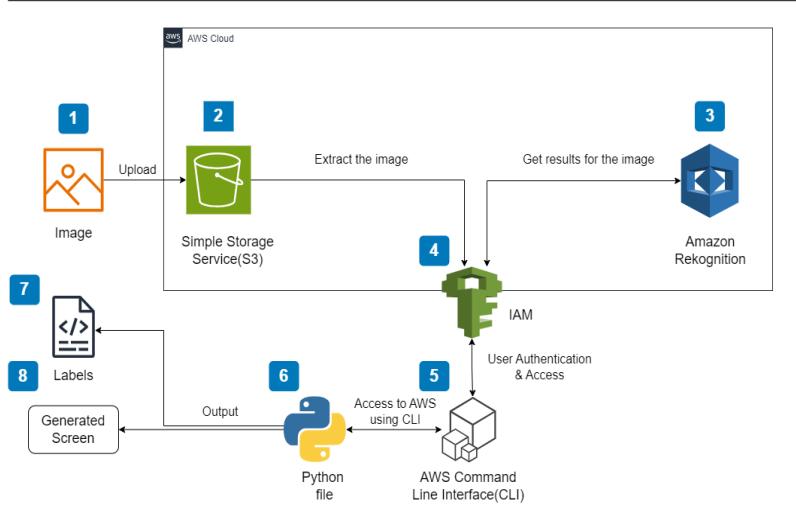


BUILD AN IMAGE LABELS GENERATOR USING AMAZON REKOGNITION

Here is the architectural plan of my application:

IMAGE LABELS GENERATOR USING AWS REKOGNITION



STEP 1:

Create an Admin user

Before i started my project i created an user admin via IAM in aws console so i don't have to use my root account for security purposes:

IAM > Users > Create user

Step 1
 Specify user details
 Step 2
 Set permissions
 Step 3
 Review and create

Specify user details

User details

User name

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

[Cancel](#) [Next](#)

Here i am logged into the aws console as an admin:



Console Home [Info](#)

[Reset to default layout](#) [+ Add widgets](#)

Recently visited [Info](#)

No recently visited services

Explore one of these commonly visited AWS services.

[EC2](#) [S3](#) [RDS](#) [Lambda](#)

[View all services](#)

Applications (0) [Info](#)

Region: US East (N. Virginia)

[Create application](#)

us-east-1 (Current Region) [Find applications](#)

Name Description Region Originat. [Create application](#)

No applications
Get started by creating an application.

[Go to myApplications](#)

Welcome to AWS

Getting started with AWS [Info](#)

 Learn the fundamentals and find valuable information to

AWS Health [Info](#)

Open issues 0 Past 7 days

Cost and usage [Info](#)



STEP 2:

Create a S3 bucket and upload images

The screenshot shows the 'Create bucket' wizard. In the 'General configuration' section, the 'Bucket name' is set to 'aws-recognition-label-image-1986'. Under 'Bucket type', 'General purpose' is selected. In the 'Object Ownership' section, 'ACLs disabled (recommended)' is selected.

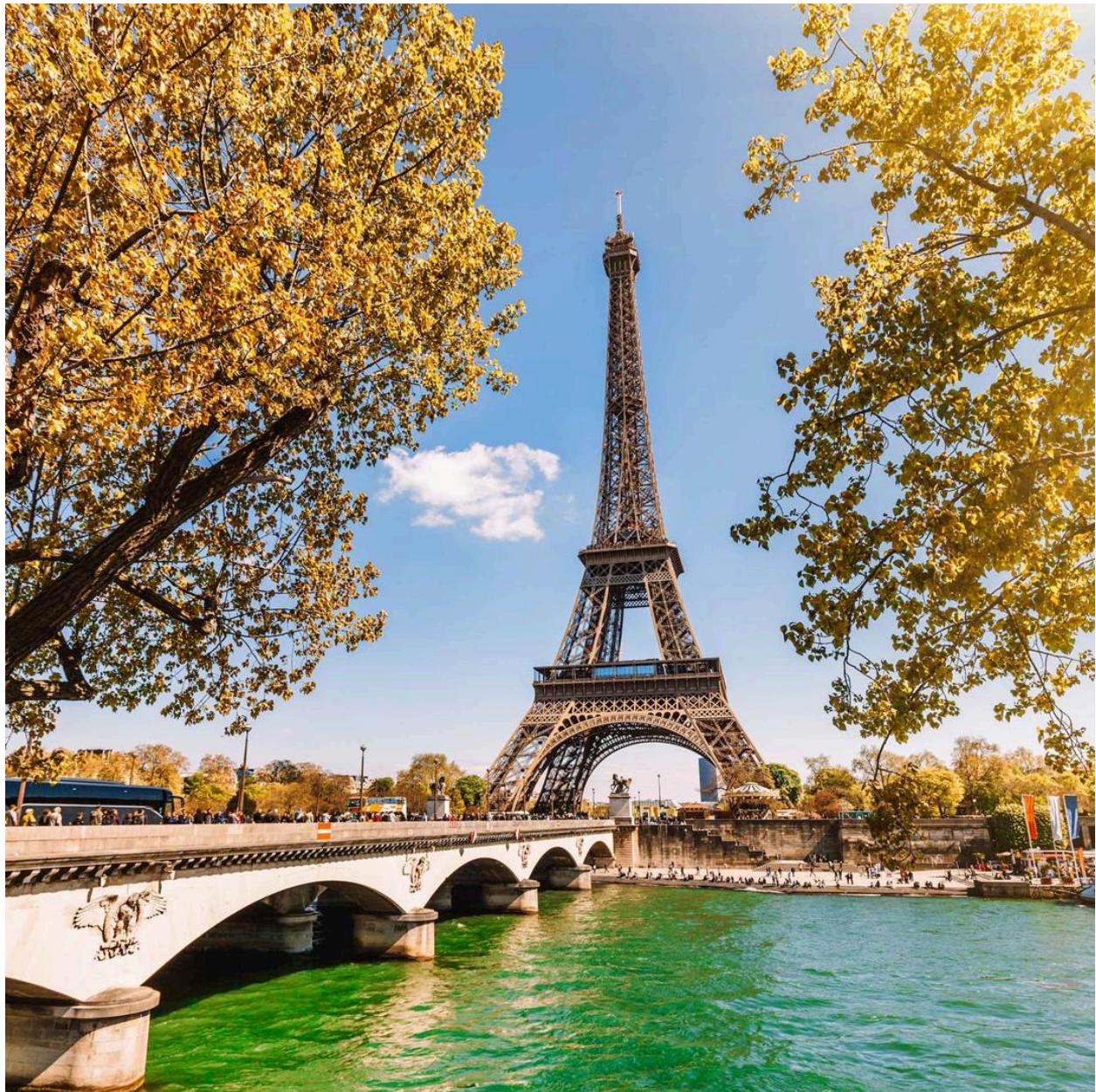
Bucket is created

The screenshot shows the 'Buckets' page. A green banner at the top indicates 'Successfully created bucket "aws-recognition-label-image-1986"'. Below it, an 'Account snapshot' provides storage usage information. The 'General purpose buckets' tab is selected, showing a table with one item: 'aws-recognition-label-image-1986'.

Name	AWS Region	IAM Access Analyzer	Creation date
aws-recognition-label-image-1986	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 1, 2025, 13:27:41 (UTC-08:00)

Now it's time to upload the image we want to analyse:

These are pictures that i uploaded:





Amazon S3 > Buckets > aws-recognition-label-image-1986

aws-recognition-label-image-1986 [Info](#)

[Objects](#) [Metadata](#) [Properties](#) [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Objects (0)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

[Upload](#)

No objects
You don't have any objects in this bucket.

[Upload](#)

Images have been uploaded

The screenshot shows the AWS S3 console interface. At the top, there's a breadcrumb navigation: Amazon S3 > Buckets > aws-recognition-label-image-1986. Below the navigation is a header with tabs: Objects (highlighted in blue), Metadata, Properties, Permissions, Metrics, Management, and Access Points. A red box highlights the 'aws-recognition-label-image-1986' bucket name. Below the header is a toolbar with actions: Copy S3 URI, Copy URL, Download, Open, Delete, Actions (with a dropdown arrow), Create folder, and Upload. A search bar labeled 'Find objects by prefix' is present. The main area displays a table of objects:

Name	Type	Last modified	Size	Storage class
france-eiffel-tower-paris.jpg	jpg	March 1, 2025, 13:37:38 (UTC-08:00)	316.7 KB	Standard
traffic_paris.webp	webp	March 1, 2025, 13:45:41 (UTC-08:00)	248.1 KB	Standard

STEP 3

Installing the AWS Command line interface(CLI) ON MAC

- a) I had to download the package via this command line (*the most challenging was to find the latest version of that package*):

```
curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
```

- b) Install the package:

```
sudo installer -pkg AWSCLIV2.pkg -target /
```

- c) Verify if the package got installed:

```
aws --version
```

Everything been successfully installed

- d) Configuration of AWS CLI:

```
aws configure
```

```
[billytismoreau@Billytiss-MacBook-Air ~ % aws configure
AWS Access Key ID [None]: ]
```

I had to create an access key and secret access key so i can get access to my aws account via AWS CLI.

From my root account I created an access key and secret access key for my Admin user so I can connect to my account via my terminal.

The screenshot shows the 'Summary' tab of a user named 'billytisSA'. It includes the ARN (arn:aws:iam::082706928695:user/billytisSA), which is highlighted with a red box. Below it are 'Console access' status ('Enabled without MFA'), 'Last console sign-in' (Today), and a button to 'Create access key' (also highlighted with a red box). Below the summary, there are tabs for 'Permissions', 'Groups', 'Tags', 'Security credentials', and 'Last Accessed'. The 'Permissions' tab is selected. Under 'Permissions policies (1)', it shows a policy named 'AdministratorAccess' attached directly. A 'Permissions boundary (not set)' section is also present.

Access key and secret access key created:

Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key

Secret access key



***** Show

My AWS CLI has been configured

```
billytismoreau@Billytiss-MacBook-Air ~ % aws configure
AWS Access Key ID [None]: [REDACTED]
AWS Secret Access Key [None]: [REDACTED]
Default region name [None]: us-est-1
Default output format [None]:
```

For the next step I wrote the python code for extracting pictures from S3 bucket and applied **detect_labels** operation from Rekognition to generate the labels with their confidence score.

STEP 4

Import Libraries

I had to add these libraries (**boto3 and matplotlib**) to my terminal but i had to make sure i have the latest version of python:

By typing these commands on my terminal:

```
python3 --version
pip3 --version
```

And then install the library with this command:

```
pip3 install boto3 matplotlib
```

And then to finish i had to check if Boto3 and Matplotlib are installed by running this command:

```
python3 -c "import boto3; print(boto3.__version__)" python3 -c "import matplotlib;
print(matplotlib.__version__)"
```

STEP5

Define Functions

- a) I define a function called **detect_labels**
- b) I created a Rekognition client using boto3
- c) I used the detect_labels method of the Rekognition client to detect labels in the given photo
- d) I printed the detected labels along with their confidence levels
- e) I loaded the image from the S3 bucket using boto3 and PIL

- f) I used matplotlib to display the image and draw bounding boxes around the detected objects

Detect_Labels_Function Code:

```

detect_labels_fuction.py  X  +
7 def detect_labels(photo, bucket):
8     # Create a Rekognition client
9     client = boto3.client('rekognition')
10
11    # Detect labels in the photo
12    response = client.detect_labels(
13        Image={'S3Object': {'Bucket': bucket, 'Name': photo}},
14        MaxLabels=10)
15
16    # Print detected labels
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
40            rect = patches.Rectangle((left, top), width, height, linewidth=1, edgecolor='r',
facecolor='none')
41            ax.add_patch(rect)
42            label_text = label['Name'] + ' (' + str(round(label['Confidence'], 2)) + '%)'
43            plt.text(left, top - 2, label_text, color='r', fontsize=8, bbox=dict(facecolor='white',
facecolor='none'))
44            ax.add_patch(rect)
45            label_text = label['Name'] + ' (' + str(round(label['Confidence'], 2)) + '%)'
46            plt.text(left, top - 2, label_text, color='r', fontsize=8, bbox=dict(facecolor='white',
alpha=0.7))
47    plt.show()
48
49    return len(response['Labels'])

```

Next, I wrote a main function to test my detect_labels function. I specify a sample photo and bucket name, then called the detect_labels function with these parameters.

MAIN_FUNCTION CODE:

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

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

Once the code was written I changed my ‘image_file_name’ and ‘bucket_name’ to my actual configured naming.

```
def main():
    photo = 'france-eiffel-tower-paris.jpg'
    bucket = 'aws-recognition-label-image-1986'
    label_count = detect_labels(photo, bucket)
    print("Labels detected:", label_count)

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

STEP 6

Running your Project

I created a file Named detect_labels.py on my MacOs terminal, then copy/Paste the code in that file, saved it, exited out. And ran that command: **python3 detect_labels.py**

Here is the final result of the

```
3.5.4
~ $ python3 detect_labels.py
Detected labels for france-eiffel-tower-paris.jpg

Label: Architecture
Confidence: 96.84970092773438

Label: Building
Confidence: 96.84970092773438

Label: Eiffel Tower
Confidence: 96.84970092773438

Label: Landmark
Confidence: 96.84970092773438

Label: Tower
Confidence: 96.84970092773438

Label: Bridge
Confidence: 77.93797302246094

Label: Bus
Confidence: 70.84432220458984

Label: Transportation
Confidence: 70.84432220458984

Label: Vehicle
Confidence: 70.84432220458984

Label: Person
Confidence: 62.174354553222656

Labels detected: 10
~ $ []
```

END

CHALLENGES DURING THIS PROJECT

The challenges that I faced during this project was when I ran the command **python3 detect_labels.py** I had an error message because the region of the bucket had to be explicit in my code and I had a syntax error at line 46.