

Name: Manav Jawrani

Roll No.: 19

Subject: Advanced DevOps

Experiment No.: 12

Experiment 12

Aim: To create a Lambda function which will log "An Image has been added" once you add a object to specific bucket in S3.

Theory:

- What are the features of Lambda function? The following key features help you develop Lambda applications that are scalable, secure and easily extensible:

1. Concurrency and Scaling Controls:

Using concurrency settings to ensure that your production applications are highly available and highly responsive. Lambda manages the infrastructure that runs that code, and scales automatically in response to incoming requests.

2. Function URLs:

Lambda offers built-in HTTP(S) endpoint support through function URLs. With function URLs, you can assign a dedicated HTTP endpoint to your Lambda function. When your function URL is configured you can use it to invoke your function through a web browser, curl, postman or any HTTP client. You can also add a function URL to an existing function.

3. Event Source mappings:

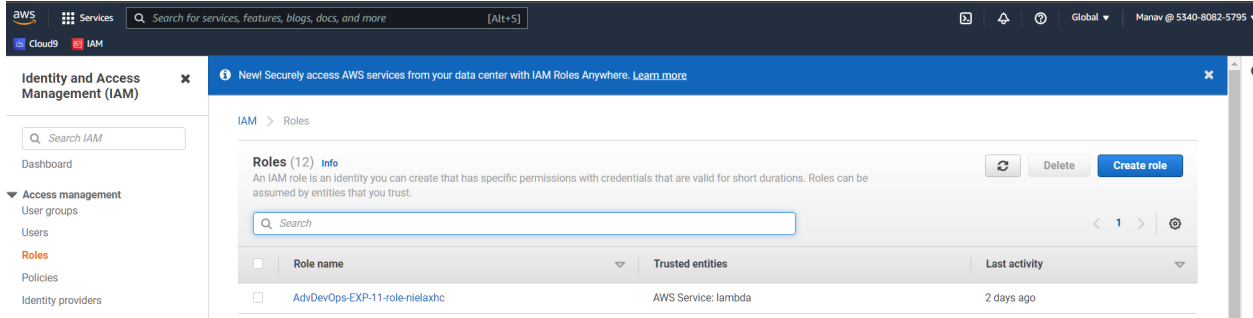
Event source mappings maintain a local queue of unprocessed items and handle retries if the function returns an error or is throttled. You can configure an event source mapping to customize batching behavior and error handling or to send a record of items that fail processing to a destination.

4. Testing and deployment tools:

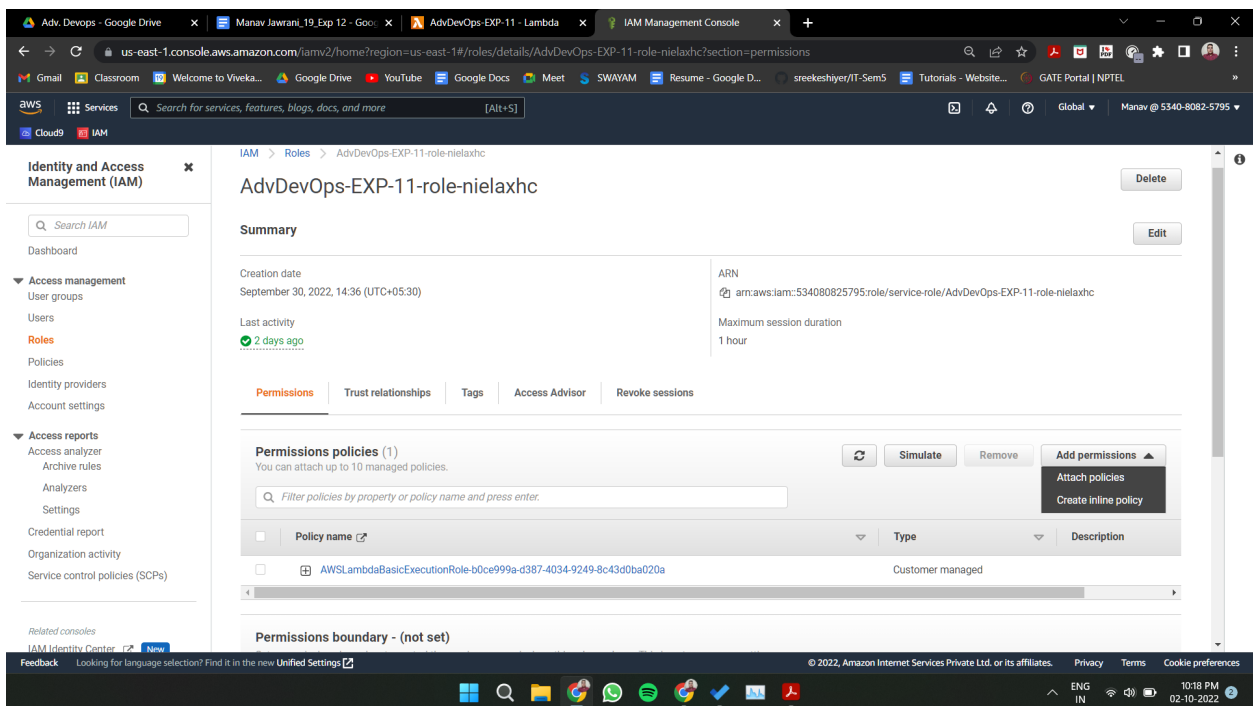
Lambda supports deploying code as is or as container images. You can use a rich tools ecosystem for authoring, building and deploying your Lambda functions using AWS and popular community tools like the Docker Command Line Interface (CLI).

Implementation:

Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.



S3-ReadOnly

Adv. DevOps - Google Drive | Manav Jawani_19_Exp 12 - Google Drive | AdvDevOps-EXP-11 - Lambda | IAM Management Console

us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/AdvDevOps-EXP-11-role-nielaxhc/attach-policies

Attach policy to AdvDevOps-EXP-11-role-nielaxhc

► Current permissions policies (1)

Other permissions policies (773)

Filter policies by property or policy name and press enter. 1 match

"S3Read" X Clear filters

<input type="checkbox"/>	Policy name	Type	Description
<input type="checkbox"/>	AmazonS3ReadOnlyAccess	AWS managed	Provides read only access to all S3 objects.

Cancel Attach policies

Feedback Looking for language selection? Find it in the new Unified Settings.

© 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

ENG IN 10:19 PM 02-10-2022

CloudWatchFull

Adv. DevOps - Google Drive | Manav Jawani_19_Exp 12 - Google Drive | AdvDevOps-EXP-11 - Lambda | IAM Management Console

us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/AdvDevOps-EXP-11-role-nielaxhc/attach-policies

Attach policy to AdvDevOps-EXP-11-role-nielaxhc

► Current permissions policies (2)

Other permissions policies (772)

Filter policies by property or policy name and press enter. 1 match

"Cloudwatchfull" X Clear filters

<input type="checkbox"/>	Policy name	Type	Description
<input type="checkbox"/>	CloudWatchFullAccess	AWS managed	Provides full access to CloudWatch.

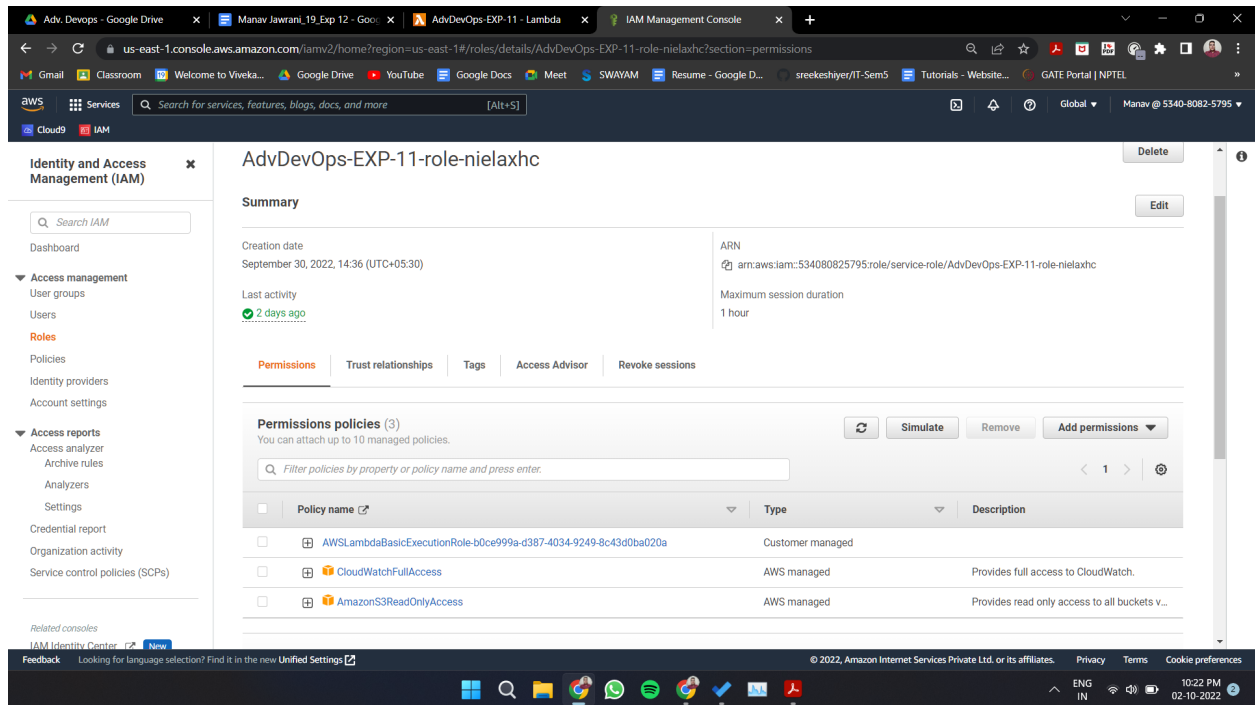
Cancel Attach policies

https://us-east-1.console.aws.amazon.com/iam/home#/policies/arn:aws:iam::aws:policy/CloudWatchFullAccess

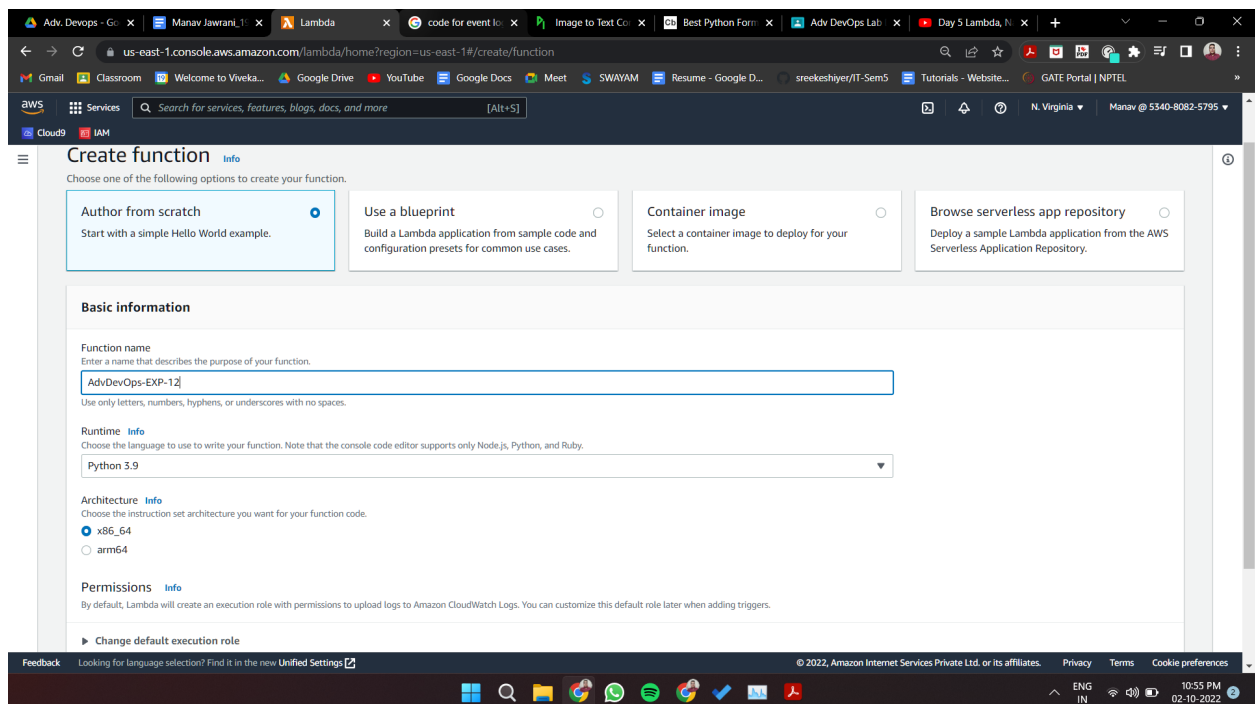
© 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

ENG IN 10:21 PM 02-10-2022

After successful attachment of policy you will see something like this you will be able to see the updated policies.



Step 3: Open up AWS Lambda and create a new Python function.



Under Execution Role, choose the existing role, then select the one which was previously created and to which we just added permissions.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☐ Create a new role with basic Lambda permissions

☒ Use an existing role

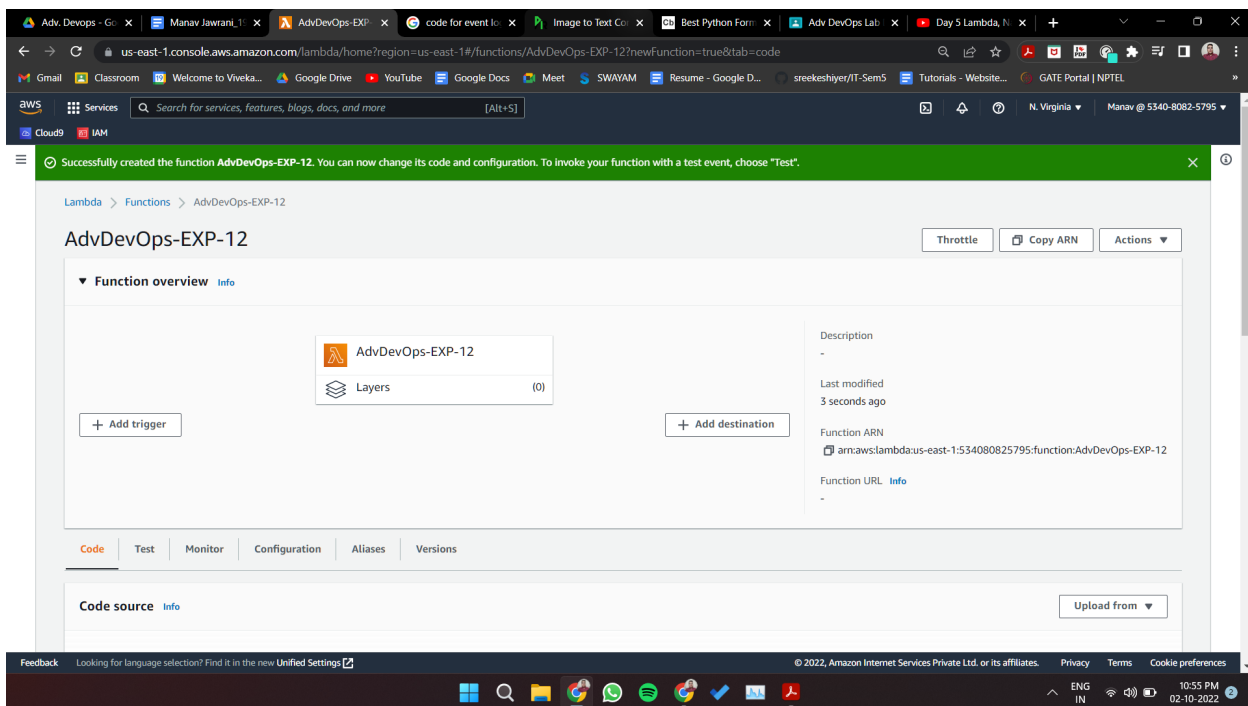
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

[View the AdvDevOps-EXP-11-role-nielaxhc role on the IAM console.](#)

► Advanced settings

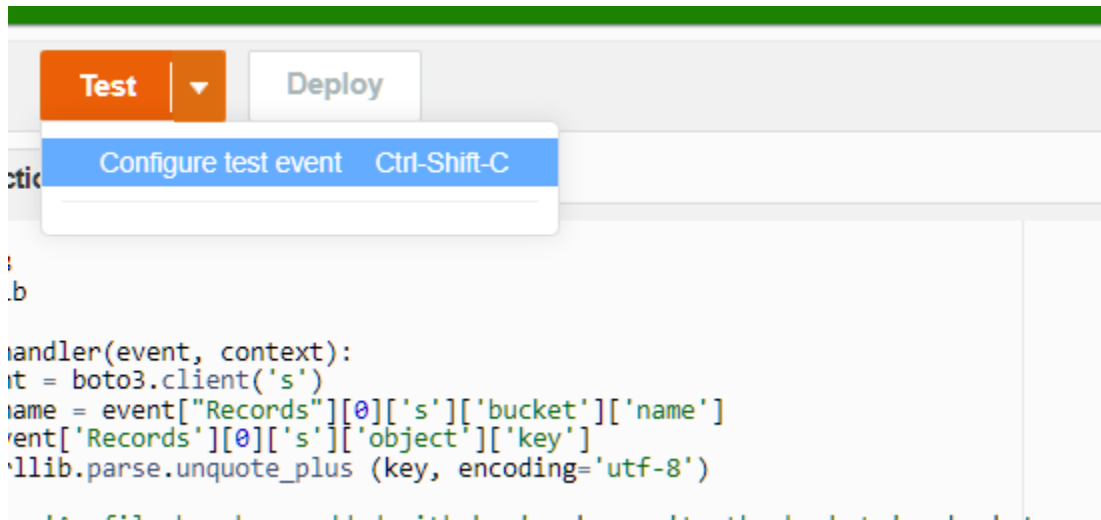
Step 4: The function is up and running.



Step 5: Make the following changes to the function and click on the deploy button. This code basically logs a message and logs the contents of a JSON file which is uploaded to an S3 Bucket and then deploy the code.

```
lambda_function ×
1 import json
2 import boto3
3 import urllib
4
5 def lambda_handler(event, context):
6
7     s3_client = boto3.client('s3')
8     bucket_name = event["Records"][0]['s3']['bucket']['name']
9     key = event["Records"][0]['s3']['object']['key']
10    key = urllib.parse.unquote_plus(key, encoding='utf-8')
11
12    message = 'An file has been added with key ' + key + ' to the bucket ' + bucket_name
13    print(message)
14
15    response = s3_client.get_object(Bucket=bucket_name, Key=key)
16    contents = response["Body"].read().decode()
17    contents = json.loads(contents)
18
19    print("These are the Contents of the File: \n", contents)
```

Step 6: Click on Test and choose the ‘S3 Put’ Template.



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, configure the JSON event, then choose Test.

Test event action

☒ Create new event

☐ Edit saved event

Event name

Test

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Q s3|



AWS

Rekognition **S3** Request

S3 Delete

S3 Put



s3-put



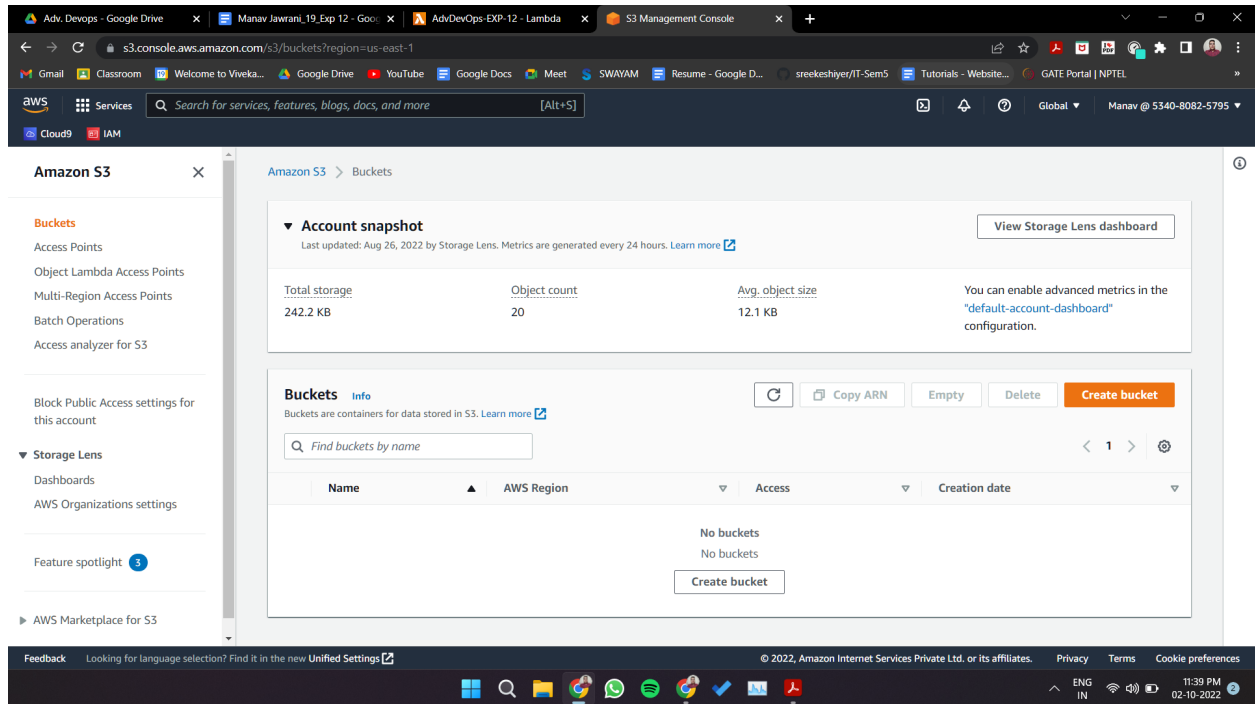
Event JSON

Format JSON

```
1 {
2   "Records": [
3     {
4       "eventVersion": "2.0",
5       "eventSource": "aws:s3",
6       "awsRegion": "us-east-1",
7       "eventTime": "1970-01-01T00:00:00.000Z",
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": "EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
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%2Fkey",
31          "size": 1024,
```

And Save it.

Step 7: Open up the S3 Console and create a new bucket.



Step 8: With all general settings, create the bucket in the same region as the function.

General configuration

Bucket name

advdevopsexp12

Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Choose bucket

Step 9: Click on the created bucket and under properties, look for events.

Amazon S3 > Buckets

Account snapshot

Last updated: Aug 26, 2022 by Storage Lens. Metrics are generated every 24 hours. [Learn more](#)

View Storage Lens dashboard

Buckets (1) Info

Buckets are containers for data stored in S3. [Learn more](#)

Refresh

Copy ARN

Empty

Delete

Create bucket

	Name	AWS Region	Access	Creation date
<input type="radio"/>	advdevopsexp12	US East (N. Virginia) us-east-1	Bucket and objects not public	October 2, 2022, 23:41:50 (UTC+05:30)

Event notifications (0)

Edit

Delete

Create event notification

Send a notification when specific events occur in your bucket. [Learn more](#)

	Name	Event types	Filters	Destination type	Destination
No event notifications					
Choose Create event notification to be notified when a specific event occurs.					
<div>Create event notification</div>					

Amazon EventBridge

For additional capabilities, use Amazon EventBridge to build event-driven applications at scale using S3 event notifications. [Learn more](#) or [see EventBridge pricing](#)

Edit

Send notifications to Amazon EventBridge for all events in this bucket

Off

Click on Create Event Notification.

Step 10: Mention an event name and check Put under event types.

General configuration

Event name
S3PutRequest
Event name can contain up to 255 characters.

Prefix - optional
Limit the notifications to objects with key starting with specified characters.
images/

Suffix - optional
Limit the notifications to objects with key ending with specified characters.
.jpg

Event types
Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.

Object creation

☐ All object create events
s3:ObjectCreated:*

☒ Put
s3:ObjectCreated:Put

☐ Post
s3:ObjectCreated:Post

Choose Lambda function as destination and choose your lambda function and save the changes.

Destination

Before Amazon S3 can publish messages to a destination, you must grant the Amazon S3 principal the necessary permissions to call the relevant API to publish messages to an SNS topic, an SQS queue, or a Lambda function. [Learn more](#)

Destination
Choose a destination to publish the event. [Learn more](#)

☒ **Lambda function**
Run a Lambda function script based on S3 events.

☐ **SNS topic**
Fanout messages to systems for parallel processing or directly to people.

☐ **SQS queue**
Send notifications to an SQS queue to be read by a server.

Specify Lambda function

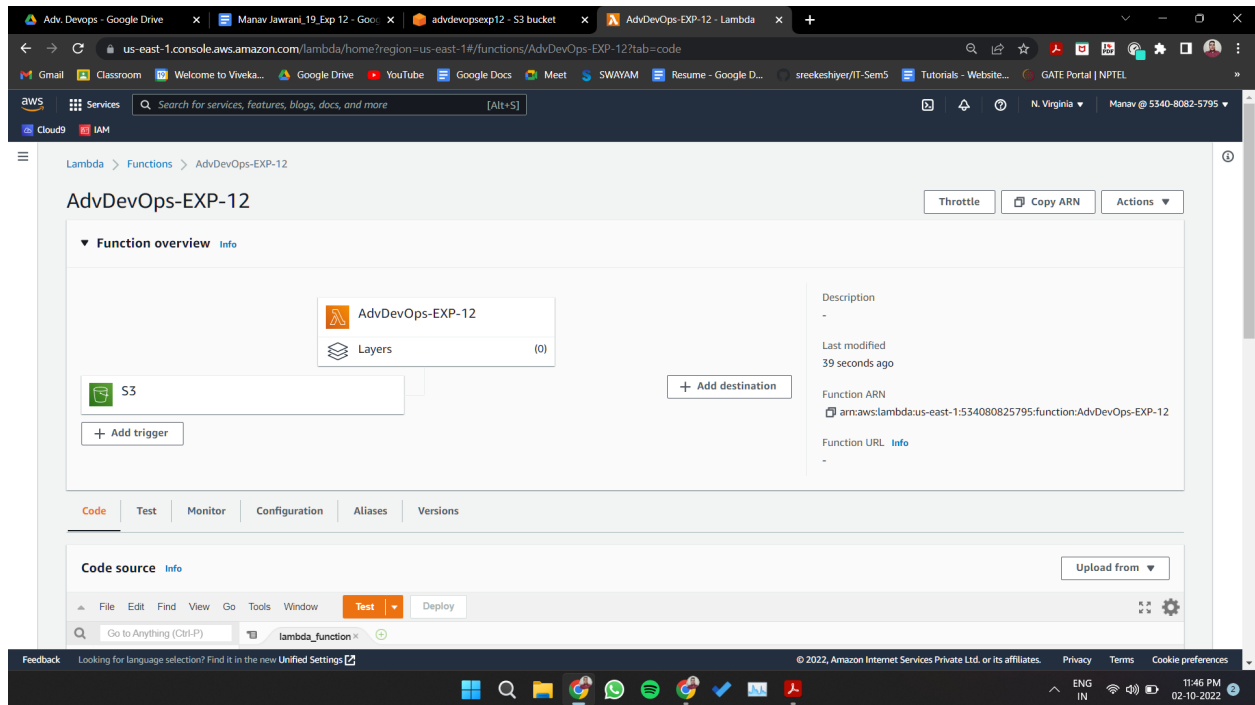
☒ Choose from your Lambda functions

☐ Enter Lambda function ARN

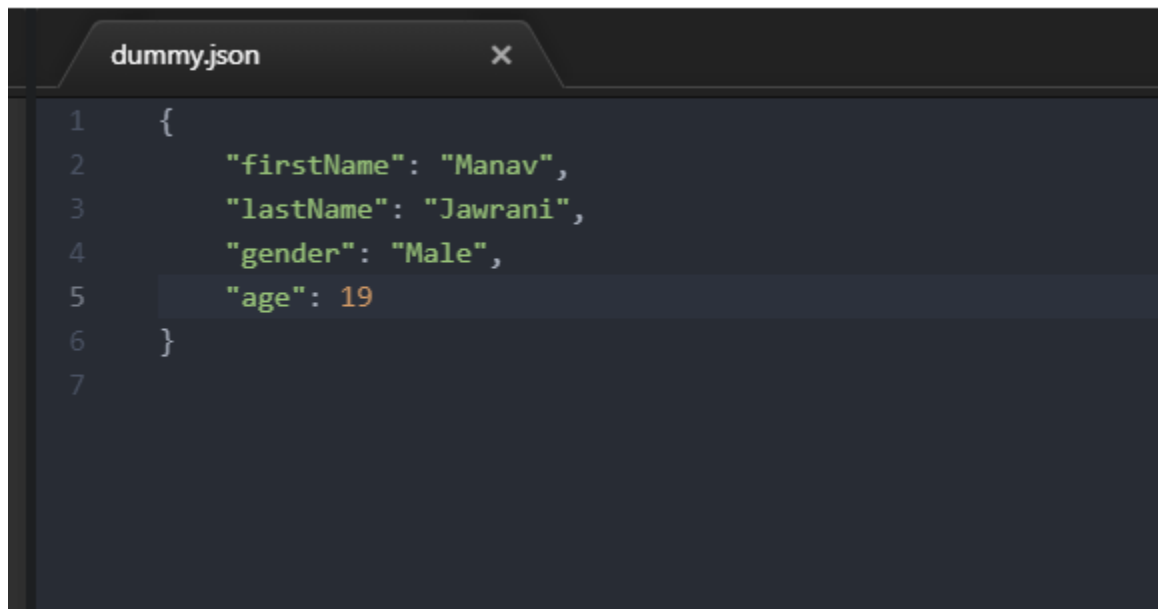
Lambda function
AdvDevOps-EXP-12

Cancel Save changes

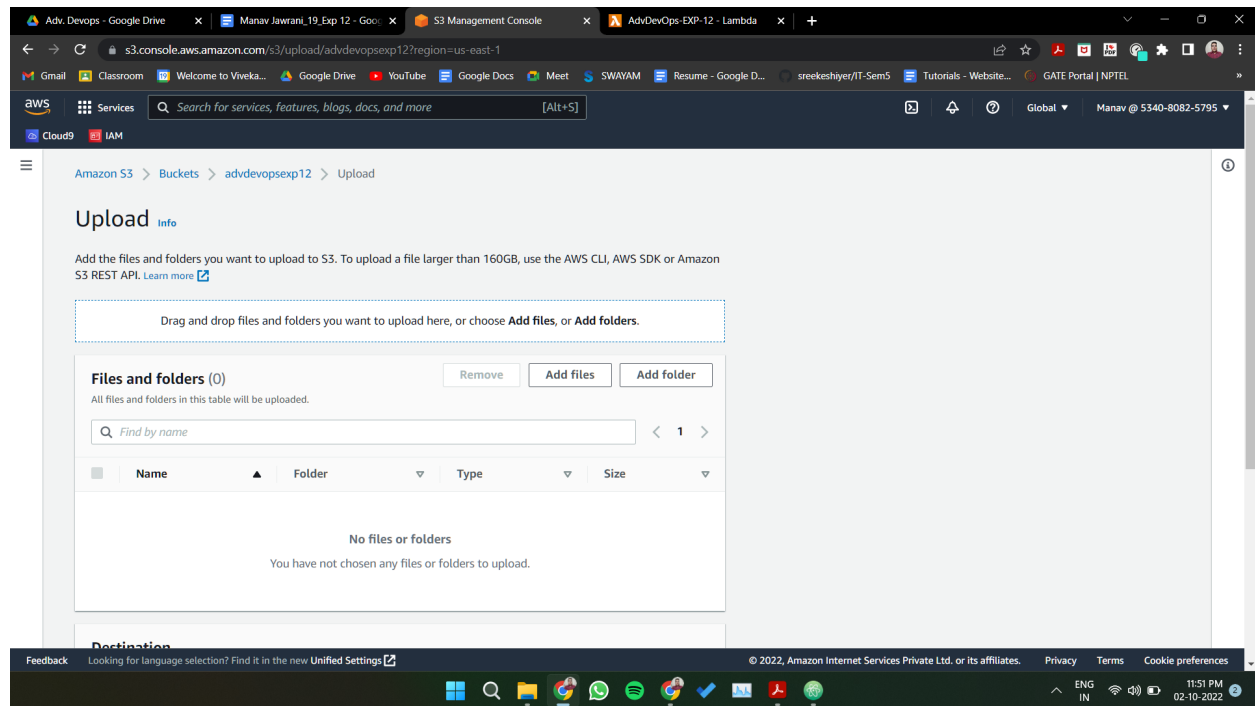
Step 11: Refresh the Lambda function console and you should be able to see an S3 Trigger in the overview.



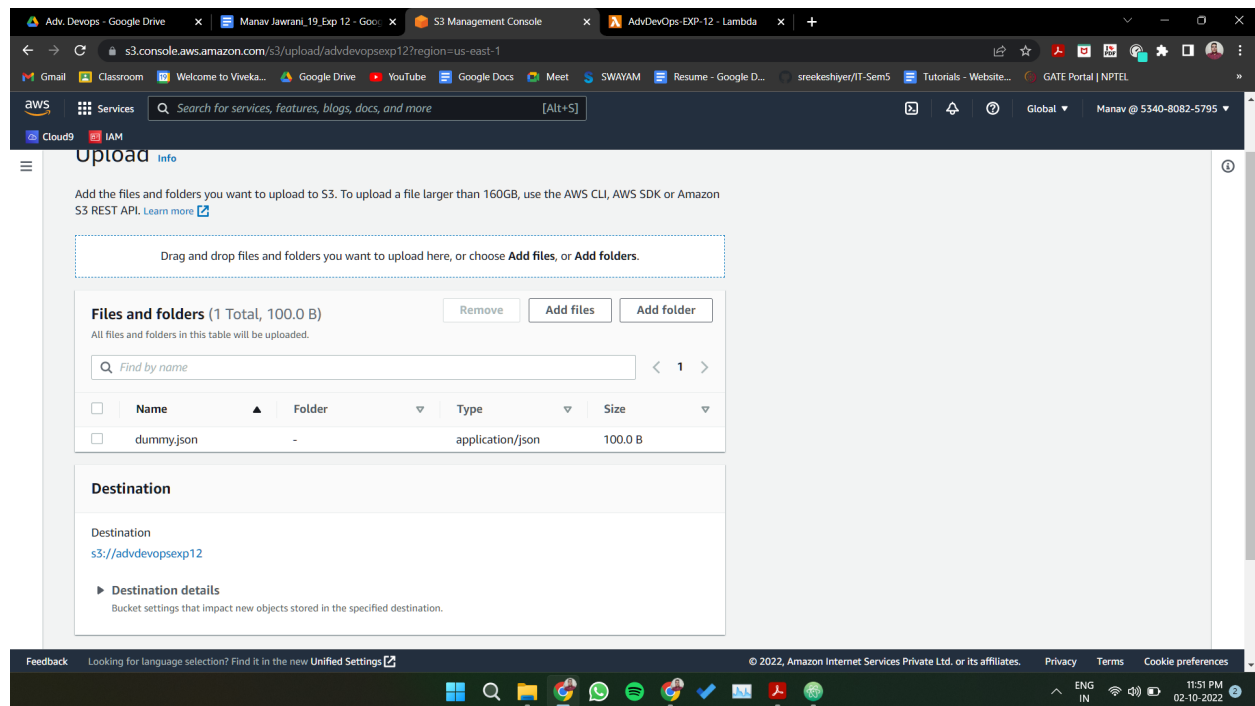
Step 12: Now, create a dummy JSON file locally.



Step 13: Go back to your S3 Bucket and click on Add Files to upload a new file.



Step 14: Select the dummy data file from your computer and click Upload.

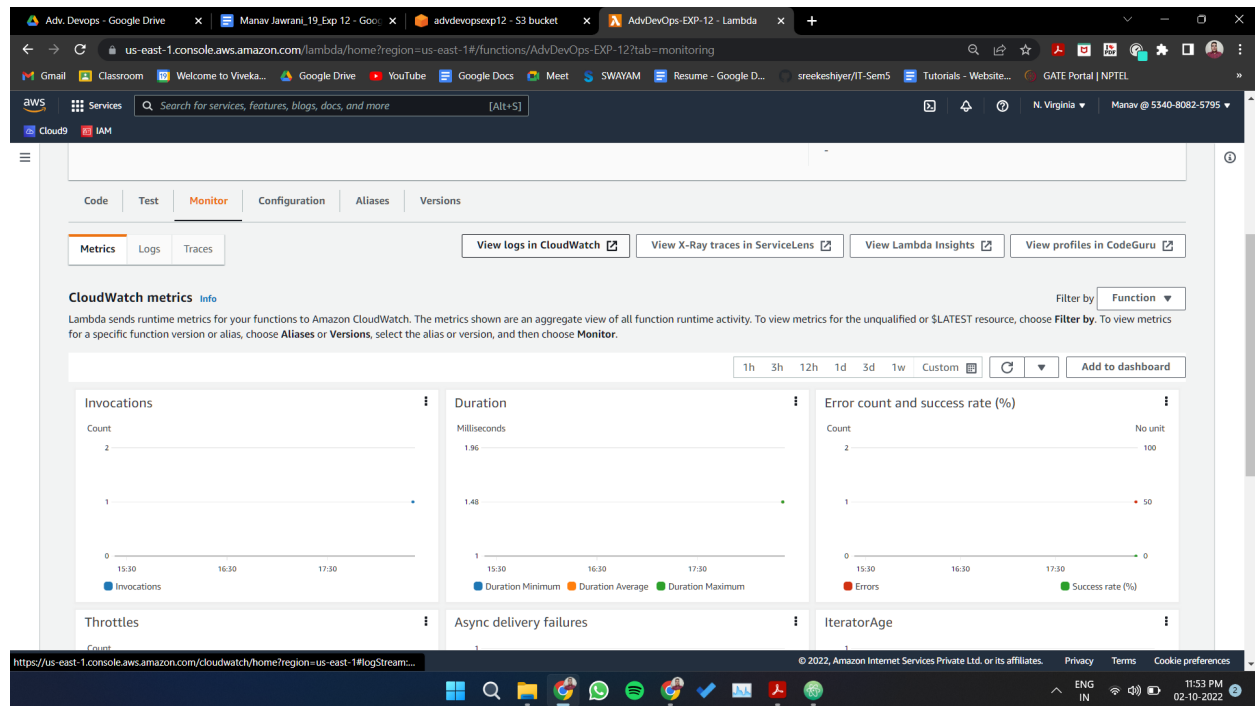


Step 15: After this make the necessary changes in the Test configuration file which we created previously by replacing the **Bucket Name** and the **ARN of Bucket**.

Event JSON Format JSON

```
1 {
2   "Records": [
3     {
4       "eventVersion": "2.0",
5       "eventSource": "aws:s3",
6       "awsRegion": "us-east-1",
7       "eventTime": "1970-01-01T00:00:00.000Z",
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": "EXAMPLE123/5678abcdefghijklmbdaisawesome/mnopqrstuvwxyzABCDEFGH"
18      },
19      "s3": {
20        "s3SchemaVersion": "1.0",
21        "configurationId": "testConfigRule",
22        "bucket": {
23          "name": "advdevopsexp12",
24          "ownerIdentity": {
25            "principalId": "EXAMPLE"
26          },
27          "arn": "arn:aws:s3:::advdevopsexp12"
28        },
29        "object": {
30          "key": "dummy.json",
31          "size": 1024,
```

Step 16: Go back to your Lambda function , Refresh it and check the Monitor tab.



Under Log streams, click on View logs in Cloudwatch to check the Function logs.

The screenshot shows the AWS CloudWatch 'Log streams' page. The 'Log streams' tab is selected. The page displays a list of log streams. The first log stream is '2022/10/02/[LATEST]fe8dd4f713af4ae09f9b6c2660aec7bc' with a last event time of '2022-10-03 00:48:59 (UTC+05:30)'. The page includes a search bar, a filter for 'Exact match', and buttons for 'Create log stream' and 'Search all log streams'.

Log stream	Last event time
2022/10/02/[LATEST]fe8dd4f713af4ae09f9b6c2660aec7bc	2022-10-03 00:48:59 (UTC+05:30)

Step 17: Click on this log Stream that was created to view what was logged by your function.

▶	2022-10-03T00:49:16.589+05:30	START RequestId: bb374eb8-671f-4ebc-accf-7d320a054081 Version: \$LATEST
▶	2022-10-03T00:49:16.613+05:30	An file has been added with key dummy.json to the bucket advdevopsexp12
▶	2022-10-03T00:49:16.808+05:30	These are the Contents of the File:
▶	2022-10-03T00:49:16.808+05:30	{'firstName': 'Manav', 'lastName': 'Jawrani', 'gender': 'Male', 'age': 19}
▶	2022-10-03T00:49:16.813+05:30	END RequestId: bb374eb8-671f-4ebc-accf-7d320a054081
▶	2022-10-03T00:49:16.813+05:30	REPORT RequestId: bb374eb8-671f-4ebc-accf-7d320a054081 Duration: 223.69 ms Billed Duration: 224 ms Memory Size: 128 MB Max Memory Used: 72 MB
No newer events at this moment. <i>Auto retry paused. Resume</i>		

As you can see, our function logged that a file was uploaded with its file name and the bucket to which it was uploaded. It also mentions the contents inside the file as our function was defined to.

Hence, we have successfully created a Python function inside AWS Lambda which logs every time an object is uploaded to an S3 Bucket.

Conclusion!

1. Here, we have successfully created a python function, inside AWS Lambda which logs everytime when an object is uploaded to an AWS S3 bucket.
2. Along with this we got to know how to create ~~a~~ logs for any of the Lambda function in future so that we can see the logs everytime whenever there is a change.