

Adv DevOps Exp-12

Aim:

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

Theory: [Exp12](#)**AWS Lambda and S3 Integration:**

AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

Workflow:**1. Create an S3 Bucket:**

- First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.

2. Create the Lambda Function:

- Set up a new Lambda function using AWS Lambda's console. You can choose a runtime environment like Python, Node.js, or Java.
- Write code that logs a message like "An Image has been added" when triggered.

3. Set Up Permissions:

- Ensure that the Lambda function has the necessary permissions to access S3.

You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

4. Configure S3 Trigger:

- Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

5. Test the Setup:

- Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message "An Image has been added" in AWS CloudWatch Logs. Prerequisites: AWS Personal Account

Prerequisites: AWS Personal Account

Steps To create the lambda function:

Step 1: Login to your AWS Personal account. Now open S3 from services and click on create S3 bucket.

The screenshot displays the Amazon S3 homepage. On the left, the text reads "Storage Amazon S3 Store and retrieve any amount of data from anywhere" followed by a description: "Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance." On the right, there is a "Create a bucket" section with the text "Every object in S3 is stored in a bucket. To upload files and folders to S3, you'll need to create a bucket where the objects will be stored." and a prominent orange "Create bucket" button. Below this, a "Pricing" section states "With S3, there are no minimum fees. You only pay for what you use. Prices are based on the location of your S3 bucket." and includes links to "Estimate your monthly bill using the AWS Simple Monthly Calculator" and "View pricing details". At the bottom left, there is a "How it works" section featuring a video thumbnail titled "Introduction to Amazon S3" with the AWS logo and a "Copy link" button.

Step 2: Now Give a name to the Bucket, select general purpose project and deselect the Block public access and keep other this to default.

Create bucket Info

Buckets are containers for data stored in S3.

General configuration

AWS Region

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


Bucket type Info

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name Info

wearekcs

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#) 

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

Choose bucket

Format: s3://bucket/prefix

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.

☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming

Successfully created bucket "warecks"

To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Amazon S3 > Buckets

Account snapshot - updated every 24 hours [All AWS Regions](#) [View Storage Lens dashboard](#)

Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

General purpose buckets | Directory buckets

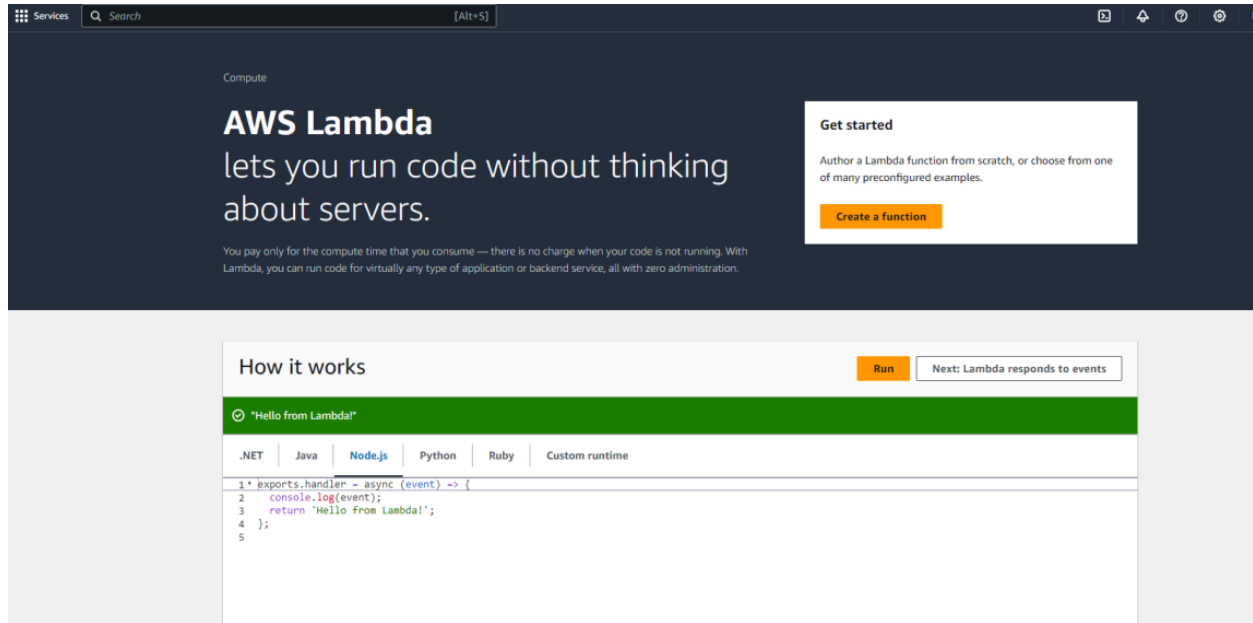
General purpose buckets (1) [Info](#) [All AWS Regions](#)

Buckets are containers for data stored in S3.

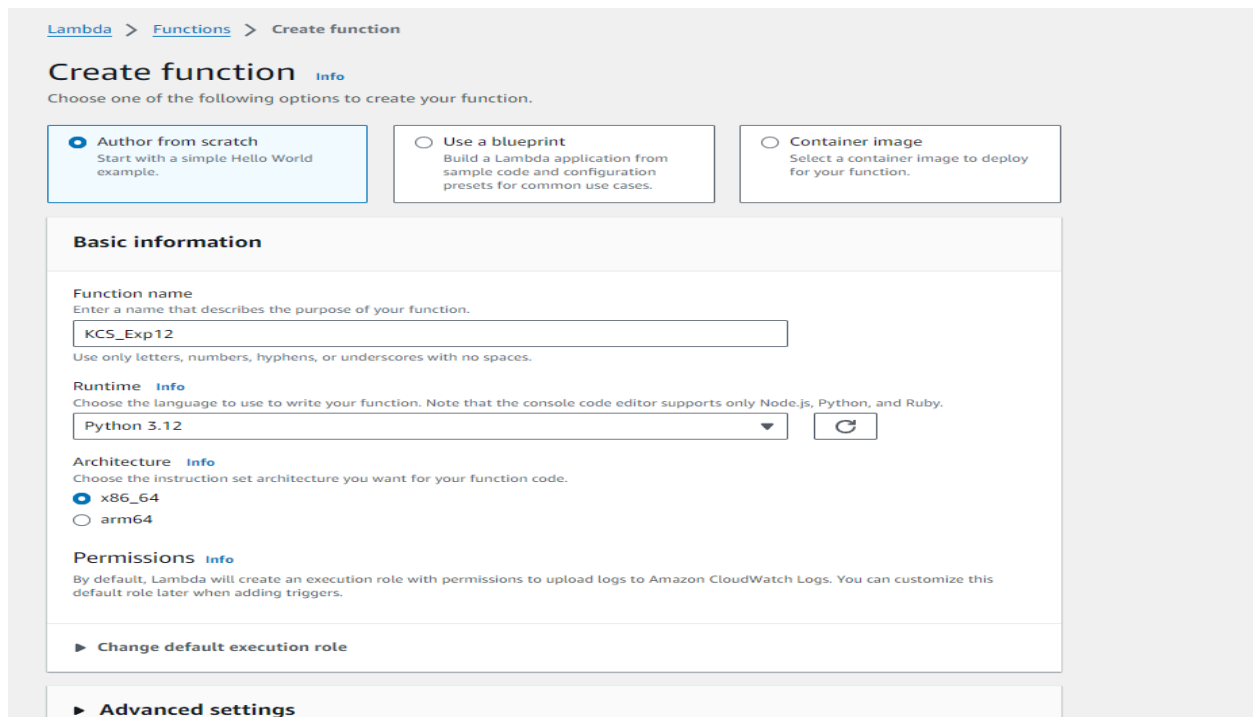
[Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

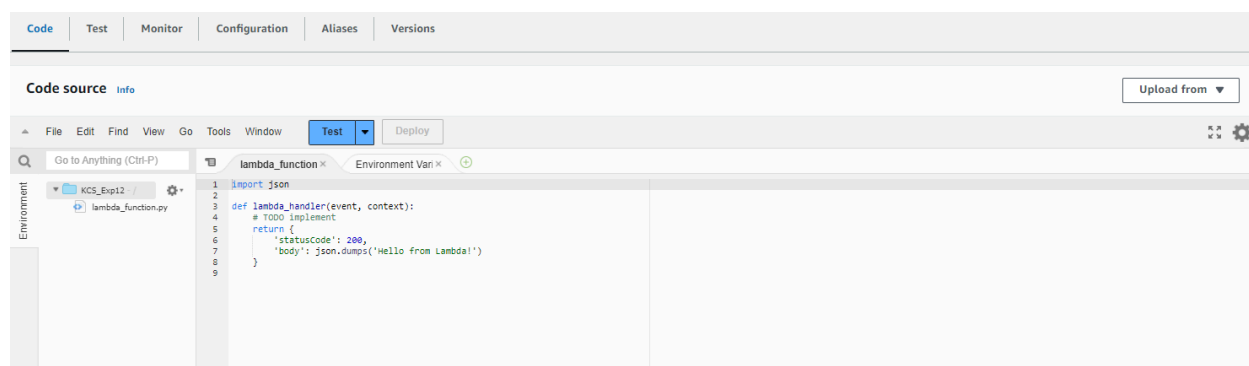
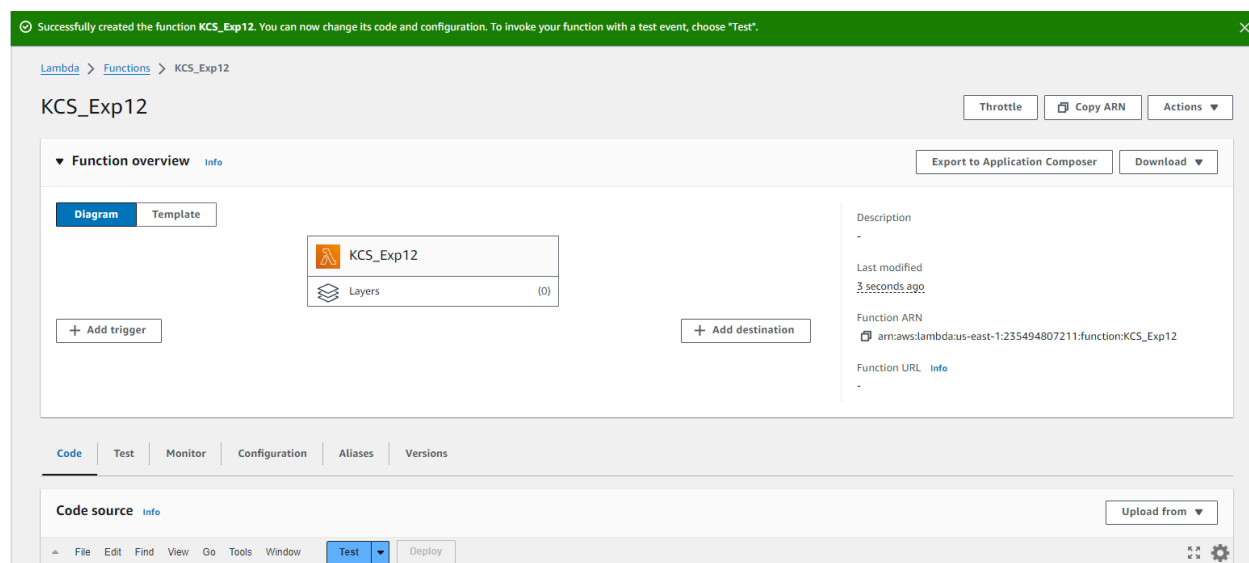
Name	AWS Region	IAM Access Analyzer	Creation date
<input type="radio"/> warecks	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 1, 2024, 13:40:40 (UTC+05:30)

Step 3: Open lambda console and click on create function button

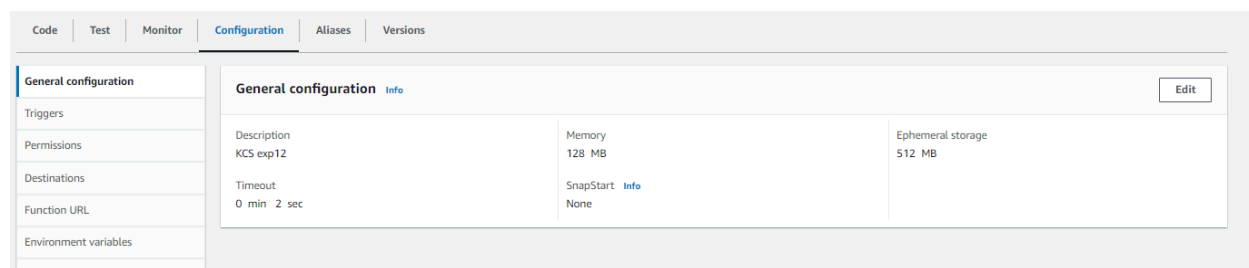


Step 4: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Execution role to Create a new role with basic Lambda permissions.





To See or Edit the basic settings go to configuration then click on edit general setting



Change any setting of your choice. Here I have set a timeout of 2 secs. Then save changes

[Lambda](#) > [Functions](#) > [KCS_Exp12](#) > Edit basic settings

Edit basic settings

Basic settings [Info](#)

Description - optional

Memory [Info](#)

Your function is allocated CPU proportional to the memory configured.

 MB

Set memory to between 128 MB and 10240 MB

Ephemeral storage [Info](#)

You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)

 MB

Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

SnapStart [Info](#)

Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#).

None ▼

Supported runtimes: Java 11, Java 17, Java 21.

Timeout

 min sec

Execution role

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

☒ Use an existing role

☐ Create a new role from AWS policy templates

Existing role

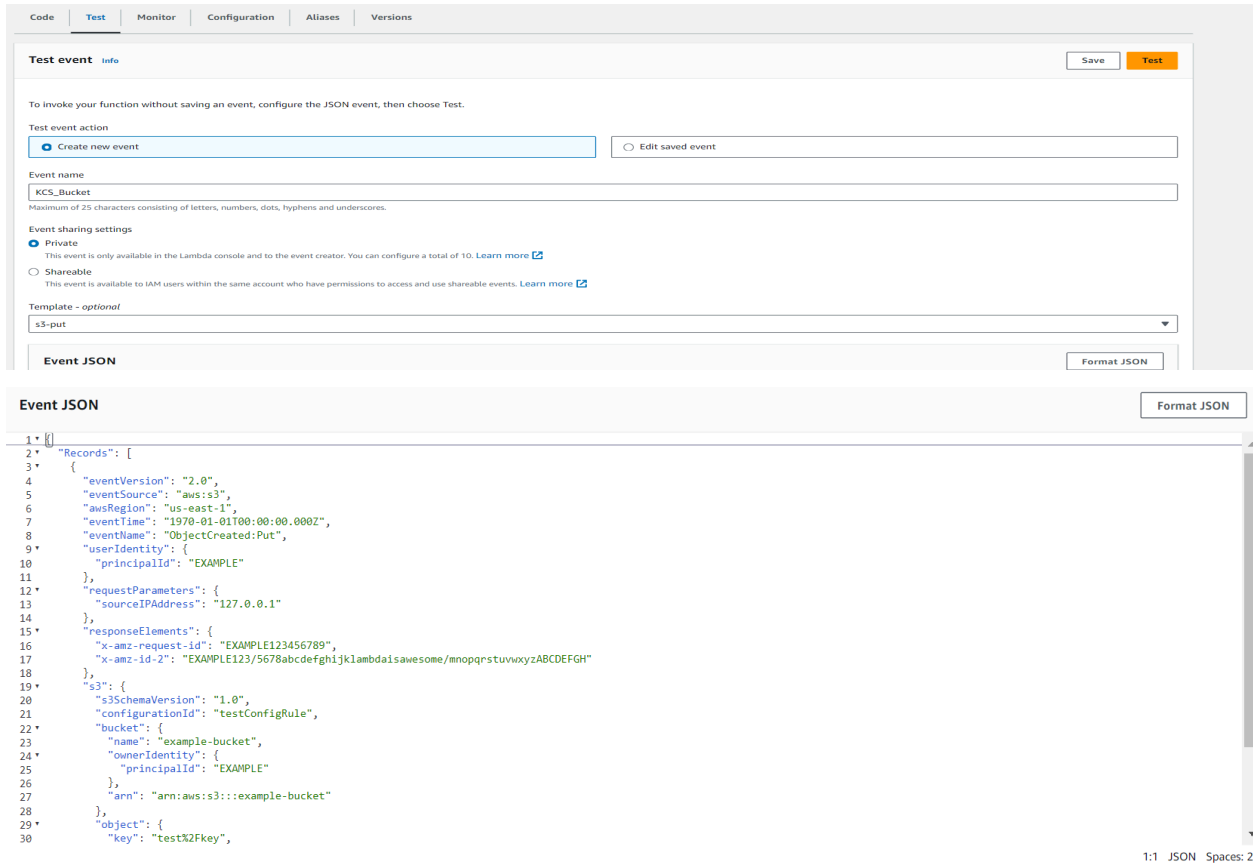
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

service-role/KCS_Exp12-role-0q6h1t4r ▼

↻

[View the KCS_Exp12-role-0q6h1t4r role](#) on the IAM console.

Step 5: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.



Test event [Info](#) Save Test

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

KCS_Bucket

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

Event sharing settings

☒ Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

☐ Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

s3-put

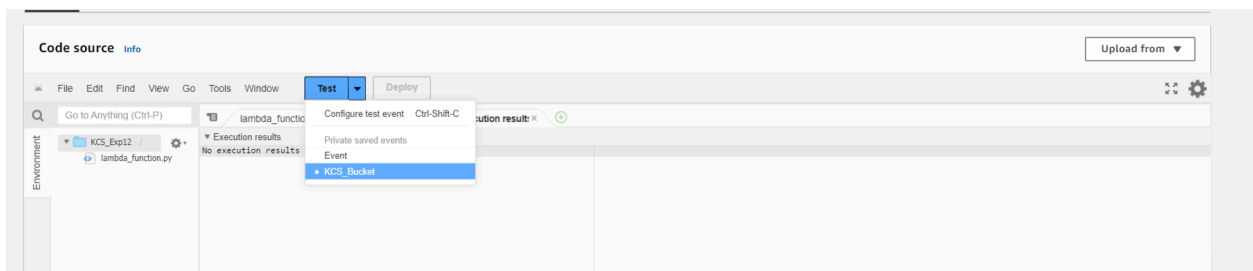
Event JSON Format JSON

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": "EXAMPLE123/5678abcdefgijklambdaawesome/mnopqrstuvxyzABCDEFGH"
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",
```

1:1 JSON Spaces: 2

Step 6: Now In the Code section select the created event from the dropdown .



Code source [Info](#) Upload from

File Edit Find View Go Tools Window **Test** Deploy

Go to Anything (Ctrl-P)

Environment

KCS_Exp12 /

lambda_function.py

Execution results

No execution results

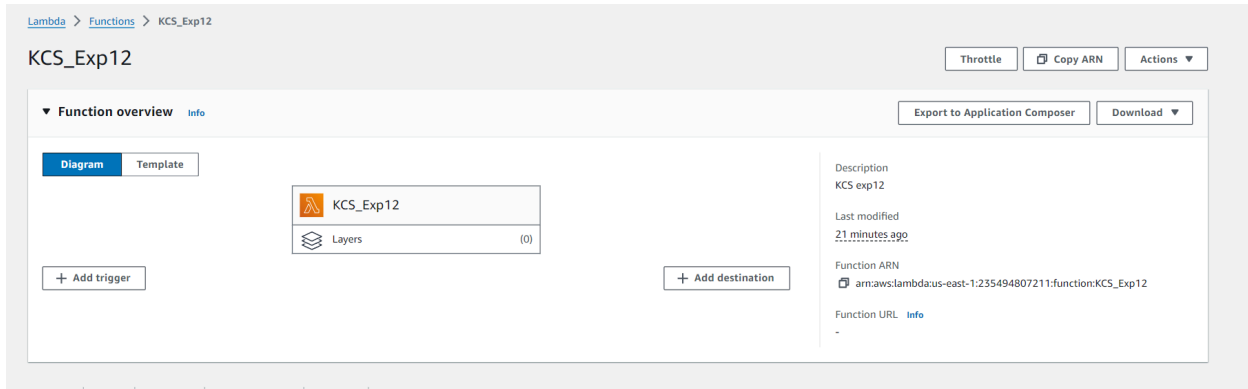
Configure test event Ctrl-Shift-C

Private saved events

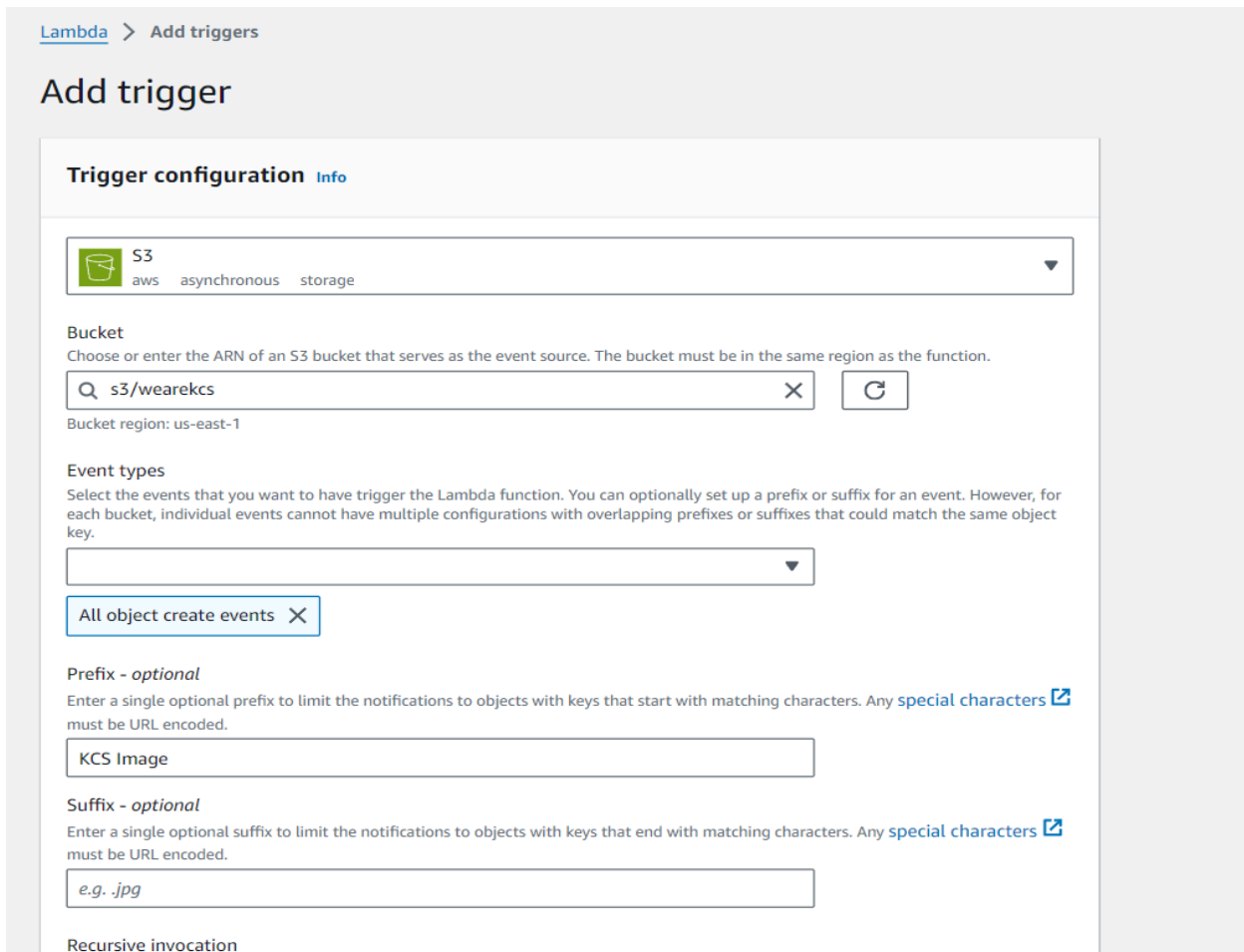
Event

KCS_Bucket

Step 7: Now In the Lambda function click on add trigger



Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to image.



KCS_Exp12

Throttle Copy ARN Actions

✓ The trigger wearekcs was successfully added to function KCS_Exp12. The function is now receiving events from the trigger.

Function overview Info

Export to Application Composer Download

Diagram Template

KCS_Exp12

Layers (0)

S3

+ Add trigger

+ Add destination

Description
KCS exp12

Last modified
26 minutes ago

Function ARN
arn:aws:lambda:us-east-1:235494807211:function:KCS_Exp12

Function URL Info

Code Test Monitor Configuration Aliases Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

Monitoring and operations tools

Concurrency and recursion detection

Asynchronous invocation

Code signing

File systems

State machines

Triggers (1) Info

Find triggers

Trigger

S3: wearekcs
arn:aws:s3:::wearekcs

Details

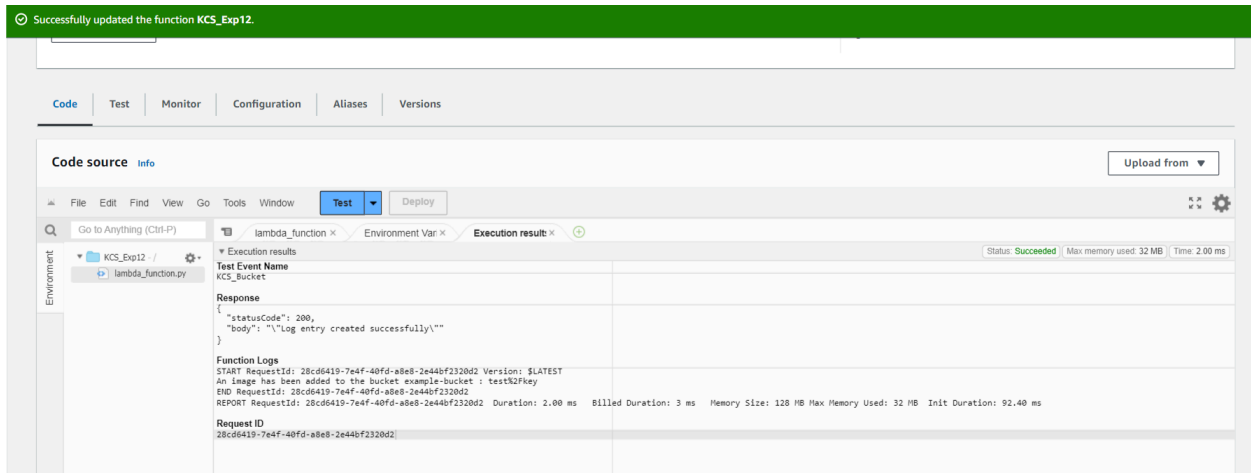
Fix errors Edit Delete Add trigger

Step 8: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.



The screenshot shows the AWS Lambda console interface for the function `KCS_Exp12`. The `Code` tab is selected, displaying the source code for `lambda_function.py`. The code is a Python lambda handler that processes an event containing a record with a bucket name and an object key. It prints a message and returns a 200 status code with a success message in the response body.

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO Implement
5     bucket_name = event['Records'][0]['s3']['bucket']['name']
6     object_key = event['Records'][0]['s3']['object']['key']
7     print(f"An image has been added to the bucket {bucket_name} : {object_key}")
8     return {
9         'statusCode': 200,
10        'body': json.dumps('Log entry created successfully')}
11
12
13
14
```



The screenshot shows the AWS Lambda console interface for the function `KCS_Exp12`, displaying the `Execution results` tab. The status is `Succeeded`. The response shows a 200 status code and a success message. The function logs show the start and end of the execution, including the request ID and duration.

Test Event Name: KCS_Bucket

Response:

```
{
  "statusCode": 200,
  "body": "\"Log entry created successfully\""
}
```

Function Logs:

```
START RequestId: 28c06419-7e4f-40fd-a8e8-2e44bf2320d2 Version: $LATEST
An image has been added to the bucket example-bucket : test12fkey
END RequestId: 28c06419-7e4f-40fd-a8e8-2e44bf2320d2
REPORT RequestId: 28c06419-7e4f-40fd-a8e8-2e44bf2320d2 Duration: 2.00 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 92.40 ms
```

Request ID: 28c06419-7e4f-40fd-a8e8-2e44bf2320d2

Step 9: Now upload any image to the bucket.

[Amazon S3](#) > [Buckets](#) > [wearekcs](#) > Upload

Upload [Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (1 Total, 957.0 KB)

[Remove](#)[Add files](#)[Add folder](#)

All files and folders in this table will be uploaded.

< 1 >

<input type="checkbox"/>	Name	Folder
<input type="checkbox"/>	F_iOUxsXgAAXB2s.jpg	-

Destination [Info](#)

Destination

[s3://wearekcs](#)

► Destination details

Bucket settings that impact new objects stored in the specified destination.

► Permissions

Grant public access and access to other AWS accounts.

► Properties

Specify storage class, encryption settings, tags, and more.

[Cancel](#)[Upload](#)

Upload succeeded
View details below.

Upload: status Close

The information below will no longer be available after you navigate away from this page.

Summary

Destination s3://wearekcs	Succeeded 1 file, 957.0 KB (100.00%)	Failed 0 files, 0 B (0%)
------------------------------	---	-----------------------------

Files and folders (1 Total, 957.0 KB)

Find by name

Name	Folder	Type	Size	Status	Error
F_I0UhsXpA...	-	image/jpeg	957.0 KB	Succeeded	-

Step 10: Now to click on test in lambda to check whether it is giving log when image is added to S3

Code Test Monitor Configuration Aliases Versions

Code source info Upload from

File Edit Find View Go Tools Window Test Deploy

Go to Anything (Ctrl-P)

Environment

- KCS_Exp12 /
- lambda_function.py

Execution results

Test Event Name
KCS_Bucket

Response

```
{
  "statusCode": 200,
  "body": "\"Log entry created successfully\""
}
```

Function Logs

```
START RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831 Version: $LATEST
An image has been added to the bucket example-bucket : test92fkey
END RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831
REPORT RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831 Duration: 1.88 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID
ba624cc5-6862-4d62-84ca-6a1bf867d831

Step 11: Now Lets see the log on Cloud watch.To see it go to monitor section and then click on view cloudwatch logs.

CloudWatch > Log groups > /aws/lambda/KCS_Exp12 > 2024/10/01/[\$LATEST]b93d5fc4cf3b4bf8802c5b106ff03bde

Log events

Actions

Start tailing

Create metric filter

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

Clear1m30m1h12hCustomUTC timezoneDisplay

	Timestamp	Message
		No older events at this moment. Retry
	2024-10-01T08:55:09.068Z	INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:188d9ca2e2714ff5637bd2bbe...
	2024-10-01T08:55:09.163Z	START RequestId: 28cd6419-7e4f-40fd-a8e8-2e44bf2320d2 Version: \$LATEST
	2024-10-01T08:55:09.164Z	An image has been added to the bucket example-bucket : test%ZFkey
	2024-10-01T08:55:09.174Z	END RequestId: 28cd6419-7e4f-40fd-a8e8-2e44bf2320d2
	2024-10-01T08:55:09.174Z	REPORT RequestId: 28cd6419-7e4f-40fd-a8e8-2e44bf2320d2 Duration: 2.00 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memor...
	2024-10-01T08:59:18.675Z	START RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831 Version: \$LATEST
	2024-10-01T08:59:18.676Z	An image has been added to the bucket example-bucket : test%ZFkey
	2024-10-01T08:59:18.678Z	END RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831
	2024-10-01T08:59:18.678Z	REPORT RequestId: ba624cc5-6862-4d62-84ca-6a1bf867d831 Duration: 1.88 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memor...
		No newer events at this moment. Auto retry paused. Resume