

IMAGE PROCESSING USING AWS LAMBDA AND AUTOMATIC MAILING SYSTEM

TEAM MEMBERS:

- HARISH KUMAR C N (163018)
- KARTHIK B (1631021)

STEPS:

1. Create a S3 bucket to store the images that you want to process:

The screenshot shows the Amazon S3 console interface. At the top, there are buttons for '+ Create bucket', 'Delete bucket', and 'Empty bucket'. Below this, a search bar contains the placeholder 'Search for buckets'. On the right, it displays '1 Buckets' (1 Public), '1 Regions', and the date 'Sep 26, 2018 8:56:48 AM GMT+0530'. A table lists one bucket: 'harishimage-rec' (Not public), located in 'US East (N. Virginia)', created on 'Sep 26, 2018 8:56:48 AM GMT+0530'. A note at the bottom states: '* Objects might still be publicly accessible due to object ACLs. [Learn more](#)'.

2. Create a IAM role to grant access permission to the Lambda service of was:

The screenshot shows the AWS IAM Roles page. The left sidebar has a 'Roles' section selected. The main area shows the 'Summary' tab for a role named 'harish-recognition-s3'. It displays the Role ARN (arn:aws:iam::617994368727:role/harish-recognition-s3), Role description (harish-recognition-s3 | Edit), Instance Profile ARNs, Path (/), Creation time (2018-09-26 08:51 UTC+0530), and Maximum CLI/API session duration (1 hour | Edit). Below this, there are tabs for 'Permissions', 'Trust relationships', 'Access Advisor', and 'Revoke sessions'. The 'Permissions' tab is active, showing 'Permissions policies (3 policies applied)'. It lists three policies: 'AmazonS3FullAccess', 'CloudWatchFullAccess', and 'AmazonRekognitionFullAccess', all of which are AWS managed policies. There are also sections for 'Add inline policy' and 'Permissions boundary (not set)'.



3. In this role give access permissions to three things, they are:

The screenshot shows the 'Permissions' tab selected in the AWS IAM console. Below it, a section titled 'Permissions policies (3 policies applied)' is expanded. A blue button labeled 'Attach policies' is visible. The list of applied policies includes:

- AmazonS3FullAccess
- CloudWatchFullAccess
- AmazonRekognitionFullAccess

4. Create SNS Topic:

The screenshot shows the 'Topic details' page for the topic 'harish-recog-sns'. The topic ARN is arn:aws:sns:us-east-1:617994368727:harish-recog-sns. The topic owner is 617994368727, located in the us-east-1 region, with a display name of reg-image.

Below the topic details, there is a 'Subscriptions' section. It shows one subscription entry:

<input type="checkbox"/>	Subscription ID	Protocol	Endpoint	Subscriber
<input type="checkbox"/>	arn:aws:sns:us-east-1:617994368727:harish-recog-sns:1dcae60d-c65a-463c-9934-ec646...	email	ciharishnagaraj@gmail.com	617994368727

5. Create subscription to send notification message to specified email:

Subscriptions

The screenshot shows the 'Subscriptions' page. It displays the same subscription entry as the previous screenshot:

<input type="checkbox"/>	Subscription ID	Protocol	Endpoint	Subscriber
<input type="checkbox"/>	arn:aws:sns:us-east-1:617994368727:harish-recog-sns:1dcae60d-c65a-463c-9934-ec646...	email	ciharishnagaraj@gmail.com	617994368727

6. Create Lambda function via python boto3 by adding S3 bucket:

The screenshot shows the AWS Lambda console's 'Functions' list. There is one function listed:

Function name	Description	Runtime	Code size	Last modified
my-recog-harish	An Amazon S3 trigger that uses rekognition APIs to detect faces	Python 2.7	2.1 kB	1 hour ago

7. Python code will be generated using boto3:

The screenshot shows the AWS Lambda function editor for 'my-recog-harish'. The code entry type is 'Edit code inline'. The runtime is 'Python 2.7' and the handler is 'lambda_function.lambda_handler'. The code editor displays the following Python script:

```
1 #!/usr/bin/python
2
3 import boto3
4 from decimal import Decimal
5 import json
6 import urllib
7
8 print('Loading function')
9
10rekognition = boto3.client('rekognition')
11client = boto3.client('sns')
12
13# ----- Helper Functions to call Rekognition APIs -----
14
15#def detect_faces(bucket, key):
16#    response =rekognition.detect_faces(Image={"S3Object": {"Bucket": bucket, "Name": key}})
17#    return response
18
19#def detect_labels(bucket, key):
20#    response =rekognition.detect_labels(Image={"S3Object": {"Bucket": bucket, "Name": key}})
21
22# Sample code to write response to DynamoDB table 'MyTable' with 'PK' as Primary Key.
23# Note: role used for executing this Lambda function should have write access to the table.
24#table = boto3.resource('dynamodb').Table('MyTable')
```

8. Connect SNS by using publish() function and by setting client to SNS:

1. messages = client.publish(TargetArn='arn:aws:sns:us-east-1:617994368727:harish-recog-sns',Message="https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logStream:group=/aws/lambda/my-recog-harish;streamFilter=typeLogStreamPrefix",Subject='Uploaded Image-rekognition Lables')

9. Then add particular image in your S3 bucket then a trigger will be initiated and the process output will be send to your specified mail id.

reg-image

Inbox - Google 10:08 AM

R

Uploaded Image-rekognition Lables

To: Harish Kumar C N

<https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logStream:group=/aws/lambda/my-recog-harish;streamFilter=typeLogStreamPrefix>

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:

<https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:617994368727:harish-recog-sns:1dcae60d-c65a-463c-9934-ec6466b81ae6&Endpoint=cnharishnagaraj@gmail.com>

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at
<https://aws.amazon.com/support>

10. Uploaded pic is:



11. The process result is:

▶	04:38:04	Loading function
▶	04:38:04	START RequestId: d0b3fa1a-c145-11e8-a7aa-516d31dae641 Version: \$LATEST
▼	04:38:06	{u'Labels': [{u'Confidence': 99.1960220336914, u'Name': u'Human'}, {u'Confidence': 99.19601440429688, u'Name': u'People'}, {u'Confidence': 99.1960220336914, u'Name': u'Person'}, {u'Confidence': 54.31345748901367, u'Name': u'Crowd'}, {u'Confidence': 52.94205893383789, u'Name': u'Leisure Activities'}, {u'Confidence': 52.8636589050293, u'Name': u'Backyard'}, {u'Confidence': 52.8636589050293, u'Name': u'Outdoors'}, {u'Confidence': 52.8636589050293, u'Name': u'Yard'}, {u'Confidence': 50.706214904785156, u'Name': u'Female'}, {u'Confidence': 50.652442932128906, u'Name': u'Architecture'}, {u'Confidence': 50.652442932128906, u'Name': u'Building'}, {u'Confidence': 50.652442932128906, u'Name': u'Housing'}, {u'Confidence': 50.652442932128906, u'Name': u'Monastery'}], 'ResponseMetadata': {'RetryAttempts': 0, 'HTTPStatusCode': 200, 'RequestId': 'f586660e-c145-11e8-a10c-5556d2bdb396', 'HTTPHeaders': {'date': 'Wed, 26 Sep 2018 04:38:06 GMT', 'x-amzn-requestid': 'f586660e-c145-11e8-a10c-5556d2bdb396', 'content-length': '708', 'content-type': 'application/x-amz-json-1.1', 'connection': 'keep-alive'}}, 'u'OrientationCorrection': u'ROTATE_0'}
▶	04:38:07	END RequestId: d0b3fa1a-c145-11e8-a7aa-516d31dae641
▶	04:38:07	REPORT RequestId: d0b3fa1a-c145-11e8-a7aa-516d31dae641 Duration: 2186.75 ms Billed Duration: 2200 ms Memory Size: 128 MB Max Memory Used: 34 MB

Code:

Python code using boto3:

```
from __future__ import print_function

import boto3
from decimal import Decimal
import json
import urllib

print('Loading function')

rekognition = boto3.client('rekognition')
client = boto3.client('sns')

def detect_labels(bucket, key):
    response = rekognition.detect_labels(Image={"S3Object": {"Bucket": bucket,
"Name": key}})
    return response

# ----- Main handler -----
def lambda_handler(event, context):

    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.unquote_plus(event['Records'][0]['s3']['object']['key'].encode('utf8'))
    try:

        response = detect_labels(bucket, key)
        print(response)
        messages = client.publish(TargetArn='arn:aws:sns:us-
east-1:617994368727:harish-recog-sns',Message="https://
console.aws.amazon.com/cloudwatch/home?region=us-east-1#logStream:group=
aws/lambda/my-recog-
```

```
harish;streamFilter=typeLogStreamPrefix",Subject='Uploaded Image-rekognition  
Labels')  
    return response  
except Exception as e:  
    print(e)  
    print("Error processing object {} from bucket {}".format(key, bucket) +  
          "Make sure your object and bucket exist and your bucket is in the same  
region as this function.")  
    raise e
```

By this above code you will able to get an email of image labels whenever you add a image file into your S3 bucket.