Topic : AI-Powered Fresh and Spoiled Fruit Detection with Custom Vision

Report

Submitted by :

Anvesh Hirve

2021-B-14072003

Group Names :

Anvesh Hirve

Rohan Siddu Samudrala

Lekha Ahirrao

Swapnali Gavali

Saniya Wagh

Presented to :

Abhay Dhanmeher

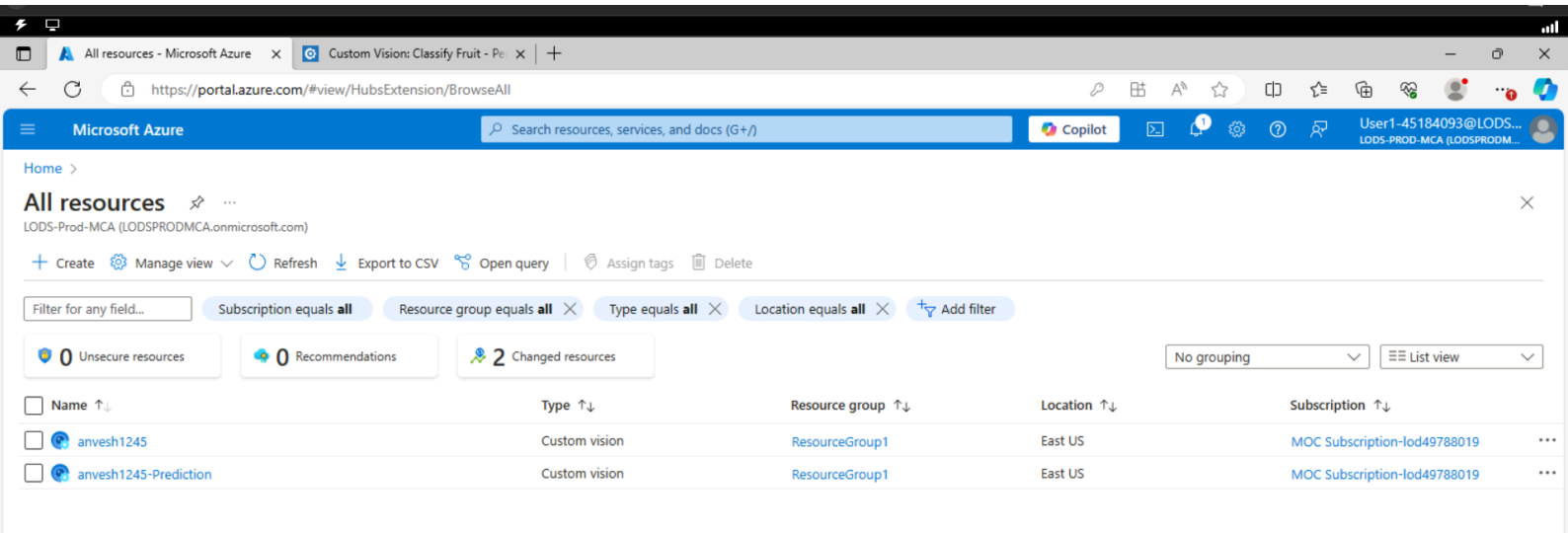
Program Trainee

Overview:

This project aims to classify fruits based on their freshness using Azure's Custom Vision, an image recognition service that leverages machine learning. By training a custom classification model, the system can accurately determine whether a fruit is fresh or spoiled. The goal is to assist industries such as agriculture, food retail, or supply chain management by providing an automated solution to assess fruit quality.

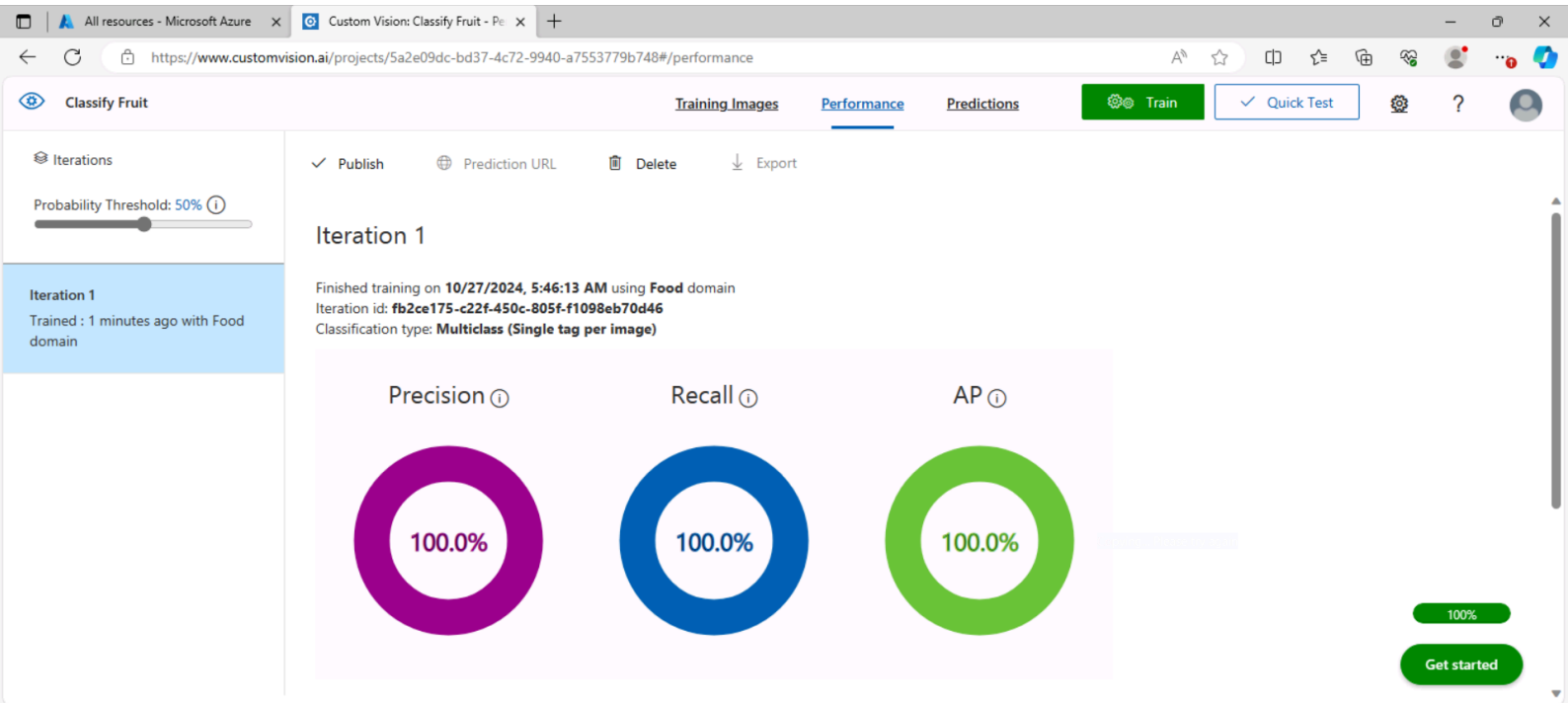
Procedure :

- Opened Visual Studio Code after opening, I cloned the repository to a local folder **https://github.com/MicrosoftLearning/AI-102-AIEngineer**
- After cloning i opened the Azure portal at **https://portal.azure.com**, and sign in using the Microsoft account with Azure subscription.
- Selected the **+ Create a resource** button, and searched for *custom vision*, and created a **Custom Vision** resource



- After creating a resource, In a new browser tab, opened the Custom Vision portal at **https://customvision.ai**. signed in using the Microsoft account with Azure subscription
- In the Custom Vision portal, created a new project with the following settings:
  - **Name**: Classify Fruit
  - **Description**: Image classification for fruit
  - **Resource**: *The Custom Vision resource you created previously*
  - **Project Types**: Classification
  - **Classification Types**: Multiclass (single tag per image)
  - **Domains**: Food
- In the new project, clicked on [+] Add images, and select all of the files in the **training-images** Then uploaded the image files which contains fresh fruit and spoiled fruit and labeled them differently.

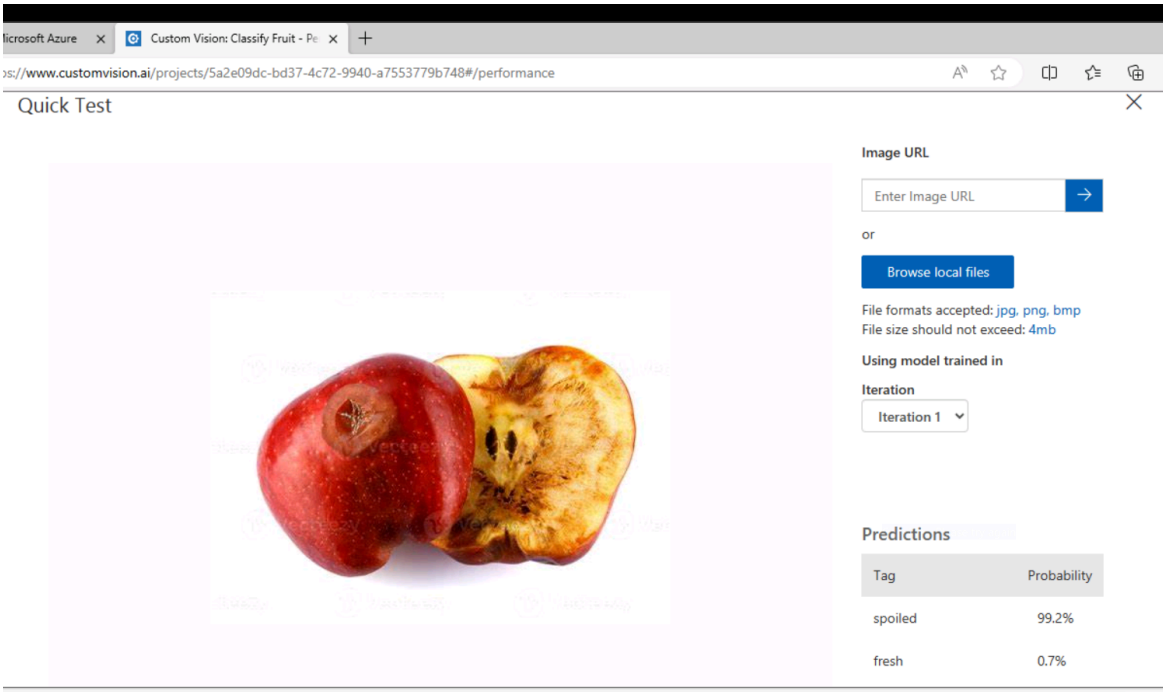
- After labeling and uploading In the Custom Vision project, trained a classification model using the tagged images. Selected the **Quick Training** option, and then waited for the training iteration to complete
- When the model iteration has been trained, reviewed the *Precision*, *Recall*, and *AP* performance metrics - the prediction accuracy of the classification model was high.

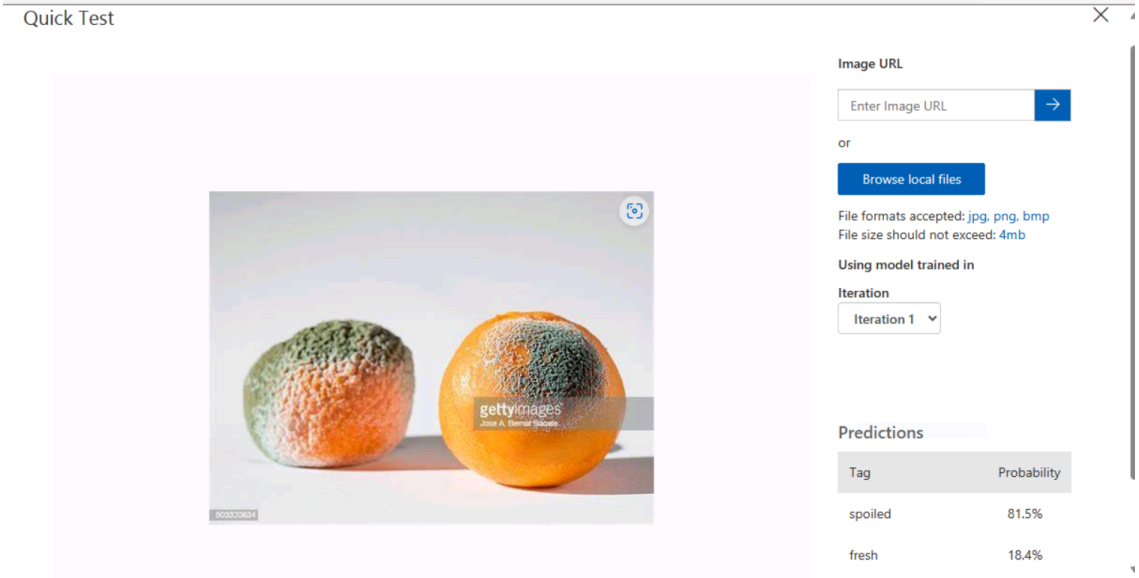


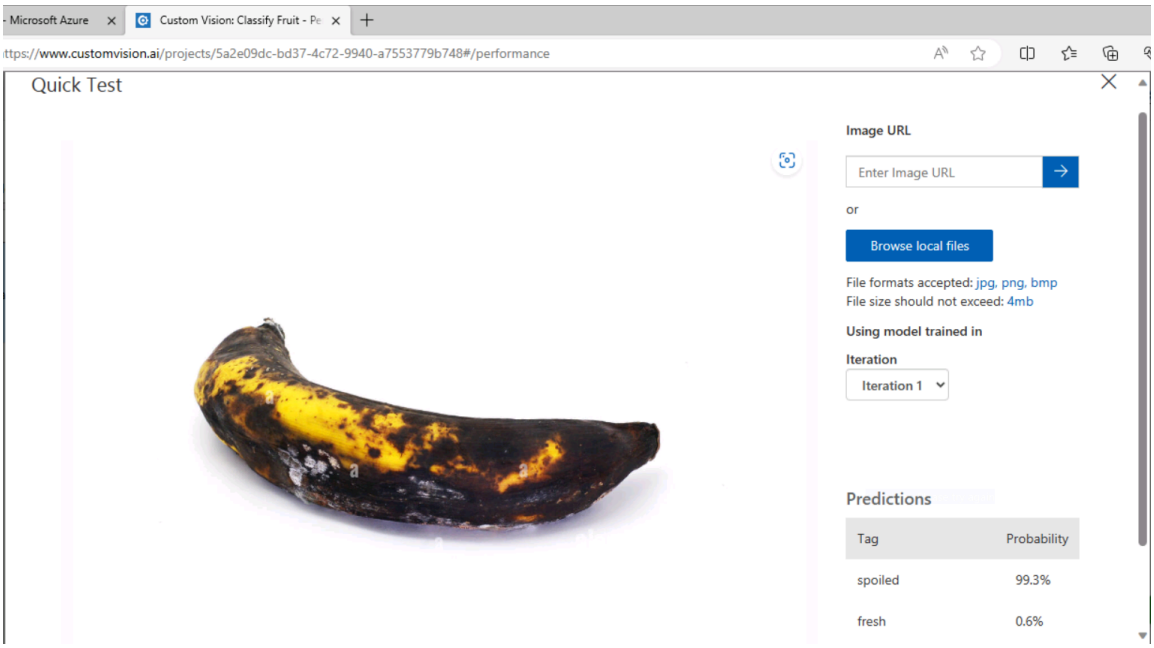
Performance Per Tag

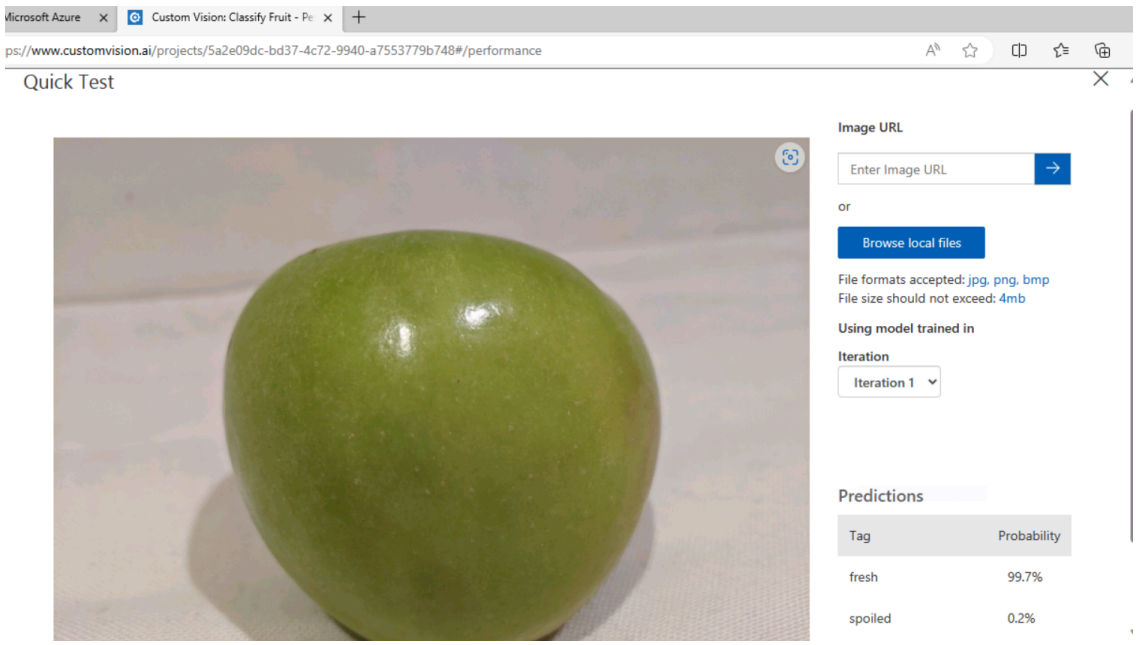
Tag	Precision	Recall	A.P.	Image count
spoiled	100.0%	100.0%	100.0%	14
fresh	100.0%	100.0%	100.0%	45

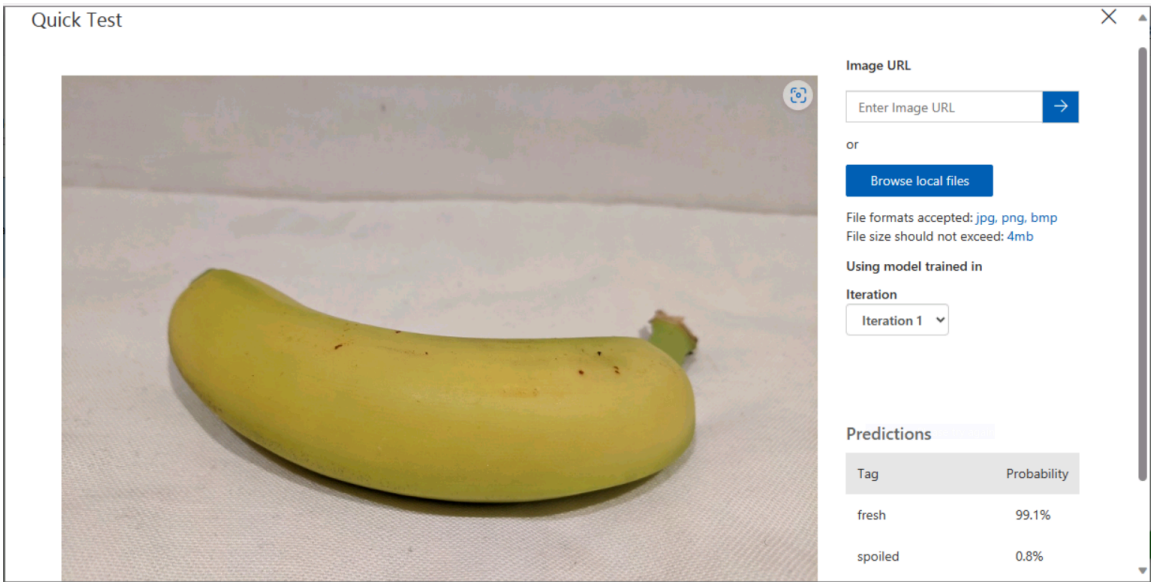
- Tested the Model
  - Following are the output of the trained Model

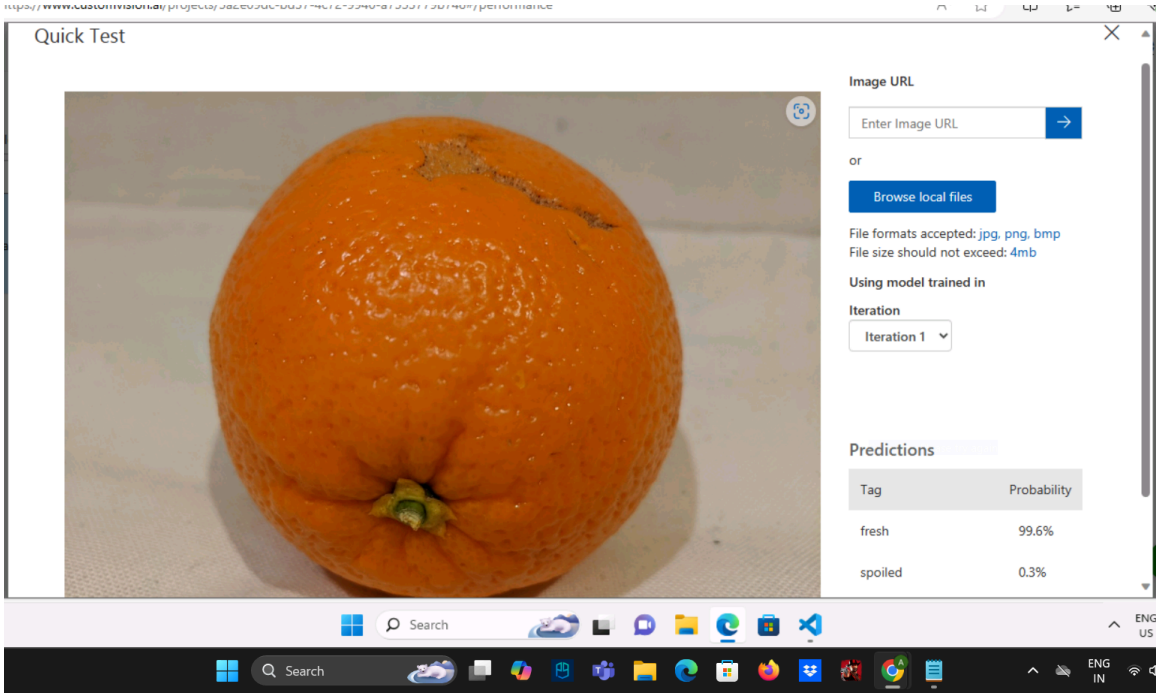





Microsoft Azure
Custom Vision: Classify Fruit - Pe
ttps://www.customvision.ai/projects/5a2e09dc-bd37-4c72-9940-a7553779b748#/performance
Quick Test
Image URL
Enter Image URL
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
spoiled
99.3%
fresh
0.6%


Microsoft Azure
Custom Vision: Classify Fruit - Pe
ps://www.customvision.ai/projects/5a2e09dc-bd37-4c72-9940-a7553779b748#/performance
Quick Test
Image URL
Enter Image URL
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
fresh
99.7%
spoiled
0.2%


Quick Test
Image URL
Enter Image URL
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
fresh
99.1%
spoiled
0.8%


Quick Test
Image URL
Enter Image URL
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
fresh
99.6%
spoiled
0.3%
Search
ENG
US
Search
ENG
IN

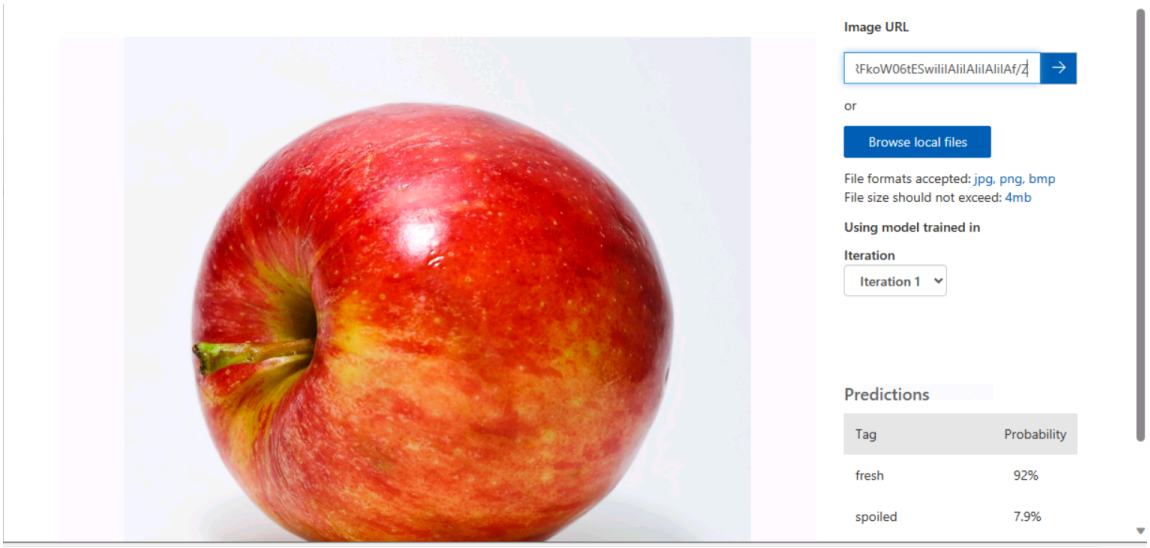
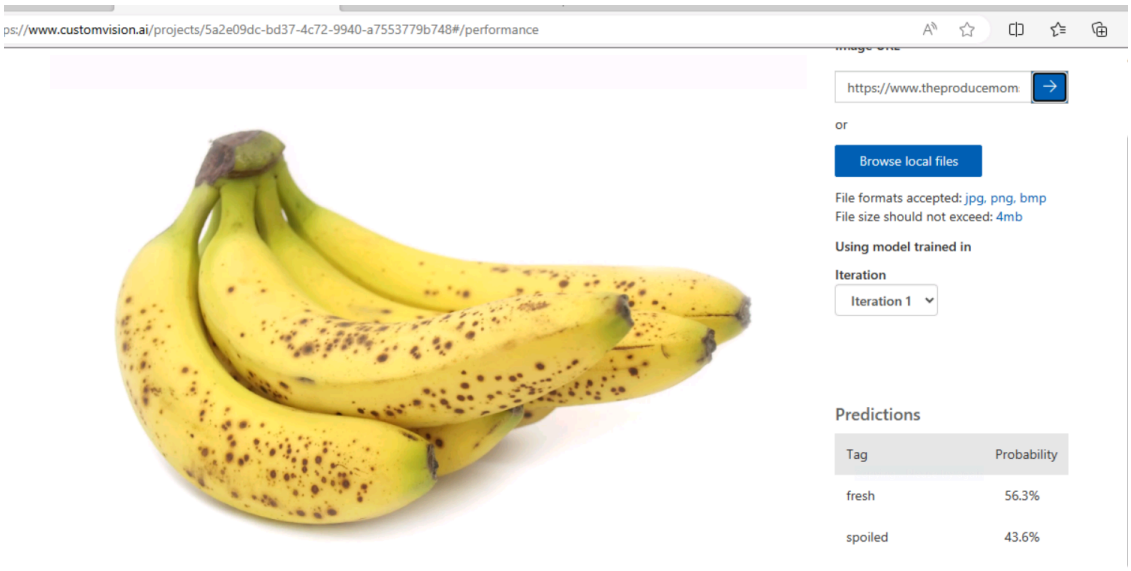


Image URL
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
fresh
92%
spoiled
7.9%


ps://www.customvision.ai/projects/5a2e09dc-bd37-4c72-9940-a7553779b748#/performance
https://www.theproducemom
or
Browse local files
File formats accepted: jpg, png, bmp
File size should not exceed: 4mb
Using model trained in
Iteration
Iteration 1
Predictions
Tag
Probability
fresh
56.3%
spoiled
43.6%

Using the *training* API

- In Visual Studio Code, in the **Explorer** pane, to the **17-image_classification** folder and expand the **Python** folder.
- Right-clicked the **train-classifier** folder and open an integrated terminal. Then install the Custom Vision Training package
- In this lab i have used python programming so according to that i installed custom vision training package

pip install azure-cognitiveservices-vision-customvision==3.1.0

- Changed the configuration setting of **train-classifier** folder i.e **Python: .env**

```
17-image-classification > Python > train-classifier > .env
1  TrainingEndpoint=https://anvesh1245.cognitiveservices.azure.com/
2  TrainingKey=9hR94daNAhHR1ZumpCJEj6drnq8CVfLexvHN9DHqjcp30TUjPjazJQQJ99AJACYeBjFXJ3w3AAAJACOGBTKA
3  ProjectID=5a2e09dc-bd37-4c72-9940-a7553779b748
```

- After saving the configuration

Python Program (train-classifier.py)

```python
from azure.cognitiveservices.vision.customvision.training import CustomVisionTrainingClient
from azure.cognitiveservices.vision.customvision.training.models import ImageFileCreateBatch,
ImageFileCreateEntry
from msrest.authentication import ApiKeyCredentials
import time
import os
from dotenv import load_dotenv

def main():
    global training_client
    global custom_vision_project

    try:
        # Load configuration settings from .env file
        load_dotenv()

        training_endpoint = os.getenv('TrainingEndpoint')
        training_key = os.getenv('TrainingKey')
        project_id = os.getenv('ProjectID')

        # Check if all environment variables are loaded properly
        if not training_endpoint or not training_key or not project_id:
            raise ValueError("Missing environment variables in .env file")

        # Authenticate a client for the training API
        credentials = ApiKeyCredentials(in_headers={"Training-key": training_key})
        training_client = CustomVisionTrainingClient(training_endpoint, credentials)

        # Get the Custom Vision project
        custom_vision_project = training_client.get_project(project_id)

        # Debug print current working directory
        print(f"Current Working Directory: {os.getcwd()}")

        # Upload and tag images
        Upload_Images('more-training-images')

        # Train the model
        Train_Model()

    except Exception as ex:
        print(f"Error: {ex}")

def Upload_Images(folder):
    print("Uploading images...")
    try:
        # Get the tags from the Custom Vision project
        tags = training_client.get_tags(custom_vision_project.id)

        # Check if there are tags defined in the project
        if not tags:
            print("No tags found in the Custom Vision project.")
            return

        # Iterate through each tag
        for tag in tags:
            print(f"Uploading images for tag: {tag.name}")

            tag_folder = os.path.join(folder, tag.name)

            # Check if the tag folder exists
            if not os.path.exists(tag_folder):
                print(f"Tag folder '{tag_folder}' does not exist.")
                continue
```

```python
            # Iterate through all images in the tag folder
            for image in os.listdir(tag_folder):
                image_path = os.path.join(tag_folder, image)

                # Check if image exists
                if not os.path.isfile(image_path):
                    print(f"Image file '{image_path}' does not exist.")
                    continue

                print(f"Processing image: {image_path}")
                with open(image_path, "rb") as image_file:
                    image_data = image_file.read()
                    image_entry = ImageFileCreateEntry(name=image, contents=image_data)

                    upload_result = training_client.create_images_from_files(
                        custom_vision_project.id,
                        ImageFileCreateBatch(images=[image_entry], tag_ids=[tag.id])
                    )

                    if not upload_result.is_batch_successful:
                        print(f"Failed to upload image {image}: {upload_result}")
                    else:
                        print(f"Successfully uploaded {image}")

    except Exception as ex:
        print(f"Error uploading images: {ex}")

def Train_Model():
    print("Training model...")
    try:
        iteration = training_client.train_project(custom_vision_project.id)

        # Keep checking the training status
        while iteration.status != "Completed":
            iteration = training_client.get_iteration(custom_vision_project.id, iteration.id)
            print(f"Training status: {iteration.status}...")
            time.sleep(5)

        print("Model trained successfully!")

        # Set the iteration as the default
        training_client.update_iteration(custom_vision_project.id, iteration.id, is_default=True)
        print("Default iteration set.")

    except Exception as ex:
        print(f"Error training model: {ex}")

if __name__ == "__main__":
    main()
```

Output :

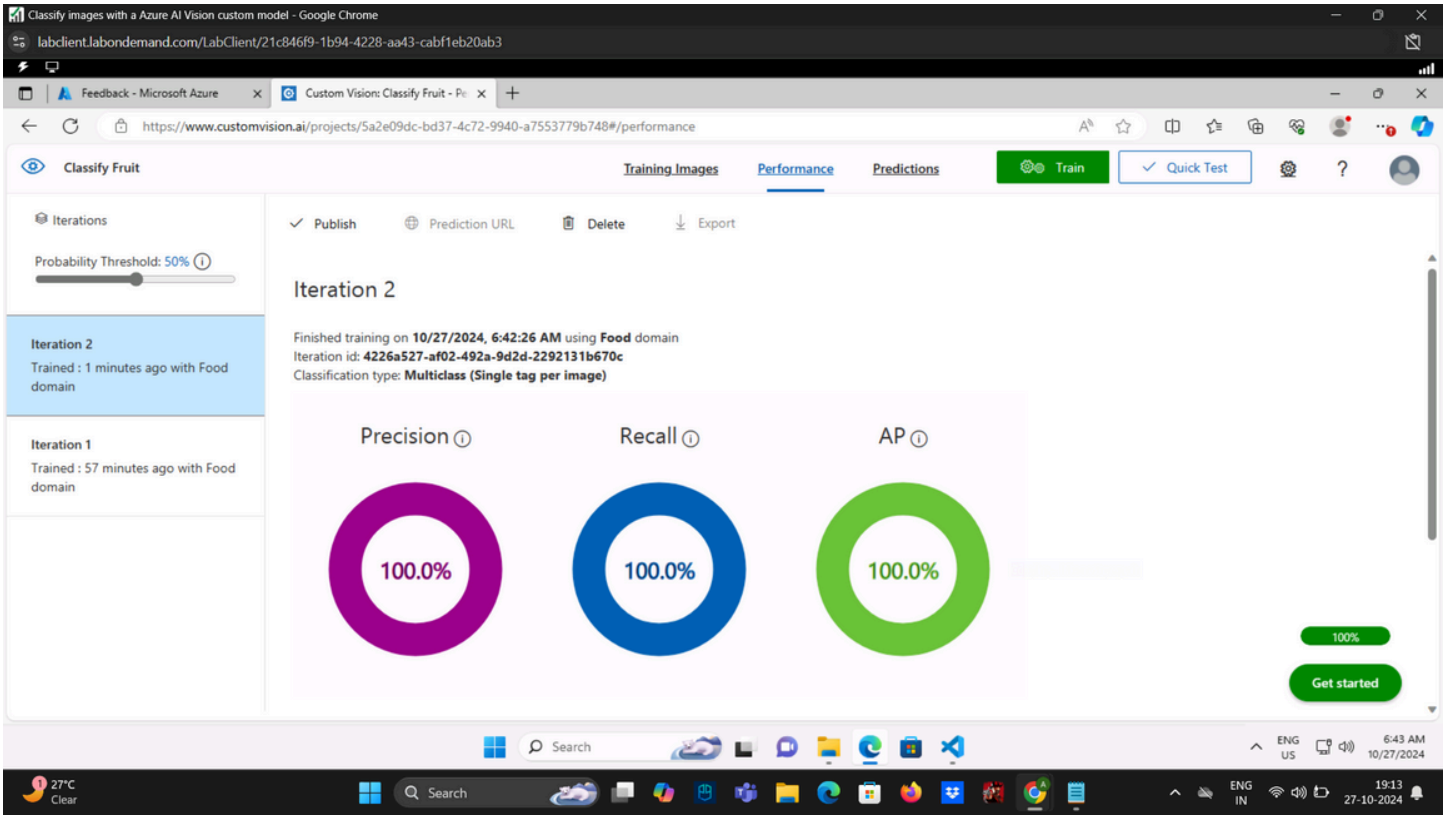
19-face
20-ocr
21-custom-form
22-Create-a-search-s...
23-custom-search-skill
24-knowledge-store
Instructions
_build.yml
_config.yml
OUTLINE
TIMELINE

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

```
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Training...
Training status: Completed...
Model trained successfully!
```

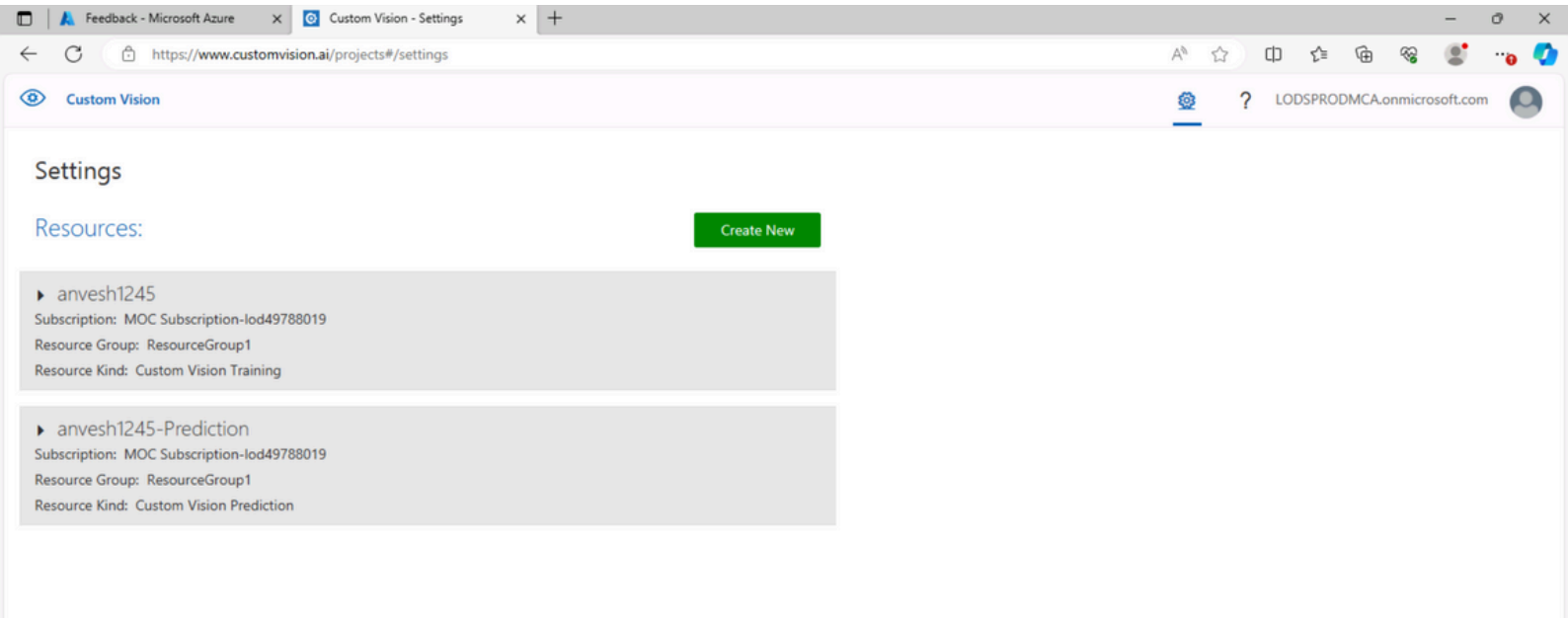
Ln 113, Col 11 Spaces: 4 UTF-8 CRLF Python 3.11.4 ('base': co

- Waited for the program to end,Verified that some new tagged images have been added to the project. Then view the Performance page and verify that a new iteration has been created.



Published the image classification model

- In the Custom Vision portal, on the Performance page, click Publish to publish the trained model with the following settings:
  - Model name: fruit-classifier
  - Prediction Resource: *The prediction resource i created previously which ends with "-Prediction"*
- clicked on the *Projects Gallery* to return to the Custom Vision portal home page, where project is now listed.
- On the Custom Vision portal home page ,in the *settings* to view the settings for Custom Vision service. Then, under Resources, found the *prediction* resource which ends with "-Prediction"to determine its Key and Endpoint values

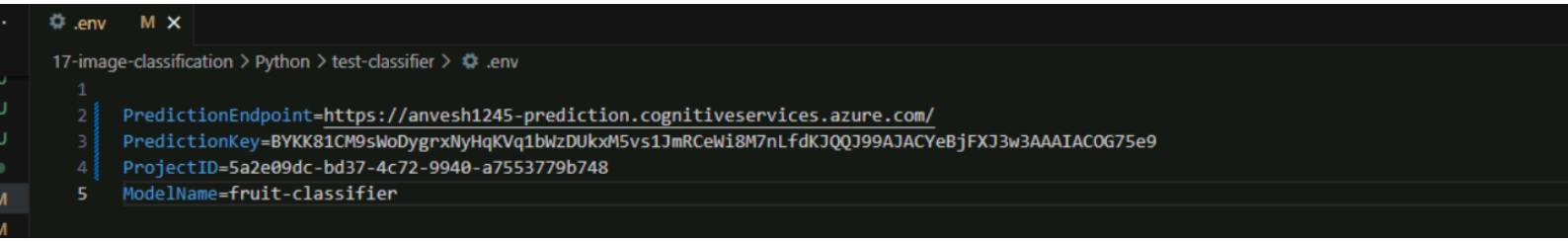


Used the image classifier from a client application

- In Visual Studio Code, in the **17-image-classification** folder **Python**, the **test-classifier** folder and opened the integrated terminal.

pip install azure-cognitiveservices-vision-customvision==3.1.0

- Changed the configuration setting of **test-classifier** folder i.e **Python: .env**

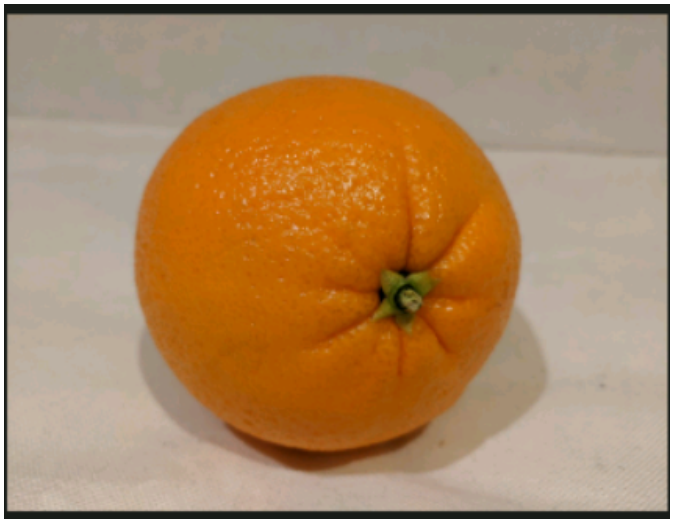
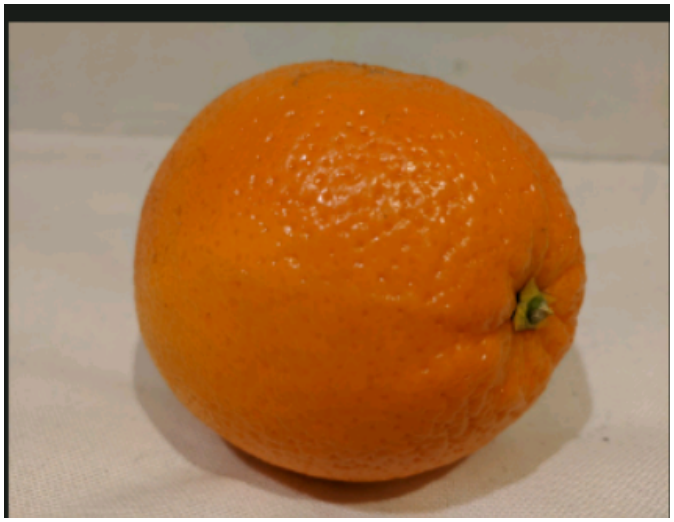


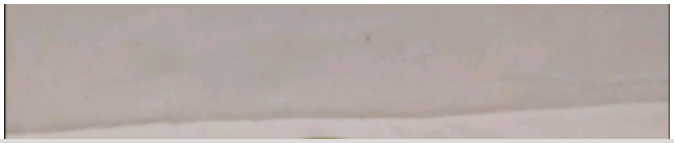
- After saving the configuration

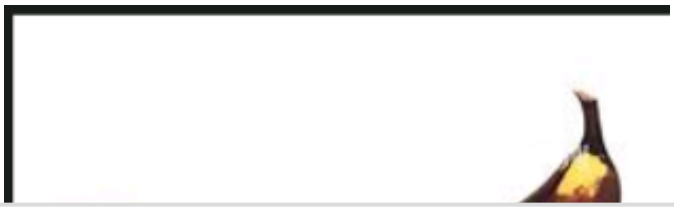
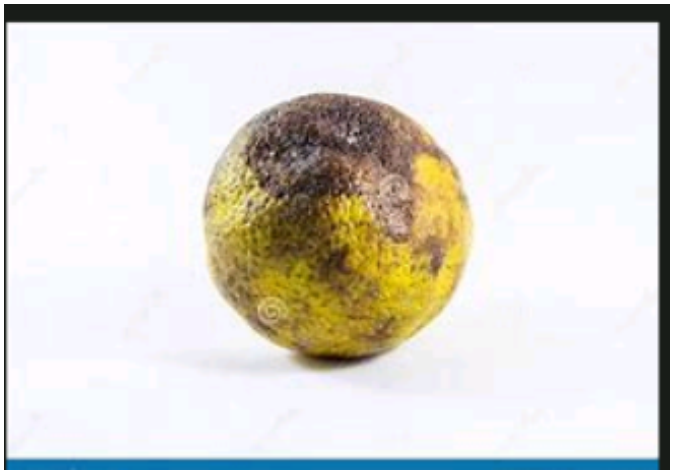
Python Program (test-classifier.py)

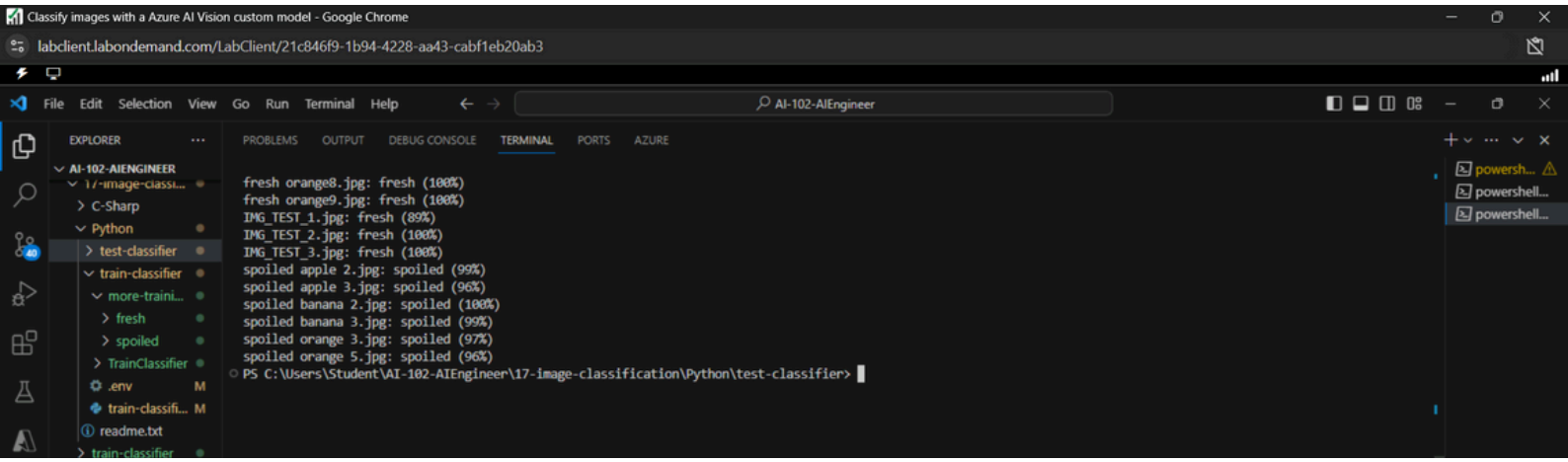
```python
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredictionClient
from msrest.authentication import ApiKeyCredentials
import os
from dotenv import load_dotenv

def main():
    try:
        # Load environment variables from a .env file
        load_dotenv()
        prediction_endpoint = os.getenv('PredictionEndpoint')
        prediction_key = os.getenv('PredictionKey')
        project_id = os.getenv('ProjectID')
        model_name = os.getenv('ModelName')

        # Check if environment variables are loaded correctly
        if not all([prediction_endpoint, prediction_key, project_id, model_name]):
            raise ValueError("One or more environment variables are missing.")

        # Authenticate a client for the prediction API
        credentials = ApiKeyCredentials(in_headers={"Prediction-key": prediction_key})
        prediction_client = CustomVisionPredictionClient(endpoint=prediction_endpoint,
credentials=credentials)

        # Classify test images
        test_images_path = 'test-images'
        if not os.path.exists(test_images_path):
            raise FileNotFoundError(f"The directory '{test_images_path}' does not exist.")

        for image in os.listdir(test_images_path):
            image_path = os.path.join(test_images_path, image)
            with open(image_path, "rb") as image_data:
                results = prediction_client.classify_image(project_id, model_name, image_data.read())

                # Loop over each label prediction and print any with probability > 50%
                for prediction in results.predictions:
                    if prediction.probability > 0.5:
                        print(f"{image}: {prediction.tag_name} ({prediction.probability:.0%})")

    except Exception as ex:
        print(f"An error occurred: {ex}")

if __name__ == "__main__":
    main()
```

Output :

	fresh orange8.jpg
	fresh orange9.jpg
	IMG_TEST_1.jpg
	IMG_TEST_2.jpg

	
	IMG_TEST_3.jpg
	spoiled apple 2.jpg
	spoiled apple 3.jpg
	spoiled banana 2.jpg

	
	spoiled banana 3.jpg
	spoiled orange 3.jpg
	spoiled banana 5.jpg



Images are provided above the output for prediction