About

In this project, we'll be building an image label generator, using Amazon Rekognition. Amazon Rekognition uses AI to recognize objects in images, to help with combing through many images without having to open and label them individually.

The steps we'll be going through to utilize Rekognition ourselves will consist of:

1. Creating an Amazon S3 Bucket
2. Uploading images to the S3 Bucket
3. Installing and configuring the AWS Command Line Interface (CLI)
4. Importing libraries
5. Adding the detect_labels function
6. Adding main function
7. Running the python file

Services Used

Amazon S3: For storing the images we'll be using for generating labels

Amazon Rekognition: To analyze our stored images and generate image labels

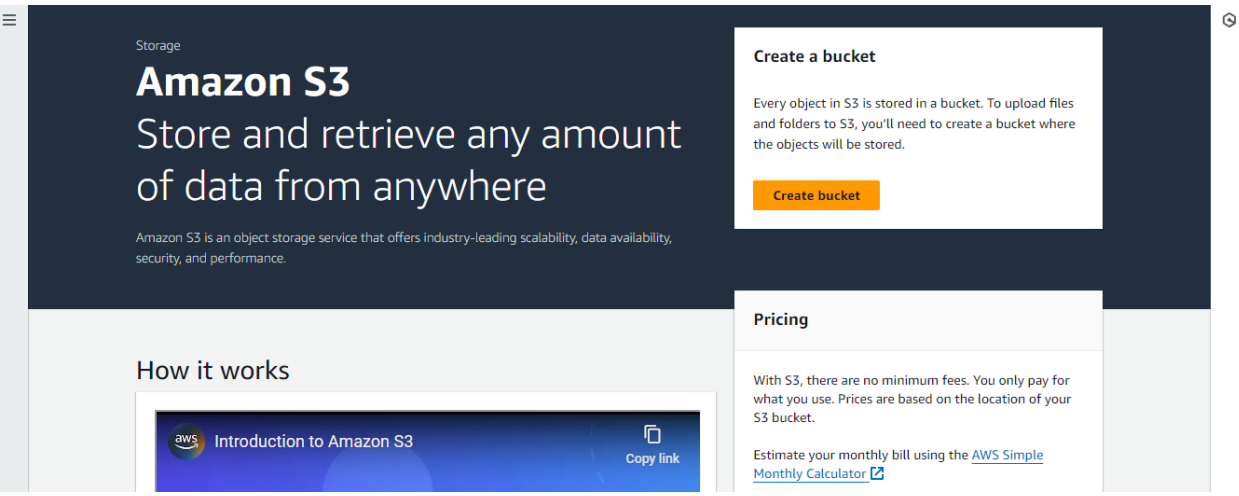
AWS CLI: Interacting with AWS via their Command Line Interface

Time & Cost

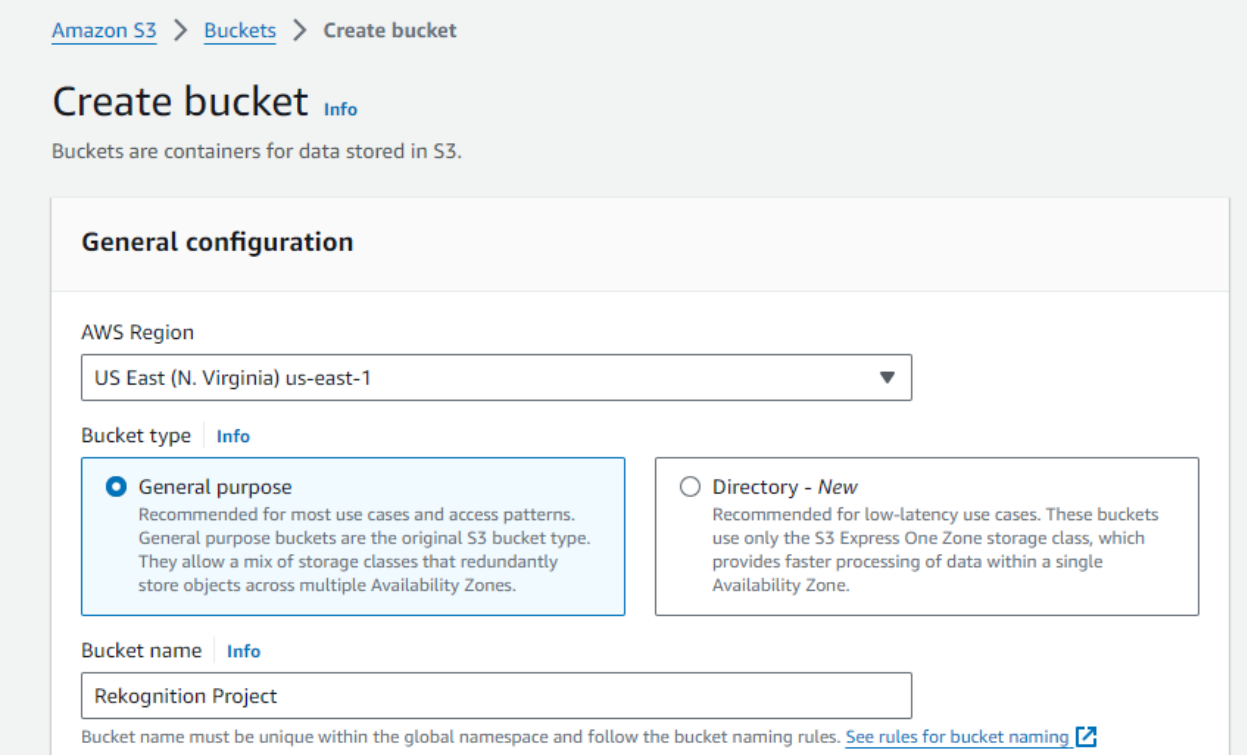
This project should take around half an hour, and the cost is free when using AWS Free Tier.

Step 1

To start, log in to your AWS Management Console and type S3 in the search bar. Clicking on it, you'll be led to this screen. Click the orange "Create bucket", which is a virtual storage box within AWS to store files for safekeeping.

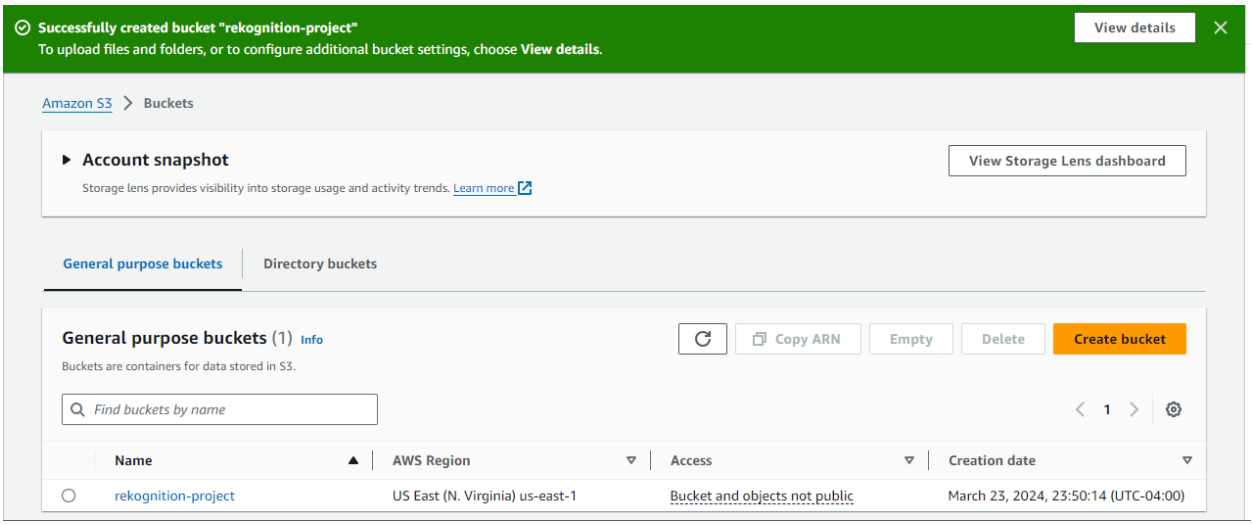


For now, we only need to configure our AWS region, and the name of our bucket.



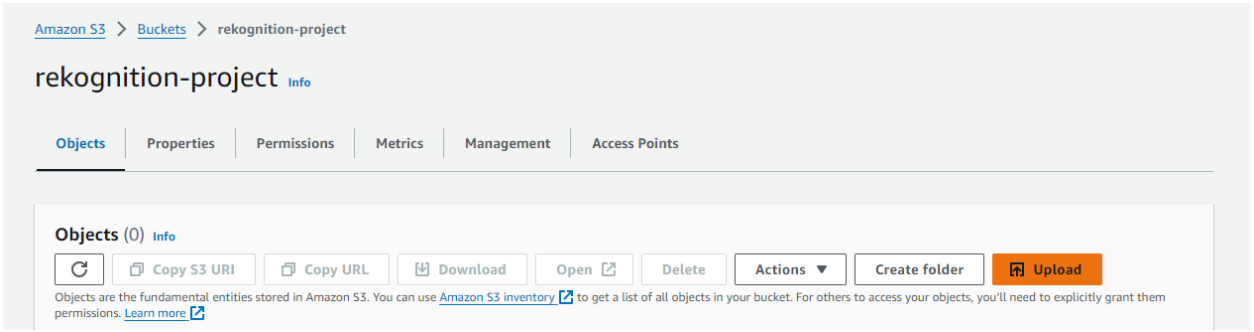
Leave everything else as default, and Create Bucket at the bottom of the page.

Once finished, a green bar will appear at the top of the screen letting you know your bucket was created successfully.



Step 2

Now that our bucket is created, click on it and on the next screen, click the orange Upload button



From there, follow the steps and upload a few photos, preferably ones with many objects for Rekognition to detect and label, such as a busy city street.

Step 3

Now that we've uploaded the photos, we need to install the AWS CLI. To install, open the command prompt or terminal on your computer and run the relevant command for your operating system.

The commands are as follows:

```
TEXT
#For Windows:
msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi

#For macOS (using Homebrew):
brew install awscli

#For Linux (using package manager):
sudo apt-get install awscli
```

Once you've entered the correct command, the AWS CLI installer will begin setup

Follow the steps for the installer. Once finished, you can check if it installed successfully by running the command 'aws –version'.

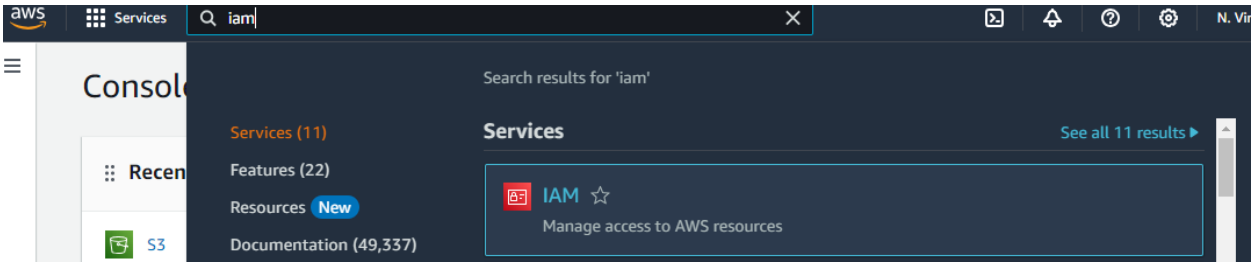
```
aws-cli/2.15.32 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
```

Now that we've installed AWS CLI, we need to configure it with our user keys in AWS to use their services.

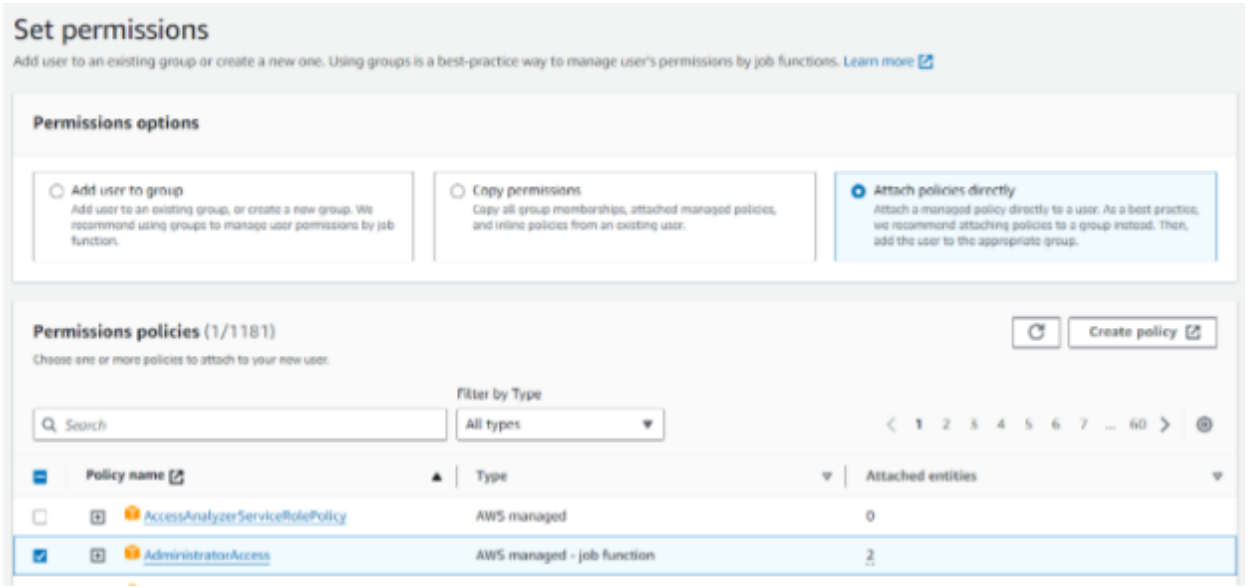
We do this by running the command 'aws configure'

You'll then be asked for an AWS Access Key, which you can obtain from the AWS Management Console.

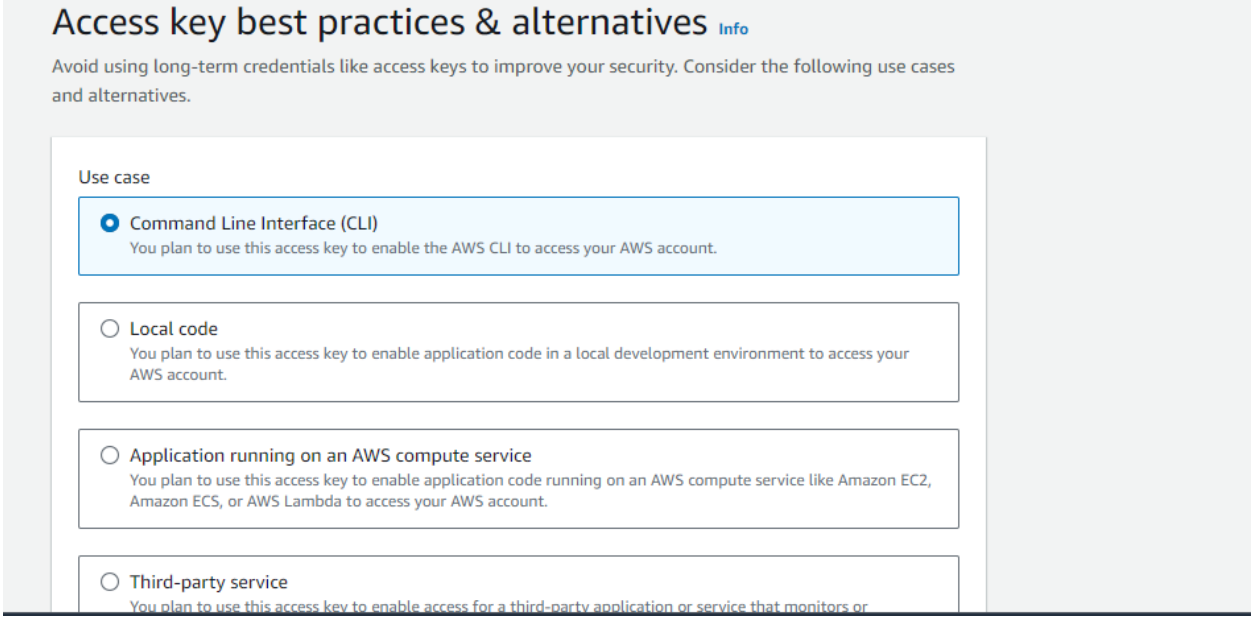
Type IAM in the search bar



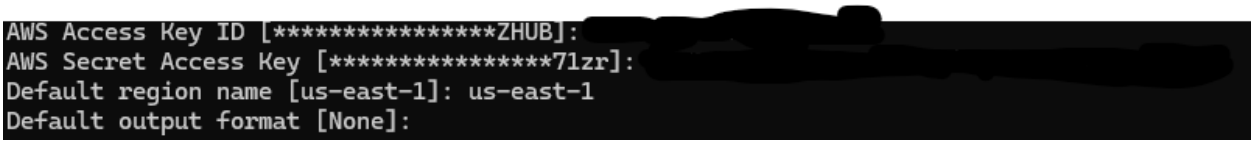
Go to **Users** and click **Create User**. Give the user a name and click **Next**. For the permission options, choose 'Attach Policies Directly' and attach the *'AdministratorAccess'* policy.



Once the user is created, click on Create access key in the Summary bar at the top. Once there, confirm your use case as CLI, and click Next.



Now you have access to your Access key and the Secret Access Key. Copy and paste them into your command console when prompted. The next question you'll be asked is the default region name, mine being *us-east-1*. The final configuration question is the default output format, which you can leave blank.



We've now configured our AWS CLI. We'll be utilizing it in the next steps by writing code for extracting pictures from our S3 bucket and applying detect_labels from Rekognition.

Step 4

Open your preferred IDE and create a .py file for our coding. I used VS Code, but there are many other options you can choose from.

Open your computer's command prompt and install the libraries we need for this project

Pip install boto3

Pip install matplotlib

Once we've installed those two in the command prompt, we need to import them. The list of the libraries and their descriptions are as follows:

1. Boto3- for interacting with AWS services

2. Matplotlib- for visualization
3. PIL (Python Imaging Library)- for handling image data
4. BytesIO- from the io module to work with image data

Add the code below in your .py file

```
1  import boto3
2  import matplotlib.pyplot as plt
3  import matplotlib.patches as patches
4  from PIL import Image
5  from io import BytesIO
```

Step 5

Then, right below the code, we'll define a function called **detect_labels**. This function takes a photo and bucket name as input parameters. Within this function we:

- Create a **Rekognition** client using boto3.
- We use the **detect_labels** method of the Rekognition client to detect labels in the given photo.
- We print the detected labels along with their confidence levels.
- We load the image from the **S3 bucket** using boto3 and PIL.
- We use **matplotlib** to display the image and draw bounding boxes around the detected objects.

The code is as follows:

```python
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('Detected labels for ' + photo)
    print()
    for label in response['Labels']:
        print("Label:", label['Name'])
        print("Confidence:", label['Confidence'])
        print()

    # Load the image from S3
    s3 = boto3.resource('s3')
    obj = s3.Object(bucket, photo)
    img_data = obj.get()['Body'].read()
    img = Image.open(BytesIO(img_data))

    # Display the image with bounding boxes
    plt.imshow(img)
    ax = plt.gca()
    for label in response['Labels']:
        for instance in label.get('Instances', []):
            bbox = instance['BoundingBox']
            left = bbox['Left'] * img.width
            top = bbox['Top'] * img.height
            width = bbox['Width'] * img.width
            height = bbox['Height'] * img.height
            rect = patches.Rectangle((left, top), width, height, linewidth=1, edgecolor='r', facecolor='none'
            ax.add_patch(rect)
            label_text = label['Name'] + ' (' + str(round(label['Confidence'], 2)) + '%)'
            plt.text(left, top - 2, label_text, color='r', fontsize=8, bbox=dict(facecolor='white', alpha=0.7
    plt.show()

    return len(response['Labels'])
```

Step 6

Next, let's write a main function to test our **detect_labels** function. We specify one of the sample photos we added to our bucket, then call the detect labels function with these parameters.

```
def main():
    photo = 'image_file_name'
    bucket = 'bucket_name'
    label_count = detect_labels(photo, bucket)
    print("Labels detected:", label_count)

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

Referring to the two highlighted lines above, be sure to change ‘image_file_name’ and ‘bucket_name’ to the configured names you used back in steps 1 & 2.

Save this as a .py file and keep track of where it’s located.

Step 7

Now we can run our project. Open your terminal in the directory where the python file we made is present and run the command

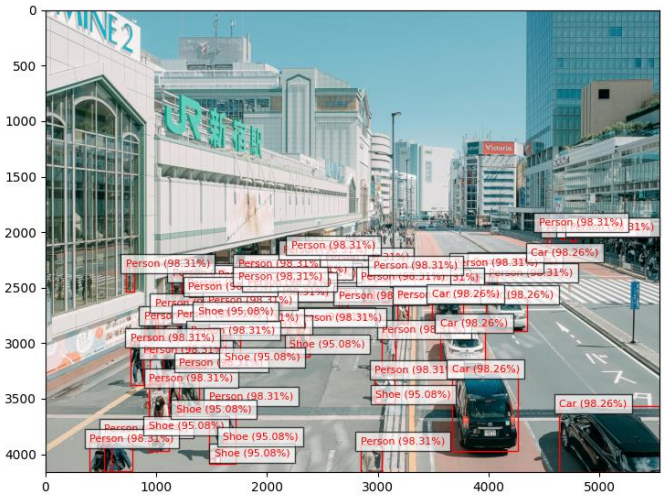
Python *name_of_python_file*.py

Make sure to change the text in red above with the actual name of your python file you saved.

If your code was successful, the result should give you an output of 10 detected labels and their confidence levels, along with a pop-up screen displaying the image uploaded to your S3 bucket with bounding boxes present on the generated labels.

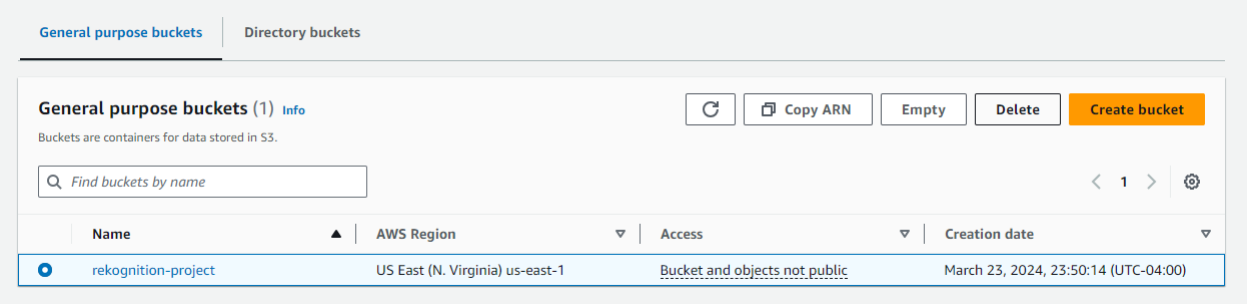
This also marks the completed of this project. If you’ve made it this far, Great Job!

```
PS C:\Users\Marcu\.vscode\Projects> python rekognition.py
Detected labels for pexels-wencheng-jiang-7161188

Label: Road
Confidence: 99.99205780029297

Label: City
Confidence: 99.98796081542969

Label: Metropolis
Confidence: 99.98336791992188

Label: Urban
Confidence: 99.98336791992188

Label: Cityscape
Confidence: 99.2630386352539

Label: Person
Confidence: 98.30790710449219

Label: Car
Confidence: 98.25601959228516

Label: Shoe
Confidence: 95.0800552368164

Label: Pedestrian
Confidence: 89.37918090820312

Label: Downtown
Confidence: 86.62726593017578
```



Clean Up

Log back in to AWS Management Console and access S3. Delete the bucket we created. AWS will first require you to empty the bucket of your images first before you delete the bucket itself.



Finally, navigate to IAM from the search bar and delete your User that was granted CLI access.

Takeaways

A good practice to consider when using Access keys is to rotate access keys regularly and disable/delete them when no longer in use. During my documentation of this project, I found I had an old access key that was approaching 80 days.

The hardest part of this project for me was configuring VS Code, the IDE I utilized during this project. The struggles I had with downloading everything, making beginner blunders, and getting used to the software could be an entire documentation project itself.

There are more functionalities that can be used in Amazon Rekognition such as video label detection, used for detecting labels in each frame. This could be useful for automating your video editing or surveillance.

Similarly, you can also configure Rekognition to get object labels in real time, not just for static photos and pre recorded videos; Rekognition even offers face recognition APIs for detecting and recognizing faces.