

Experiment 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:

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

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 shows the Amazon S3 console interface. On the left is a navigation sidebar with options like Buckets, Access Grants, and Storage Lens. The main area is titled 'Amazon S3' and shows an 'Account snapshot' banner. Below this, the 'General purpose buckets' tab is selected, displaying a list of four buckets. The table has columns for Name, AWS Region, IAM Access Analyzer, and Creation date. The 'Create bucket' button is highlighted in orange in the top right of the bucket list area.

Name	AWS Region	IAM Access Analyzer	Creation date
codepipeline-eu-north-1-823007647292	Europe (Stockholm) eu-north-1	View analyzer for eu-north-1	August 8, 2024, 23:54:38 (UTC+05:30)
codepipeline-us-east-1-934567252759	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 11, 2024, 22:46:59 (UTC+05:30)
elasticbeanstalk-eu-north-1-010928205712	Europe (Stockholm) eu-north-1	View analyzer for eu-north-1	August 9, 2024, 00:03:29 (UTC+05:30)
elasticbeanstalk-us-east-1-010928205712	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 11, 2024, 20:15:18 (UTC+05:30)

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.

The screenshot shows the 'Create bucket' wizard in the AWS S3 console. The 'General configuration' section is active, showing the 'General purpose' bucket type selected. The bucket name 'lab12_bucket48' is entered in the 'Bucket name' field. The 'Object Ownership' section shows 'ACLs disabled (recommended)' selected. The 'Copy settings from existing bucket' section is also visible.

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](#)
lab12_bucket48

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.

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

aws

Services

Search

[Alt+S]

N. Virginia

Bhushankor

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.

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

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 public

AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

- ☐ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

✔ **Successfully created bucket "lab12bucket48"**
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
Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

General purpose buckets | **Directory buckets**

General purpose buckets (2) Info All AWS Regions
Buckets are containers for data stored in S3.

Find buckets by name

	Name	AWS Region
<input type="radio"/>	elasticbeanstalk-us-east-1-017820672175	US East (N. Virginia) us-east-1
<input type="radio"/>	lab12bucket48	US East (N. Virginia) us-east-1

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

AWS Lambda lets you run code without thinking about servers.

You pay only for the compute time that you consume — there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service, all with zero administration.

Get started
Author a Lambda function from scratch, or choose from one of many preconfigured examples.
[Create a function](#)

How it works [Run](#) [Next: Lambda responds to events](#)

✔ "Hello from Lambda!"

.NET | Java | **Node.js** | Python | Ruby | Custom runtime

```
1 * exports.handler = async (event) => {  
2   console.log(event);  
3   return 'Hello from Lambda!';  
4 };  
5
```

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.

Create function [Info](#)

Choose one of the following options to create your function.

☒ **Author from scratch**
Start with a simple Hello World example.

☐ **Use a blueprint**
Build a Lambda application from sample code and configuration presets for common use cases.

☐ **Container image**
Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

lab12_48

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.12

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ 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

🔔 Successfully created the function lab12_48. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > lab12_48

lab12_48

Throttle Copy ARN Actions ▼

▼ Function overview [Info](#)

Export to Application Composer Download ▼

Diagram Template

lab12_48

Layers (0)

+ Add trigger

+ Add destination

Description -

Last modified 1 minute ago

Function ARN am:aws:lambda:us-east-1:017820672175:function:lab12_48

Function URL [Info](#)

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

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

General configuration [Info](#)

Edit

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout	SnapStart Info	
0 min 3 sec	None	

Edit basic settings

Basic settings [Info](#)

Description - optional

Memory [Info](#)

Your function is allocated CPU proportional to the memory configured.

128

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](#)

512

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

0

 min

1

 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/lab12_48-role-3mhrlau0

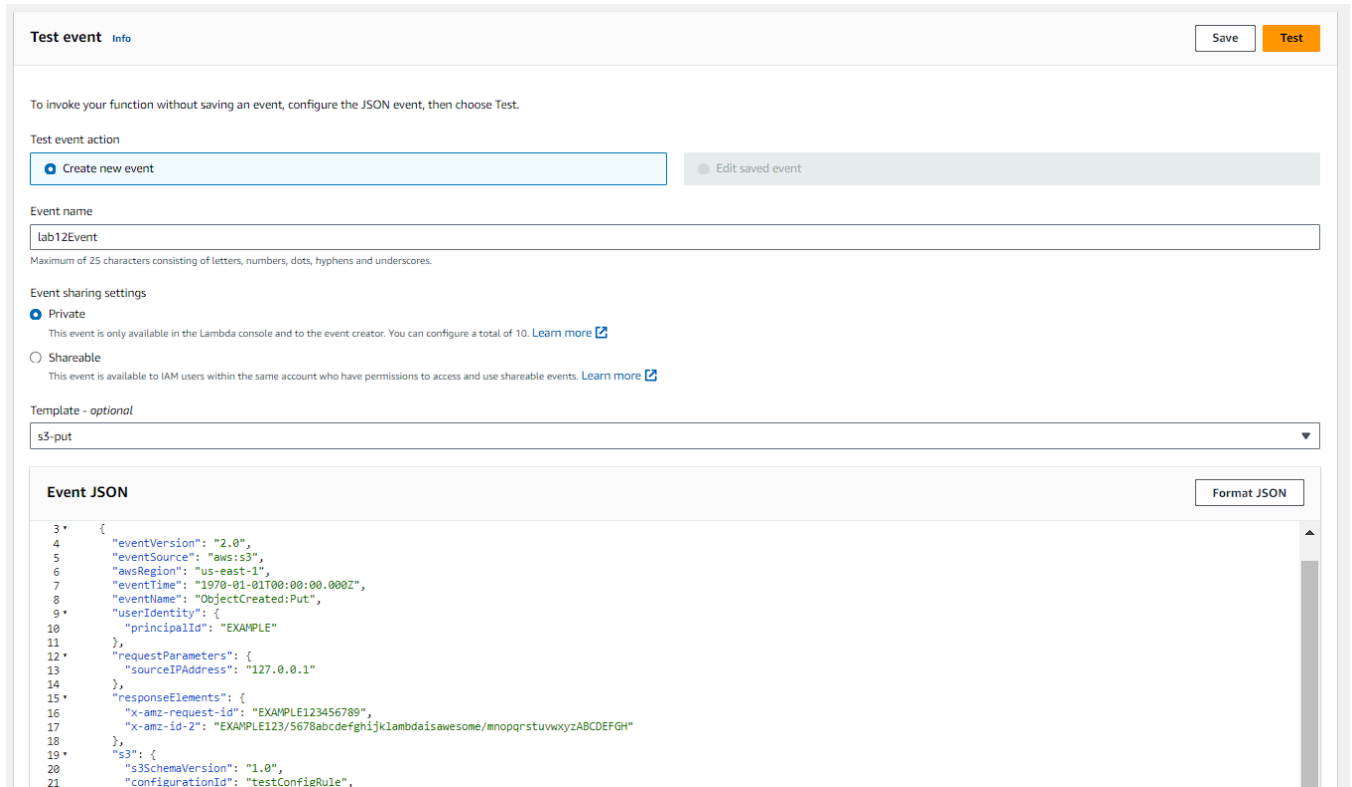
[View the lab12_48-role-3mhrlau0 role](#) on the IAM console.

Cancel

Save

Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

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

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

lab12Event

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](#)

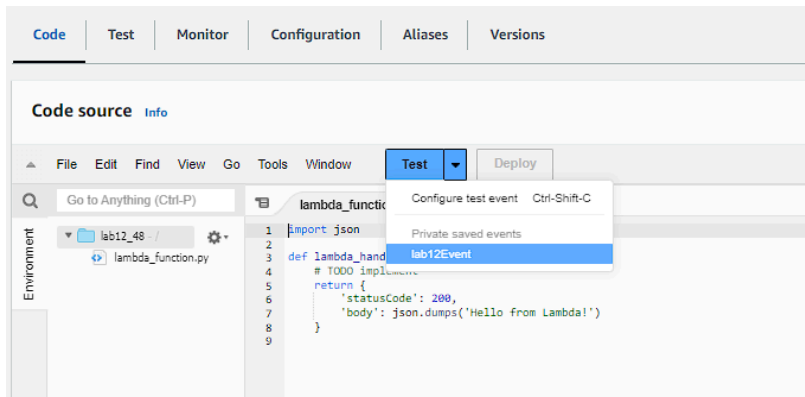
Template - optional

s3-put

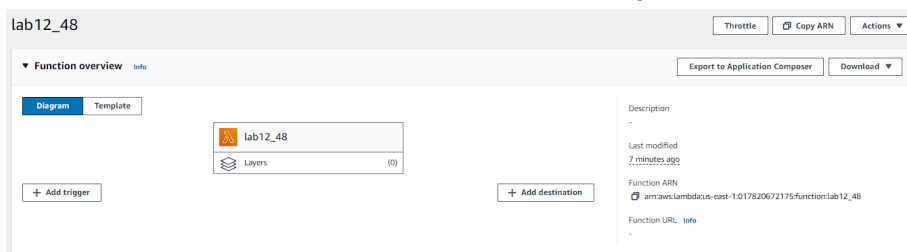
Event JSON Format JSON

```
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": "EXAMPLE123/5678abcdefgijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
18  },
19  "s3": {
20    "s3SchemaVersion": "1.0",
21    "configurationId": "testConfigRule",
```

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



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



Add trigger

Trigger configuration [Info](#)

**S3**

aws

asynchronous

storage



Bucket

Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.



Bucket region: us-east-1

Event types

Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events

Prefix - optional

Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any [special characters](#) must be URL encoded.

Suffix - optional

Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters. Any [special characters](#) must be URL encoded.

Recursive invocation

If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

- ☒ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

[Cancel](#)[Add](#)

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

Step 9: Now upload any image to the bucket.

```
1 import json
2
3 def lambda_handler(event, context):
4     # Extract the bucket name and object key from the S3 event
5     bucket_name = event['Records'][0]['s3']['bucket']['name']
6     object_key = event['Records'][0]['s3']['object']['key']
7
8     print(f"An image was uploaded to bucket {bucket_name}:{object_key}")
9
10    return {
11        'statusCode': 200,
12        'body': json.dumps('Log entry created successfully!')
13    }
14
```

Code | Test | Monitor | **Configuration** | Aliases | Versions

General configuration

Triggers

Permissions

Destinations


Function URL

Environment variables

Triggers (1) Info Fix errors Edit Delete Add trigger

Find triggers

Trigger

 **S3: lab12bucket48**
arn:aws:s3:::lab12bucket48

[Details](#)

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, 30.4 KB) Remove Add files Add folder

All files and folders in this table will be uploaded.

Find by name

< 1 >

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

Destination [Info](#)

Destination

<s3://lab12bucket48>

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: status Close

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

Summary

Destination s3://lab12bucket48	Succeeded 1 file, 30.4 KB (100.00%)	Failed 0 files, 0 B (0%)
-----------------------------------	--	-----------------------------

Files and folders | Configuration

Files and folders (1 Total, 30.4 KB)

Find by name

Name	Folder	Type	Size	Status	Error
about.jpg	-	image/jpeg	30.4 KB	Succeeded	-

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

Execution results Status: Succeeded Max memory used: 32 MB Time: 2.07 ms

Test Event Name
lab12Event

Response

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

Function Logs
START RequestId: d4539d35-bc4d-4539-99c0-a4475259b020 Version: \$LATEST
An image was uploaded to bucket example-bucket:test%2Fkey
END RequestId: d4539d35-bc4d-4539-99c0-a4475259b020
REPORT RequestId: d4539d35-bc4d-4539-99c0-a4475259b020 Duration: 2.07 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 95.17 ms

Request ID
d4539d35-bc4d-4539-99c0-a4475259b020

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/lab12_48 > 2024/10/07/[LATEST]f2914f94a7824006b9d673478b103e95

Log events Clear 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](#)

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

Timestamp	Message
	No older events at this moment. Retry
2024-10-07T16:44:34.740Z	INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:1880ca2e2714ff5637bd2bbe06ceb81ec3bc488a0f277dab304c14cd8140881
2024-10-07T16:44:34.838Z	START RequestId: d4539d35-bc4d-4539-99c0-a4475259b020 Version: \$LATEST
2024-10-07T16:44:34.839Z	An image was uploaded to bucket example-bucket:test%2Fkey
2024-10-07T16:44:34.841Z	END RequestId: d4539d35-bc4d-4539-99c0-a4475259b020
2024-10-07T16:44:34.841Z	REPORT RequestId: d4539d35-bc4d-4539-99c0-a4475259b020 Duration: 2.07 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 95.17 ms
	No newer events at this moment. Auto retry paused. Resume

Conclusion: In this experiment, we successfully created an AWS Lambda function that logs a message when an image is uploaded to an S3 bucket. It is important to note that we have to select S3-put template in event other wise code will give an error. The function was successfully triggered by S3 object uploads, validating the functionality of Lambda's event-driven architecture. This experiment demonstrated how Lambda can efficiently respond to S3 events and how to troubleshoot common issues with event structure.