12/19/2021

Project Documentation

Proxima Centauri- SkipQ

Waheed Ahmad
SKIPQ

Table of Contents

1. Introduction

1.1. AWS:

Amazon Web Services (AWS) is the world's most comprehensive and broadly adopted cloud platform, offering over 200 fully featured services from data centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.

1.2. Cloud computing

Cloud computing is the on-demand delivery of IT resources over the Internet with pay-as-you-go pricing. Instead of buying, owning, and maintaining physical data centers and servers, you can access technology services, such as computing power, storage, and databases, on an as-needed basis from a cloud provider like Amazon Web Services (AWS).

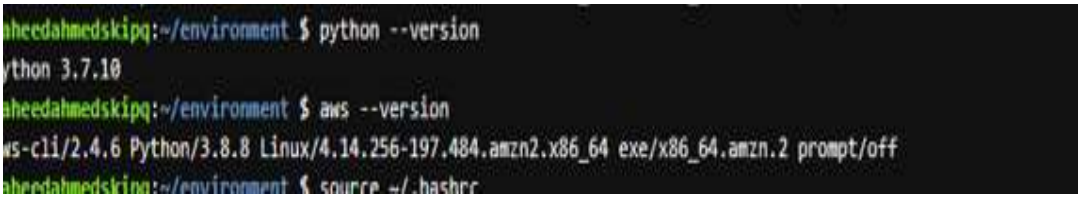
Sprint1

1. Day_1

1.1. Setting up environment:

The environment in AWS console was created in cloud9

The specifications of environment were RAM=1 GB and CPU= 1, all the required packages were installed, and necessary updates were made.



Issues faced: In updating AWS version and python version, it was solved by changing *'alias python=python3'* in the code.

1.2. Hello Lambda!

After setting up the environment the next task was writing a lambda function for hello world , it was simple task the program was tested and proper output

PcwaheedprojectSt
constants.py
hwlambda.py
webhealthwaheed.py

Execution results

Status: Su

Test Event Name
hello

Response
"Hello waheed ahmad!"

Function Logs
```
START RequestId: 4f4e27a7-811a-4051-aa5c-8f80b4abb8c3 Version: $LATEST
END RequestId: 4f4e27a7-811a-4051-aa5c-8f80b4abb8c3
REPORT RequestId: 4f4e27a7-811a-4051-aa5c-8f80b4abb8c3  Duration: 1.07 ms   Billed Duration: 2 ms   M
```

Request ID

webhealthmonitor.py | pcwaheedproject_st | cloudmatricdata.py | constants.py | dynamodb_lambda.p | hwlambda.py

```
AWS: Add Debug Configuration | AWS: Edit Debug Configuration
def lambda_handler(event, context):
    return 'Hello {} {}!'.format(event['first_name'], event['last_name'])
```

Day_2:

1.3. Webhealth_lambda:

This function is programmed to check whether a website is performing okay or not, in terms of availability and latency ,we defined two functions for availability and latency and deployed it.

```
AWS: Add Debug Configuration | AWS: Edit Debug Configuration
def lambda_handler(events,context):
    values= dict()
    avail= get_availability()
    latency= get_latency()
    values.update({"Availability": avail,"Latency":latency})
    return values

AWS: Add Debug Configuration | AWS: Edit Debug Configuration
def get_availability():
    http=urllib3.PoolManager()
    response=http.request("GET",URL_to_Monitor)
    if response.status==200:
        return 1
    else:
```

Issues faced: while writing in stack , there was an issue with lambda handler, which was rectified.

1.4. Webhealth_Monitor/ periodic lambda:

This is an extension of web _health ,and it creates graph of availability and latency metrics for monitoring the status of a website , metrics were created using this lambda function and can be seen in cloudwatch , these latency and availability values can further be used to raise alarms.

```
URL_to_Monitor='www.bbc.com'

AWS: Add Debug Configuration | AWS: Edit Debug Configuration
def lambda_handler(events,context):
    values = dict()
    cw= cloudmetric1();


    #dynamo_table=self.create
    avail= get_availability()   #now we are putting matrices to cloudwatch
    dimensions=[
        {'Name': 'URL' , 'Value': constants.URL_to_Monitor },
        {'Name': 'Region' , 'Value': "DUB"}

        ]
    cw.put_data(constants.URL_MONITOR_NAMESPACE ,constants.URL_MONIROR_NAME_AVAILABILITY , dimensions,avail)
```

Issues faced: Importing from constants.py file caused an error, which was rectified by changing the names of variables

1.5. Cloudmetric data:

This function is created to load the availability and latency metric graphs .

webhealthmonito × cloudmatricdata. ×

Execution results

Test Event Name

```
heelo
```

Response

```
{
  "Availability": 1,
  "Latency": 0.211912
}
```

Function Logs

```
START RequestId: c4840c3a-4edc-403b-b133-6def2c00db6
END RequestId: c4840c3a-4edc-403b-b133-6def2c00db6f
```

Day_3:

1.6. Alarms

Now that the metrics for the availability and latency are defined, we can set threshold to them and raise an alarm when a certain threshold is reached

ALARM: "PcwaheedprojectStack-LatencyAlarm5394FC57-PVKRNHVNSDIM" in US East (Ohio) External Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com> 6:49 PM (6 minutes ago)
to me

You are receiving this email because your Amazon CloudWatch Alarm "PcwaheedprojectStack-LatencyAlarm5394FC57-PVKRNHVNSDIM" in the U region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [0.21814875 (19/12/21 13:48:00)] was greater thar (0.2) (minimum 1 datapoint for OK -> ALARM transition)." at "Sunday 19 December, 2021 13:49:25 UTC".

View this alarm in the AWS Management Console:
https://us-east-2.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-east-2#alarmsV2:alarm/PcwaheedprojectStack-LatencyAlarm5394FC PVKRNHVNSDIM

Alarm Details:

1.7. SNS

After setting up alarms to be raised when a certain threshold is reached, we need to add subscriptions to it , to be notified in case of an alarm is breached

Lambdafunction subscription:

```
#####module code for sending sns notifications#################################################
topic =sns.Topic(self, "webhealth")
topic.add_subscription(subscriptions_.EmailSubscription('waheed.ahmad.s@skipq.org'))
topic.add_subscription(subscription_.LambdaSubscription(fn=db_lambda))
# import sys
```

Day_4:

1.8. Working with DynamoDB

The DynamoDB table is created to store the alarm values ,

```
table = dynamodb.create_table(
    TableName='Movies',
    KeySchema=[
        {
            'AttributeName': 'alarmID',
            'KeyType': 'HASH'
        },
        {
            'AttributeName': 'alarm',
            'KeyType': 'RANGE'
        }
        ],
    AttributeDefinitions=[
        {
            'AttributeName': 'alarmID',
            'AttributeType': 'N'
        },
        {
            'AttributeName': 'title'
```