import json

import boto3

import time

import pandas as pd

from trp import Document

from sys import argv

from PyPDF2 import PdfFileWriter, PdfFileReader

import os

from io import StringIO

s3\_client = boto3.resource('s3')

def insertS3File(output,JobId,filename):

print("Writing File to S3")

s3\_buffer = StringIO()

print('s3 buffer')

print(output.shape[0])

output.to\_csv(s3\_buffer, index=False)

print('s3 buffer completed')

paths3 = "textractresults/{}/{}.csv".format(JobId, filename)

print(paths3)

response = s3\_client.Object('rpapdfbucket',paths3).put(Body=s3\_buffer.getvalue(),ACL='bucket-owner-full-control',ServerSideEncryption='AES256')

print("Finished writing file to S3")

def startJob(s3BucketName, objectName):

response = None

client = boto3.client('textract')

print(s3BucketName)

print(objectName)

response = client.start\_document\_analysis(

DocumentLocation={

'S3Object': {

'Bucket': s3BucketName,

'Name': objectName

}},FeatureTypes=["FORMS"])

return response

def isJobComplete(jobId):

time.sleep(5)

client = boto3.client('textract')

response = client.get\_document\_analysis(JobId=jobId)

status = response["JobStatus"]

print("Job status: {}".format(status))

while (status == "IN\_PROGRESS"):

time.sleep(30)

response = client.get\_document\_analysis(JobId=jobId)

status = response["JobStatus"]

print("Job status: {}".format(status))

return status

def ParseResponse(doc,JobId,filename):

values = []

i = 0

for page in doc.pages:

i += 1

for field in page.form.fields:

if field.value is None:

continue

else:

temp = [i, field.key.text, field.value.text]

values.append(temp)

df = pd.DataFrame(data=values, columns=['Page Number', 'Key', 'Value'])

csv\_buffer = StringIO()

df.to\_csv(csv\_buffer, index=False)

insertS3File(df,JobId,filename)

def getJobResults(jobId):

pages = []

time.sleep(5)

client = boto3.client('textract')

response = client.get\_document\_analysis(JobId=jobId)

doc = Document(response)

pages.append(response)

print("Result set page recieved : {}".format(len(pages)))

nextToken = None

if ('NextToken' in response):

nextToken = response['NextToken']

while (nextToken):

time.sleep(5)

response = client.get\_document\_analysis(JobId=jobId, NextToken=nextToken)

pages.append(response)

print("Resultset page recieved: {}".format(len(pages)))

nextToken = None

if ('NextToken' in response):

nextToken = response['NextToken']

return doc

def lambda\_handler(event, context):

# TODO implement

print(event)

s3BucketName = event["BucketName"]

path = "s3://"

filename = event["FileName"]

response = startJob(s3BucketName, filename)

JobId = response['JobId']

print("JobId is : ", JobId)

print("Started job with id: {}".format(JobId))

if isJobComplete(JobId):

response = getJobResults(JobId)

ParseResponse(response,JobId,filename)

print("Script execution complete")

return {

'statusCode': 200,

'body': json.dumps('Textraction Completed')

}