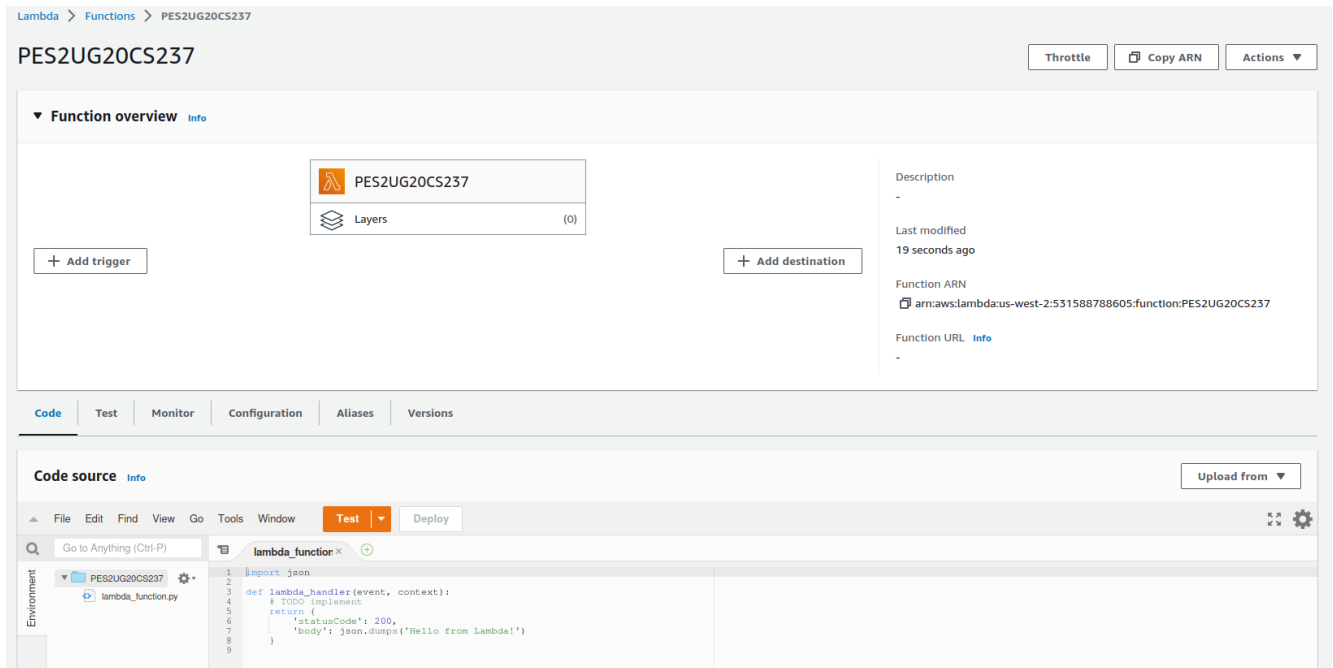


Details :

- Name : P K Navin Shrinivas
- SRN : PES2UG20CS237
- Section : D

Creating lambda function



Note : Note the name of lambda function, it is the SRN.

API gateway, the trigger

Note : the gateways is a normal HTTP gateway with cors enabled.

API Gateway

+ Add trigger

+ Add destination

1 minute ago

Function ARN
arn:aws:lambda:us-west-2:531588788605:function:PES2UG20CS237

Function URL [Info](#)

-

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

Monitoring and operations tools

Concurrency

Asynchronous invocation

Code signing

Database proxies

File systems


State machines

Triggers (1) [Info](#)

< 1 >

☐

Trigger



API Gateway: PES2UG20CS237-API

arn:aws:execute-api:us-west-2:531588788605:giklzxsdw5/"/PES2UG20CS237

API endpoint: <https://giklzxsdw5.execute-api.us-west-2.amazonaws.com/default/PES2UG20CS237>

▼ Details

☐

API type: HTTP

Authorization: NONE

CORS: Yes

Detailed metrics enabled: No

Method: ANY

Resource path: /PES2UG20CS237

Service principal: apigateway.amazonaws.com

Stage: default

Statement ID: lambda-99ddea17-8e4a-474c-b87f-86cb253dc639

The function

Note : this image doesn't cover the full code, hence I have pasted the code as well.

Successfully updated the function PES2UG20CS237.

File Edit Find View Go Tools Window

Test

Deploy

Go to Anything (Ctrl-F)

lambda_function.py

```
1 import json
2 import base64
3 import csv
4 def lambda_handler(event, context):
5     req_json_obj = event
6     if req_json_obj["httpMethod"] == "GET" :
7         key = req_json_obj["queryStringParameters"]["key"]
8         return{
9             "statusCode" : 200,
10            "body" : json.dumps("PES2UG20CS237:"+key)
11        }
12     elif req_json_obj["httpMethod"] == "POST":
13         sample_string_bytes = str(base64.b64decode(req_json_obj["body"])).split('\n')[4:-3]
14         csv_string = ""
15         for i in sample_string_bytes:
16             # print(i)
17             csv_string+=i+"\n"
18         csv_obj = csv.reader(csv_string.split('\n'))
19         res = dict()
20         row_num = 0
21         col_names = []
22         for i in csv_obj:
23             if row_num == 0:
24                 # Col names :
25                 for j in i :
26                     res[j] = 0
27                     col_names = i
28                     row_num+=1
29             else:
30                 for j in range(0,len(i)):
31                     res[col_names[j]] = res[col_names[j]]+int(i[j])
32                     row_num+=1
33         for i in res :
34             res[i] = res[i]/(row_num-1)
35         return{
36             "statusCode" : 200,
37             "body" : json.dumps(res)
38         }
39     else:
40         return{
41             "statusCode" : 400,
42             "body" : json.dumps("Invalid request")
43         }
```

2020 Python Spaces: 4

```
import json
import base64
import csv
def lambda_handler(event, context):
    req_json_obj = event
    if req_json_obj["httpMethod"] == "GET" :
        key = req_json_obj["queryStringParameters"]["key"]
        return{
            "statusCode" : 200,
            "body" : json.dumps("PES2UG20CS237:"+key)
```

```

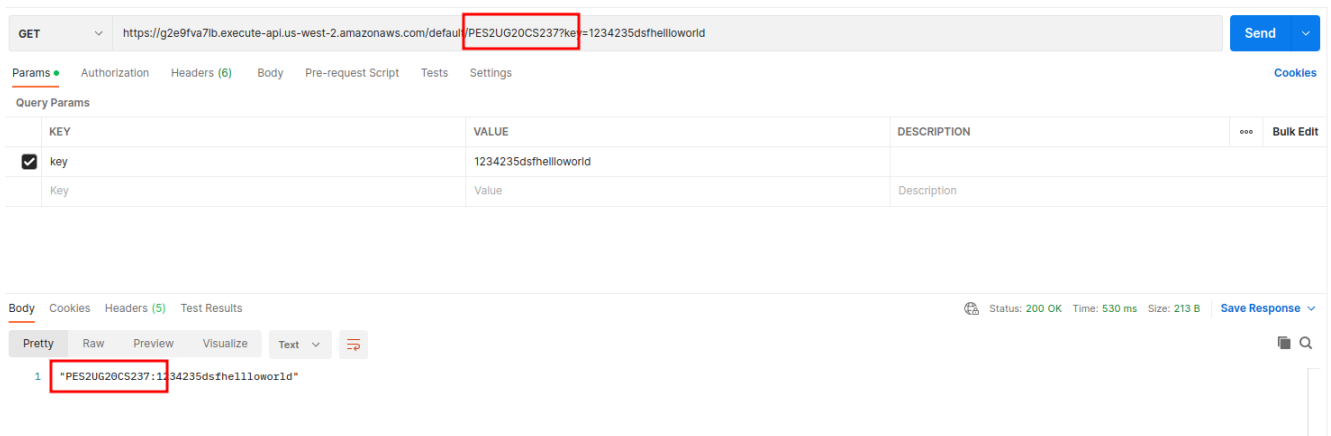
    }
    elif req_json_obj["httpMethod"] == "POST":
        sample_string_bytes =
str(base64.b64decode(req_json_obj["body"])).split('\n')[4:-3]
        csv_string = ""
        for i in sample_string_bytes:
            # print(i)
            csv_string+=i+"\n"
        csv_obj = csv.reader(csv_string.split('\n'))
        res = dict()
        row_num = 0
        col_names = []
        for i in csv_obj:
            if row_num == 0:
                # Col names :
                for j in i :
                    res[j] = 0
                col_names = i
                row_num+=1
            else:
                for j in range(0,len(i)):
                    res[col_names[j]] = res[col_names[j]]+int(i[j])
                row_num+=1
        for i in res :
            res[i] = res[i]/(row_num-1)

        return{
            "statusCode" : 200,
            "body" : json.dumps(res)
        }
    else:
        return{
            "statusCode" : 200,
            "body" : json.dumps("Only GET and POST are supported")
        }

# return {
#     'statusCode': 200,
#     'body': json.dumps(event,indent=4)
# }

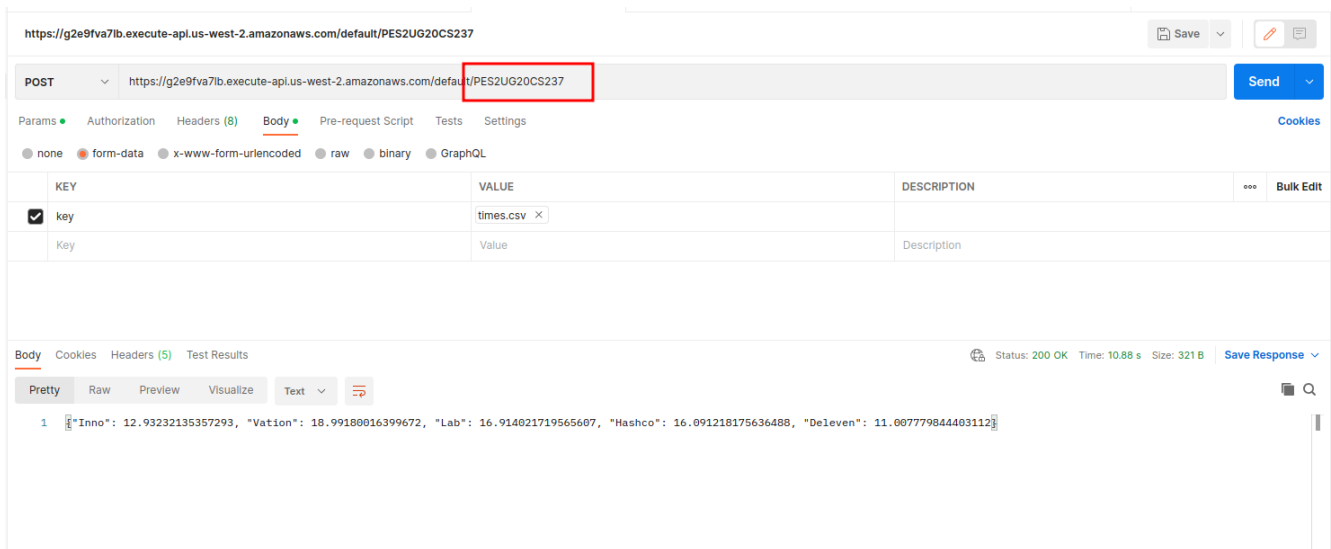
```

GET request in postman



POST request in postman

Note : the files is being loaded as a form-data in the postman body



Methods other than POST and GET

