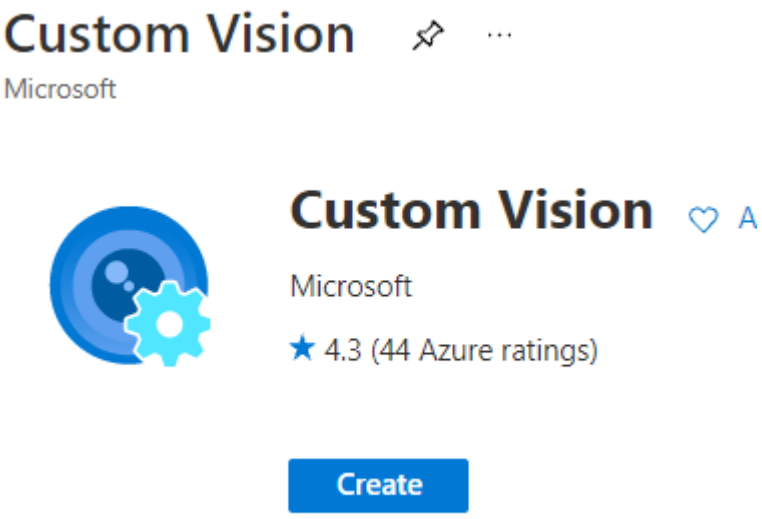
Tags:
- person
- clothing
- shop
- convenience store
- retail
- woman
- store
- text
- customer
- supermarket
- bottle
- grocery store
- human face
- shopping
- trade
- outdoor
- shelf
- indoor
- girl
- lady
- street

10. hasil RUN dari code nomor 9 adalah seperti gambar diatas. terlihat gambar sudah di specify descriptionnya, tags nya, adult (dewasa)-nya, dan facesnya (wajah, untuk menentukan umur).

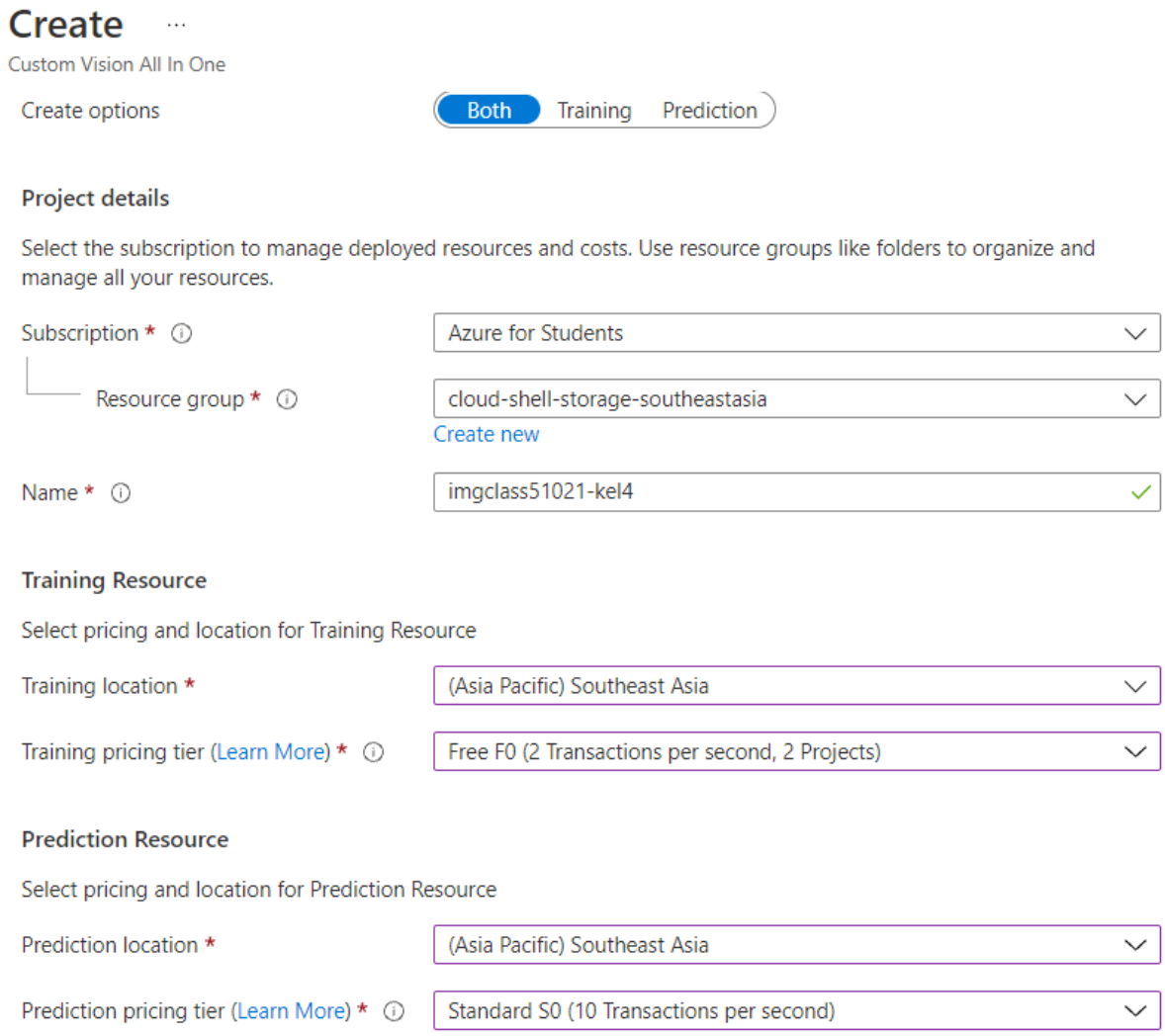
PERCOBAAN 2

Classify images with the Custom Vision service

Seperti percobaan lab sebelumnya, pada microsoft learn klik launch lab, dan menunggu virtual machine terbuat. ketika sudah terbuat, lakukan login dan klik GET_FILES. kalau sudah, klik Visual studio Code, dan klik 02 - Image Classification.ipynb dan lakukan langkah-langkah pada notebook tersebut. Di lab ini kita akan menggunakan layanan Custom Vision di Azure untuk mengidentifikasi sati item dunia nyata dalam sebuah gambar.

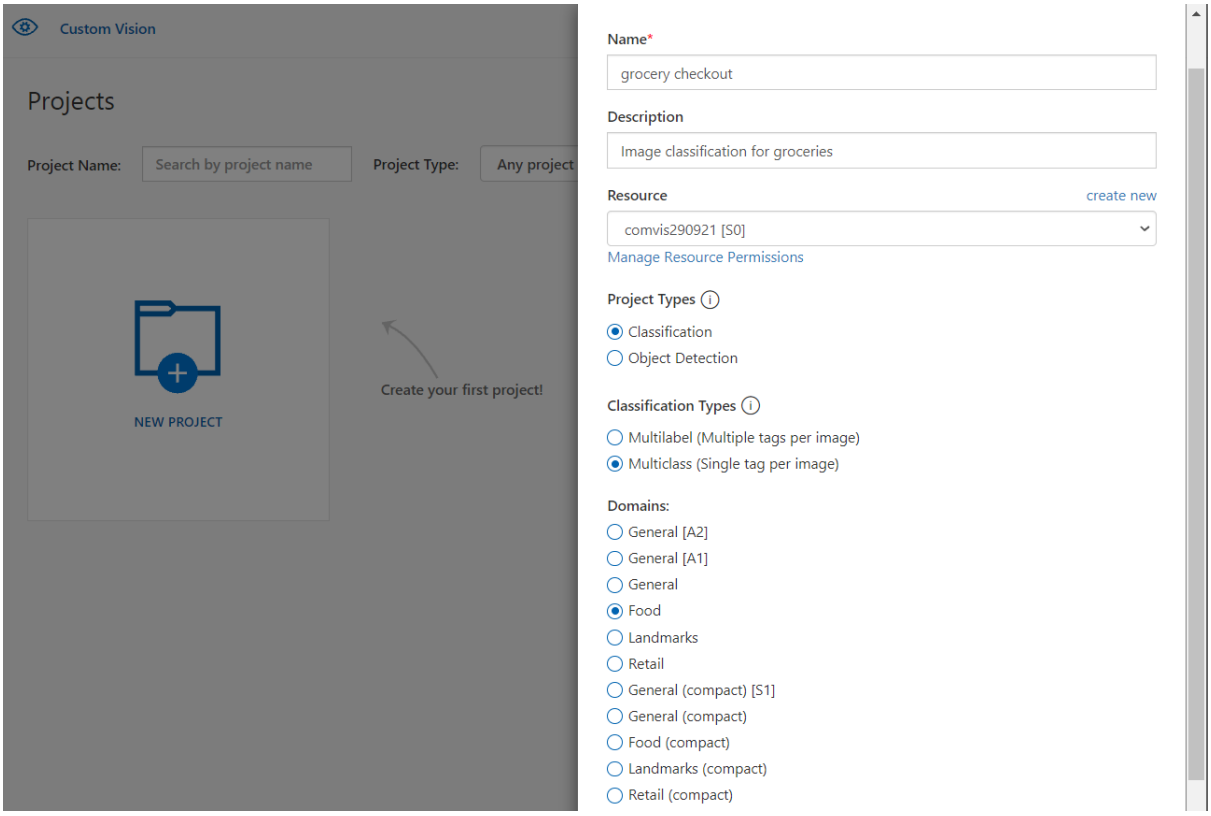


1. Masuk ke akun portal azure. Di Azure, pilih Custom Vision seperti pilihan pada gambar



2. Kemudian pilih tombol create a resource, cari custom vision dan buat custom vision resource dengan pengaturan sebagai berikut:
   - **Create options**: Both
   - **Subscription**: *Azure for student,* (karena kami menggunakan layanan untuk Azure pelajar)
   - **Resource group**: *cloud-shell-storage-southeastasia*
   - **Name**: *imgclass51021-kel4*
   - **Training location**: *(Asia Pacific) Southeast Asia*
   - **Training pricing tier**: F0
   - **Prediction location**: *(Asia Pacific) Southeast Asia*
   - **Prediction pricing tier**: F0

   Kemudian klik create dan tunggu hingga resource dibuat dan perhatikan bahwa dua custom vision telah tersedia, yang satu untuk pelatihan dan satu lagi untuk prediksi



3. Pada custom vision portal buat proyek baru dengan pengaturan sebagai berikut :
   - **Name**: Grocery Checkout
   - **Description**: Image classification for groceries
   - **Resource**: *comvis290921 [SO]*
   - **Project Types**: Classification
   - **Classification Types**: Multiclass (single tag per image)
   - **Domains**: Food

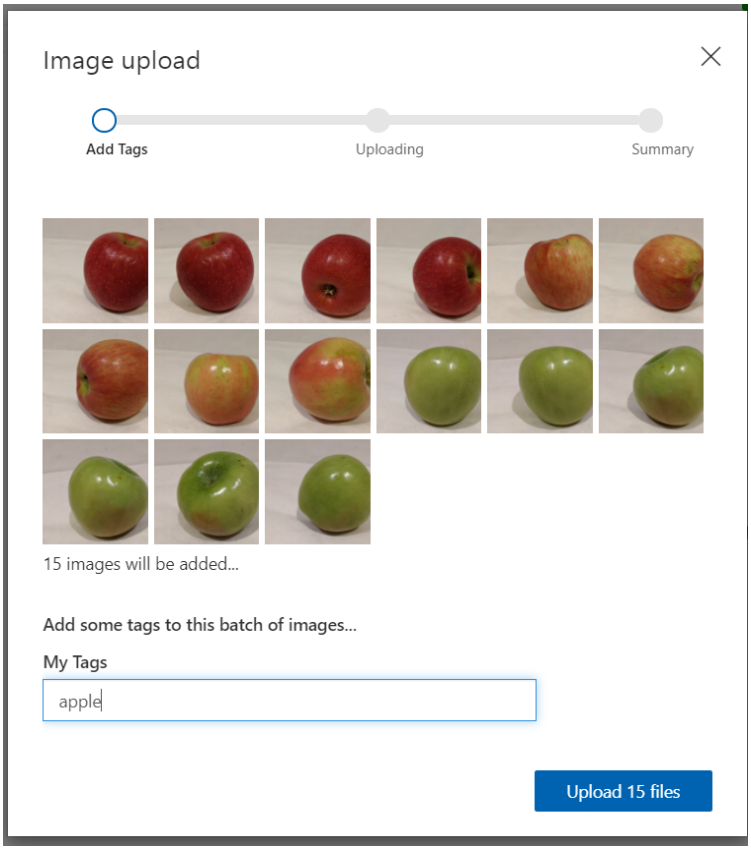
Looks like you don't have any images here!

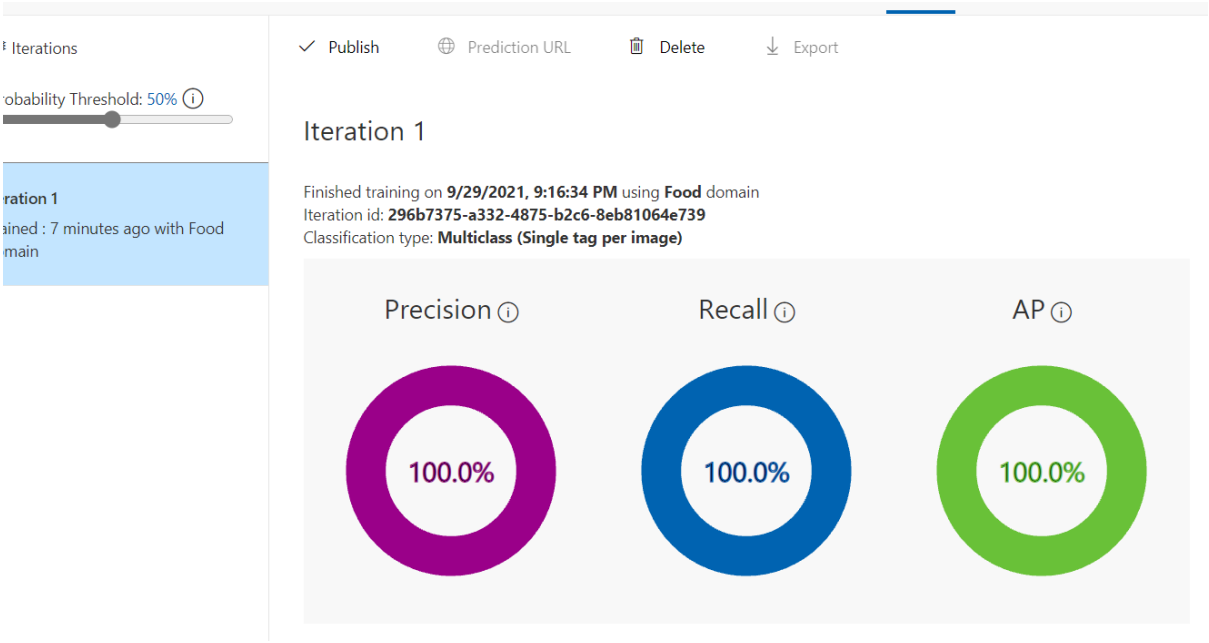
Go ahead and browse for images to upload to your project, tag them, and they will be ready to be trained.

Add images

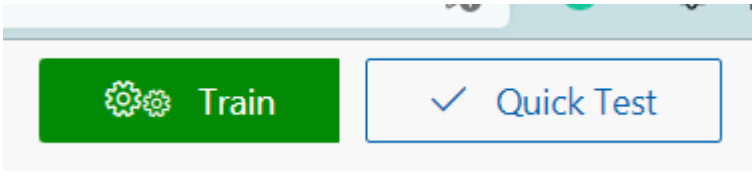
.JPG, .PNG, .BMP format, up to 6 MB per image

4. Klik tambahkan gambar dan kami memilih semua gambar apel yang sudah diekstrak sebelumnya kemudian unggah. Lalu berikan tag nama “apple” seperti gambar dibawah.

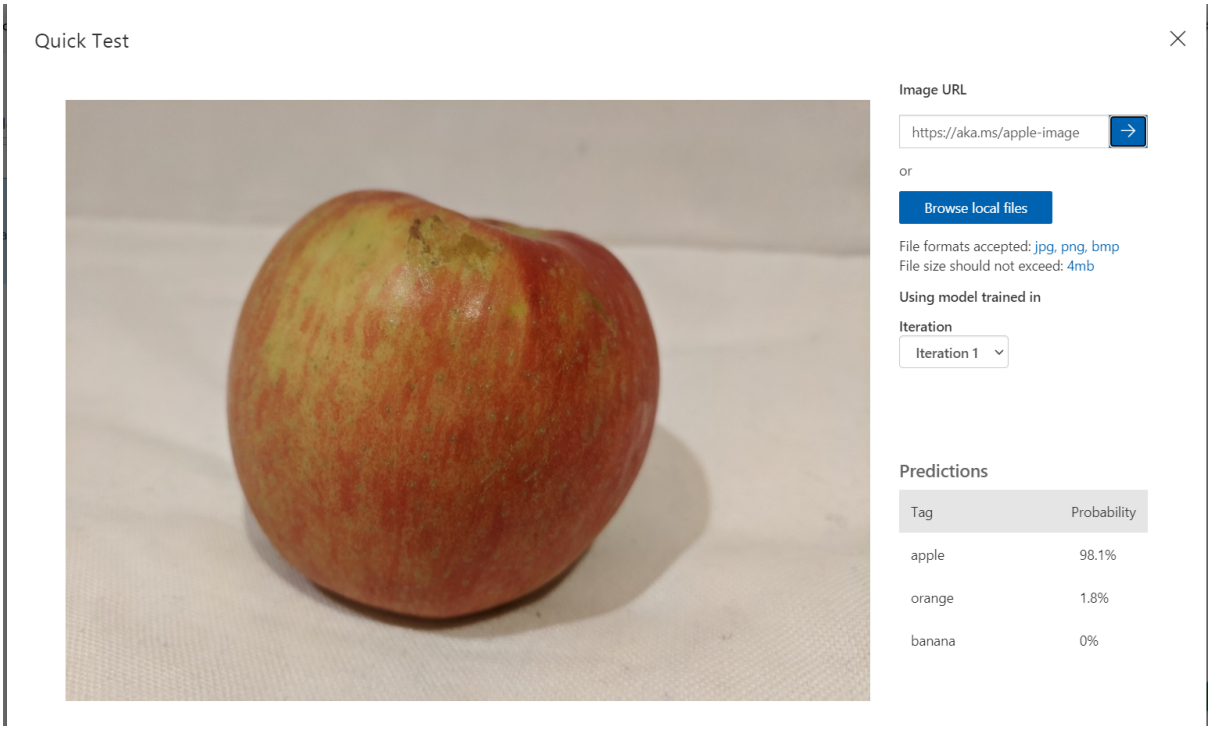




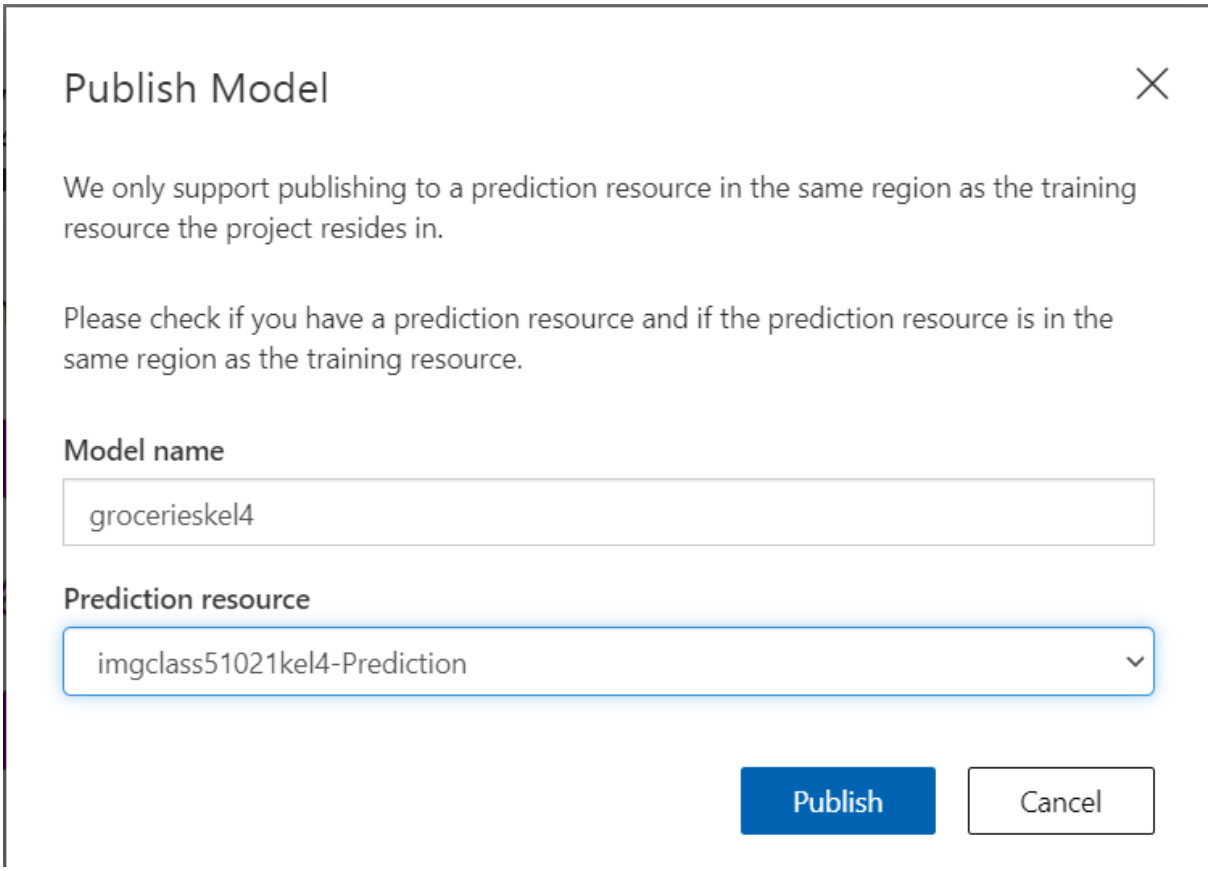
5. Selanjutnya, dalam custom vision project klik train untuk melatih model klasifikasi yang telah diberi tag lalu klik publish



6. Setelah melakukan training selanjutnya melakukan quick test untuk menguji hasil training



7. Hasil quick test menunjukan bahwa pada gambar tersebut terdeteksi terdapat objek apel dengan probabilitas tertinggi yaitu 98,1%



8. Supaya model dapat dimanfaatkan secara luas, maka kami melakukan publikasi model. Beri nama model dan pilih prediction resource yang sudah di buat sebelumnya

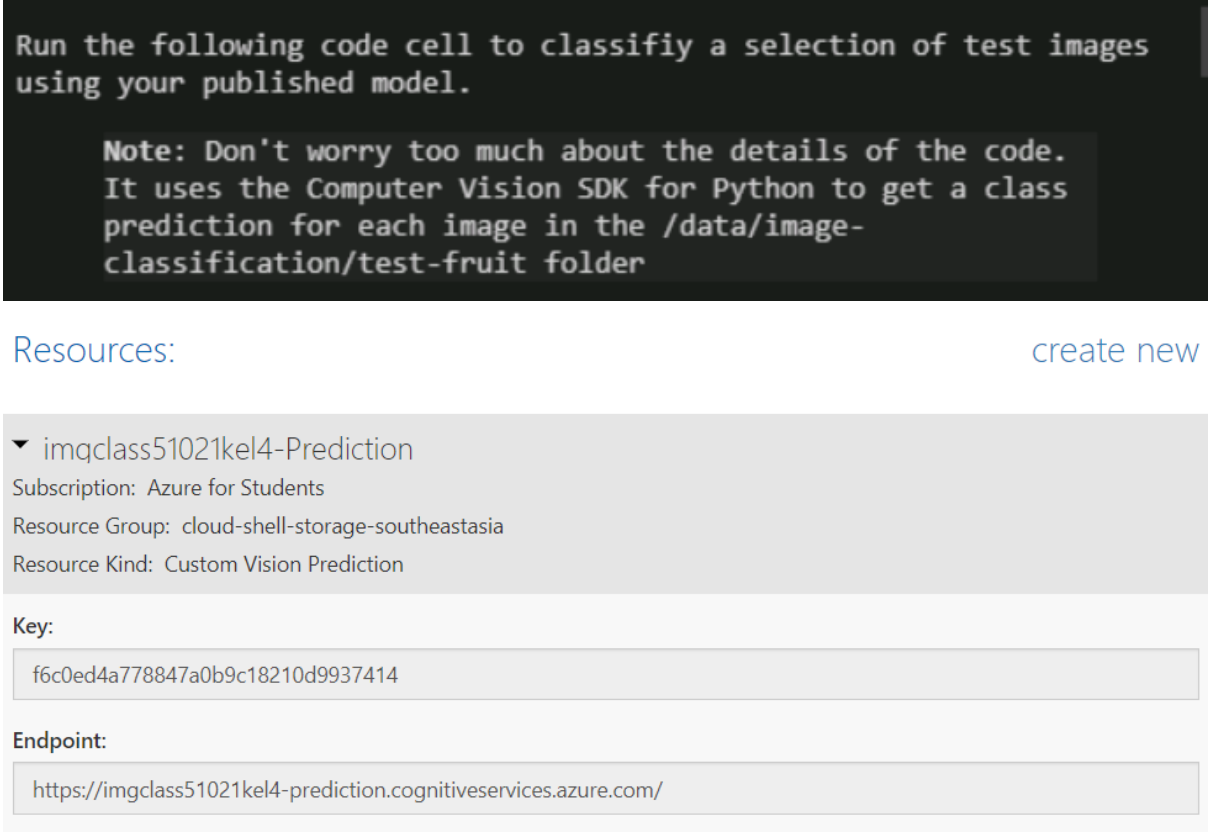
```
project_id = '1a89172f-cfc5-4506-95ec-06548db7eea5'
cv_key = 'f6c0ed4a778847a0b9c18210d9937414'
cv_endpoint = 'https://
imgclass51021kel4-prediction.cognitiveservices.azure.com/'

model_name = 'groceries kel4' # this must match the model
name you set when publishing your model iteration (it's
case-sensitive)!
print('Ready to predict using model {} in project {}'.format
(model_name, project_id))
```

{}

Now you can use your key and endpoint with a Custom Vision client to connect to your custom vision classification model.

Run the following code cell to classifiy a selection of test images using your published model.

> **Note:** Don't worry too much about the details of the code. It uses the Computer Vision SDK for Python to get a class prediction for each image in the /data/image-classification/test-fruit folder

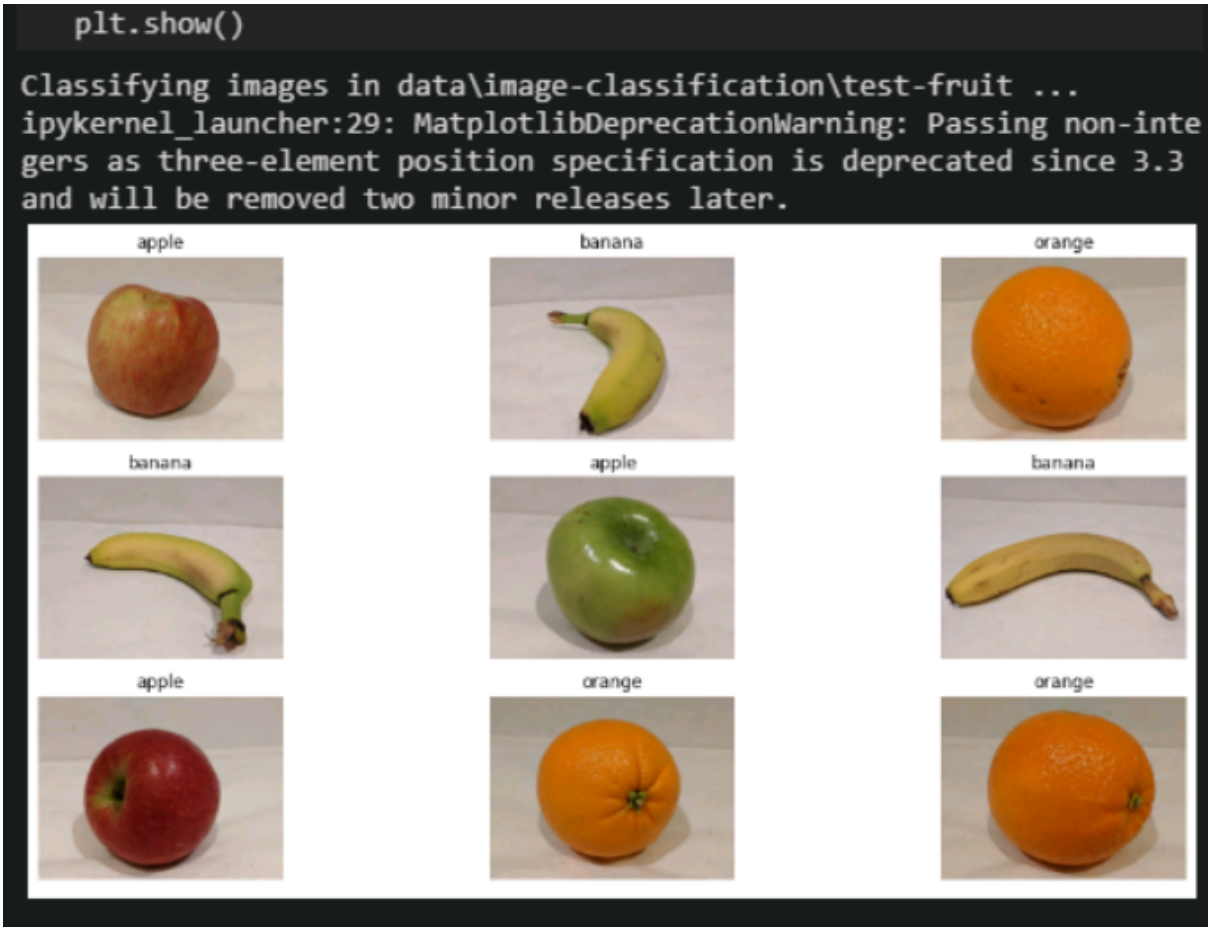


9. Untuk memeriksa apakah program berjalan, jalankan kode seperti diatas, untuk yang menggunakan custom vision key dan endpointnya menggunakan endpoint prediction beserta key nya juga. Untuk project id dapat dilihat di project lalu klik gambar setting. Pastikan model name yang ditulis sudah tepat dengan yang kita buat karena itu merupakan hal yang sensitif.

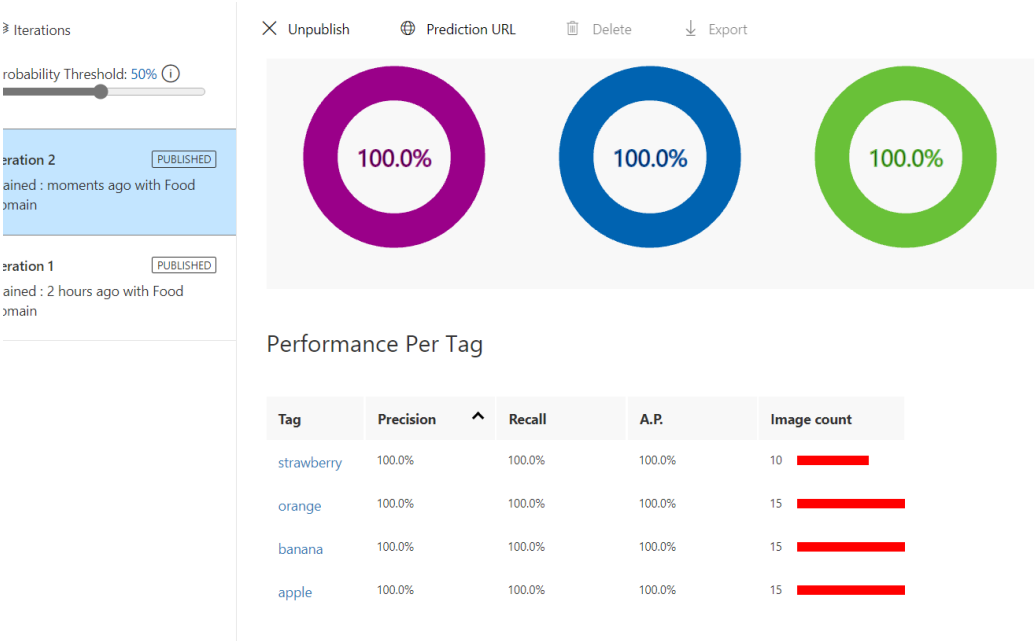
```
from azure.cognitiveservices.vision.customvision.prediction
import CustomVisionPredictionClient
from msrest.authentication import ApiKeyCredentials
import matplotlib.pyplot as plt
from PIL import Image
import os
%matplotlib inline

# Get the test images from the data/vision/test folder
test_folder = os.path.join('data', 'image-classification',
'test-fruit')
test_images = os.listdir(test_folder)

# Create an instance of the prediction service
credentials = ApiKeyCredentials(in_headers=
{"Prediction-key": cv_key})
custom_vision_client = CustomVisionPredictionClient
(endpoint=cv_endpoint, credentials=credentials)

# Create a figure to display the results
fig = plt.figure(figsize=(16, 8))

# Get the images and show the predicted classes for each one
print('Classifying images in {} ...'.format(test_folder))
```

```
for i in range(len(test_images)):
    # Open the image, and use the custom vision model to
classify it
    image_contents = open(os.path.join(test_folder,
test_images[i]), "rb")
    classification = custom_vision_client.classify_image
(project_id, model_name, image_contents.read())
    # The results include a prediction for each tag, in
descending order of probability - get the first one
    prediction = classification.predictions[0].tag_name
    # Display the image with its predicted class
    img = Image.open(os.path.join(test_folder, test_images
[i]))
    a=fig.add_subplot(len(test_images)/3, 3,i+1)
    a.axis('off')
    imgplot = plt.imshow(img)
    a.set_title(prediction)
plt.show()
```

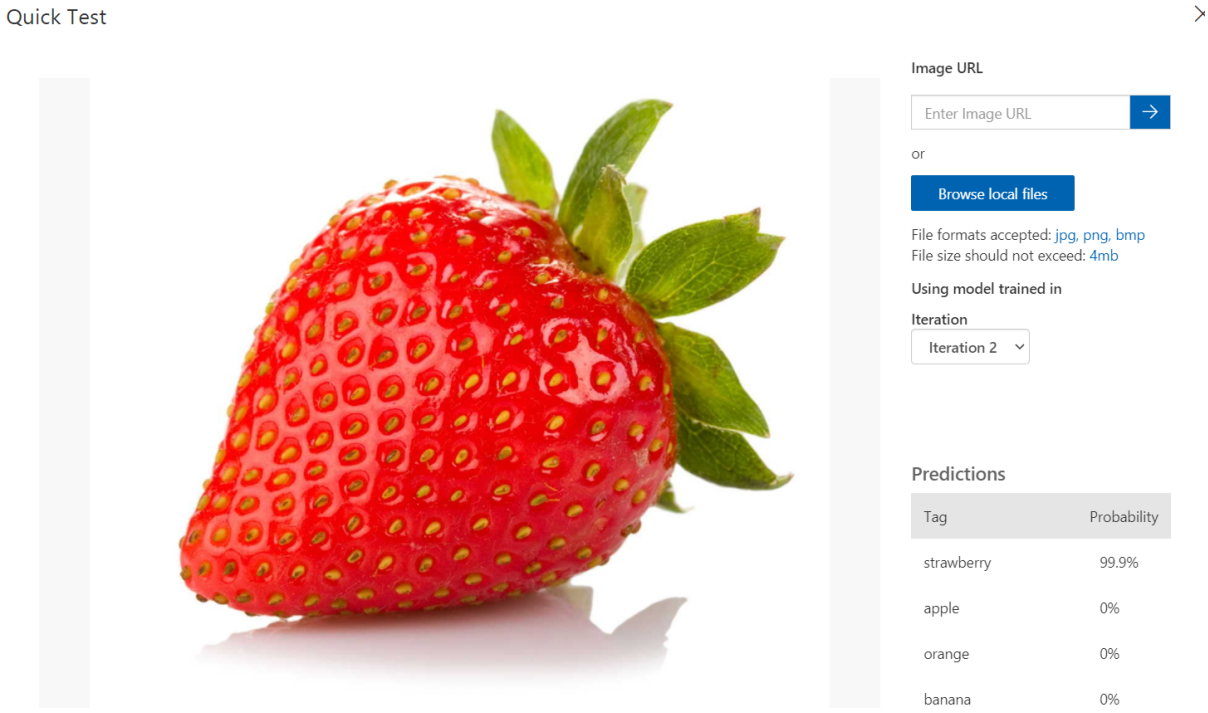
10. Selanjutnya, jalankan code yang tersedia



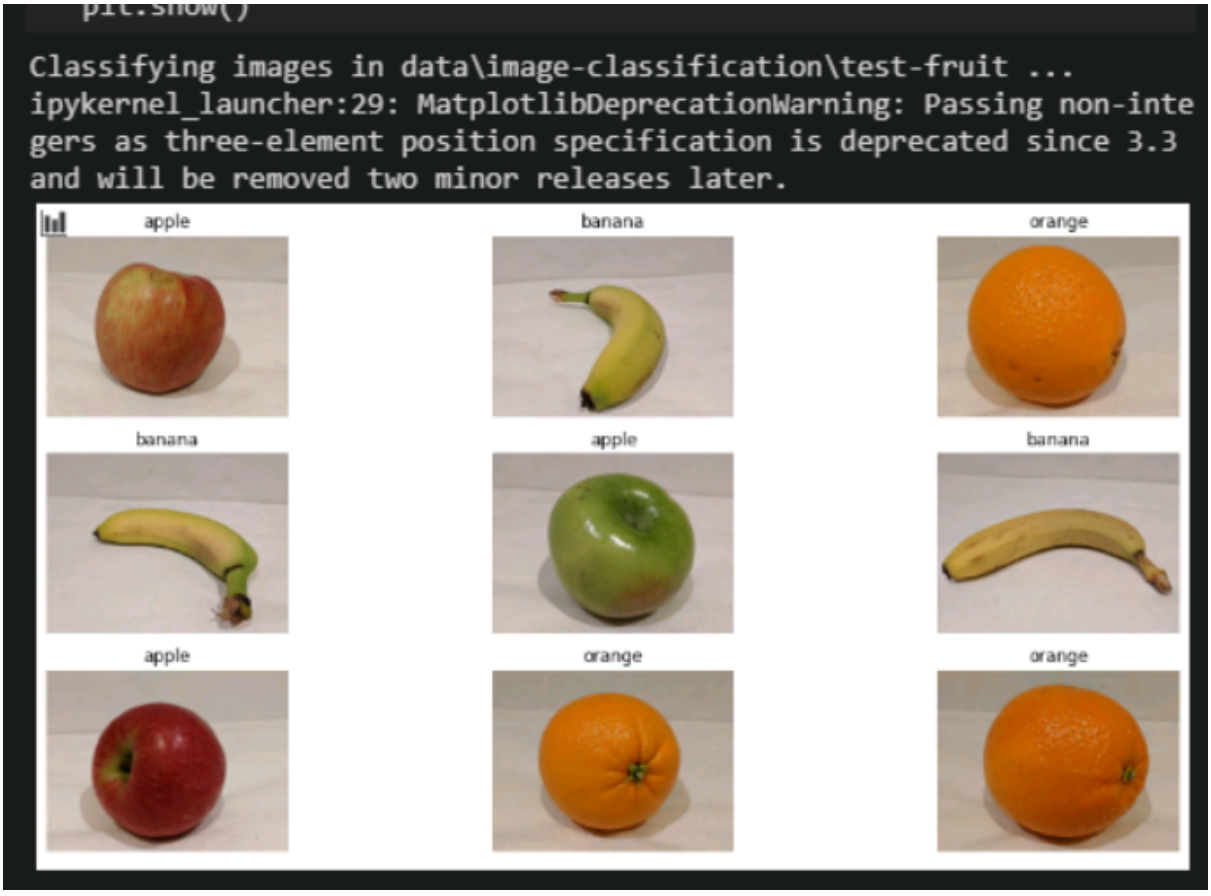
11. Maka akan muncul klasifikasi gambar yang sudah sesuai dengan jenis nya



12. Setelah percobaan persis seperti di module telah selesai, kami mencoba melakukan sedikit modifikasi. Kami mencoba menambahkan 1 buah baru yaitu strawberry. Langkah percobaan pun diulangi. Dengan modifikasi tambahan seperti jumlah data-set yang berbeda (hanya menggunakan 10 foto strawberry), kami pun men-train model. Hasilnya ada pada gambar di atas.



13. Setelah berhasil menambah data-set dan juga melatih model, kami juga menjalankan quick-test dengan tujuan mengetahui apakah model sudah mampu mengenali foto strawberry yang diuji, **gambar diambil dari local files** dan hasilnya adalah ada pada gambar. Strawberry berhasil terdeteksi dengan probability 99.9%.

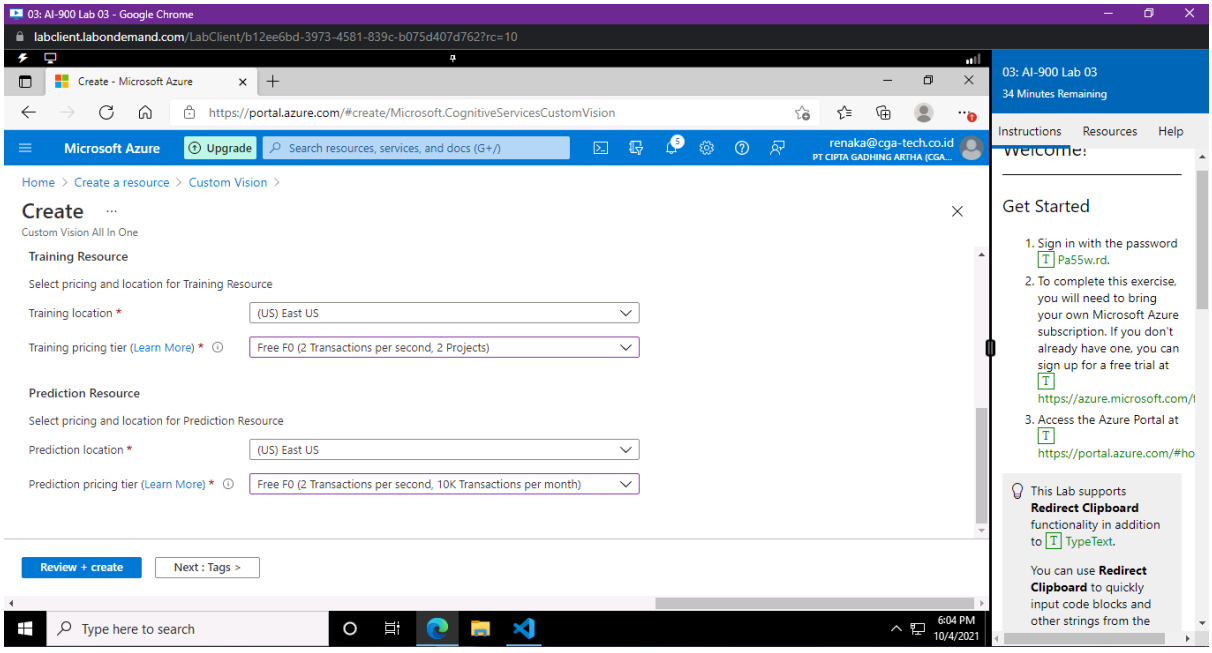


14. Gambar di atas adalah hasil dari proses paling terakhir yaitu setelah publish. Dalam proses melabeli buah secara satu per satu, hasil modifikasinya tidak dapat melabeli gambar strawberry dengan label “strawberry”. Penyebabnya adalah kemungkinan besar ada pada kesalahan dimana data-test belum disesuaikan dengan data training yang baru. Jadi, di data training strawberrynya sudah ada, tetapi di data untuk testnya belum ada (masih menggunakan data test bawaan yang hanya ada banana, orange, dan apple).

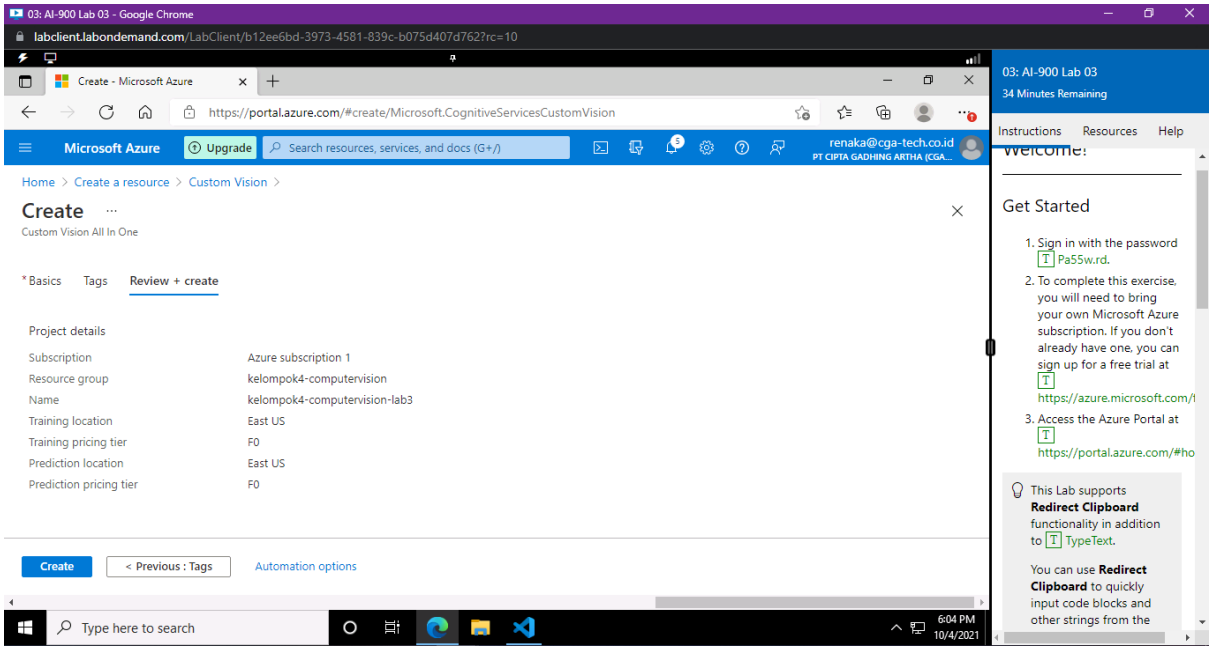
PERCOBAAN 3

Detect objects in images with the Custom Vision service

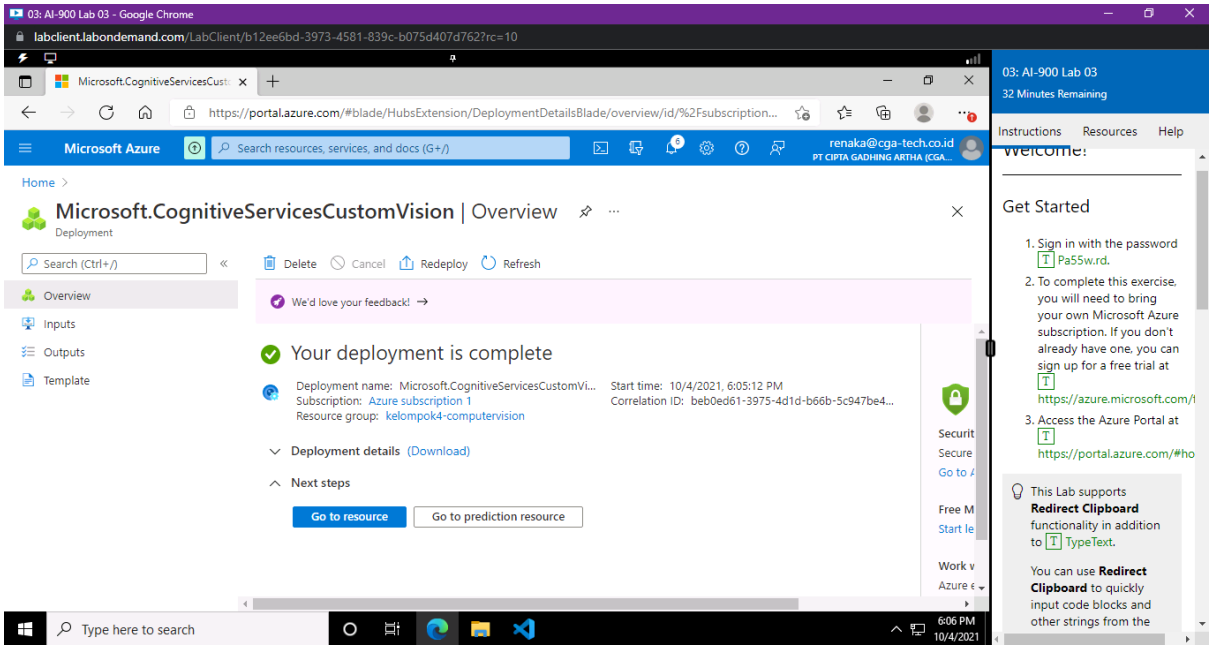
Seperti percobaan lab sebelumnya, pada microsoft learn klik launch lab, dan menunggu virtual machine terbuat. ketika sudah terbuat, lakukan login dan klik GET_FILES. kalau sudah, klik Visual studio Code, dan klik 03 - Object Detection.ipynb dan lakukan langkah-langkah pada notebook tersebut. Di lab ini kita akan menggunakan layanan custom vision pada Azure untuk mendeteksi objek dalam gambar.



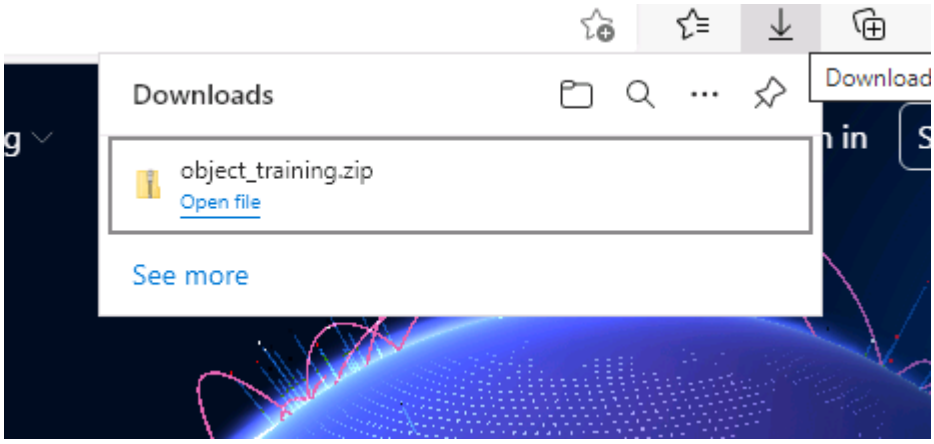
1. Membuat Custom Vision project dengan nama “kelompok4-customvision-lab3” dan melakukan konfigurasi terkait dengan training pricing dan prediction pricing



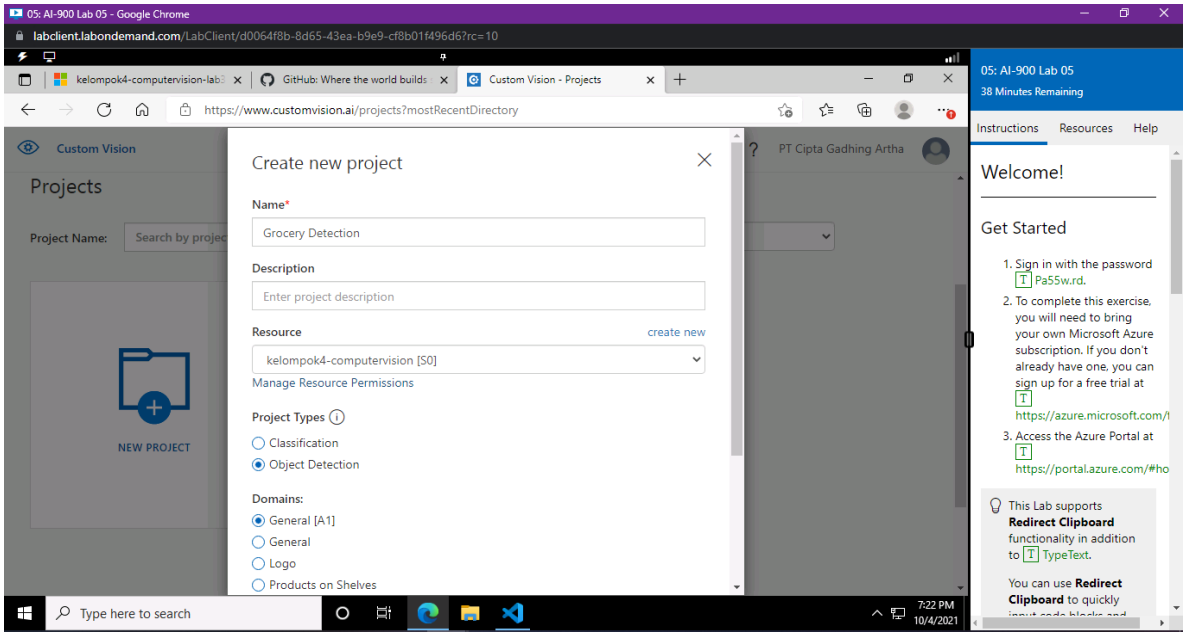
2. Melakukan review untuk validasi dan mengcreate service Custom Vision



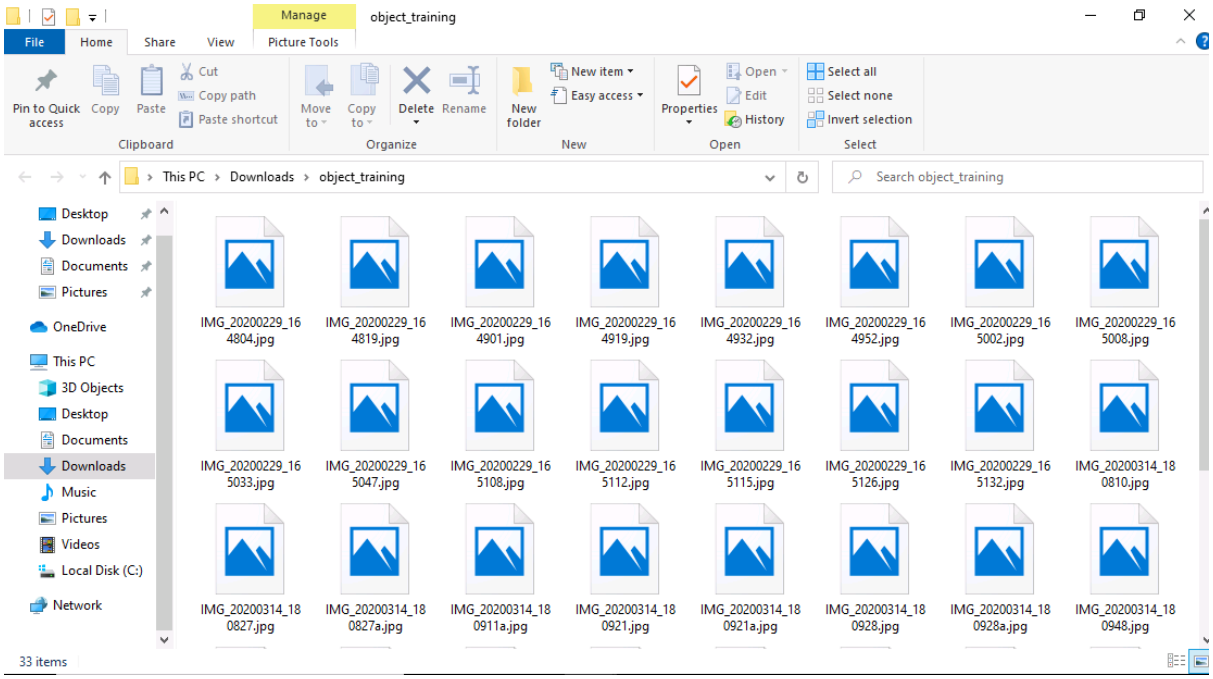
3. Menunggu hingga proses deployment selesai



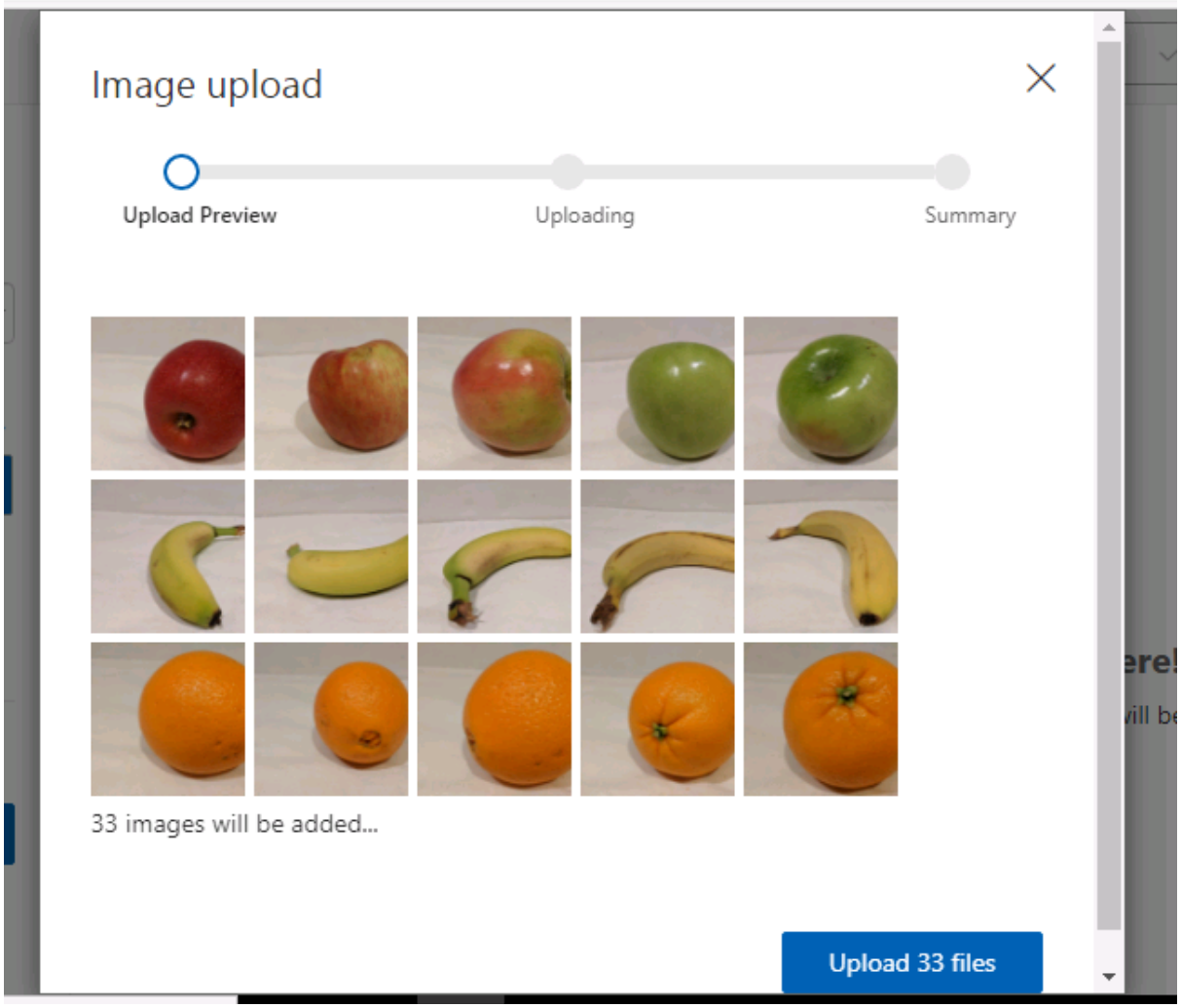
4. Mengunduh dataset gambar untuk dilakukan training



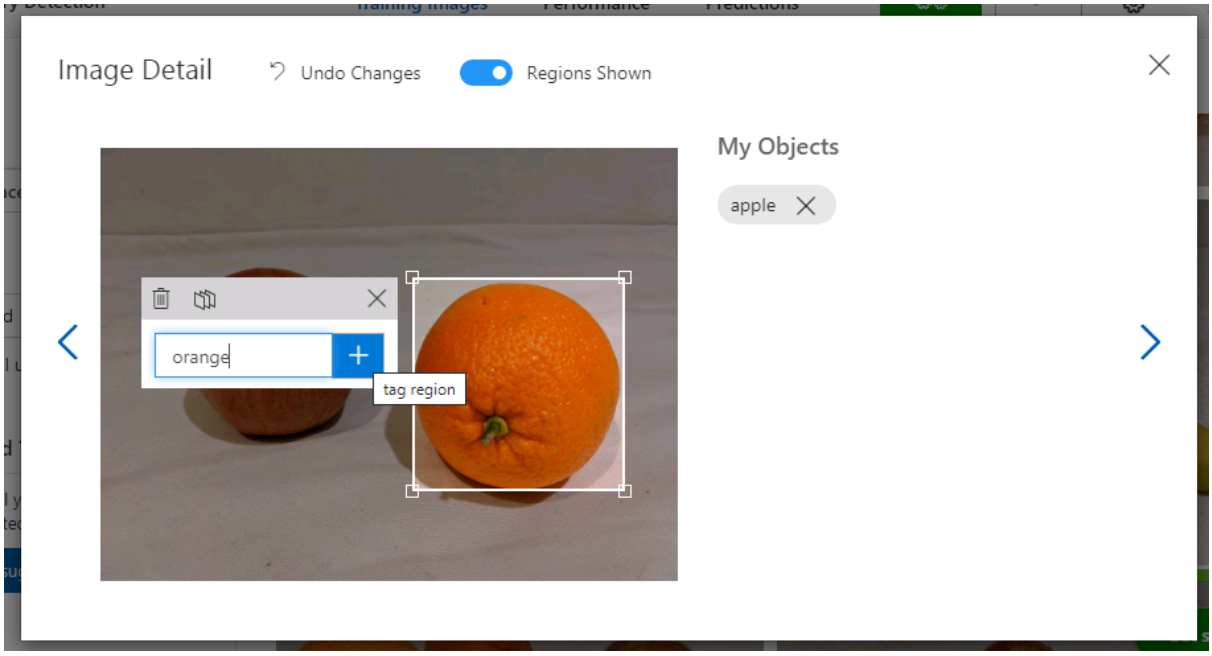
5. Membuat project baru situs www.customvision.ai dengan nama Grocery Detection



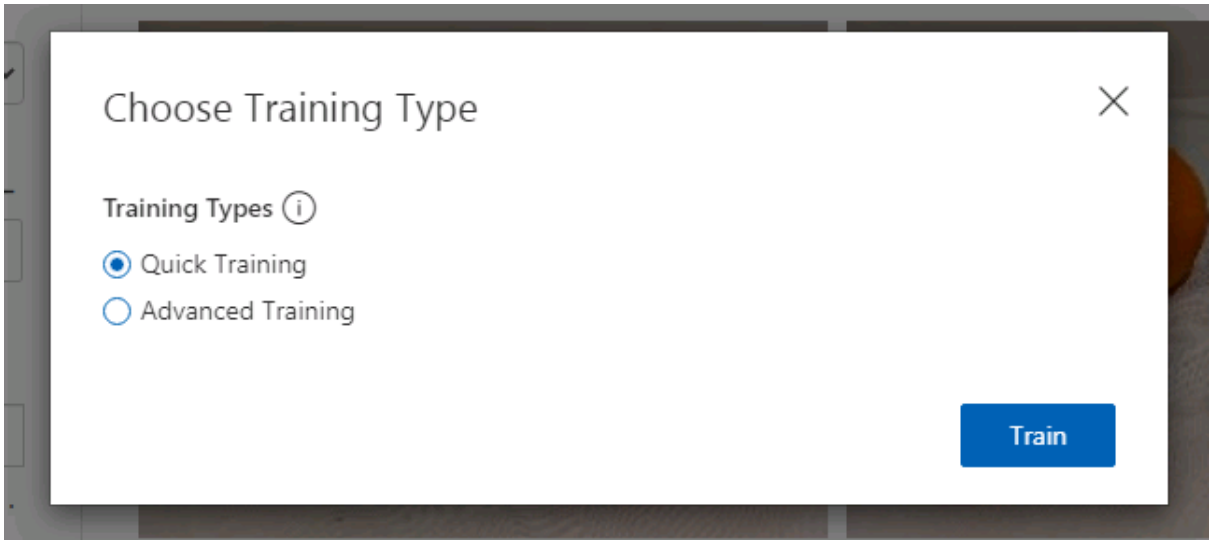
6. Mengekstrak seluruh gambar ke dalam satu folder



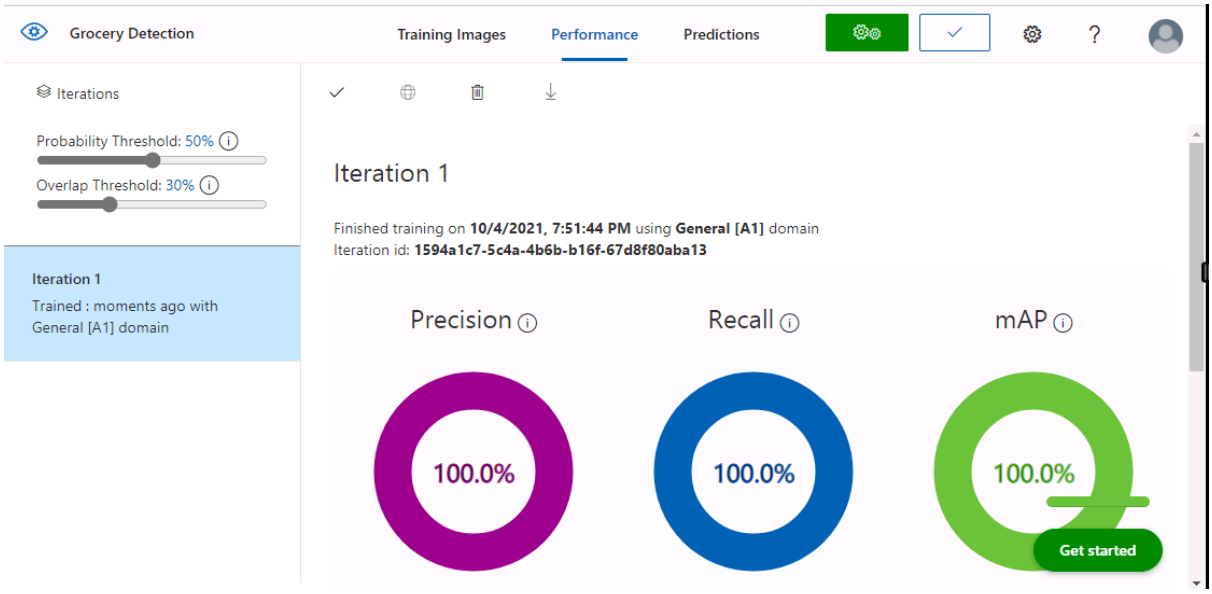
7. Mengunggah seluruh gambar dengan menekan tombol add images



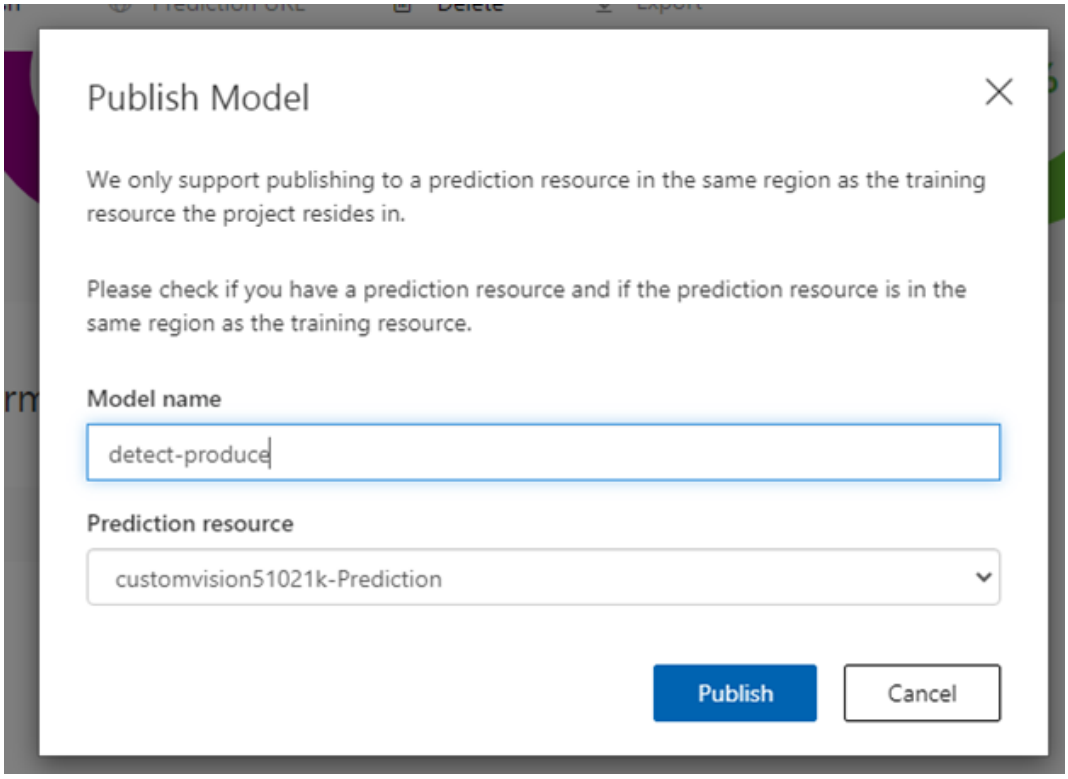
8. Melakukan proses labeling pada gambar dengan mengisi label sesuai dengan nama buah (apple/orange/banana)



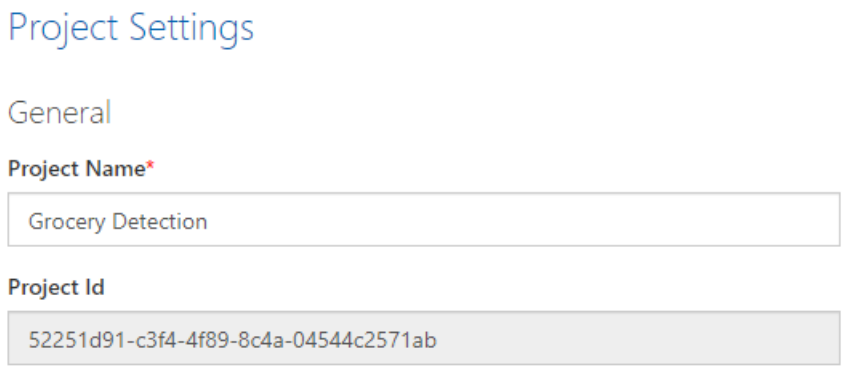
9. Melakukan proses training dengan opsi Quick Training



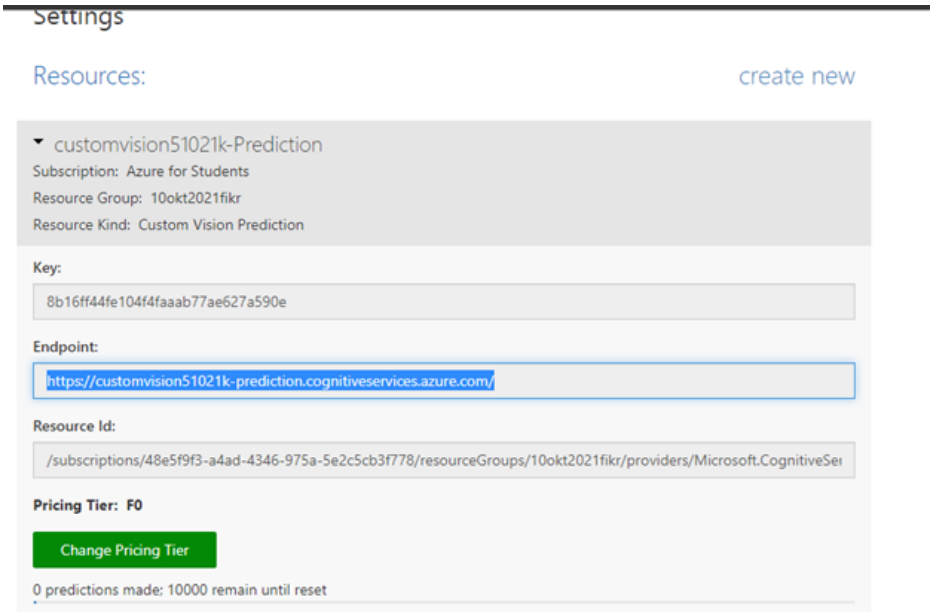
10. Setelah proses training selesai maka akan menampilkan hasil seperti diatas



11. Publish model yang sudah dibuat



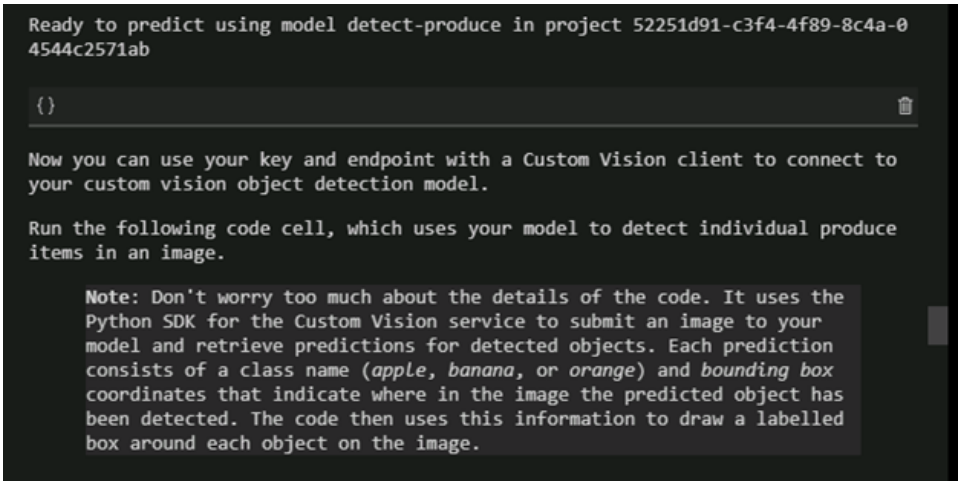
12. Di setting dapat terlihat Project Id nya



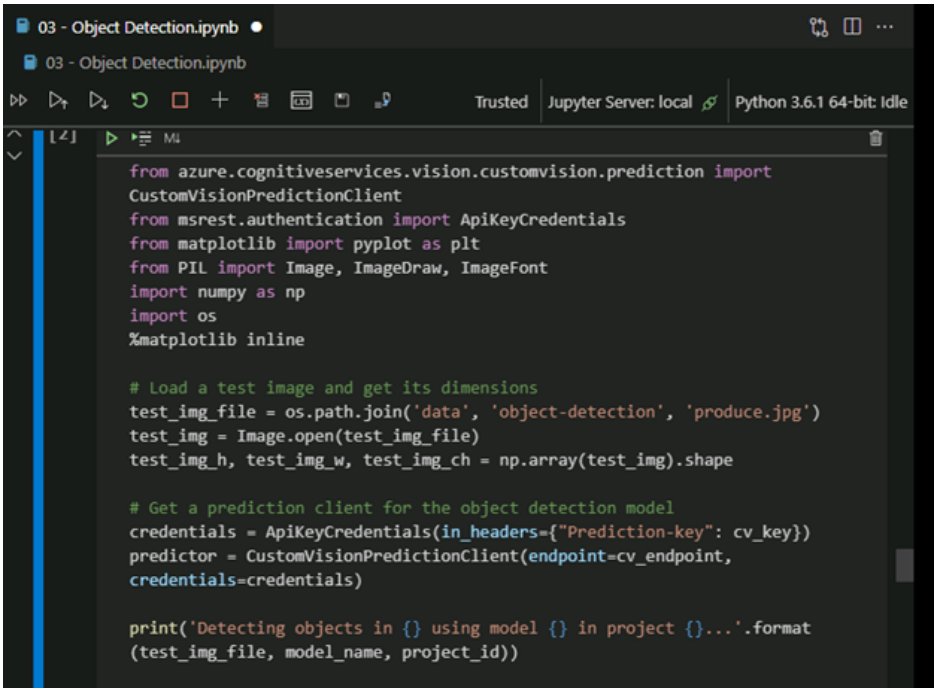
13. dari resource setting didapatkan key dan endpoint

```
project_id = '52251d91-c3f4-4f89-8c4a-04544c2571ab' # Replace with your
project ID
cv_key = '8b16ff44fe104f4faaab77ae627a590e' # Replace with your
prediction resource primary key
cv_endpoint = 'https://
customvision51021k-prediction.cognitiveservices.azure.com/' # Replace
with your prediction resource endpoint

model_name = 'detect-produce' # this must match the model name you set
when publishing your model iteration exactly (including case)!
print('Ready to predict using model {} in project {}'.format(model_name,
 project_id))
```

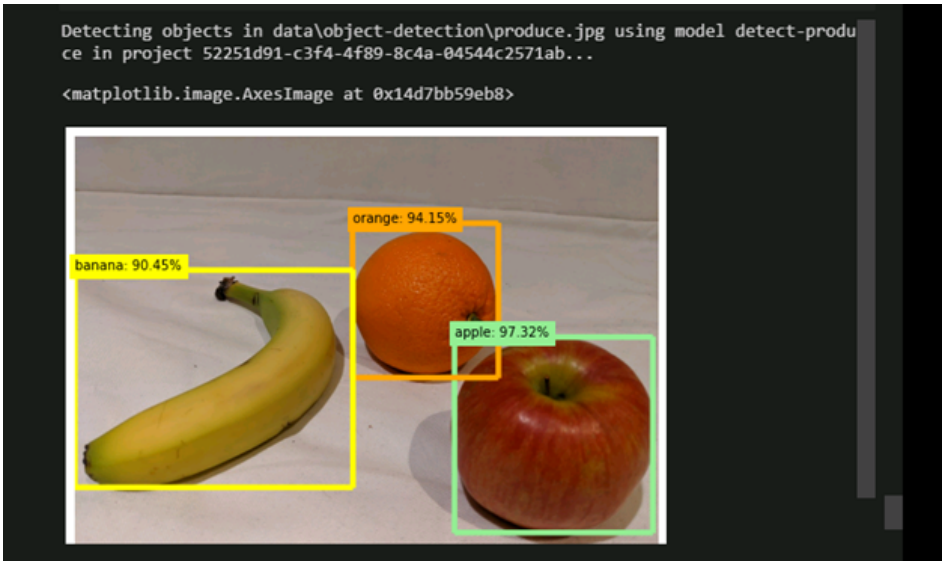
14. Masukkan project id, endpoint, dan key di notebook yang sudah disediakan di lab



15. Jika semua telah terkonfigurasi dengan benar maka akan tampil statement seperti diatas



16. Jalankan kode diatas untuk melihat prediksi yang dihasilkan, yang menunjukkan objek yang terdeteksi dan probabilitas untuk setiap prediksi

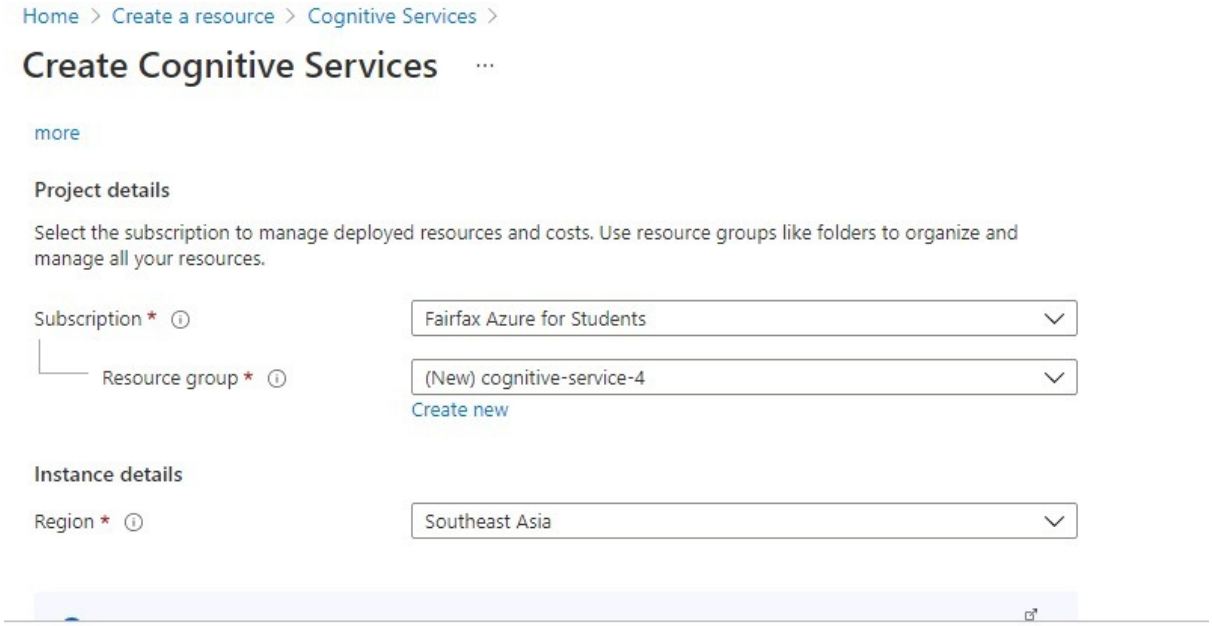


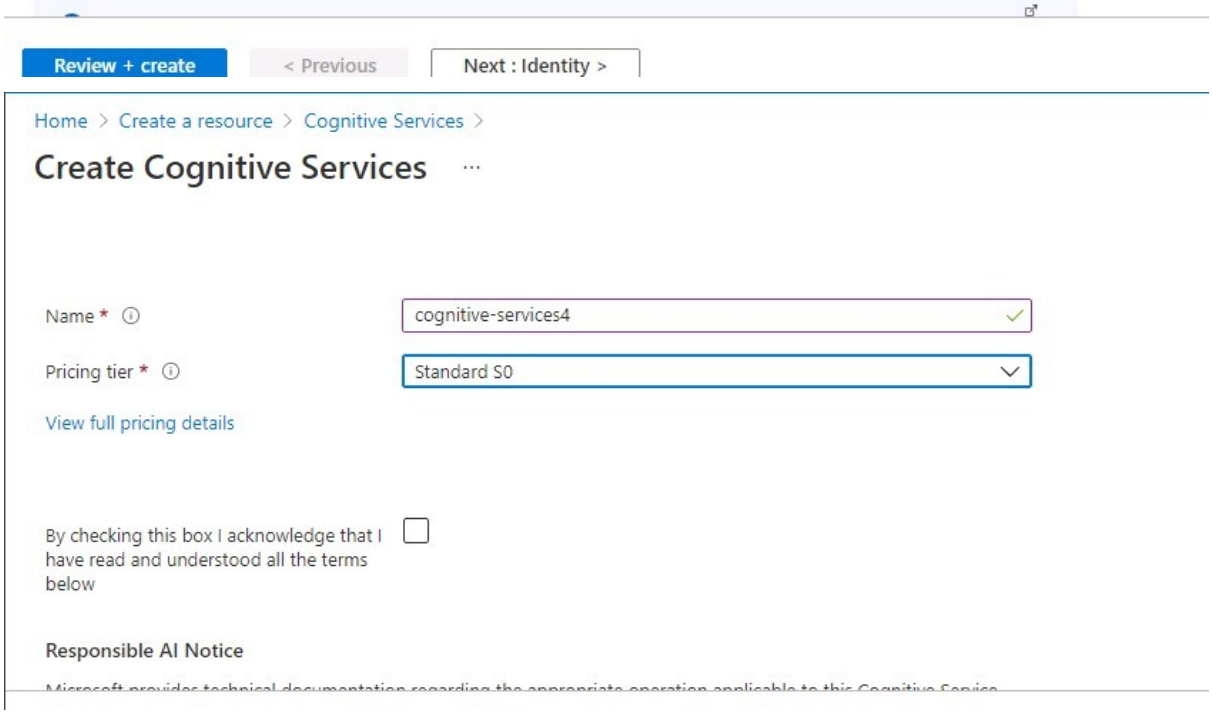
17. Gambar diatas menunjukan hasil akhirnya

PERCOBAAN 4

Detect and analyze faces with the Face service

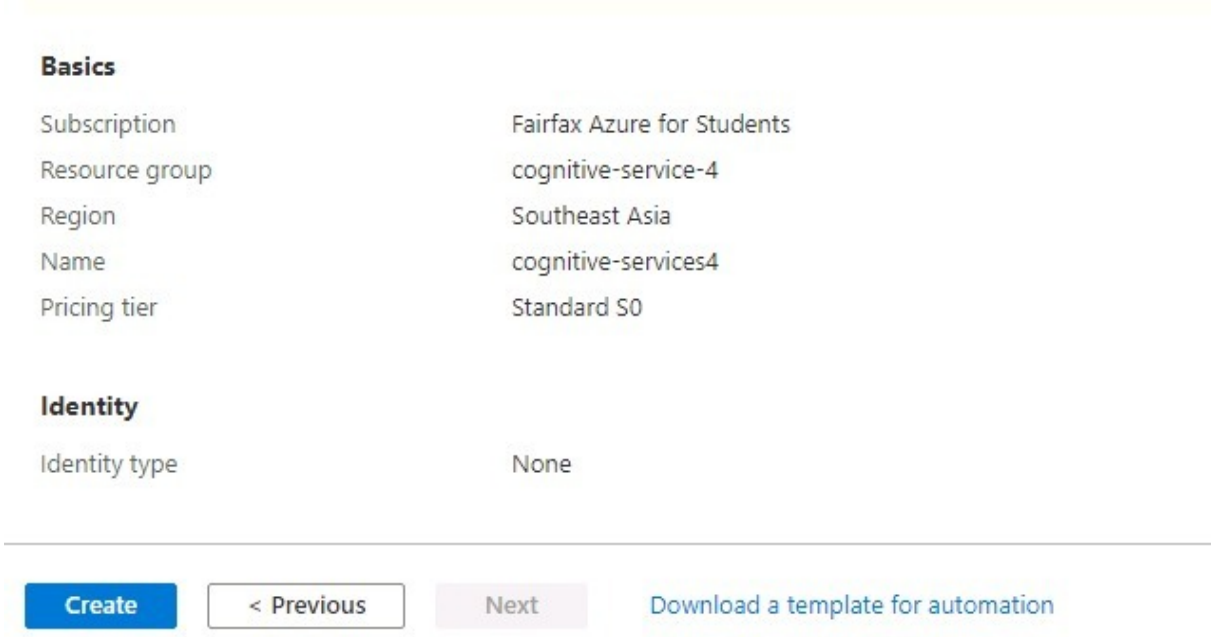
Seperti percobaan lab sebelumnya, pada microsoft learn klik launch lab, dan menunggu virtual machine terbuat. ketika sudah terbuat, lakukan login dan klik GET_FILES. kalau sudah, klik Visual studio Code, dan klik 04 - face analysis.ipynb dan lakukan langkah-langkah pada notebook tersebut. Di lab ini kita akan menggunakan layanan face pada Azure untuk mendeteksi dan menganalisis wajah.



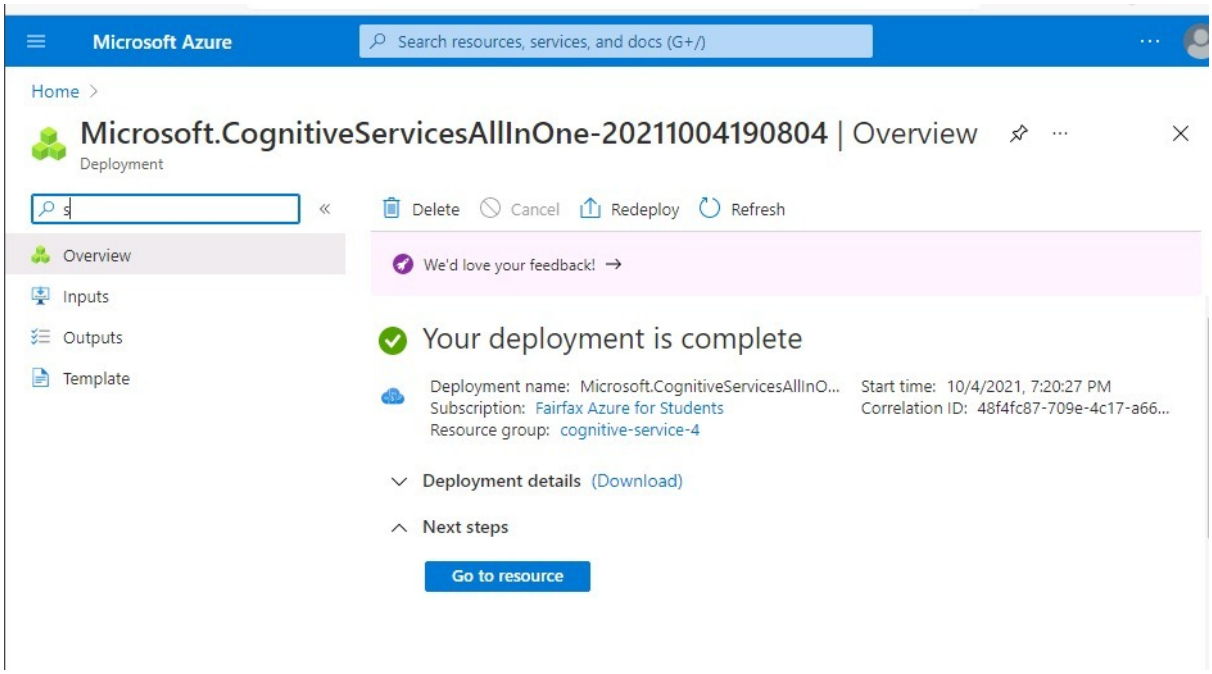


1. Klik tombol create a resource pada custom vision, cari layanan kognitif dan buat resource dengan pengaturan sebagai berikut:
   - **Subscription:** *Fairfax Azure for Students*
   - **Resource group:** *(New) cognitive-service-4*
   - **Region:** *Southeast Asia*
   - **Name:** Cognitive-service4

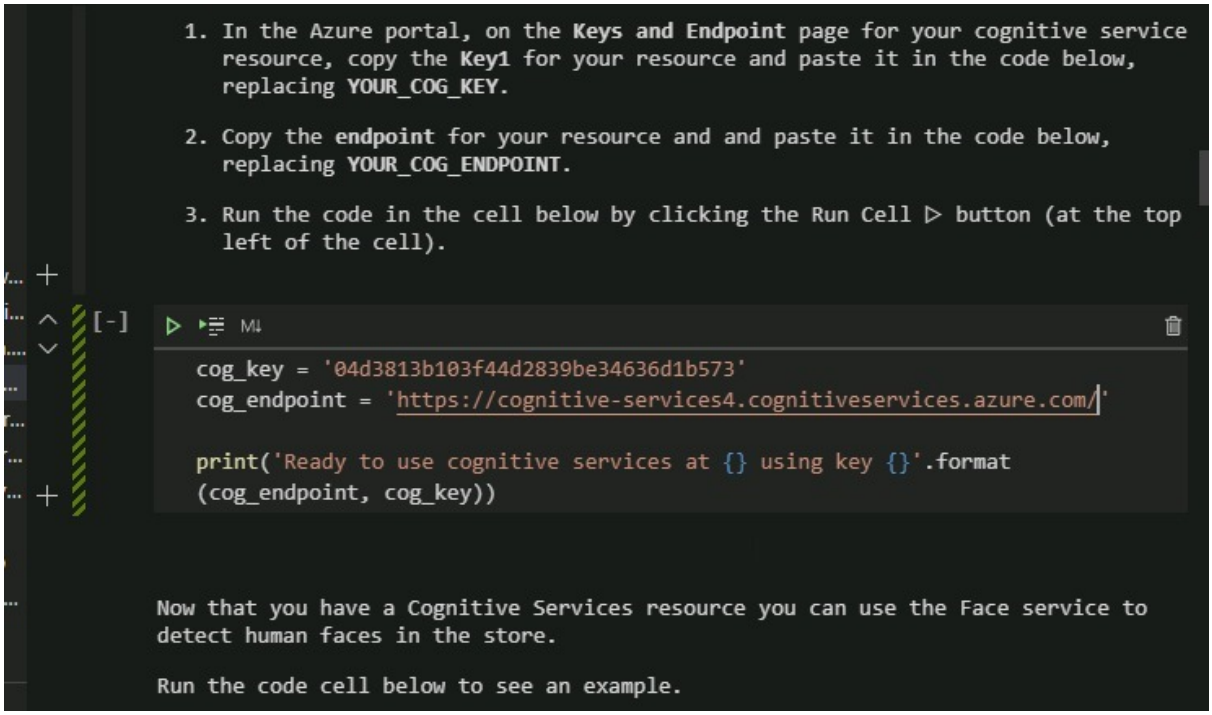
- **Pricing tier:** S0
- **I confirm I have read and understood the notices:** Selected.



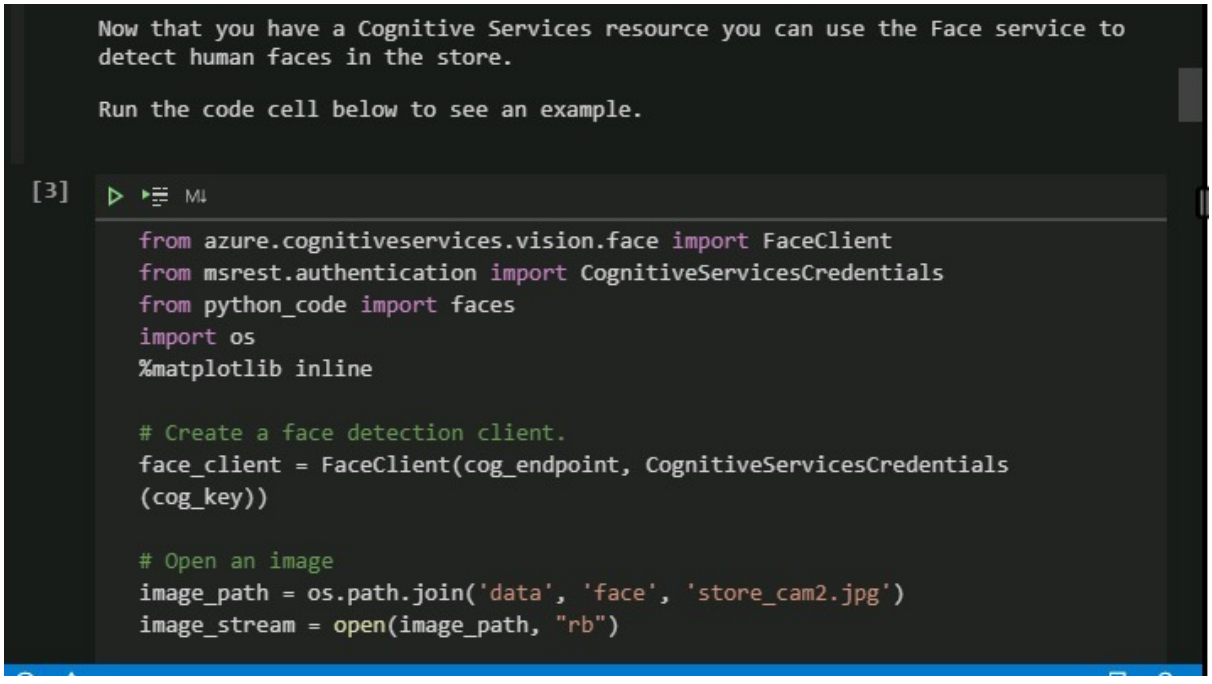
2. Selanjutnya klik review + create dan tunggu hingga penerapan selesai



3. Ini merupakan halaman ketika kita berhasil membuat sebuah resource

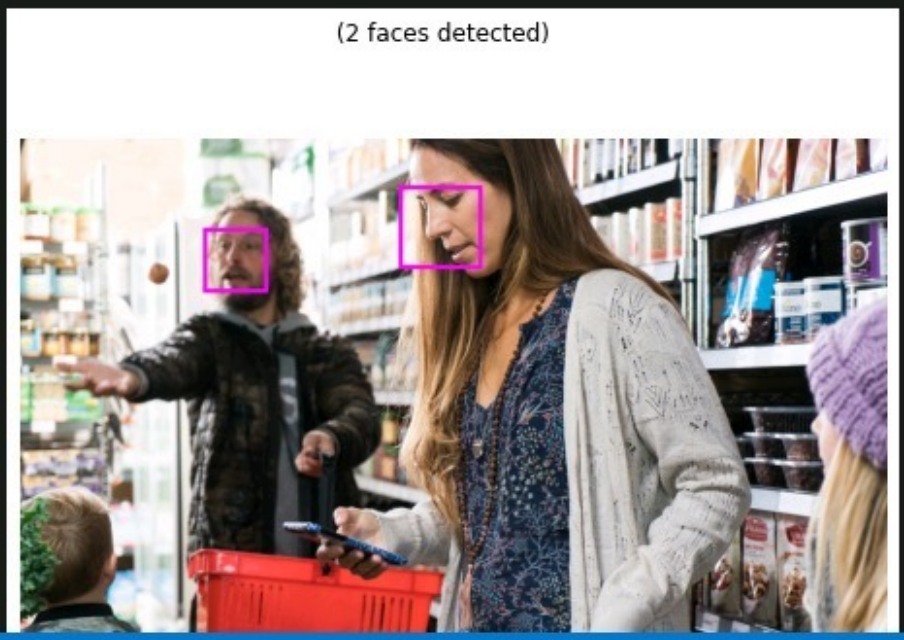


4. Untuk menggunakan cognitive service resource aplikasi ini membutuhkan endpoint dan authentication key, salin kunci1 pada cog_key dan salin endpoint pada cog_endpoint



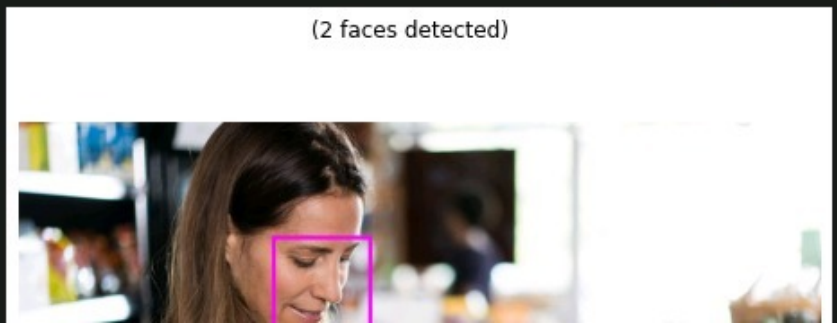
5. Selanjutnya, untuk melihat hasilnya jalankan kode berikut

```
# Display the faces (code in python_code/faces.py)
faces.show_faces(image_path, detected_faces)
```



6. Setelah di run, maka akan muncul 2 wajah yang terdeteksi

```
# Open an image
image_path = os.path.join('data', 'face', 'store_cam3.jpg')
image_stream = open(image_path, "rb")

# Detect faces
detected_faces = face_client.face.detect_with_stream(image=image_stream)

# Display the faces (code in python_code/faces.py)
faces.show_faces(image_path, detected_faces, show_id=True)
```



7. Code selanjutnya di run dan terlihat hasil seperti gambar diatas


```
# Get the ID of the first face in image 1
image_1_path = os.path.join('data', 'face', 'store_cam3.jpg')
image_1_stream = open(image_1_path, "rb")
image_1_faces = face_client.face.detect_with_stream(image=image_1_stream)
face_1 = image_1_faces[0]

# Get the face IDs in a second image
image_2_path = os.path.join('data', 'face', 'store_cam2.jpg')
image_2_stream = open(image_2_path, "rb")
image_2_faces = face_client.face.detect_with_stream(image=image_2_stream)
image_2_face_ids = list(map(lambda face: face.face_id, image_2_faces))

# Find faces in image 2 that are similar to the one in image 1
similar_faces = face_client.face.find_similar(face_id=face_1.face_id,
face_ids=image_2_face_ids)

# Show the face in image 1, and similar faces in image 2(code in
python_code/face.py)
faces.show_similar_faces(image_1_path, face_1, image_2_path,
image_2_faces, similar_faces)
```

8. Code diatas merupakan code untuk menganalisis apakah ada face similarity pada dua image. Klik RUN untuk menjalanlannya.

```
# Show the face in image 1, and similar faces in image 2(code in
python_code/face.py)
faces.show_similar_faces(image_1_path, face_1, image_2_path,
image_2_faces, similar_faces)
```

{}

9. Gambar diatas adalah hasil ketika code diatas di RUN. Terlihat bahwa sistem dapat mendeteksi bahwa ada kesamaan wajah (face similarity) pada kedua gambar, yaitu pada ibu-ibu.

Recognize faces

So far you've seen that Face can detect faces and facial features, and can identify two faces that are similar to one another. You can take things a step further by inplementing a *facial recognition* solution in which you train Face to recognize a specific person's face. This can be useful in a variety of scenarios, such as automatically tagging photographs of friends in a social media application, or using facial recognition as part of a biometric identity verification system.

To see how this works, let's suppose the Northwind Traders company wants to use facial recognition to ensure that only authorized employees in the IT department can access secure systems.

We'll start by creating a *person group* to represent the authorized employees.

10. Layanan pada face services di Azure yang selanjutnya adalah Recognize faces atau mengenali suatu wajah.

[11]

```
group_id = 'employee_group_id'
try:
    # Delete group if it already exists
    face_client.person_group.delete(group_id)
except Exception as ex:
    print(ex.message)
finally:
    face_client.person_group.create(group_id, 'employees')
    print ('Group created!')
```

```
Group created!
```

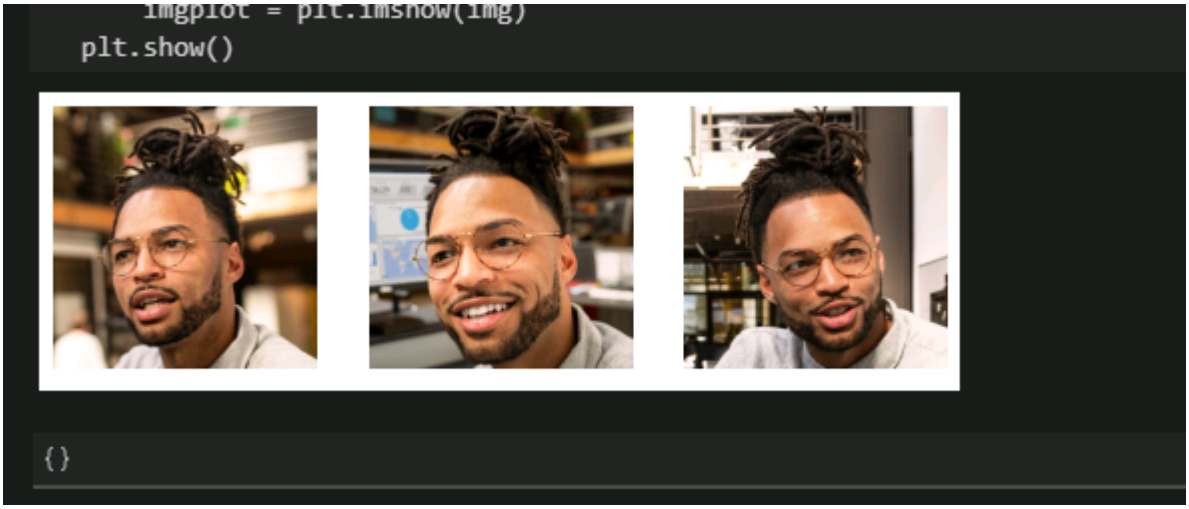
11. Code diatas adalah code untuk membuat group yang nantinya akan diisi oleh input dataset. Klik RUN untuk menjalankannya.

```
We'll add a single employee called Wendell, and register three photographs of the employee.
```

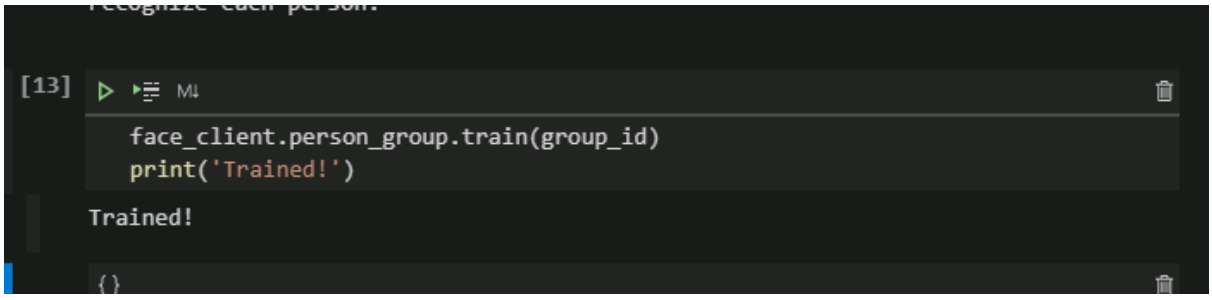
[12]

```
import matplotlib.pyplot as plt
from PIL import Image
import os
%matplotlib inline

# Add a person (Wendell) to the group
wendell = face_client.person_group_person.create(group_id, 'Wendell')

# Get photo's of Wendell
folder = os.path.join('data', 'face', 'wendell')
wendell_pics = os.listdir(folder)

# Register the photos
i = 0
fig = plt.figure(figsize=(8, 8))
for pic in wendell_pics:
```

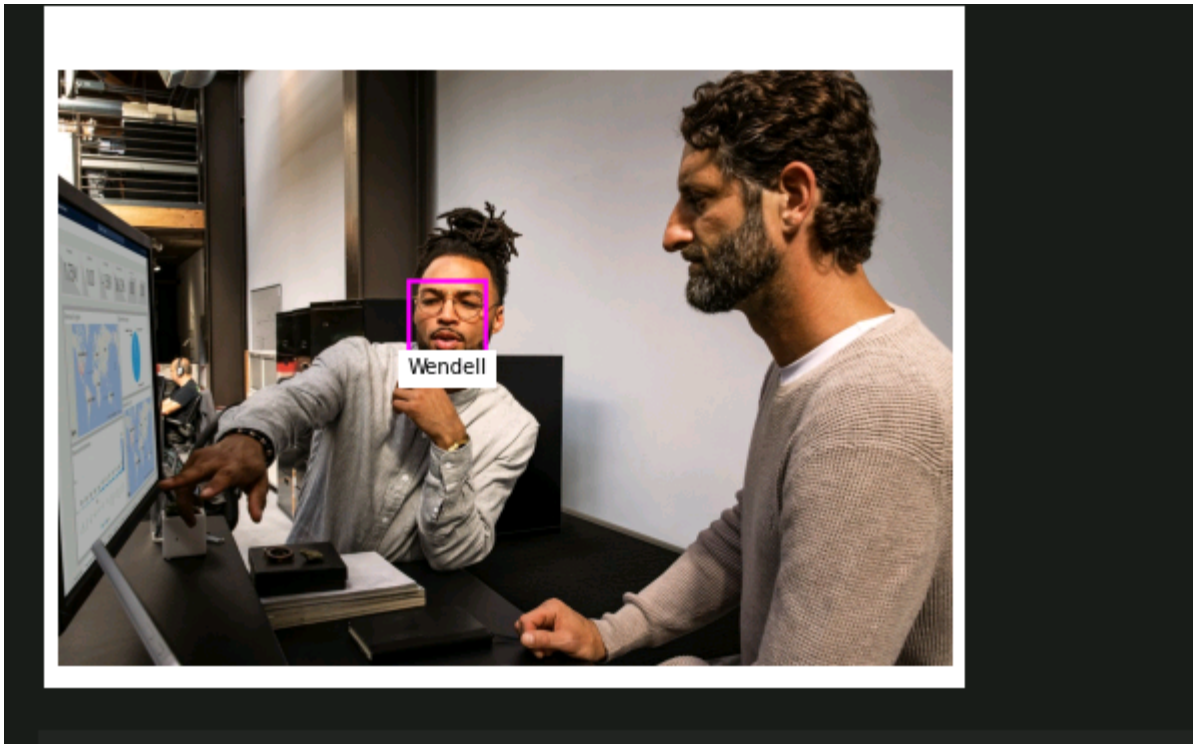
12. Code diatas adalah code untuk menaruh beberapa input dataset dari satu orang bernama Wendell ke dalam group yang tadi telah dibuat. Kemudian untuk menjalankannya klik RUN.



13. Ketika code sebelumnya sudah di RUN, muncul beberapa input dataset dari orang bernama Wendell.



14. Jalankan kode diatas dan klik RUN. code ini berguna untuk melakukan training pada input dataset.



15. Gambar diatas merupakan hasil dari RUN code sebelumnya.