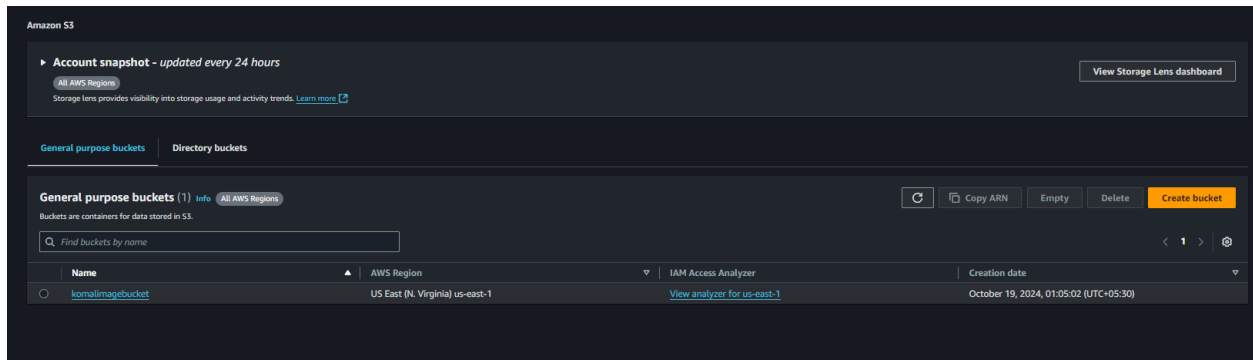


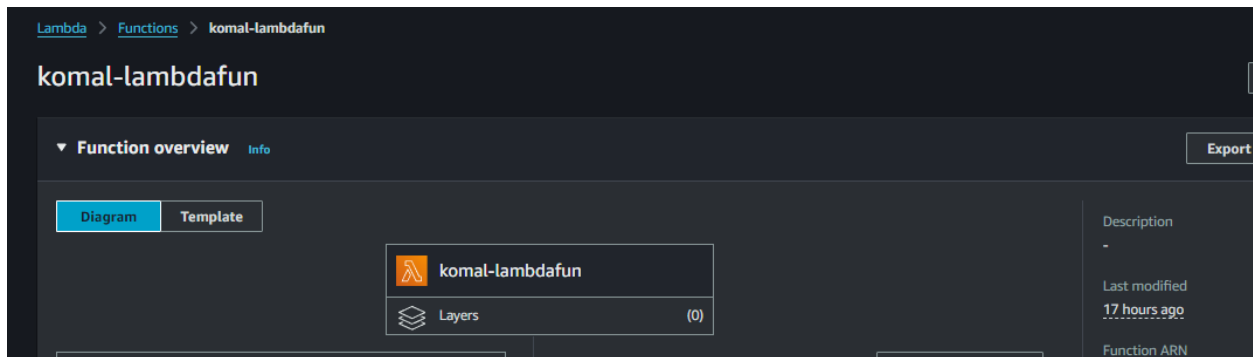
CREATION OF S3 BUCKET:

CREATE AN S3 BUCKET OF IN AWS

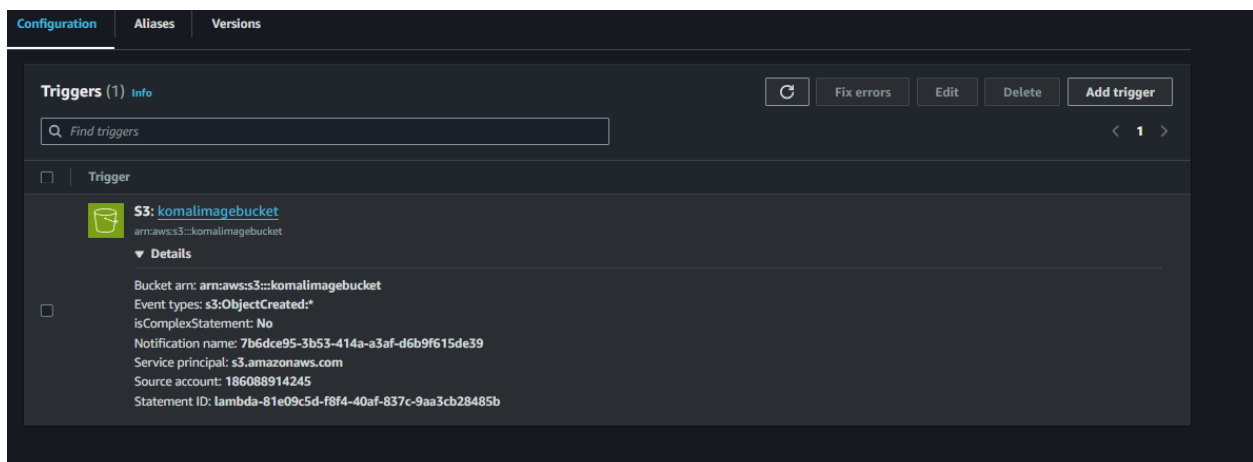


CREATION OF LAMBDA FUNCTION

CREATE A NEW LAMBDA FUNCTION AND CONFIGURE IT.



ADD AN S3 TRIGGER TO GET UPDATED WHEN A NEW IMAGE IS UPLOADED IN THE S3 BUCKET



UPDATE THE PYTHON SCRIPT IN THE CODE SOURCE TO WRITE A CODE THAT PRINTS “AN IMAGE IS UPLOADED” WHEN THE LAMBDA FUNCTION IS TRIGGERED BY S3.

Write the below code in the script.

```
import json

import logging

# Set up logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

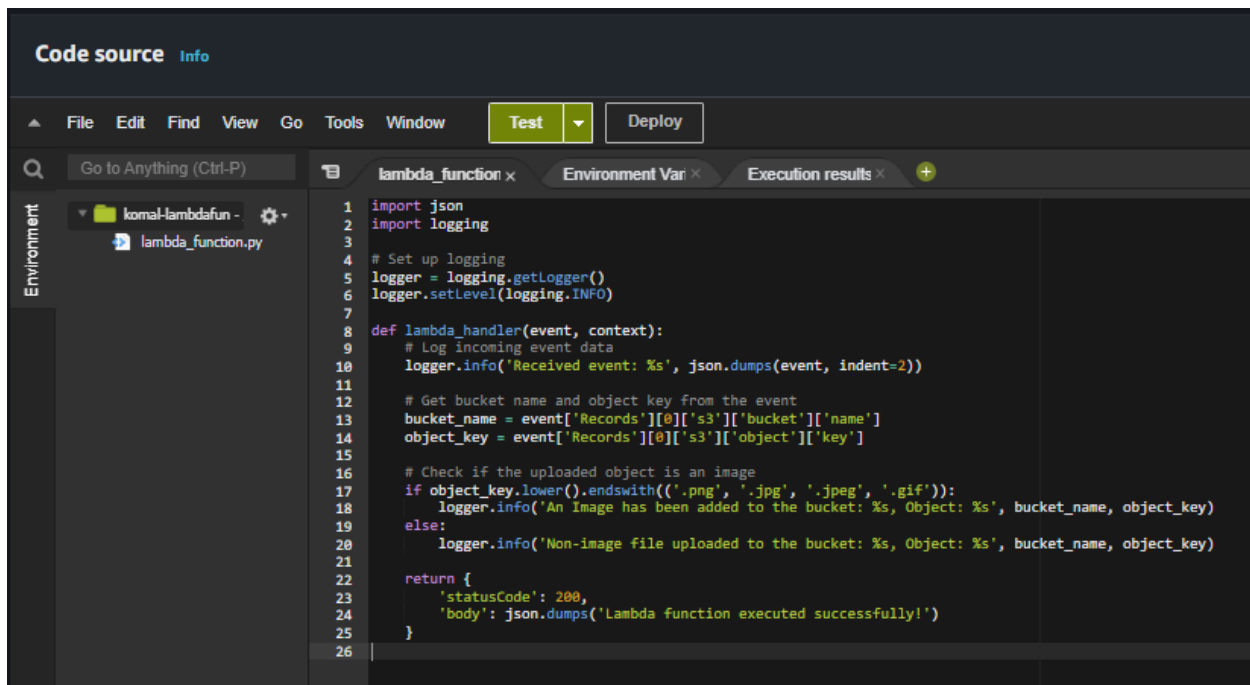
def lambda_handler(event, context):

    # Log incoming event data
    logger.info('Received event: %s', json.dumps(event, indent=2))

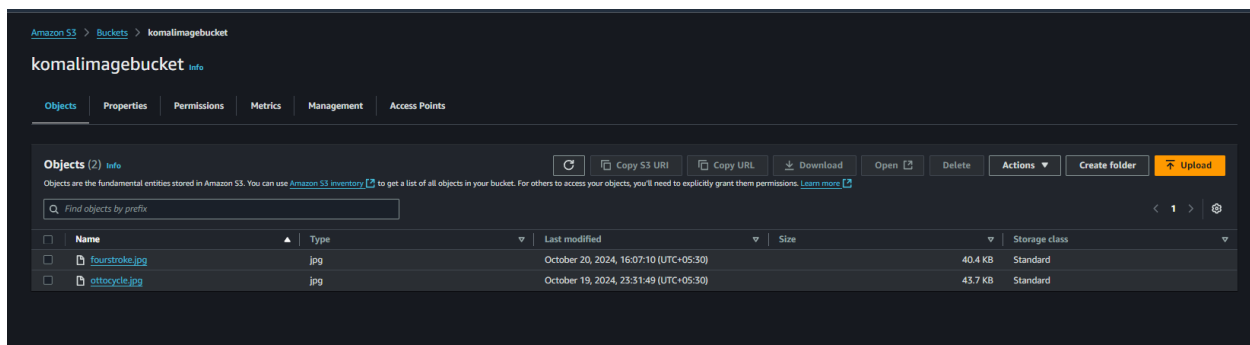
    # Get bucket name and object key from the event
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    object_key = event['Records'][0]['s3']['object']['key']

    # Check if the uploaded object is an image
    if object_key.lower().endswith(('.png', '.jpg', '.jpeg', '.gif')):
        logger.info('An Image has been added to the bucket: %s, Object: %s', bucket_name, object_key)
    else:
        logger.info('Non-image file uploaded to the bucket: %s, Object: %s', bucket_name, object_key)

    return {
        'statusCode': 200,
        'body': json.dumps('Lambda function executed successfully!')
    }
```

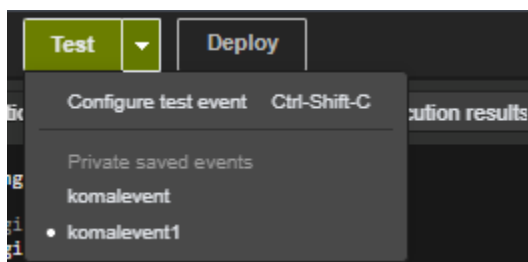


1)UPLOAD AN IMAGE IN THE S3 BUCKET:



NOW GO BACK TO LAMBDA FUNCTION AND

DEPLOY THE CODE BY CLICKING ON THE DEPLOY OPTION AND TEST IT.



-USE THE SUITABLE EVENT FOR YOUR CODE.

THE EVENT SHOULD INCLUDE BELOW SCRIPT.

```
{
  "Records": [
    {
      "eventVersion": "2.1",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "2024-10-19T17:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/456EXAMPLE="
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "example-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::example-bucket"
        }
      }
    }
  ]
}
```

```

    },
    "object": {
      "key": "test-image.jpg",
      "size": 1024,
      "eTag": "0123456789abcdef0123456789abcdef",
      "sequencer": "0A1B2C3D4E5F678901"
    }
  }
}
}
}}

```

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save changes.

Test event action

☐ Create new event
 ☒ Edit saved event

Event name

komalevent1

Delete

Event JSON

Format JSON

```

1 * 0
2 * "Records": [
3 * {
4 *   "eventVersion": "2.1",
5 *   "eventSource": "aws:s3",
6 *   "awsRegion": "us-east-1",
7 *   "eventTime": "2024-10-19T17:00:00.000Z",
8 *   "eventName": "ObjectCreated:Put",
9 *   "userIdentity": {
10 *     "principalId": "EXAMPLE"
11 *   },
12 *   "requestParameters": {
13 *     "sourceIPAddress": "127.0.0.1"
14 *   },
15 *   "responseElements": {
16 *     "x-amz-request-id": "EXAMPLE123456789",
17 *     "x-amz-id-2": "EXAMPLE123/456EXAMPLE="
18 *   },
19 *   "s3": {
20 *     "s3SchemaVersion": "1.0",
21 *     "configurationId": "testConfigRule",
22 *     "bucket": {
23 *       "name": "example-bucket",
24 *       "ownerIdentity": {
25 *         "principalId": "EXAMPLE"
26 *       },
27 *       "arn": "arn:aws:s3:::example-bucket"
28 *     },
29 *     "object": {
30 *       "key": "test-image.jpg",

```

1:1 JSON Spaces: 2

Cancel

Invoke

Save

THE RESULT OF EXECUTION WILL LOOK LIKE:

```
Execution results
Test Event Name
komalEvent1

Response
{
  "statusCode": 200,
  "body": "\\Lambda function executed successfully!\\\"
}

Function Logs
START RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Version: $LATEST
[INFO] 2024-10-20T11:30:12.312Z 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Received event: {
  "Records": [
    {
      "eventVersion": "2.1",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "2024-10-19T17:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/456EXAMPLE="
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "komalimagebucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::komalimagebucket"
        },
        "object": {
          "key": "ottocycle.jpg",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",

```

```

[INFO] 2024-10-20T11:30:12.312Z 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg
END RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99
REPORT RequestId: 9dccd201-07da-4a2c-8b7d-119fe2a3ed99 Duration: 4.97 ms Billed Duration: 5 ms Memory Size: 128 MB Max Memory Used: 40 MB

Request ID |
9dccd201-07da-4a2c-8b7d-119fe2a3ed99
```

-the output of this code is of before the image fourstroke.jpg was uploaded.

2)CHECK THE CLOUDWATCH LOGS

Once the image is uploaded. The lambda function is triggered and it can be checked in the logs.

Go to cloudwatch>log group name>log stream

CloudWatch > Log groups > /aws/lambda/komal-lambdafun > 2024/10/20/[SLATEST]Ja1eb2c02645f446ab79825d2059cc189

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Q Filter events - press enter to search Clear 1m 30m 1h 12h Custom Local timezone Display

| Timestamp | Message |
|--|---|
| No older events at this moment. Retry | |
| 2024-10-20T16:58:05.921+05:30 | INIT_START Runtime Version: python:3.8.v59 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:31ddcc9689d292f9af49a11d58846fad8a5c21d1a933704c7257bcefb6b0 |
| 2024-10-20T16:58:06.097+05:30 | START RequestId: 7d1bf8a5-a3df-4dc2-93eb-1487c596ba33 Version: \$LATEST |
| 2024-10-20T16:58:06.098+05:30 | [INFO] 2024-10-20T11:28:06.098Z 7d1bf8a5-a3df-4dc2-93eb-1487c596ba33 Received event: { "Records": [{ "eventVersion": "2.1", "eventSource": "aws:s3", "awsRegion": "us-east-1", "eventTime": "2024-10-19T17:00:00.000Z", "eventName": "Obje |
| 2024-10-20T16:58:06.098+05:30 | [INFO] 2024-10-20T11:28:06.098Z 7d1bf8a5-a3df-4dc2-93eb-1487c596ba33 An Image has been added to the bucket: example-bucket, Object: test-image.jpg |
| 2024-10-20T16:58:06.118+05:30 | END RequestId: 7d1bf8a5-a3df-4dc2-93eb-1487c596ba33 |
| 2024-10-20T16:58:06.118+05:30 | REPORT RequestId: 7d1bf8a5-a3df-4dc2-93eb-1487c596ba33 Duration: 2.54 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 40 MB Init Duration: 174.16 ms |
| 2024-10-20T17:00:12.312+05:30 | START RequestId: 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 Version: \$LATEST |
| 2024-10-20T17:00:12.313+05:30 | [INFO] 2024-10-20T11:30:12.313Z 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 Received event: { "Records": [{ "eventVersion": "2.1", "eventSource": "aws:s3", "awsRegion": "us-east-1", "eventTime": "2024-10-19T17:00:00.000Z", "eventName": "Obje |
| 2024-10-20T17:00:12.313+05:30 | [INFO] 2024-10-20T11:30:12.313Z 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg |
| 2024-10-20T17:00:12.313+05:30 | [INFO] 2024-10-20T11:30:12.313Z 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg |
| 2024-10-20T17:00:12.317+05:30 | END RequestId: 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 |
| 2024-10-20T17:00:12.317+05:30 | REPORT RequestId: 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 Duration: 4.97 ms Billed Duration: 5 ms Memory Size: 128 MB Max Memory Used: 40 MB |
| No newer events at this moment. Auto retry paused Resume | |

2024-10-20T17:00:12.313+05:30 [INFO] 2024-10-20T11:30:12.313Z 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg

[INFO] 2024-10-20T11:30:12.313Z 9dccc201-07da-4a2c-8b7d-119fe2a3ed99 An Image has been added to the bucket: komalimagebucket, Object: ottocycle.jpg

AFTER UPLOADING ONE MORE IMAGE:

2024-10-20T16:07:11.478+05:30 [INFO] 2024-10-20T10:37:11.478Z e1324b12-b1ee-437a-bf9b-3cf4a16e471d An Image has been added to the bucket: komalimagebucket, Object: fourstroke.jpg

[INFO] 2024-10-20T10:37:11.478Z e1324b12-b1ee-437a-bf9b-3cf4a16e471d An Image has been added to the bucket: komalimagebucket, Object: fourstroke.jpg

3)FOR MONITORING THE LAMBDA FUNCTION BY USING NAGIOS ON EC2:

PREREQUISITES: INSTALL AND CONFIGURE NAGIOS ON YOUR EC2 INSTANCE.

1)GO TO THE LIBEXEC DIRECTORY `cd /usr/local/nagios/libexec`

```
[ec2-user@ip-172-31-43-5 ~]$ cd /usr/local/nagios/libexec  
[ec2-user@ip-172-31-43-5 libexec]$ sudo chmod +x check_lambda.sh
```

2) RUN THE COMMAND `sudo nano check_lambda.sh`

3) WRITE THE BELOW CODE IN THE SCRIPT

```
#!/bin/bash
```

```
# AWS Lambda function name
```

```
LAMBDA_FUNCTION_NAME="komal-lambdafun"
```

```
# S3 bucket name (replace this with your actual bucket name)
```

```
S3_BUCKET_NAME="your-s3-bucket-name"
```

```
# Log group for the Lambda function (CloudWatch logs)
```

```
LOG_GROUP_NAME="/aws/lambda/$LAMBDA_FUNCTION_NAME"
```

```
# Get the current time to filter logs from this point onward
```

```
CURRENT_TIME=$(date -u +"%Y-%m-%dT%H:%M:%S.000Z")
```

```
# Monitor Lambda for S3 trigger (image upload)
```

```
echo "Monitoring Lambda function '$LAMBDA_FUNCTION_NAME' for image uploads to S3 bucket  
'$S3_BUCKET_NAME'."
```

```
# Monitor S3 bucket for any new object (image upload)
```

```
echo "Checking if there is a new image uploaded to the S3 bucket..."
```



```
LATEST_OBJECT=$(aws s3api list-objects --bucket $S3_BUCKET_NAME --query 'Contents[?contains(Key,
`.jpg`) || contains(Key, `.png`)] | sort_by(@, &LastModified)[-1].Key' --output text)
```

```
if [ "$LATEST_OBJECT" == "None" ]; then
```

```
    echo "No image found in the bucket '$S3_BUCKET_NAME'."
```

```
    exit 1
```

```
else
```

```
    echo "Latest image uploaded: $LATEST_OBJECT"
```

```
fi
```

```
# Check the Lambda function's logs for recent activity (triggered by the S3 event)
```

```
echo "Checking Lambda function logs for the latest execution..."
```

```
# Fetch log streams (sorted by latest event time)
```

```
LATEST_LOG_STREAM=$(aws logs describe-log-streams --log-group-name $LOG_GROUP_NAME --order-
by LastEventTime --descending --limit 1 --query 'logStreams[0].logStreamName' --output text)
```

```
if [ "$LATEST_LOG_STREAM" == "None" ]; then
```

```
    echo "No logs found for Lambda function '$LAMBDA_FUNCTION_NAME'."
```

```
    exit 2
```

```
else
```

```
    echo "Fetching logs from stream: $LATEST_LOG_STREAM"
```

```
fi
```

```
# Get logs for the latest Lambda execution
```

```
LOG_EVENTS=$(aws logs get-log-events --log-group-name $LOG_GROUP_NAME --log-stream-name
$LATEST_LOG_STREAM --start-time $(date --date="$CURRENT_TIME" +%s%3N))
```

```
# Check if log events were found
```

```
if [ -z "$LOG_EVENTS" ]; then
```

```
    echo "No log events found for Lambda function execution after image upload."
    exit 3
else
    echo "Log events from Lambda function triggered by S3 upload:"
    echo "$LOG_EVENTS"
    exit 0
fi
```

-IT MONITORS YOUR LAMBDA FUNCTION AND THE S3 BUCKET'S LATEST CHANGES.

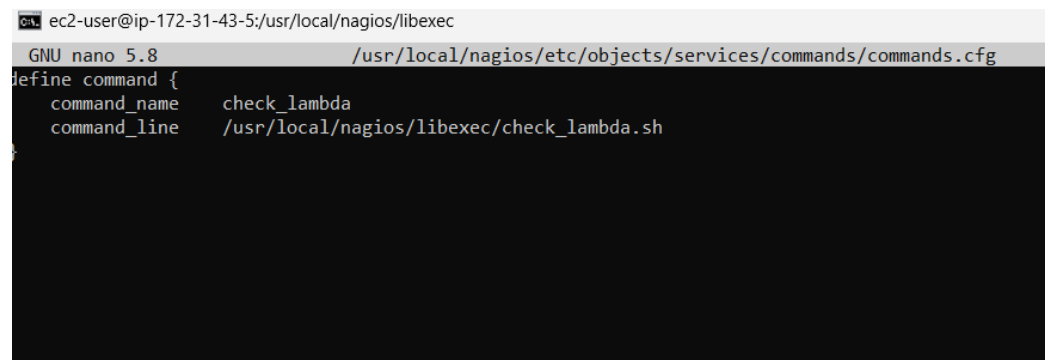
3)RUN *sudo chmod +x check_lambda.sh*

4) RUN THE FOLLOWING COMMAND TO ACCESS THE COMMANDS SCRIPT

sudo nano /usr/local/nagios/etc/objects/services/commands/commands.cfg

5)WRITE THE BELOW CODE IN THE SCRIPT

```
define command {
    command_name    check_lambda
    command_line    /usr/local/nagios/libexec/check_lambda.sh
}
```



```
ec2-user@ip-172-31-43-5:/usr/local/nagios/libexec
GNU nano 5.8 /usr/local/nagios/etc/objects/services/commands/commands.cfg
define command {
  command_name    check_lambda
  command_line    /usr/local/nagios/libexec/check_lambda.sh
}
```

6)RUN THE FOLLOWING COMMAND TO ACCESS THE SERVICES SCRIPT

sudo nano /usr/local/nagios/etc/objects/services/services.cfg

```
ec2-user@ip-172-31-43-5:/usr/local/nagios/libexec
GNU nano 5.8 /usr/local/nagios/etc/objects/services/services.cfg
define service {
    use                generic-service
    host_name          localhost
    service_description Check Lambda Function
    check_command       check_lambda
}
```

7)WRITE THE BELOW CODE IN THE SCRIPT

```
define service {

    use                generic-service

    host_name          localhost

    service_description Check Lambda Function

    check_command       check_lambda

}
```

8)SET THE AWS CREDENTIALS TO ACCESS AND MONITOR THE CREATED LAMBDA FUNCTION.

```
export AWS_SECRET_ACCESS_KEY="your_secret_key"
export AWS_SESSION_TOKEN="your_session_token"
export AWS_SESSION_TOKEN="your_session_token"
```

```
ec2-user@ip-172-31-43-5 libexec]$ export AWS_ACCESS_KEY_ID="ASIASMU6DPVCRK2RKYT7"
ec2-user@ip-172-31-43-5 libexec]$ export AWS_SESSION_TOKEN="Iqo3B3pZ21uxZVJEAcacQVzLxdlc1QlH11JNEYCQCd+3e0tv+R4FYRdW6n8E12ybHYLYLCKcgsy+36mbwQ1hAJ076off9/13D7vdhLAmgjs0Da09wtDvUjJh0m
2bgnKXUC4QAO8RWTg2DgD40TE0hj011gpl0krmu0dc27JaR0c0kplz4mmEU3h7Cxcac0/KattwRkKwCB22mubutnd0Wktutut0MT9o59plensAEGJum561bno7jipRk3c729bht2aGS80UgLO666wBgcce9eGR03smnbYlGnFTR9y0u5798BgfFip1Bu
3V91WV/N75f1VEBT3yLteVwQXoxS9Rur0dMFHf1rfu1M80JkgcAF8J0r5B31eWfntZcXPMGtoTop8Mir3QVjp1Zz+AbmJES+9CtAwitbfVop0ILfAw2uxqBPHt1v4e8rW0ZwBAZ7fnoLBunEpfrr+j0paiE9e7VEVniM2cnpJtaIDv0c0N1+E
G5S569ZYLjlaSkpH5M4oJD8McF649qW4J+SnaaJhp88MPj08rg60pw8Nni0fZKVS1Uda/etB15dGLI9eFTTICyCdaPncD0F2kUf3V2V1b1bcd100rSbwtTbEXsXx41PBmTj26m24qumj82ftaXkf99uW5WYKZkrebo0v7c611/v850pJQAEOLC3ald
hWqt/AeS9+20kDr1I6LhdJv+seEqCnH03DfOKDZ1YX893wHJQy3T6DfV41HC+vcQVzaFv1zpkS"TZUkk0XeF1qMYWoJXA0dtLCJQ0d082B1hgP1awBSS1n1tqA1C2LXC2FkX2TQxG2Fw29L1FGVzq2BTRZcplPEX2BBdwmmrGumB8r2Z2B8gMC5
ec2-user@ip-172-31-43-5 libexec]$ ^Cuh1965q8wG1ovgK0wFVY35eExwCr14feb0rCxm9LkX2BEH554KsEIQAKX2B2Lcmvh35rK9BVUJY3V3MzrQC4%2F3ueNB0x1VPInqEkumNucT6KIDGs0lrnpmba%2Bk185TrFW6HF8iDxFP270W11bIk
ec2-user@ip-172-31-43-5 libexec]$ ^C2EhbTG361SMYQSwVx2F0XTg13U1pHMPDSN3pyjmcwsZTTUm6Hk2Bz8R1wpJn1PX2FDZ2ccodgnleJUAviF32FD08tUUA424ZAcSxEXKtbo91c4yV2H8FcL%2F0ed790Kk313zZATVMX6yJBMVF54uFj
```

```
ec2-user@ip-172-31-43-5 libexec]$ export AWS_SECRET_ACCESS_KEY="uzjnpR3BdeJnPktvBZxxxzc4GHiCdmmJ9T+wGKL"
```

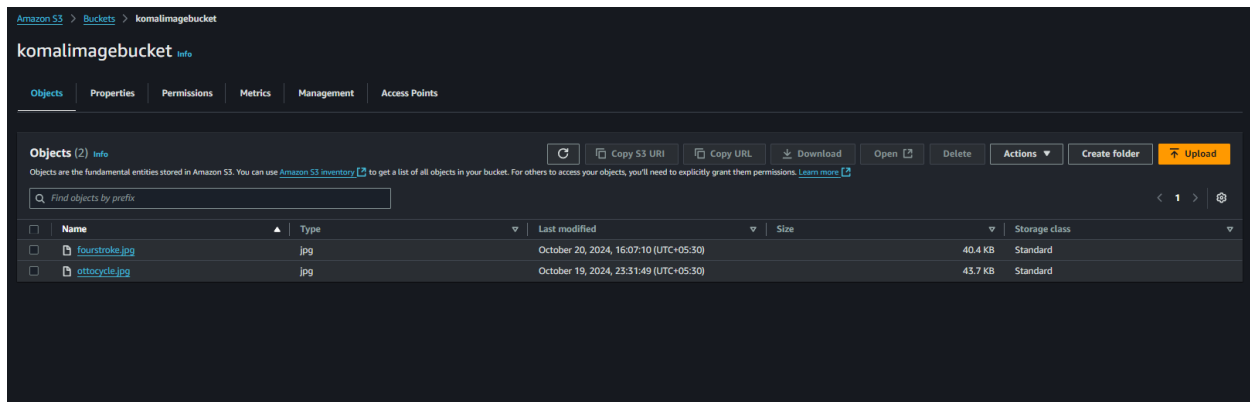
9) RESTART NAGIOS TO GET THE CHANGES UPDATED *sudo systemctl restart nagios*

10)RUN BELOW COMMAND TO CHECK THE MONITORING OF YOUR LAMBDA FUNCTION

./check_lambda.sh

```
ec2-user@ip-172-31-43-5 libexec]$ ./check_lambda.sh
Monitoring Lambda function 'komal-lambdafun' for image uploads to S3 bucket 'komalimagebucket'.
Checking if there is a new image uploaded to the S3 bucket...
Latest image uploaded: ottocycle.jpg
Checking Lambda function logs for the latest execution...
Fetching logs from stream: 2024/10/19/[$LATEST]6fffb263855042b295ab4aabd7f7f3fb
Log events from Lambda function triggered by S3 upload:
{
    "events": [],
    "nextForwardToken": "f/38567525873567295377391290756440763726736881459105103872/s",
    "nextBackwardToken": "b/38567365242593073582827578416871117500402558849515520000/s"
}
```

ALSO TRY UPLOADING A NEW IMAGE IN THE S3 BUCKET TO CHECK IF THE MONITORING IS WORKING



```
Monitoring Lambda function 'komal-lambdafun' for image uploads to S3 bucket 'komalimagebucket'.
Checking if there is a new image uploaded to the S3 bucket...
Latest image uploaded: fourstroke.jpg
Checking Lambda function logs for the latest execution...
Fetching logs from stream: 2024/10/20/[$LATEST]2385120a48774ac5aac4a8c8a8f44d49
Log events from Lambda function triggered by S3 upload:
{
  "events": [],
  "nextForwardToken": "f/38567529563337092200275542639233030527771534431094833152/s",
  "nextBackwardToken": "b/38567368944516776538911019911800350760030568842264576000/s"
}
[ec2-user@ip-172-31-43-5 libexec]$
Broadcast message from root@localhost (Sun 2024-10-20 10:37:49 UTC):

The system will power off now!

Connection to ec2-98-83-23-82.compute-1.amazonaws.com closed by remote host.
Connection to ec2-98-83-23-82.compute-1.amazonaws.com closed.
```

-IT ALSO NOTIFIES YOU ABOUT SYSTEM BEING POWERED OFF BY THE HOST.

11) RUN `aws lambda get-function --function-name your_function_name` TO GET THE CONFIGURATION OF THE LAMBDA FUNCTION.

```
ec2-user@ip-172-31-43-5 libexec]$ aws lambda get-function --function-name komal-lambdafun
```

```
{
  "Configuration": {
    "FunctionName": "komal-lambdafun",
    "FunctionArn": "arn:aws:lambda:us-east-1:186088914245:function:komal-lambdafun",
    "Runtime": "python3.8",
    "Role": "arn:aws:iam:186088914245:role/LabRole",
    "Handler": "lambda_function.lambda_handler",
    "CodeSize": 557,
    "Description": "",
    "Timeout": 3,
    "MemorySize": 128,
    "LastModified": "2024-10-19T17:57:43.000+0000",
    "CodeSha256": "SUR6xL78G9bHk3QX3UFHBHP1jj0Cm2cujsk9Wr9d=",
    "Version": "$LATEST",
    "TracingConfig": {
      "Mode": "PassThrough"
    },
    "RevisionId": "0376bc2b-e931-4909-b7cc-e09fcd5ea146",
    "State": "Active",
    "LastUpdateStatus": "Successful",
    "PackageType": "Zip",
    "Architectures": [
      "x86_64"
    ],
    "EphemeralStorage": {
      "Size": 512
    },
    "SnapStart": {
      "ApplyOn": "None",
      "OptimizationStatus": "Off"
    },
    "RuntimeVersionConfig": {
      "RuntimeVersionArn": "arn:aws:lambda:us-east-1::runtime:31ddccb9689d29d2fa6f49a11d508466fad8a5c21d1a9337d4cf257bcef8ebb0"
    },
    "LoggingConfig": {
      "LogFormat": "Text",
      "LogGroup": "/aws/lambda/komal-lambdafun"
    }
  },
  "Code": {
    "RepositoryType": "s3",
    "Location": "https://prod-04-2014-tasks.s3.us-east-1.amazonaws.com/snapshots/186088914245/komal-lambdafun-dbc2793b-3ebf-49f3-afe9-5f2cbaefcfc0?versionId=Q2bTAvh1gWg2vaw86p18cmwhJVoa36.t&X-Amz-Security-Token=IQoJb3JpZ21uX2VjEAoACXVzLWVhc3QtMSJHMEUCIPV8uNF8s2bYXCQ2BHZH8ANoLd2TRf7VER7dED3sAsuIAiEA91gyQINPv9TVI8Sk4AJt9s5dyHBC32FVXTvtFXsSZ3PUCqsgUichAAGgw3NDk2Nzg5MDI4Mzk"
```