SMART CAMERA APPLICATION WITH VOICE CHAT CAPABILITY USING RASPBERRY PI AND AWS

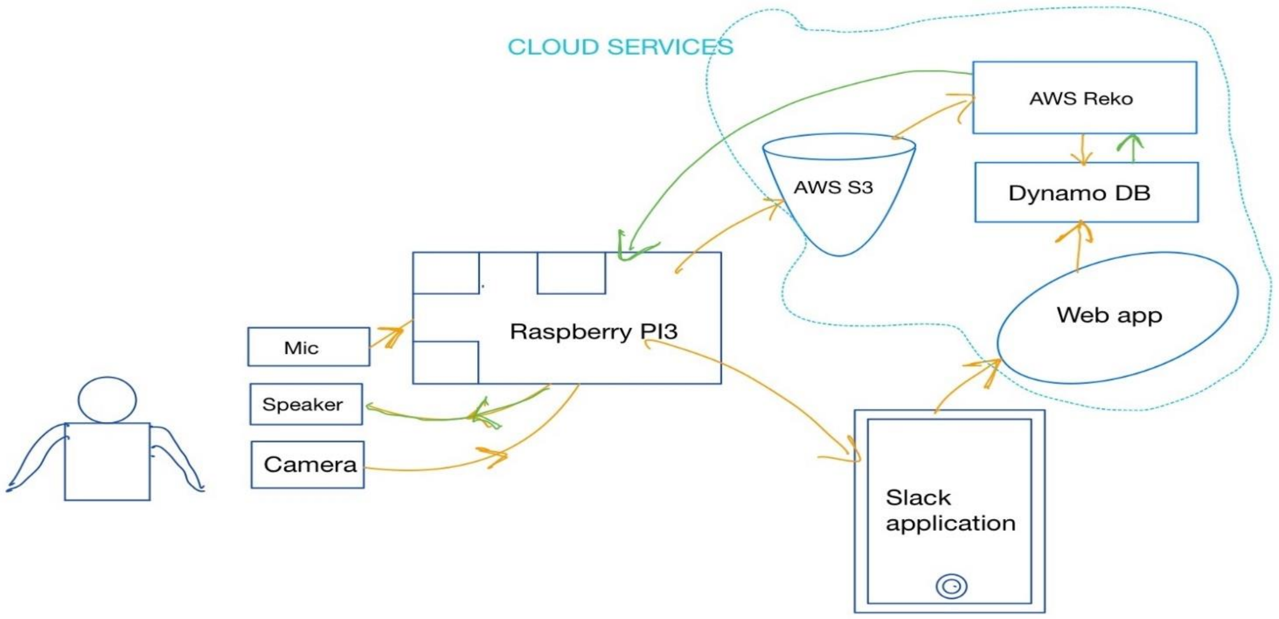
Introduction:

In this project, we designed a smart camera which detects the human faces and objects along with voice chat capability. This project includes hardware components Raspberry pi Model 3B, Pi Camera, Speaker, and Mic.

Software Installations to the Raspberry PI:

- The raspberry pi is configured with the operating system version stretch.
- Open CV modules are installed and configured to integrate the PI camera with the raspberry pi.
- Installed python3 and pip3 on Raspberry pi.
- Installed Sox on Raspberry Pi for integrating mic and speaker to the raspberry pi.

Architecture:



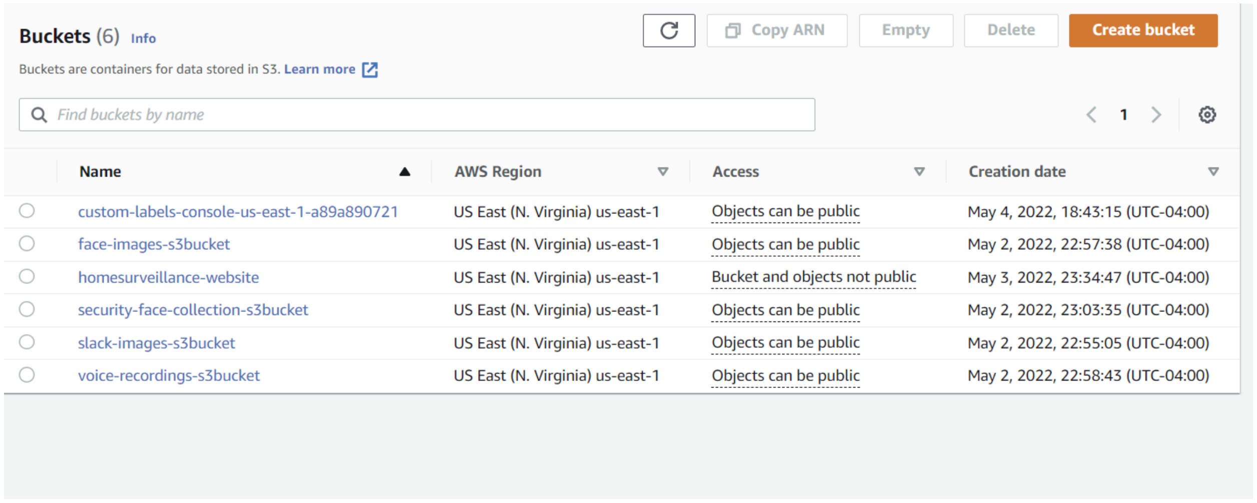
Created an AWS account to use the web services for storing the data to the cloud. The AWS services used for the project:

- S3 storage buckets
- Dynamo DB
- AWS Rekognition

S3 Storage:

In our application we created three buckets,

- “slack-images-s3bucket” to store images captured by raspberry pi.
- “voice-recordings-s3bucket” to store visitors voice recordings by pi.
- “face-images-s3bucket” to store indexed faces by AWS Rekognition.



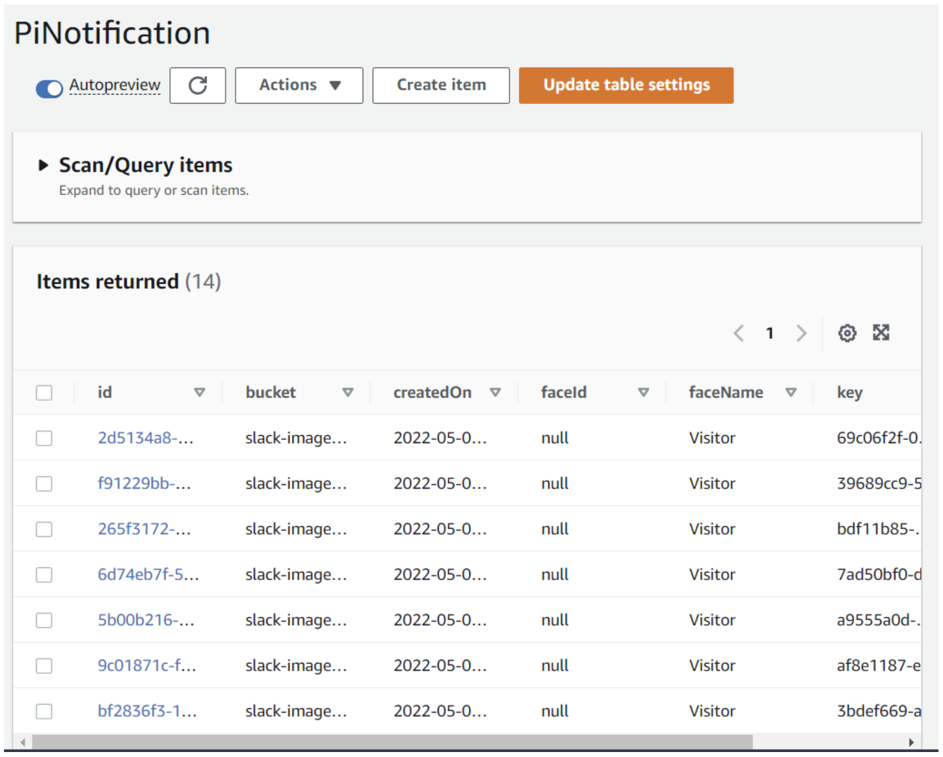
Name	AWS Region	Access	Creation date
custom-labels-console-us-east-1-a89a890721	US East (N. Virginia) us-east-1	Objects can be public	May 4, 2022, 18:43:15 (UTC-04:00)
face-images-s3bucket	US East (N. Virginia) us-east-1	Objects can be public	May 2, 2022, 22:57:38 (UTC-04:00)
homesurveillance-website	US East (N. Virginia) us-east-1	Bucket and objects not public	May 3, 2022, 23:34:47 (UTC-04:00)
security-face-collection-s3bucket	US East (N. Virginia) us-east-1	Objects can be public	May 2, 2022, 23:03:35 (UTC-04:00)
slack-images-s3bucket	US East (N. Virginia) us-east-1	Objects can be public	May 2, 2022, 22:55:05 (UTC-04:00)
voice-recordings-s3bucket	US East (N. Virginia) us-east-1	Objects can be public	May 2, 2022, 22:58:43 (UTC-04:00)

Dynamo DB:

We created a dynamo db tables to store all the slack images and voice recordings.

- “PI Notifications”
- “PI Messages”

The Pi Notification tables stores all the facial images that were captured by the pi camera which got registered in the slack.



PiNotification

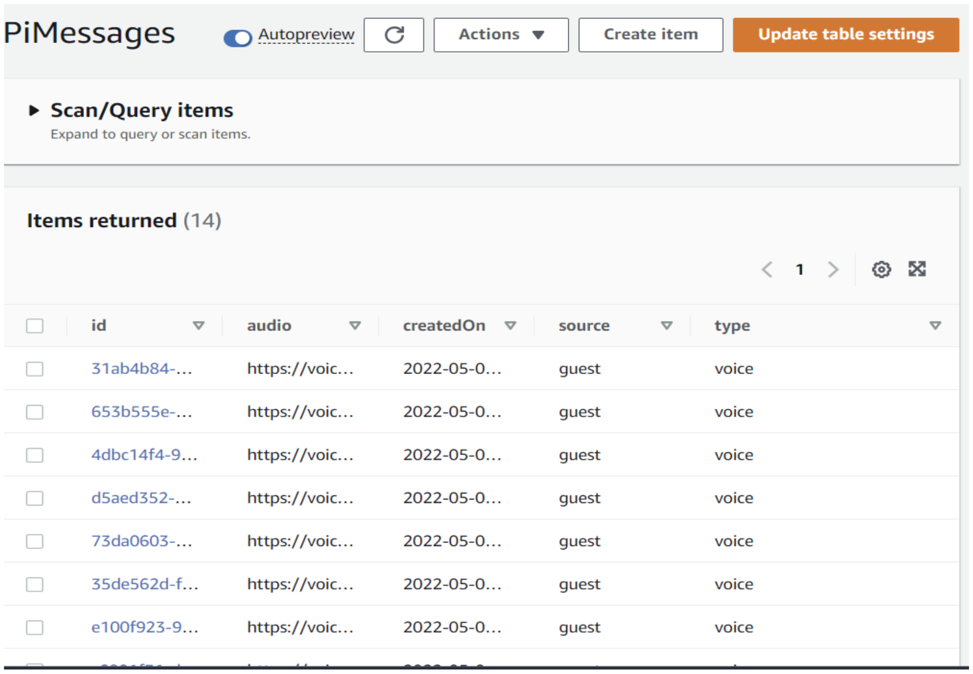
Autopreview | Actions | Create item | Update table settings

Scan/Query items
Expand to query or scan items.

Items returned (14)

id	bucket	createdOn	faceId	faceName	key
2d5134a8-…	slack-image…	2022-05-0…	null	Visitor	69c06f2f-0.
f91229bb-…	slack-image…	2022-05-0…	null	Visitor	39689cc9-5
265f3172-…	slack-image…	2022-05-0…	null	Visitor	bdf11b85-.
6d74eb7f-5…	slack-image…	2022-05-0…	null	Visitor	7ad50bf0-d
5b00b216-…	slack-image…	2022-05-0…	null	Visitor	a9555a0d-.
9c01871c-f…	slack-image…	2022-05-0…	null	Visitor	af8e1187-e
bf2836f3-1…	slack-image…	2022-05-0…	null	Visitor	3bdef669-a

The dynamo DB table Pi Messages will store the voice recordings of humans when they speak in front of the camera which were recorded through mic



PiMessages

Autopreview | Actions | Create item | Update table settings

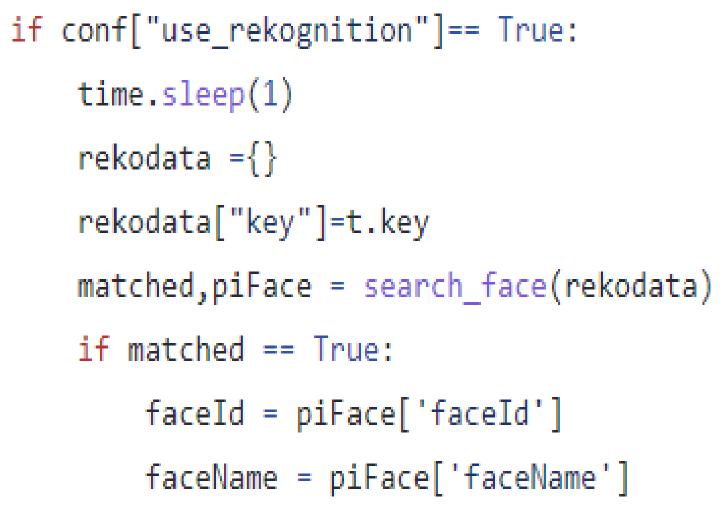
Scan/Query items
Expand to query or scan items.

Items returned (14)

id	audio	createdOn	source	type
31ab4b84-…	https://voic…	2022-05-0…	guest	voice
653b555e-…	https://voic…	2022-05-0…	guest	voice
4dbc14f4-9…	https://voic…	2022-05-0…	guest	voice
d5aed352-…	https://voic…	2022-05-0…	guest	voice
73da0603-…	https://voic…	2022-05-0…	guest	voice
35de562d-f…	https://voic…	2022-05-0…	guest	voice
e100f923-9…	https://voic…	2022-05-0…	guest	voice

AWS Rekognition:

The Amazon Rekognition service is used for identify the objects, people in the images and as well as to detect any inappropriate things in the video surveillance. The below code snippet represents the rekog services.



```
if conf["use_rekognition"]== True:
    time.sleep(1)
    rekodata ={}
    rekodata["key"]=t.key
    matched,piFace = search_face(rekodata)
    if matched == True:
        faceId = piFace['faceId']
        faceName = piFace['faceName']
```

Created slack webhook:

Here we are using Slack to get notified when someone appears in front of the camera. The Pi camera will detect the face and send the image to the slack channel.

Set up for incoming webhook:

Step 1: Registered to the slack account using https://api.slack.com/apps?new_app=1 .

Step 2: Once you login to the slack, From the Features page, toggle Activate Incoming Webhooks on.

Step 3: Click Add New Webhook to Workspace.

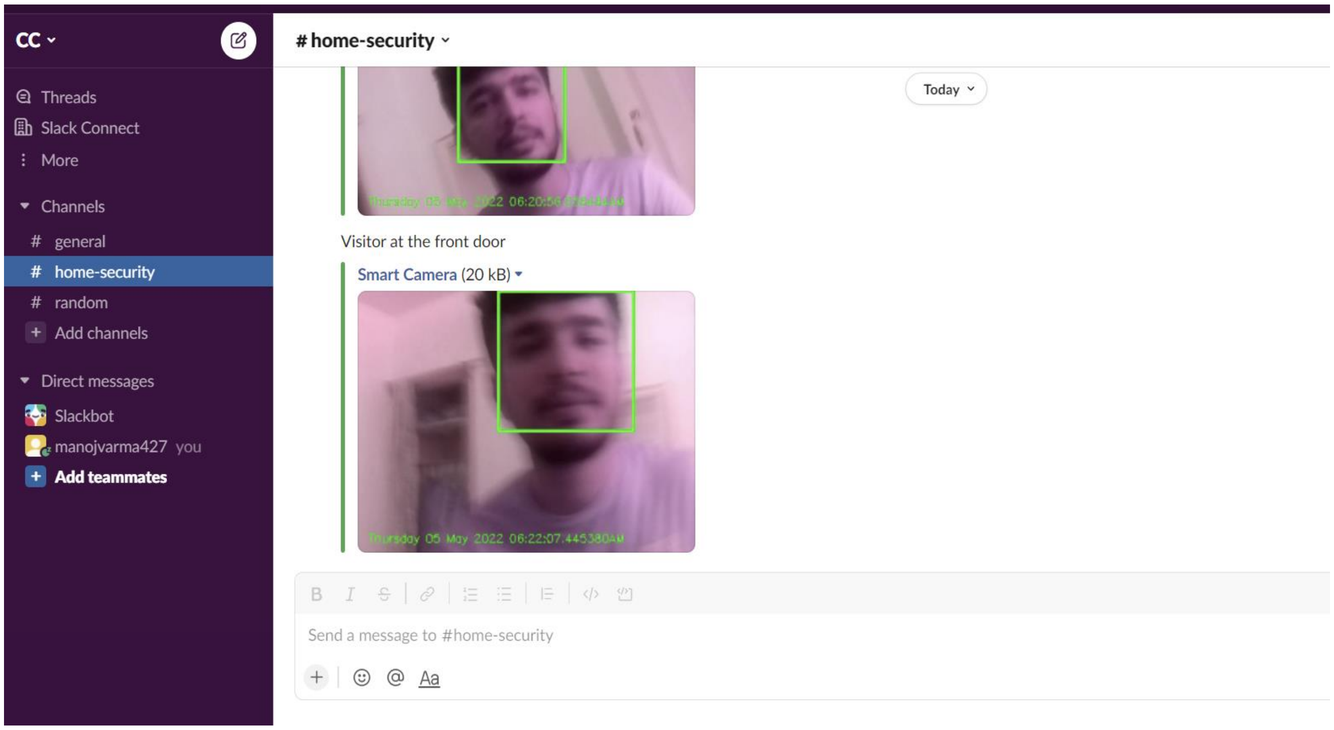
Step 4: Pick a channel that the app will post to, then click Authorize.

Step 5: Now under the Webhook URLs in Workspace section, we can find an URL https://hooks.slack.com/services/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

Step 6: Now the slack is integrated with the Raspberry PI by configuring this URL in config.json

When the PI camera captures the images, it will be sent to slack. In slack the owner of the house find an URL “SMART CAMERA” which redirects him to a web application where he can register the details of that person. These details will be useful when the person re appears in front of the camera. When the person re-appears he will be greeted with his name through the speaker connected to the application.

Snapshots captured during the demo:

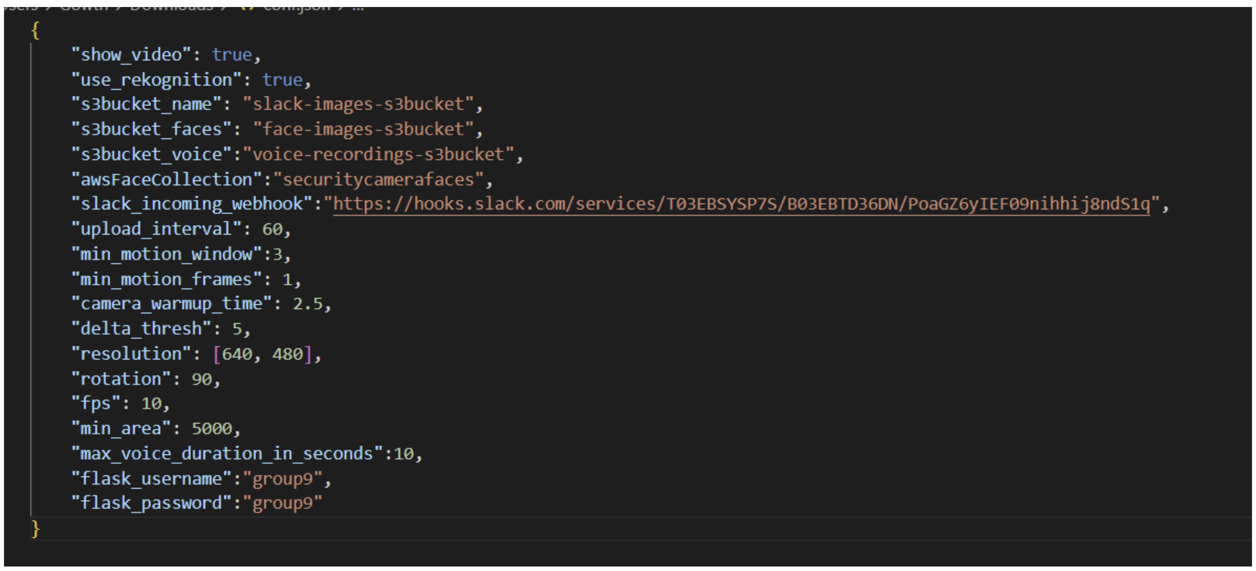


Visitor at the front door

Smart Camera (20 kB)

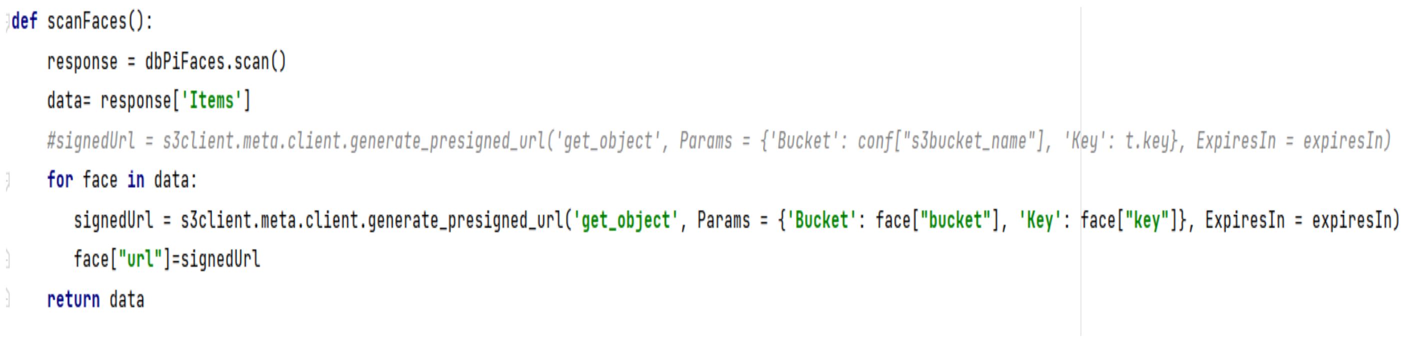
Thursday 05 May 2022 06:22:07.445380AM

The AWS services and slack are integrated to the application in config.json file. Below is the code snippet of config.json.



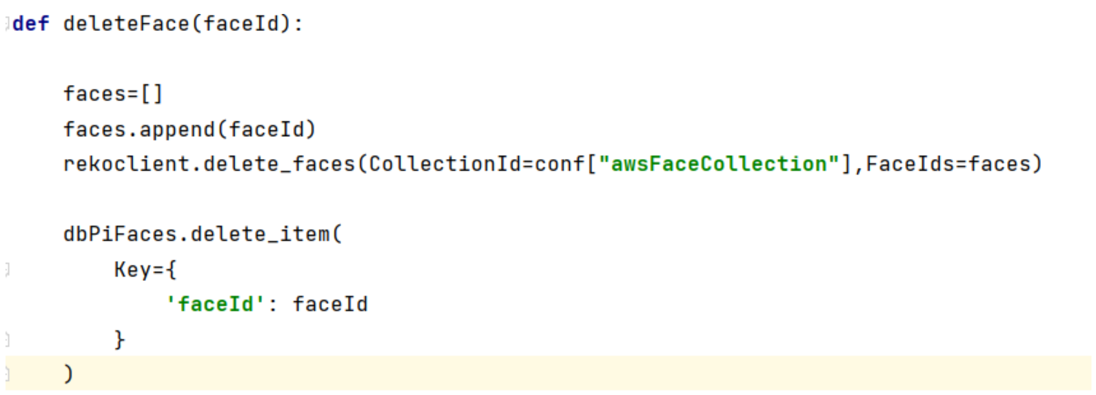
```
{
    "show_video": true,
    "use_rekognition": true,
    "s3bucket_name": "slack-images-s3bucket",
    "s3bucket_faces": "face-images-s3bucket",
    "s3bucket_voice":"voice-recordings-s3bucket",
    "awsFaceCollection":"securitycamerafaces",
    "slack_incoming_webhook":"https://hooks.slack.com/services/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",
    "upload_interval": 60,
    "min_motion_window":3,
    "min_motion_frames": 1,
    "camera_warmup_time": 2.5,
    "delta_thresh": 5,
    "resolution": [640, 480],
    "rotation": 90,
    "fps": 10,
    "min_area": 5000,
    "max_voice_duration_in_seconds":10,
    "flask_username":"group9",
    "flask_password":"group9"
}
```

Code snippet for storing scanned images into the bucket:



```
def scanFaces():
    response = dbPiFaces.scan()
    data= response['Items']
    #signedUrl = s3client.meta.client.generate_presigned_url('get_object', Params = {'Bucket': conf["s3bucket_name"], 'Key': t.key}, ExpiresIn = expiresIn)
    for face in data:
        signedUrl = s3client.meta.client.generate_presigned_url('get_object', Params = {'Bucket': face["bucket"], 'Key': face["key"]}, ExpiresIn = expiresIn)
        face["url"]=signedUrl
    return data
```

Code snippet for deleting face Id’s from AWS:

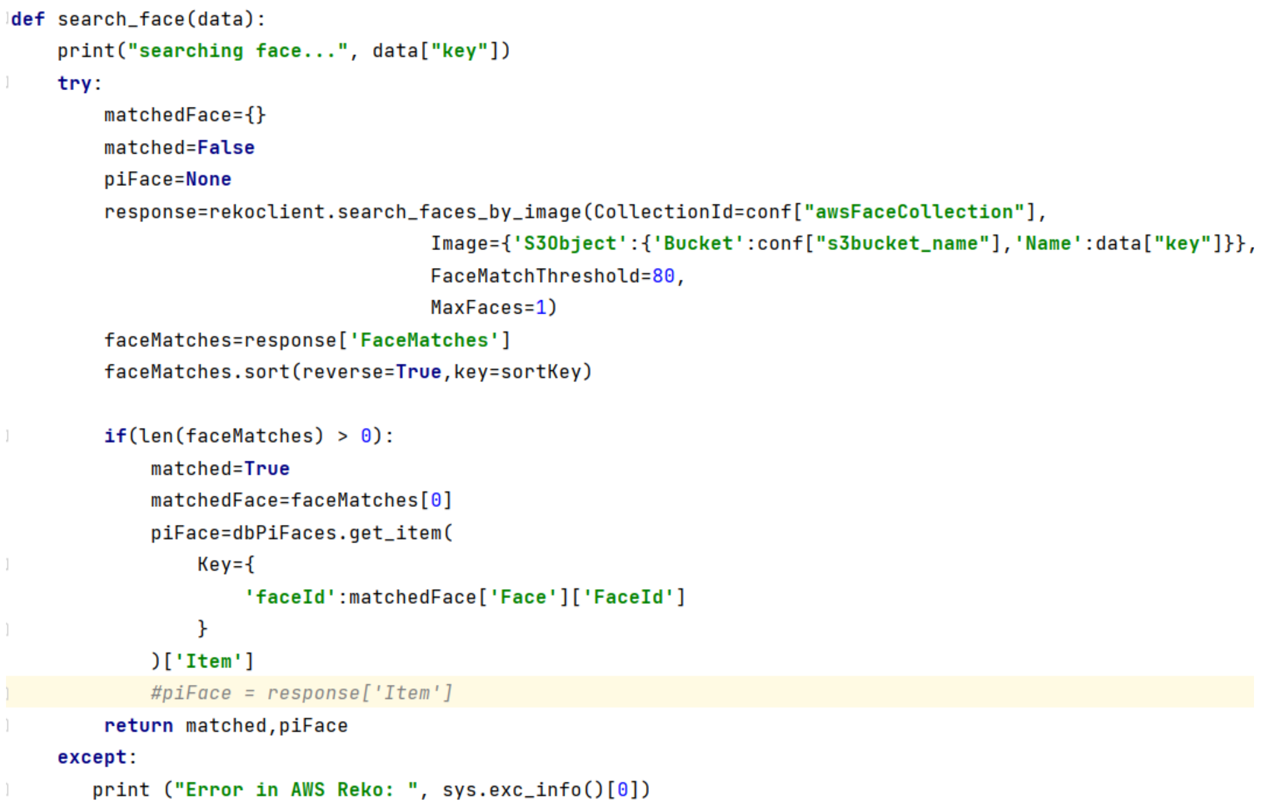


```
def deleteFace(faceId):

    faces=[]
    faces.append(faceId)
    rekoclient.delete_faces(CollectionId=conf["awsFaceCollection"],FaceIds=faces)

    dbPiFaces.delete_item(
        Key={
            'faceId': faceId
        }
    )
```

Code Snippet for searching the data over AWS:



```
def search_face(data):
    print("searching face...", data["key"])
    try:
        matchedFace={}
        matched=False
        piFace=None
        response=rekoclient.search_faces_by_image(CollectionId=conf["awsFaceCollection"],
                                    Image={'S3Object':{'Bucket':conf["s3bucket_name"],'Name':data["key"]}},
                                    FaceMatchThreshold=80,
                                    MaxFaces=1)
        faceMatches=response['FaceMatches']
        faceMatches.sort(reverse=True,key=sortKey)

        if(len(faceMatches) > 0):
            matched=True
            matchedFace=faceMatches[0]
            piFace=dbPiFaces.get_item(
                Key={
                    'faceId':matchedFace['Face']['FaceId']
                }
            )['Item']
            #piFace = response['Item']
        return matched,piFace
    except:
       print ("Error in AWS Reko: ", sys.exc_info()[0])
```

Conclusion:

Finally, we successfully implemented security camera with voice capability using Raspberry Pi and AWS cloud services for accessing across the globe. Future scope will be implementing different objects detection for monitoring pets at home and creating a healthy and safe environment.

References:

- https://thepihut.com/blogs/raspberry-pi-tutorials/16021420-how-to-install-use-the-raspberry-pi-camera
- https://www.pyimagesearch.com/2017/09/04/raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/
- https://get.slack.help/hc/en-us/articles/115005265063-Incoming-WebHooks-for-Slack
- https://docs.aws.amazon.com/AmazonS3/latest/userguide/create-bucket-overview.html