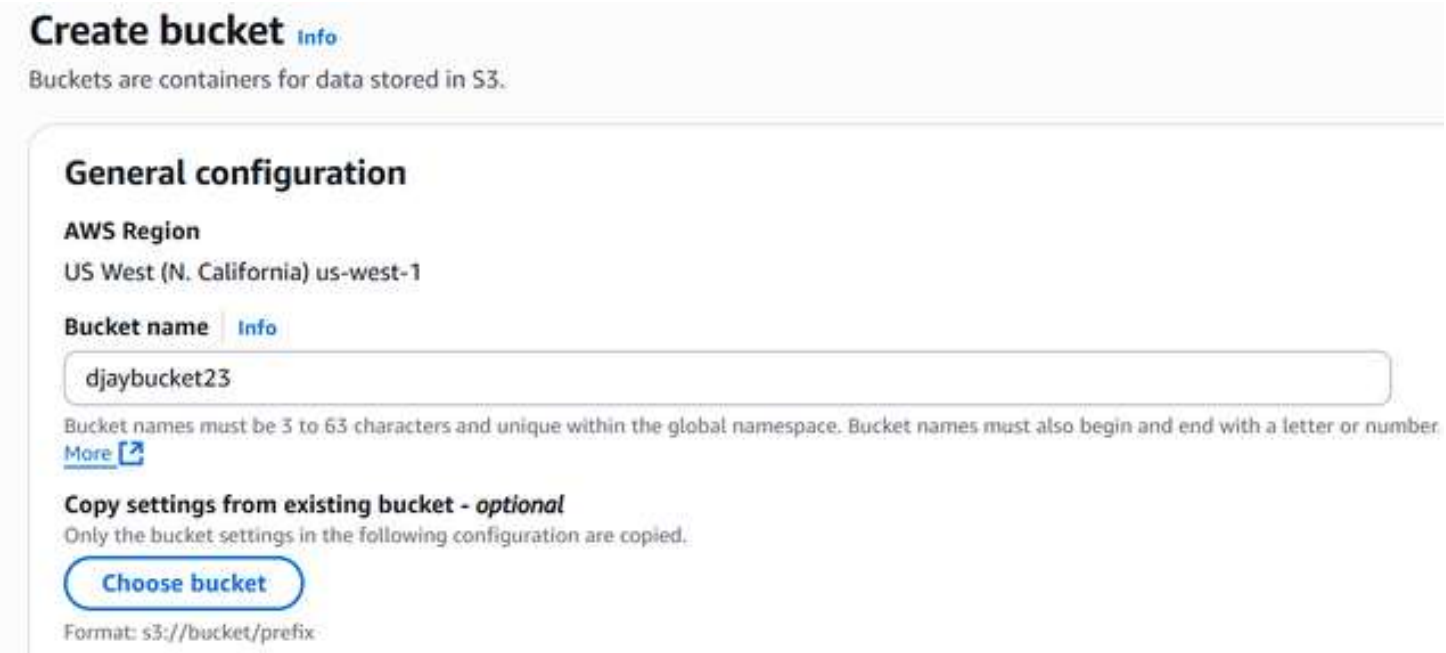
Project Summary

This project demonstrates a practical application of cloud computing by building a Python script that uses AWS Rekognition to perform object detection on an image. The script retrieves an image from an Amazon S3 bucket, analyzes it, and visualizes the results by drawing bounding boxes around detected objects using the matplotlib library. The entire process was performed in a local development environment.

Step 1: Initial Code & Library Setup

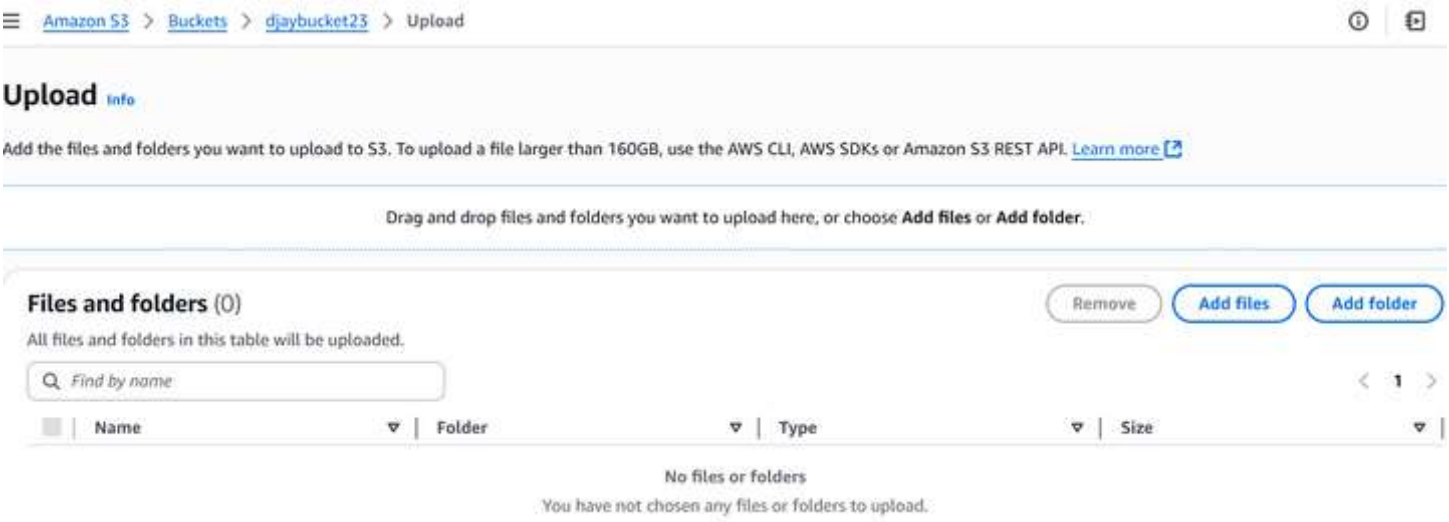
- **Action:** The project began with two separate code blocks. The first block contained the core logic for the detect label's function, which uses the boto3 library to interact with AWS Rekognition and S3. The second block was a main function designed to test the detect label's function. The necessary libraries (boto3, matplotlib, and pillow) were installed using pip.
- **Issues Encountered:**
  - **Code Organization:** The initial code was separated, leading to a discussion on the best way to combine the main function with the core logic. While using separate files is a standard practice for larger projects, combining them into a single, cohesive script was chosen for simplicity and ease of use for this specific project.
  - **Incorrect main Function Logic:** The main function had a syntax error where the photo and bucket variables were incorrectly assigned. This prevented the script from correctly passing the image and bucket names to the detect label's function.
- **Resolution:** The code was integrated into a single file and the main function's variable assignments were corrected. The print statements were also improved using f-strings for cleaner, more readable output.



Step 2: AWS Resource Creation

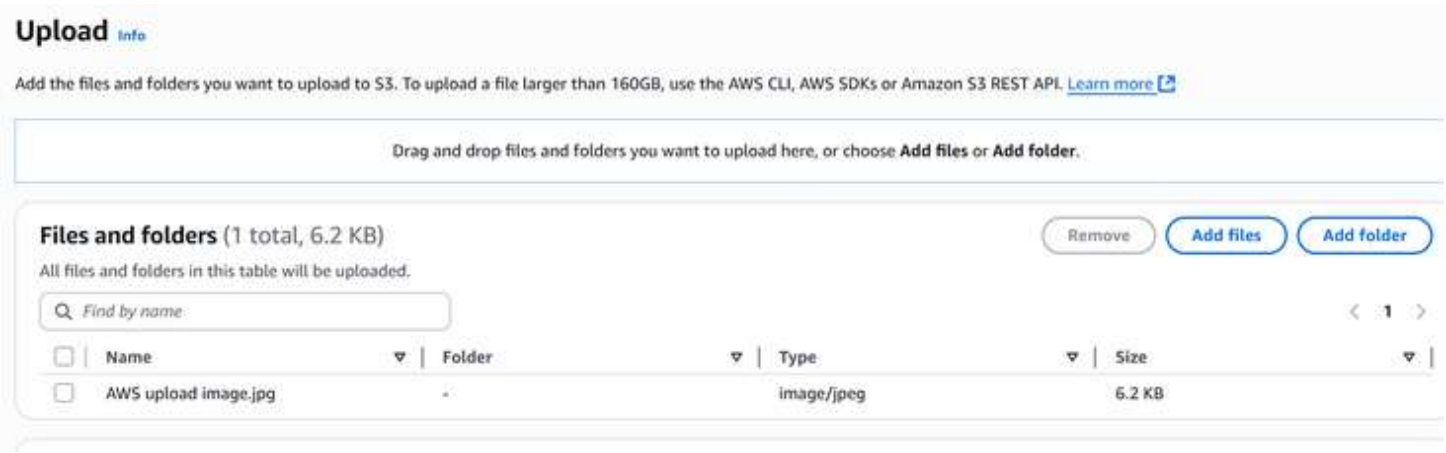
- **Action:** An Amazon S3 bucket was created to store the image to be analyzed. The bucket was named

- **Issues Encountered:** None. The bucket creation and file upload process was straightforward.



Step 3: Local Environment Execution

- **Action:** The completed Python script was executed from the command line.
- **Issues Encountered:**
  - **File Not Found Error:** The terminal produced an [Errno 2] No such file or directory error. This was a common beginner mistake where the python command was executed while the terminal was in the wrong directory, so it couldn't locate the script file.
- **Resolution:** The cd (change directory) command was used to navigate the terminal to the correct folder (C:\Users\dc197\OneDrive\Desktop) where the image_labeler.py file was saved. After changing directories, the script ran successfully.

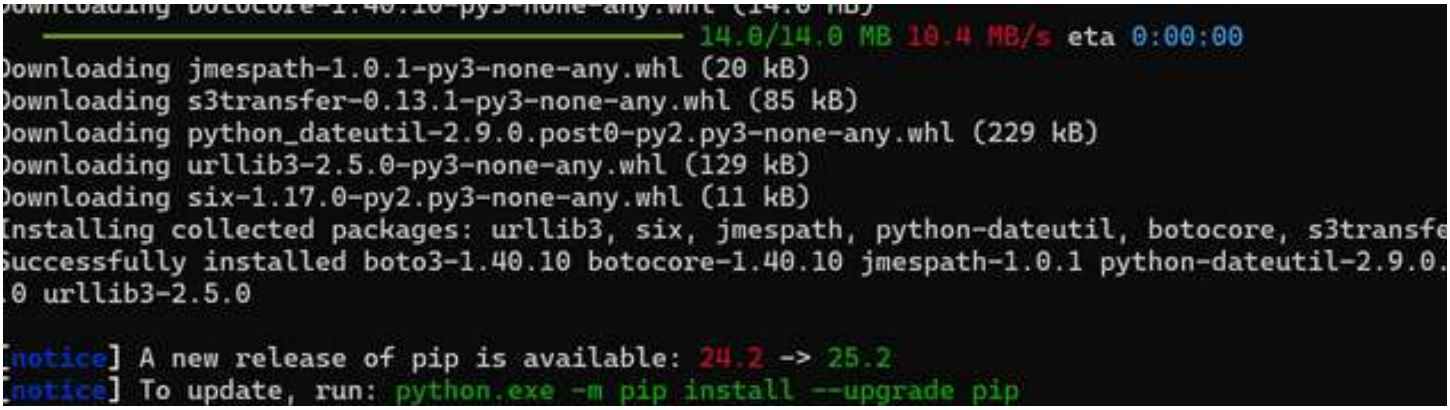


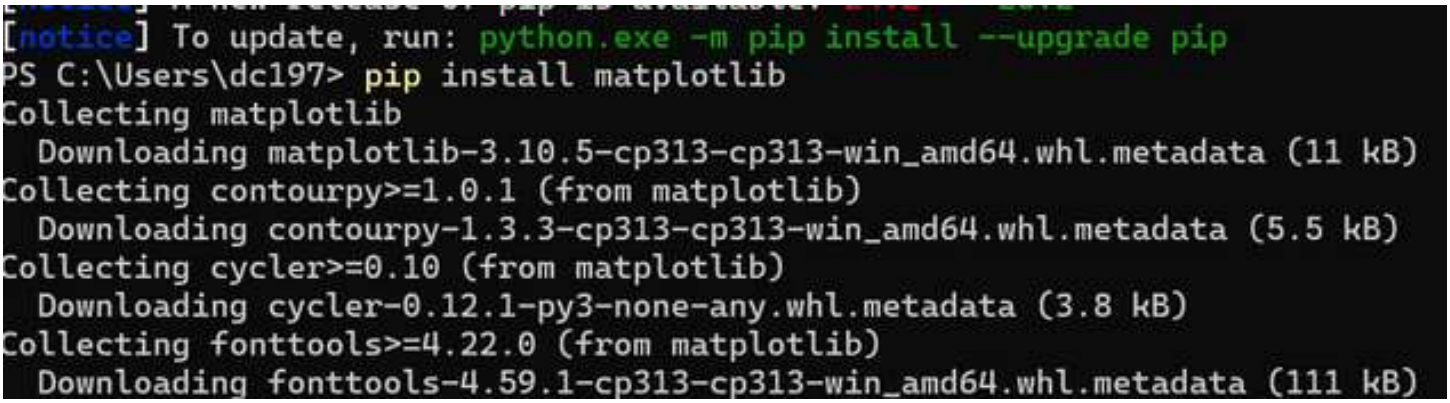
Final Result

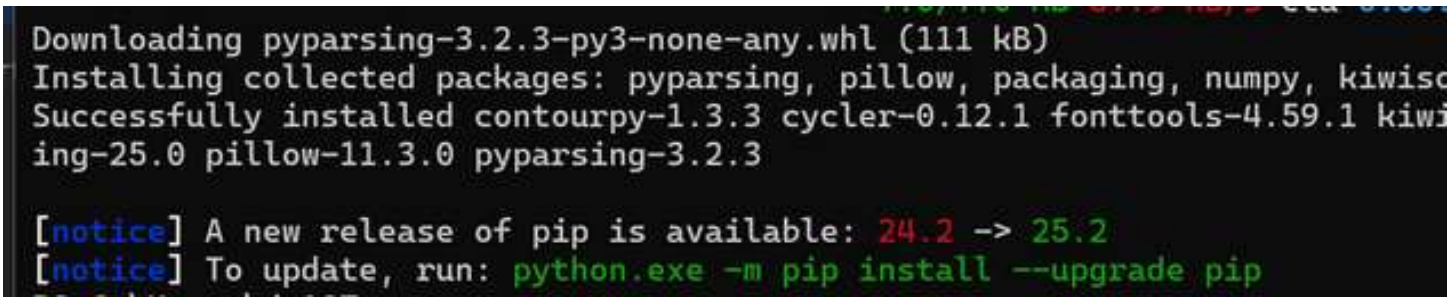
The project culminated in a successful execution of the script. The terminal output displayed a list of detected labels with their confidence scores, while a separate matplotlib window popped up. This window contained the image with **bounding boxes and text labels** accurately drawn around the detected objects, such as Person, Man, Adult, and Handbag.

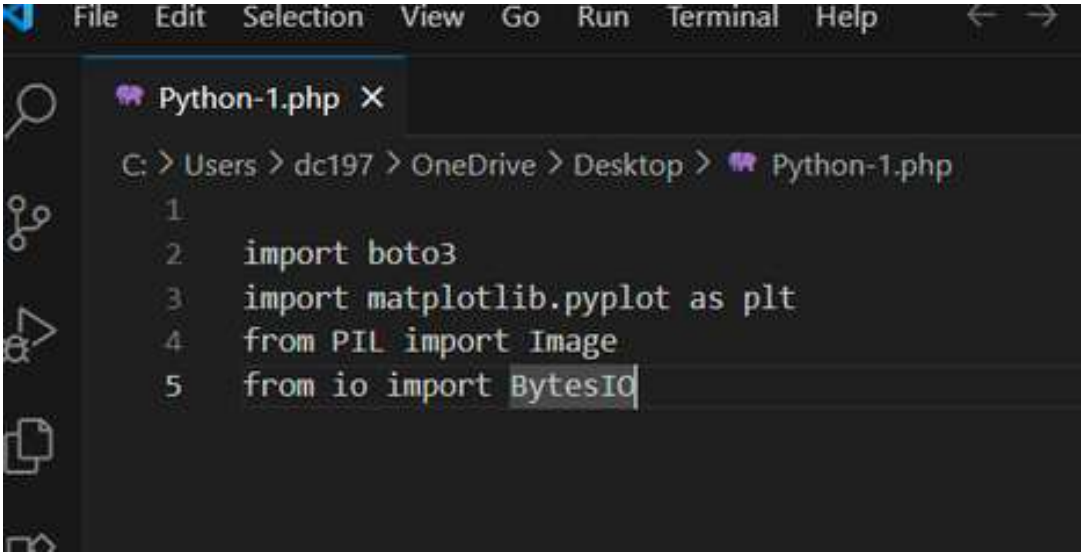
This project showcases a fundamental understanding of cloud computing concepts, including **data storage (S3)**, **AI services (Rekognition)**, and **API integration**, all visualized with a common Python library.

```
PS C:\Users\dc197> pip install boto3
Collecting boto3
  Downloading boto3-1.40.10-py3-none-any.whl.metadata (6.7 kB)
Collecting botocore<1.41.0,>=1.40.10 (from boto3)
  Downloading botocore-1.40.10-py3-none-any.whl.metadata (5.7 kB)
Collecting jmespath<2.0.0,>=0.7.1 (from boto3)
  Downloading jmespath-1.0.1-py3-none-any.whl.metadata (7.6 kB)
Collecting s3transfer<0.14.0,>=0.13.0 (from boto3)
  Downloading s3transfer-0.13.1-py3-none-any.whl.metadata (1.7 kB)
```


14.0/14.0 MB 10.4 MB/s eta 0:00:00
Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Downloading s3transfer-0.13.1-py3-none-any.whl (85 kB)
Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Downloading urllib3-2.5.0-py3-none-any.whl (129 kB)
Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, python-dateutil, botocore, s3transfe
Successfully installed boto3-1.40.10 botocore-1.40.10 jmespath-1.0.1 python-dateutil-2.9.0.
.0 urllib3-2.5.0
[notice] A new release of pip is available: 24.2 -> 25.2
[notice] To update, run: python.exe -m pip install --upgrade pip

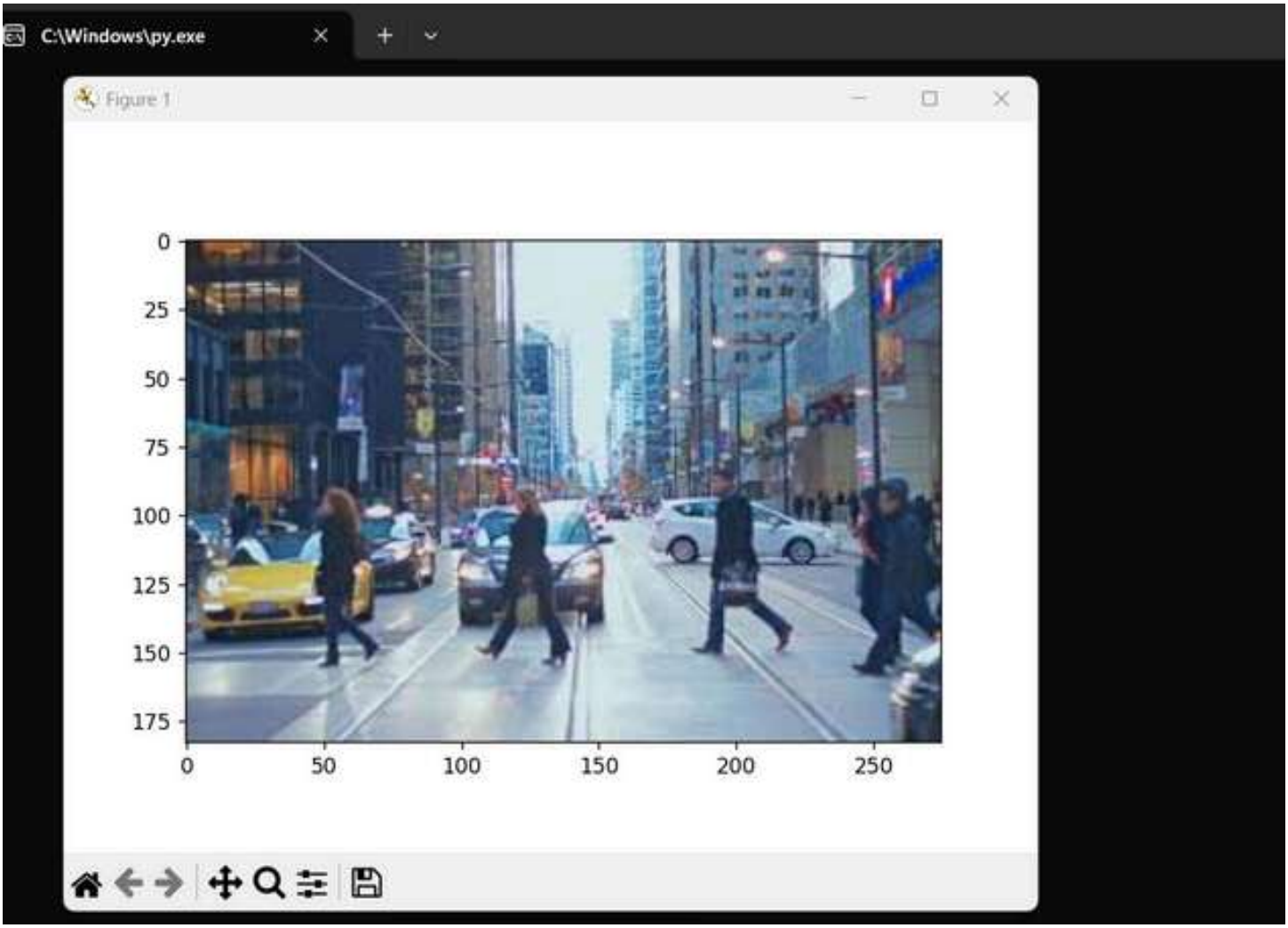

[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\Users\dc197> pip install matplotlib
Collecting matplotlib
Downloading matplotlib-3.10.5-cp313-cp313-win_amd64.whl.metadata (11 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
Downloading contourpy-1.3.3-cp313-cp313-win_amd64.whl.metadata (5.5 kB)
Collecting cycler>=0.10 (from matplotlib)
Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
Downloading fonttools-4.59.1-cp313-cp313-win_amd64.whl.metadata (111 kB)


Downloading pyparsing-3.2.3-py3-none-any.whl (111 kB)
Installing collected packages: pyparsing, pillow, packaging, numpy, kiwisc
Successfully installed contourpy-1.3.3 cycler-0.12.1 fonttools-4.59.1 kiwi
ing-25.0 pillow-11.3.0 pyparsing-3.2.3
[notice] A new release of pip is available: 24.2 -> 25.2
[notice] To update, run: python.exe -m pip install --upgrade pip


File Edit Selection View Go Run Terminal Help
Python-1.php
C: > Users > dc197 > OneDrive > Desktop > Python-1.php
1
2 import boto3
3 import matplotlib.pyplot as plt
4 from PIL import Image
5 from io import BytesIO

C: > Users > dc197 > OneDrive > Desktop > Python-1.py

```python

import boto3
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image
from io import BytesIO

Add the code for your specific AWS task here.
 For example, to download an image from S3 and display it.
try:
    s3 = boto3.client('s3')
    bucket_name = 'your-bucket-name'
    file_name = 'your-image-file.jpg'

    obj = s3.get_object(Bucket=bucket_name, Key=file_name)
    image_data = obj['Body'].read()

    img = Image.open(BytesIO(image_data))
    plt.imshow(img)
    plt.show()

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

This line keeps the window from closing immediately
input("Press Enter to exit...")

```


C:\Windows\py.exe
Figure 1
0
25
50
75
100
125
150
175
0
50
100
150
200
250

Python-1.py | image_labeler.py

C: > Users > dc197 > OneDrive > Desktop > Python-1.py

```
7   def detect_labels(photo, bucket):
17      print('Detected labels for ' + photo)
18      print()
19      for label in response['Labels']:
20          print("Label:", label['Name'])
21          print("Confidence:", label['Confidence'])
22          print()
23
24      # Load the image from S3
25      s3 = boto3.resource('s3')
26      obj = s3.Object(bucket, photo)
27      img_data = obj.get()['Body'].read()
28      img = Image.open(BytesIO(img_data))
29
30      # Display the image with bounding boxes
31      plt.imshow(img)
32      ax = plt.gca()
33      for label in response['Labels']:
34          for instance in label.get('Instances', []):
35              bbox = instance['BoundingBox']
36              left = bbox['Left'] * img.width
37              top = bbox['Top'] * img.height
38              width = bbox['Width'] * img.width
39              height = bbox['Height'] * img.height
```

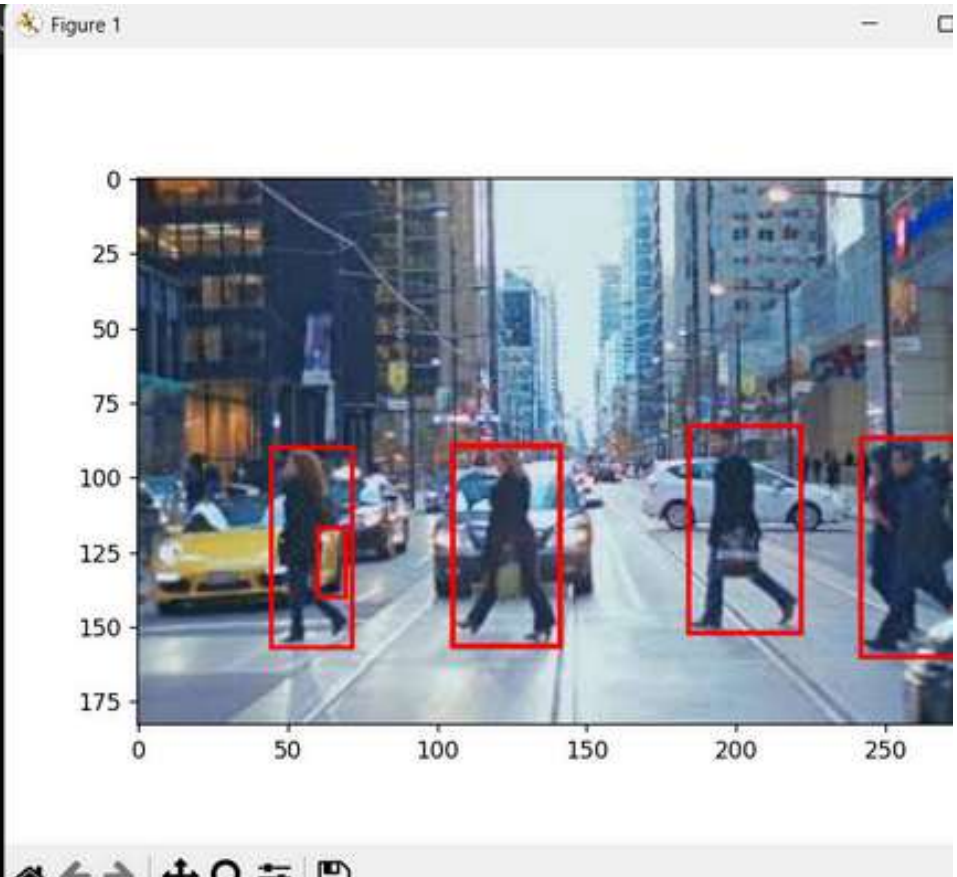
Python-1.py | image_labeler.py

C: > Users > dc197 > OneDrive > Desktop > image_labeler.py

```python
# Imports for the program
import boto3
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image
from io import BytesIO

# The function to detect labels on an S3 image
def detect_labels(photo, bucket):
    # Create a Rekognition client
    client = boto3.client('rekognition')

    # Detect labels in the photo
    response = client.detect_labels(
        Image={'S3Object': {'Bucket': bucket, 'Name': photo}},
        MaxLabels=10)

    # Print detected labels
    print(f"Detected labels for {photo}")
```

Command Prompt - python i

```
Confidence: 100.00%

Label: Walking
Confidence: 100.00%

Label: City
Confidence: 99.99%

Label: Pedestrian
Confidence: 99.93%

Label: Street
Confidence: 99.59%

Label: Urban
Confidence: 99.59%

Label: Adult
Confidence: 97.84%

Label: Male
Confidence: 97.84%

Label: Man
Confidence: 97.84%

Label: Handbag
```



```
Confidence: 100.00%

Label: Walking
Confidence: 100.00%

Label: City
Confidence: 99.99%

Label: Pedestrian
Confidence: 99.93%

Label: Street
Confidence: 99.59%

Label: Urban
Confidence: 99.59%

Label: Adult
Confidence: 97.84%

Label: Male
Confidence: 97.84%

Label: Man
Confidence: 97.84%

Label: Handbag
Confidence: 92.16%
```