

CS 551: Assignment 6

Ishaan Roychowdhury

Function handler code for part 4:

```
import json
import urllib.parse
import boto3
s3 = boto3.client('s3')
def fibo_homework6(n):
    fib_table_ishaanHW = [0] * (n + 1)
    fib_table_ishaanHW[0] = 0
    if n > 0:
        fib_table_ishaanHW[1] = 1
    for i in range(2, n + 1):
        fib_table_ishaanHW[i] = fib_table_ishaanHW[i-1] + fib_table_ishaanHW[i-2]
    return fib_table_ishaanHW[n]

def lambda_handler(event, context):
    if 'Records' not in event or not event['Records']:
        print("No records")
        return "No records"
    fetch_Rec = event['Records'][0]
    if 's3' not in fetch_Rec or 'bucket' not in fetch_Rec['s3'] or 'name' not in fetch_Rec['s3']['bucket']:
        print("Invalid S3 scene bro")
        return "Invalid S3 scene"
    bucket = fetch_Rec['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(fetch_Rec['s3']['object']['key'], encoding='utf-8')
    get_response = s3.get_object(Bucket=bucket, Key=key)
    if 'Body' not in get_response:
        print(f"Missing 'Body' in S3 object for {key}")
        return f"Missing data in S3 object {key}"
    fetch_json = json.loads(get_response['Body'].read().decode('utf-8'))
    if 'n' not in fetch_json:
        print("JSON does not contain 'n'")
        return "Invalid JSON content"
    n = int(fetch_json['n'])
    retVal_for_Fib = fibo_homework6(n)
    final_json_to_upload_to_S3 = {'n': n, 'fibonacci_result': retVal_for_Fib}
    key_for_homework = f"fibonacci_result_{n}.json"
    s3.put_object(Bucket=bucket, Key=key_for_homework,
Body=json.dumps(final_json_to_upload_to_S3))
    print(f"I have computed fibonacci for this homework and the results for n={n} is: {retVal_for_Fib}")
    return f"I have computed the results of fibonacci in the assignment and those results for n={n} have been uploaded to S3: {key_for_homework}"
```

This was my code for function handler in part 4:

```
app.py > lambda_handler
1 import json
2 import urllib.parse
3 import boto3
4 s3 = boto3.client('s3')
5 def fibo_homework6(n):
6     fib_table_ishaanHW = [0] * (n + 1)
7     fib_table_ishaanHW[0] = 0
8     if n > 0:
9         fib_table_ishaanHW[1] = 1
10    for i in range(2, n + 1):
11        fib_table_ishaanHW[i] = fib_table_ishaanHW[i-1] + fib_table_ishaanHW[i-2]
12    return fib_table_ishaanHW[n]
13
14 def lambda_handler(event, context):
15     if 'Records' not in event or not event['Records']:
16         print("No records")
17         return "No records"
18     fetch_Rec = event['Records'][0]
19     if 's3' not in fetch_Rec or 'bucket' not in fetch_Rec['s3'] or 'name' not in fetch_Rec['s3']['bucket']:
20         print("Invalid S3 scene bro")
21         return "Invalid S3 scene"
22     bucket = fetch_Rec['s3']['bucket']['name']
23     key = urllib.parse.unquote_plus(fetch_Rec['s3']['object']['key'], encoding='utf-8')
24     get_response = s3.get_object(Bucket=bucket, Key=key)
25     if 'Body' not in get_response:
26         print(f"Missing 'Body' in S3 object for {key}")
27         return f"Missing data in S3 object {key}"
28     fetch_json = json.loads(get_response['Body'].read().decode('utf-8'))
29     if 'n' not in fetch_json:
30         print("JSON does not contain 'n'")
31         return "Invalid JSON content"
32     n = int(fetch_json['n'])
33     retVal_for_Fib = fibo_homework6(n)
34     final_json_to_upload_to_S3 = {'n': n, 'fibonacci_result': retVal_for_Fib}
35     key_for_homework = f"fibonacci_result_{n}.json"
36     s3.put_object(Bucket=bucket, Key=key_for_homework, Body=json.dumps(final_json_to_upload_to_S3))
37     print(f"I have computed fibonacci for this homework and the results for n={n} is: {retVal_for_Fib}")
38     return f"I have computed the results of fibonacci in the assignment and those results for n={n} have been uploaded to S3: {key_for_homework}"
39
```

I made a policy which gave access to GetObject and PutObject for S3:

The screenshot shows the AWS IAM console interface. On the left is a navigation menu with sections like 'Identity and Access Management (IAM)', 'Access management', 'Access reports', and 'Related consoles'. The main area displays the 'ishaan_lambda' policy details. A table shows the policy's type as 'Customer managed', its creation and edited times as 'November 13, 2023, 20:16 (UTC-05:00)', and its ARN as 'arn:aws:iam:044112898403:policy/ishaan_lambda'. Below this, the 'Permissions defined in this policy' section shows a JSON document with two statements. The first statement allows logging actions on the 'arn:aws:logs:*:*:*' resource, and the second statement allows S3 GetObject and PutObject actions on the 'arn:aws:s3:::/*/*' resource.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "logs:PutLogEvents",
8         "logs:CreateLogGroup",
9         "logs:CreateLogStream"
10      ],
11      "Resource": "arn:aws:logs:*:*:*"
12    },
13    {
14      "Effect": "Allow",
15      "Action": [
16        "s3:GetObject",
17        "s3:PutObject"
18      ],
19      "Resource": "arn:aws:s3:::/*/*"
20    }
21  ]
22 }
```

I made a role that used this policy:

The screenshot shows the AWS IAM console interface. On the left is a navigation sidebar with sections like 'Identity and Access Management (IAM)', 'Access management', 'Access reports', and 'Related consoles'. The main content area shows the details for the role 'ishaan_lambda_role'. It includes a 'Summary' section with creation date, last activity, ARN, and maximum session duration. Below this is a 'Permissions' section with tabs for 'Permissions policies', 'Trust relationships', 'Tags', 'Access Advisor', and 'Revoke sessions'. The 'Permissions policies' tab is active, showing a table of attached policies. One policy, 'ishaan_lambda', is listed with a 'Customer managed' type and is attached to 1 entity. There are also buttons for 'Simulate', 'Remove', and 'Add permissions'.

Policy name	Type	Attached entities
ishaan_lambda	Customer managed	1

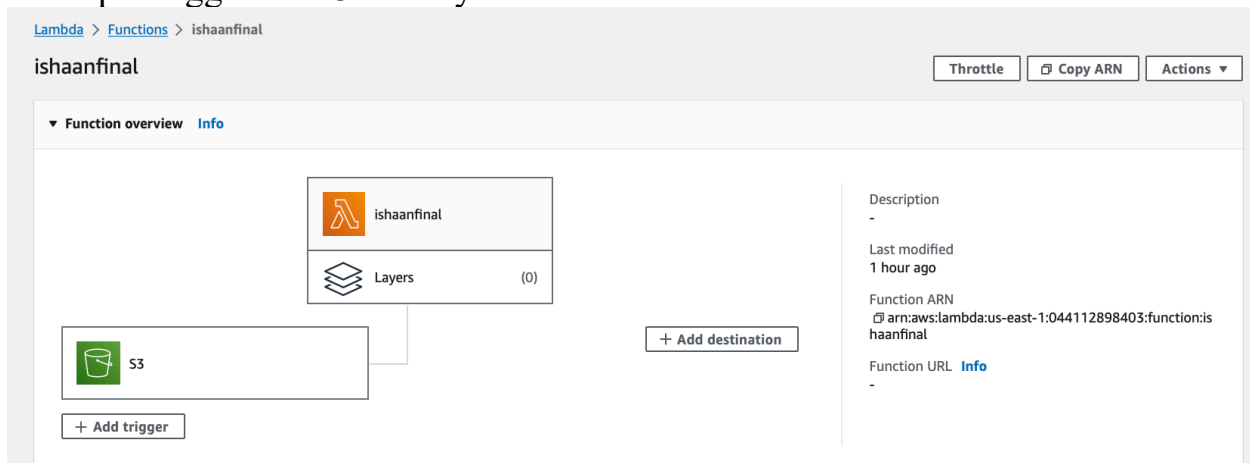
I made a S3 bucket for uploading json files, both from my end and from lambda's end. I used the same bucket to write the JSON result:

The screenshot shows the Amazon S3 console interface for the bucket 'ishaanfinalbuck'. It has tabs for 'Objects', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. The 'Objects' tab is active, displaying a list of objects. Above the list are buttons for 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and 'Upload'. The objects table has columns for Name, Type, Last modified, Size, and Storage class. Two objects are listed: 'fibonacci_result_10.json' and 'test.json'.

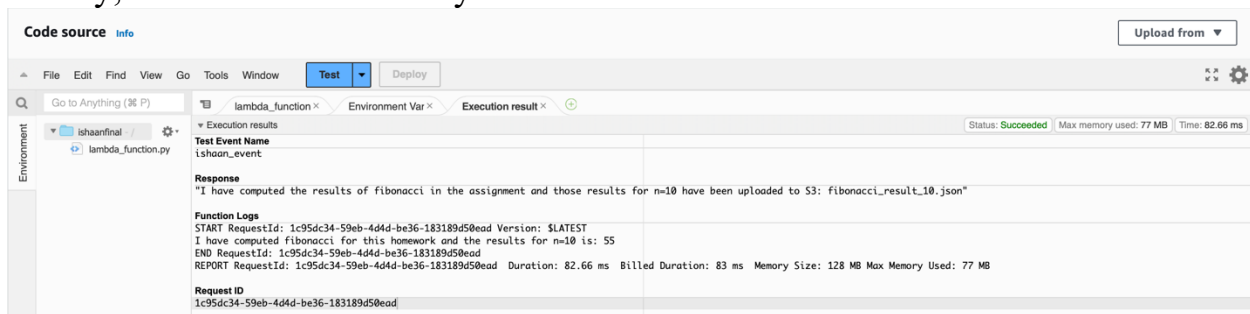
Name	Type	Last modified	Size	Storage class
fibonacci_result_10.json	json	November 13, 2023, 21:07:40 (UTC-05:00)	33.0 B	Standard
test.json	json	November 13, 2023, 20:28:57 (UTC-05:00)	16.0 B	Standard

(In this I uploaded, test.json and once I ran the lambda function, fibonacci_result_10.json got added to the same bucket with the result)

I set up a trigger for S3 and my bucket in Lambda:



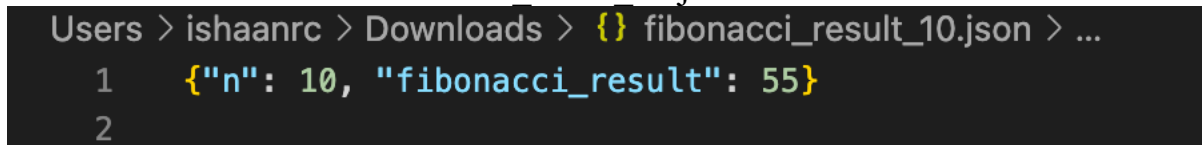
Finally, I ran a test to see if my code works and it does!!:



test.json that I uploaded:



When I downloaded the fibonacci_result_10.json file that was created:



This successfully shows me the 10th number in the Fibonacci Series which is 55. Hence this works!! 😊