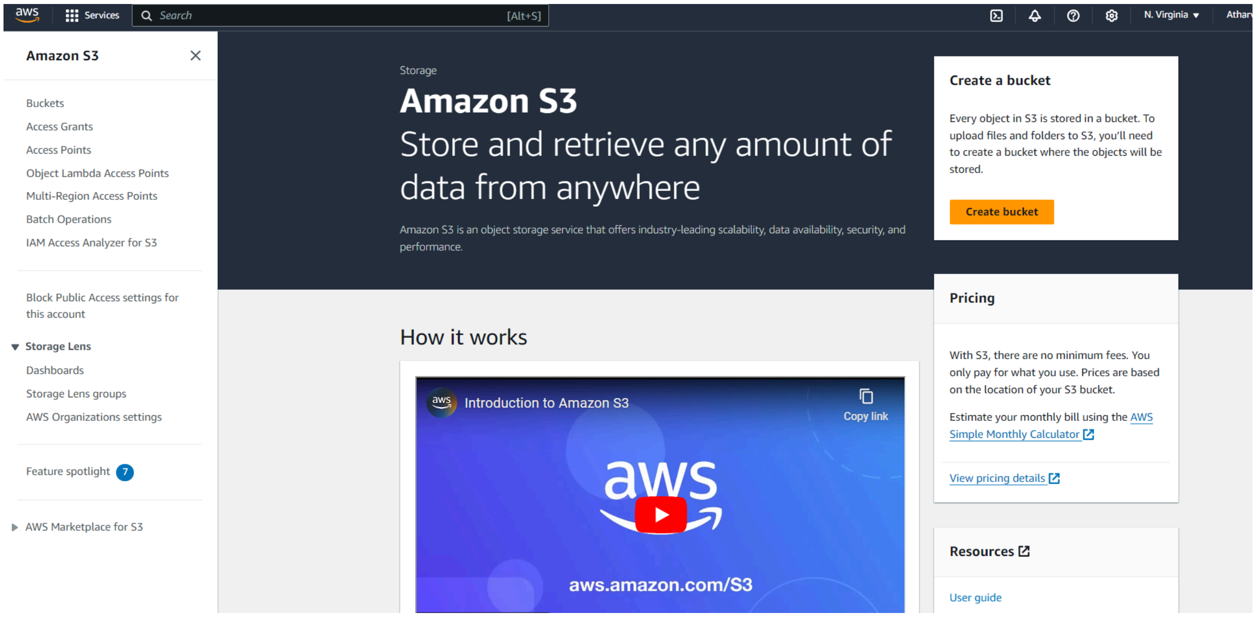
Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

Step 1: Create a s3 bucket. 1) Search for S3 bucket in the services search. Then click on create bucket.



Amazon S3 ✕

Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3

Block Public Access settings for this account

▼ Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings

Feature spotlight 7

▶ AWS Marketplace for S3

Storage

Amazon S3

Store and retrieve any amount of data from anywhere

Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance.

How it works

Create a bucket

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.

Create bucket

Pricing

With S3, there are no minimum fees. You only pay for what you use. Prices are based on the location of your S3 bucket.

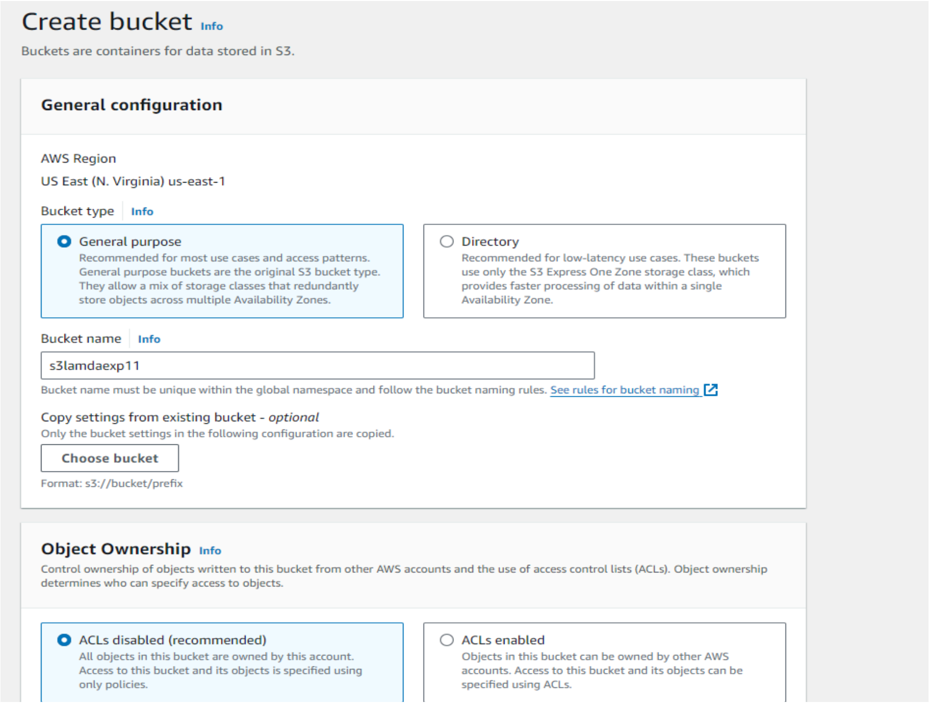
Estimate your monthly bill using the AWS Simple Monthly Calculator

View pricing details

Resources

User guide

2) Keep the bucket as a general purpose bucket. Give a name to your bucket.



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
s3lamdaexp11
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

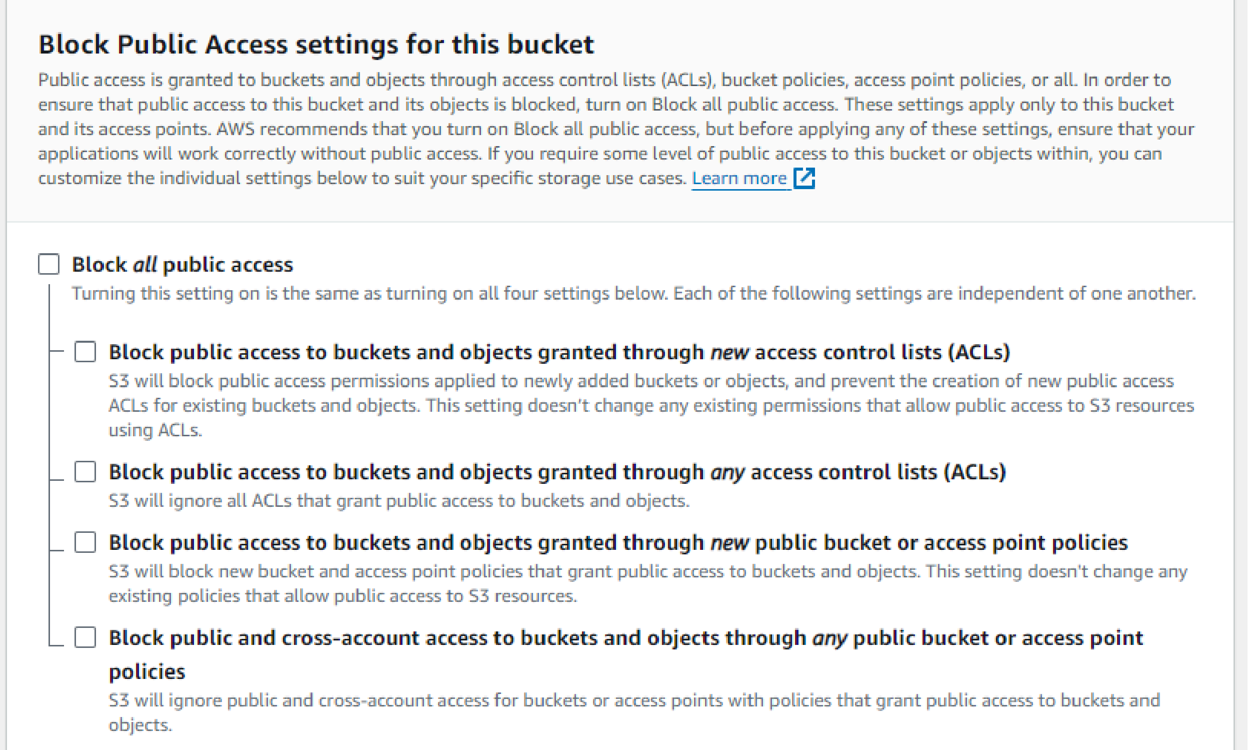
Object Ownership Info

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.

3) Uncheck block all public access



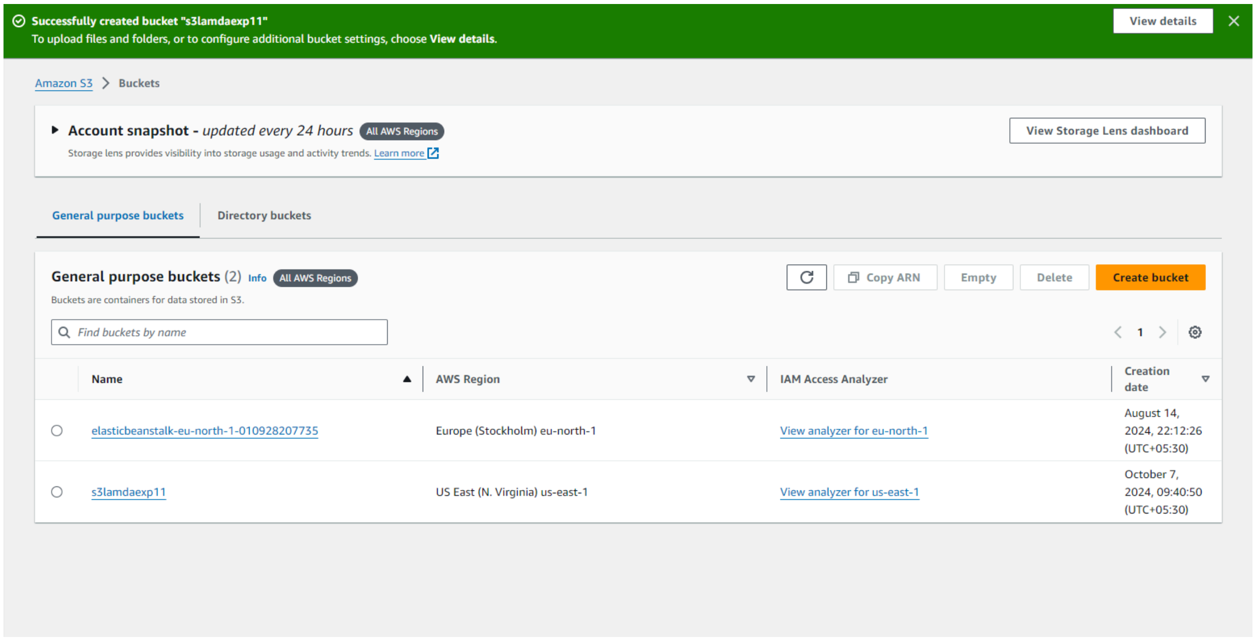
Block Public Access settings for this bucket

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.

4) Keeping all other options same, click on create. This would create your bucket. Now click on the name of the bucket



Successfully created bucket "s3lamdaexp11"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

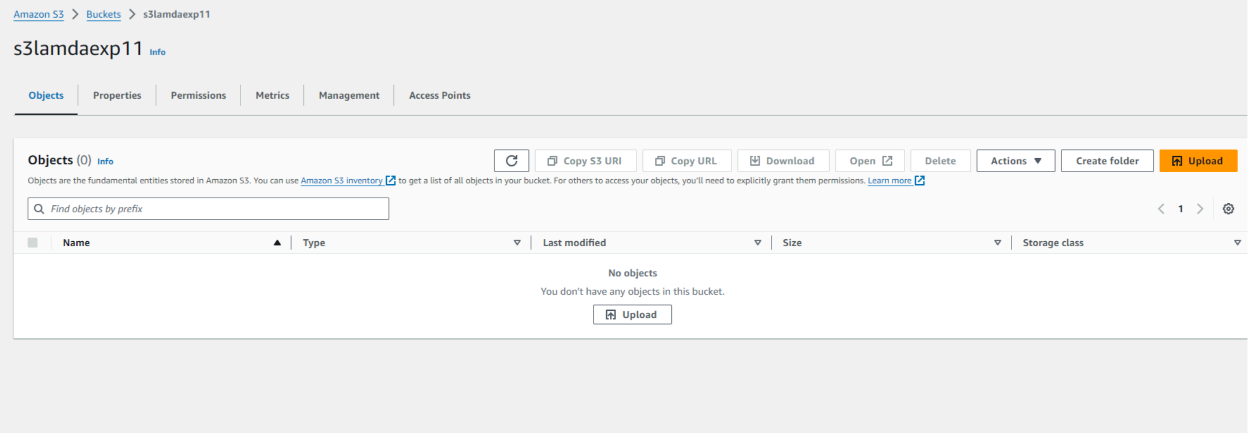
View details
Amazon S3 > Buckets
Account snapshot - *updated every 24 hours* All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets

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

Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1

	Name	AWS Region	IAM Access Analyzer	Creation date
○	elasticbeanstalk-eu-north-1-010928207735	Europe (Stockholm) eu-north-1	View analyzer for eu-north-1	August 14, 2024, 22:12:26 (UTC+05:30)
○	s3lamdaexp11	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 7, 2024, 09:40:50 (UTC+05:30)

5) Here, click on upload, then add files. Select any image that you want to upload in the bucket and click on upload



s3lamdaexp11

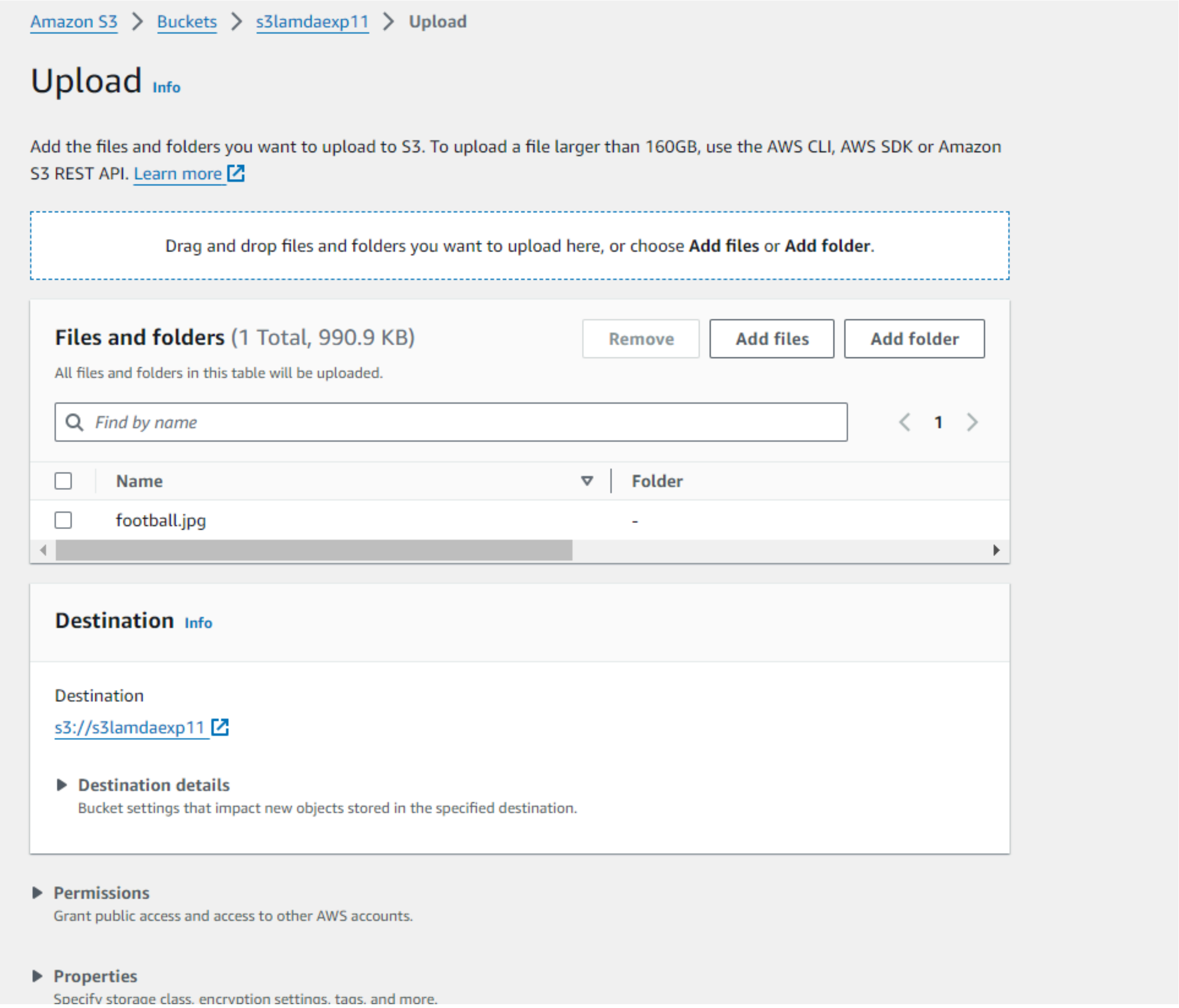
Objects (0)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name ▲	Type ▽	Last modified ▽	Size ▽	Storage class ▽

No objects

You don't have any objects in this bucket.



Upload

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.

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

Files and folders (1 Total, 990.9 KB)

All files and folders in this table will be uploaded.

□	Name ▽	Folder
□	football.jpg	-

Destination

Destination

s3://s3lamdaexp11

▶ Destination details

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

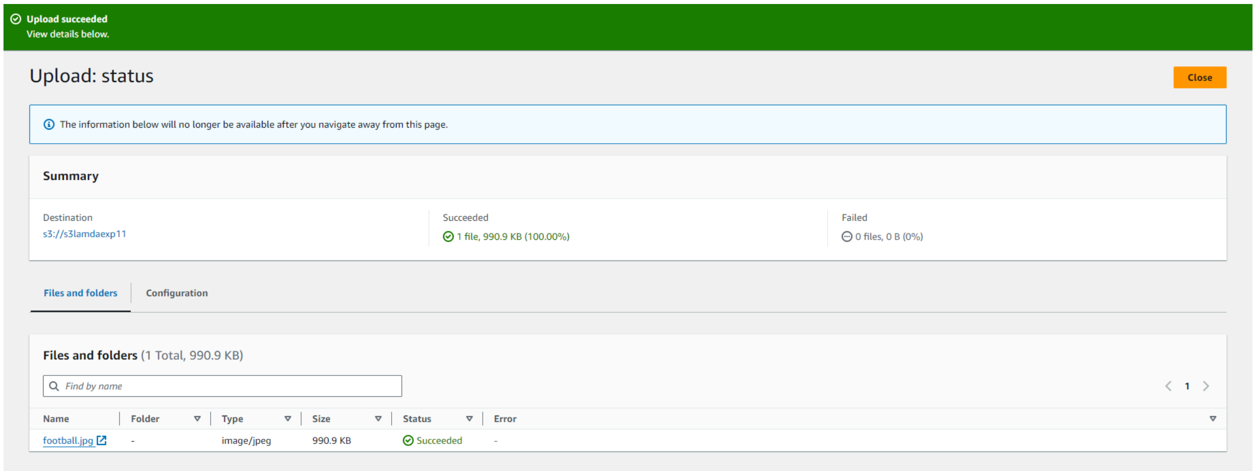
▶ Permissions

Grant public access and access to other AWS accounts.

▶ Properties

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

6) The image has been uploaded to the bucket



Upload succeeded
View details below.

Upload: status

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

Summary

Destination
s3://s3lamdaexp11

Succeeded
1 file, 990.9 KB (100.00%)

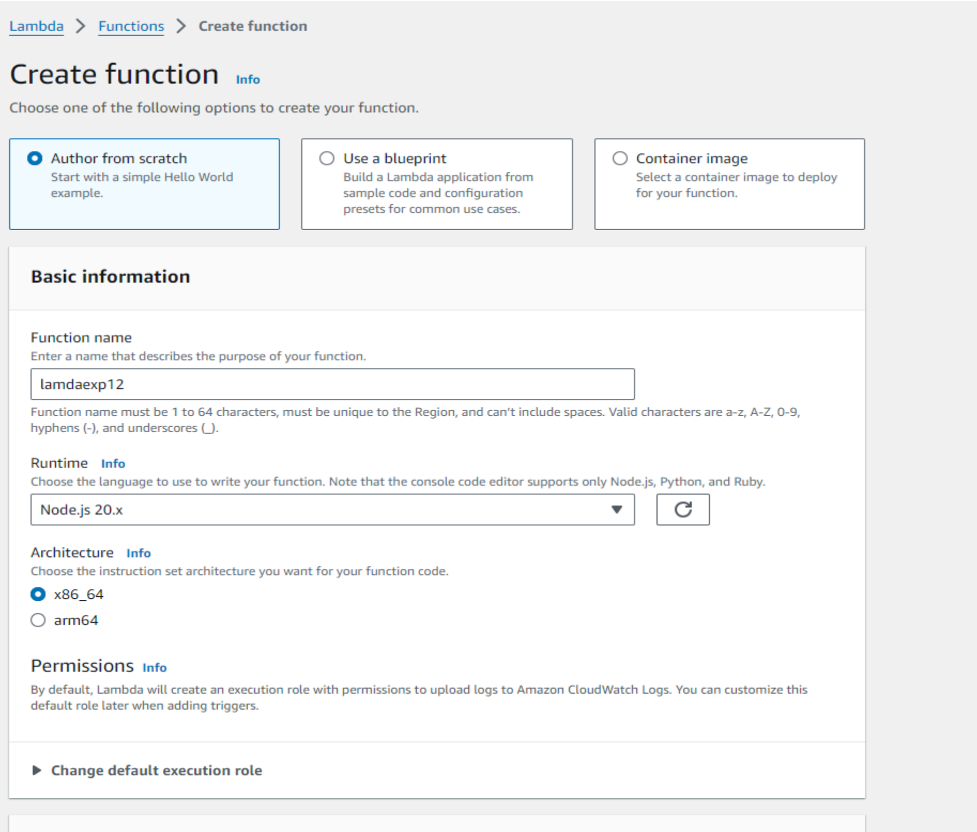
Failed
0 files, 0 B (0%)

Files and folders (1 Total, 990.9 KB)

Name	Folder	Type	Size	Status	Error
football.jpg	-	image/jpeg	990.9 KB	Succeeded	-

Step 2: Configure Lambda function

1) Go to the lambda function you had created berfor. (Services → Lambda → Click on name of function). Here, click on add trigger



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.
lamdaexp12
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 (_).

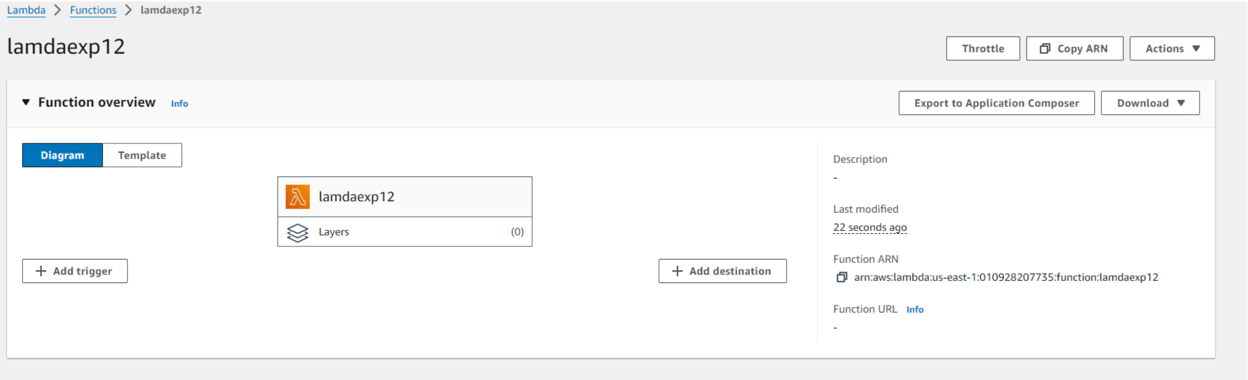
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Node.js 20.x

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

2) Under trigger configuration, search for S3 and select it.



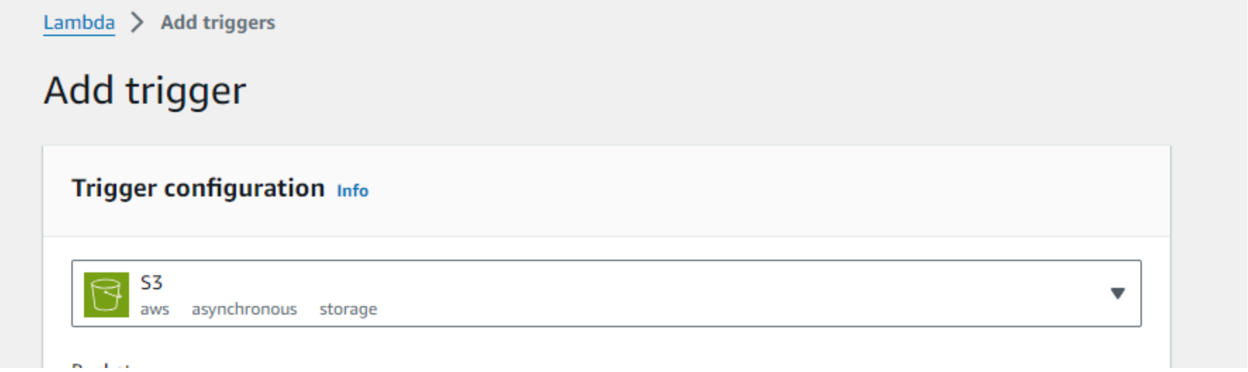
Description
-

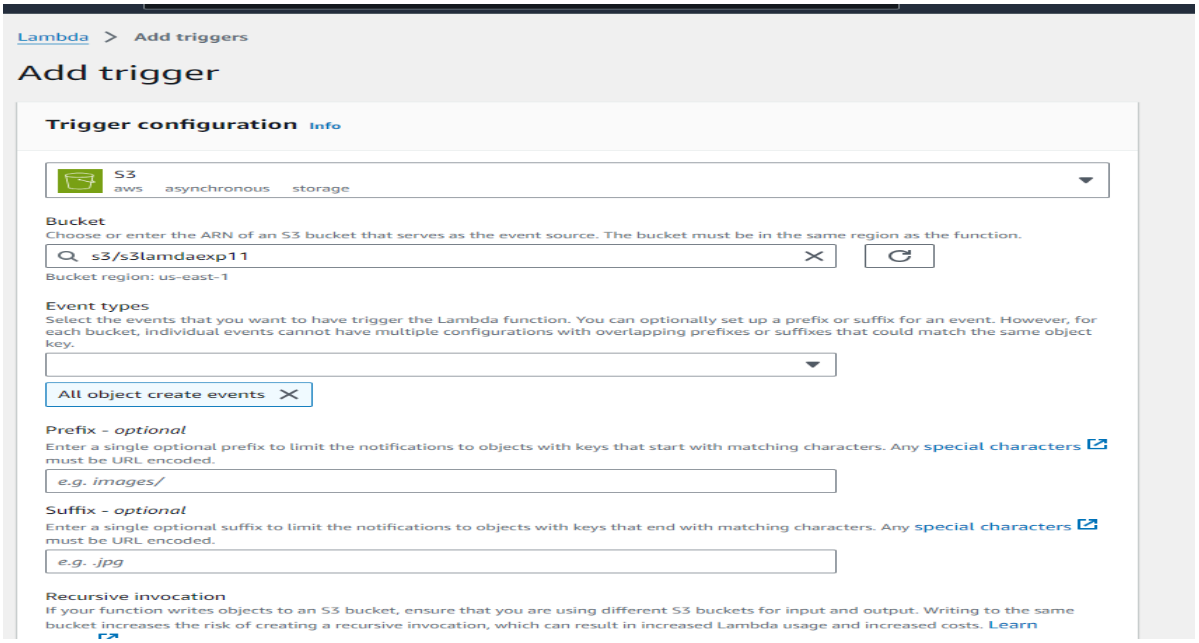
Last modified
22 seconds ago

Function ARN
arn:aws:lambda:us-east-1:010928207735:function:lamdaexp12

Function URL Info
-

3) Here, select teh S3 bucket you created for this experiment. Acknowledge the condition given by AWS. then click on Add. This will add the S3 bucket trigger to your function





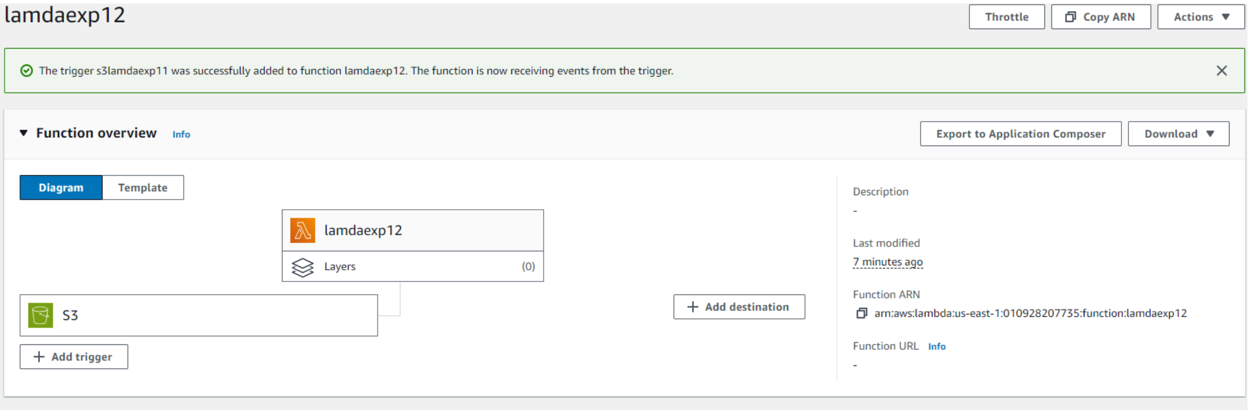
Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
s3/s3lamdaexp11
Bucket region: us-east-1

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.
e.g. images/

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.
e.g. .jpg

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



The trigger s3lamdaexp11 was successfully added to function lamdaexp12. The function is now receiving events from the trigger.

Description
-

Last modified
7 minutes ago

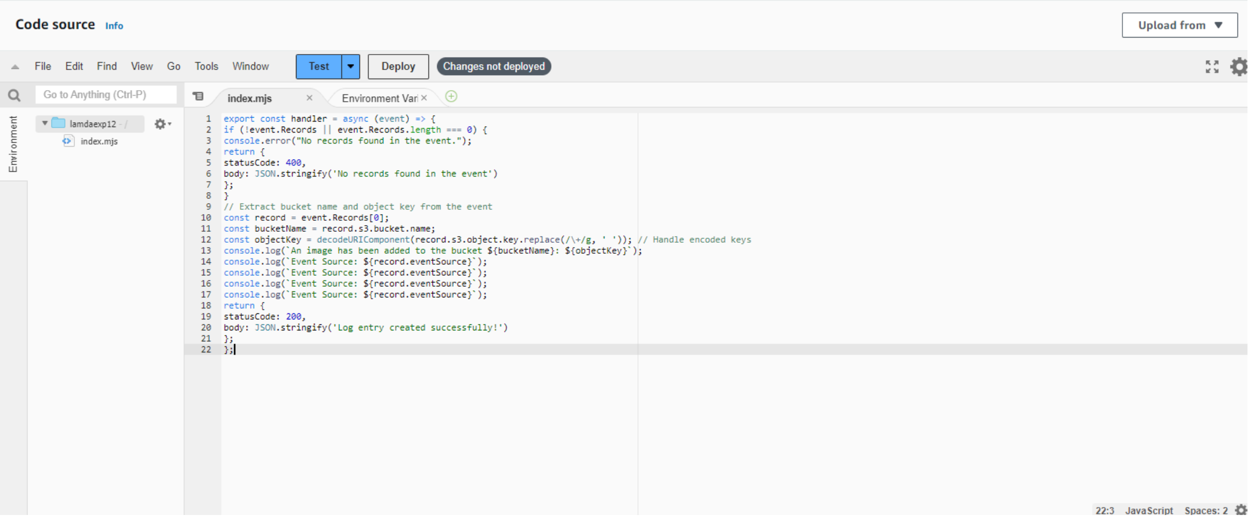
Function ARN
arn:aws:lambda:us-east-1:010928207735:function:lamdaexp12

Function URL Info
-

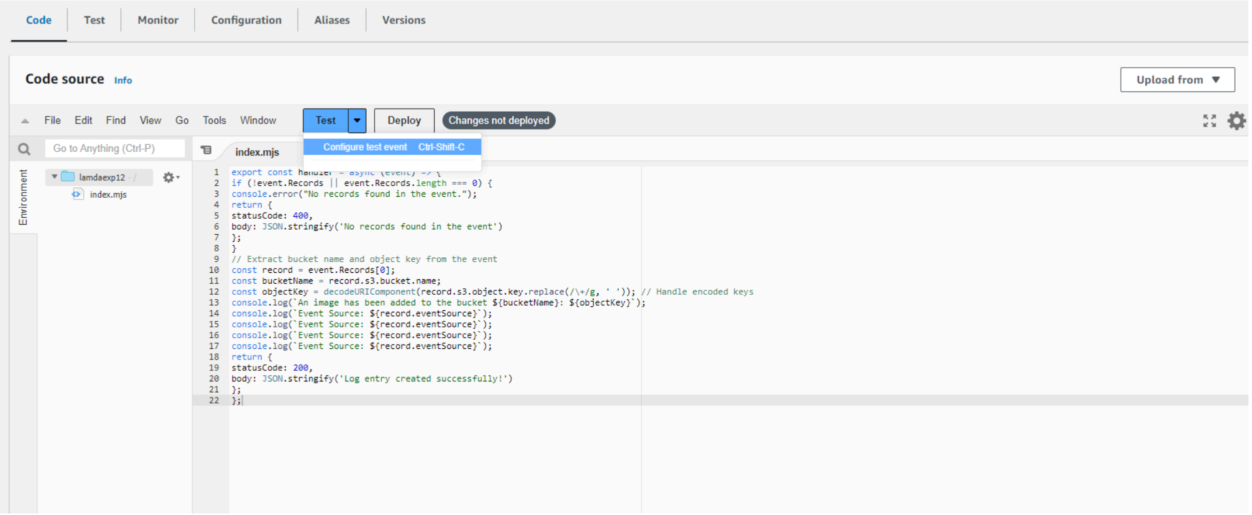
4) Scroll down to the code section of the function. Add the following javascript code to the code area by replacing the existing code

```
export const handler = async (event) => {
if (!event.Records || event.Records.length === 0) {
console.error("No records found in the event.");
return {
statusCode: 400,
body: JSON.stringify('No records found in the event')
};
}
// Extract bucket name and object key from the event
const record = event.Records[0];
const bucketName = record.s3.bucket.name;
const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys
console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
return {
statusCode: 200,
body: JSON.stringify('Log entry created successfully!')
};
};
```

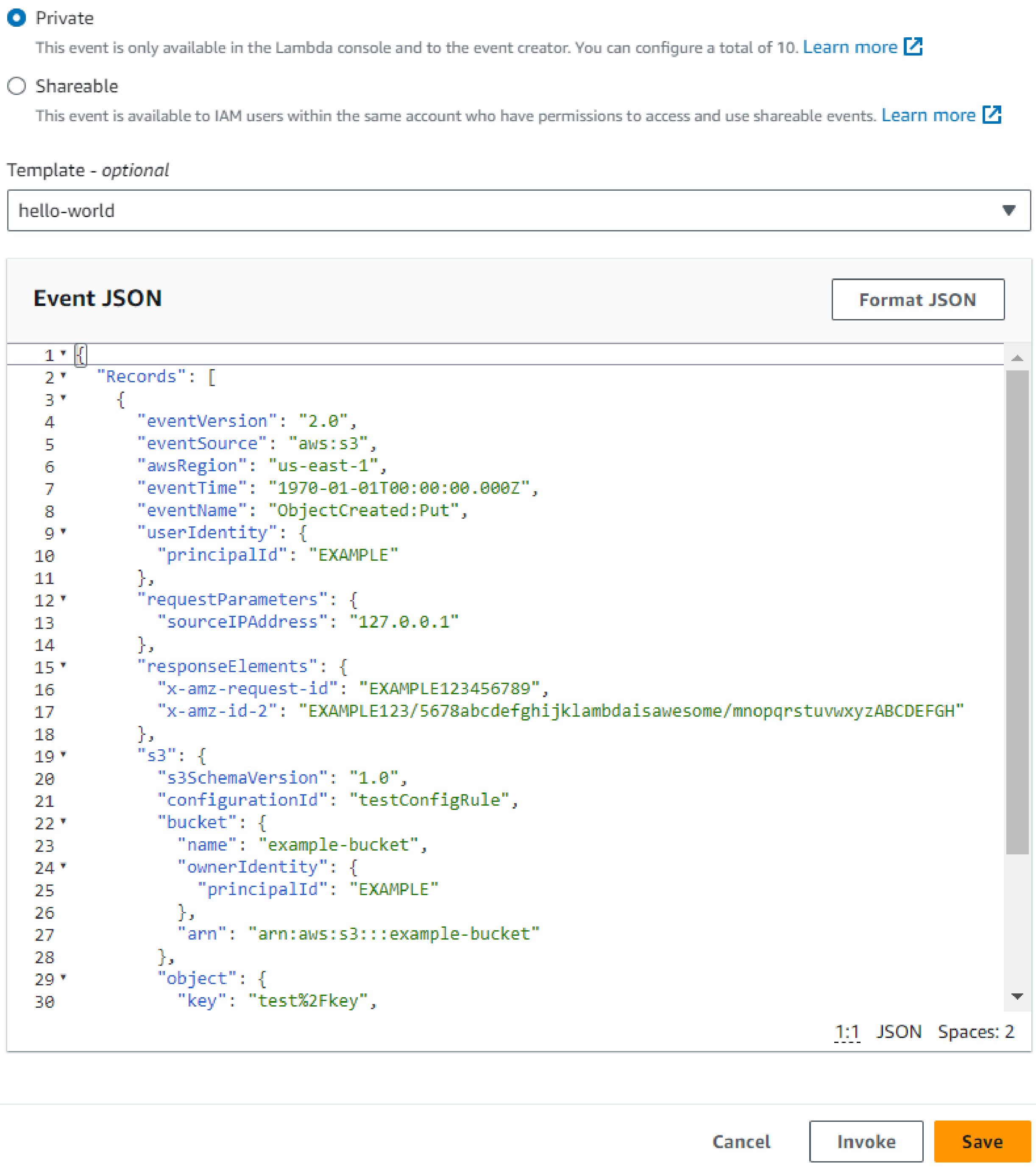
This JSON structure represents an S3 event notification triggered when an object is uploaded to an S3 bucket. It contains details about the event, including the bucket name (example-bucket), the object key (test/key), and metadata like the object's size, the event source (aws:s3), and the event time.



```
export const handler = async (event) => {
if (!event.Records || event.Records.length === 0) {
console.error("No records found in the event.");
return {
statusCode: 400,
body: JSON.stringify('No records found in the event')
};
}
// Extract bucket name and object key from the event
const record = event.Records[0];
const bucketName = record.s3.bucket.name;
const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys
console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
return {
statusCode: 200,
body: JSON.stringify('Log entry created successfully!')
};
};
```



```
export const handler = async (event) => {
if (!event.Records || event.Records.length === 0) {
console.error("No records found in the event.");
return {
statusCode: 400,
body: JSON.stringify('No records found in the event')
};
}
// Extract bucket name and object key from the event
const record = event.Records[0];
const bucketName = record.s3.bucket.name;
const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys
console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
console.log(`Event Source: ${record.eventSource}`);
return {
statusCode: 200,
body: JSON.stringify('Log entry created successfully!')
};
};
```



● 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

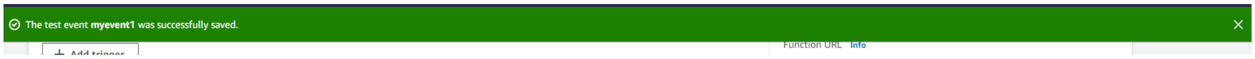
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more

Template - *optional*

hello-world

Event JSON

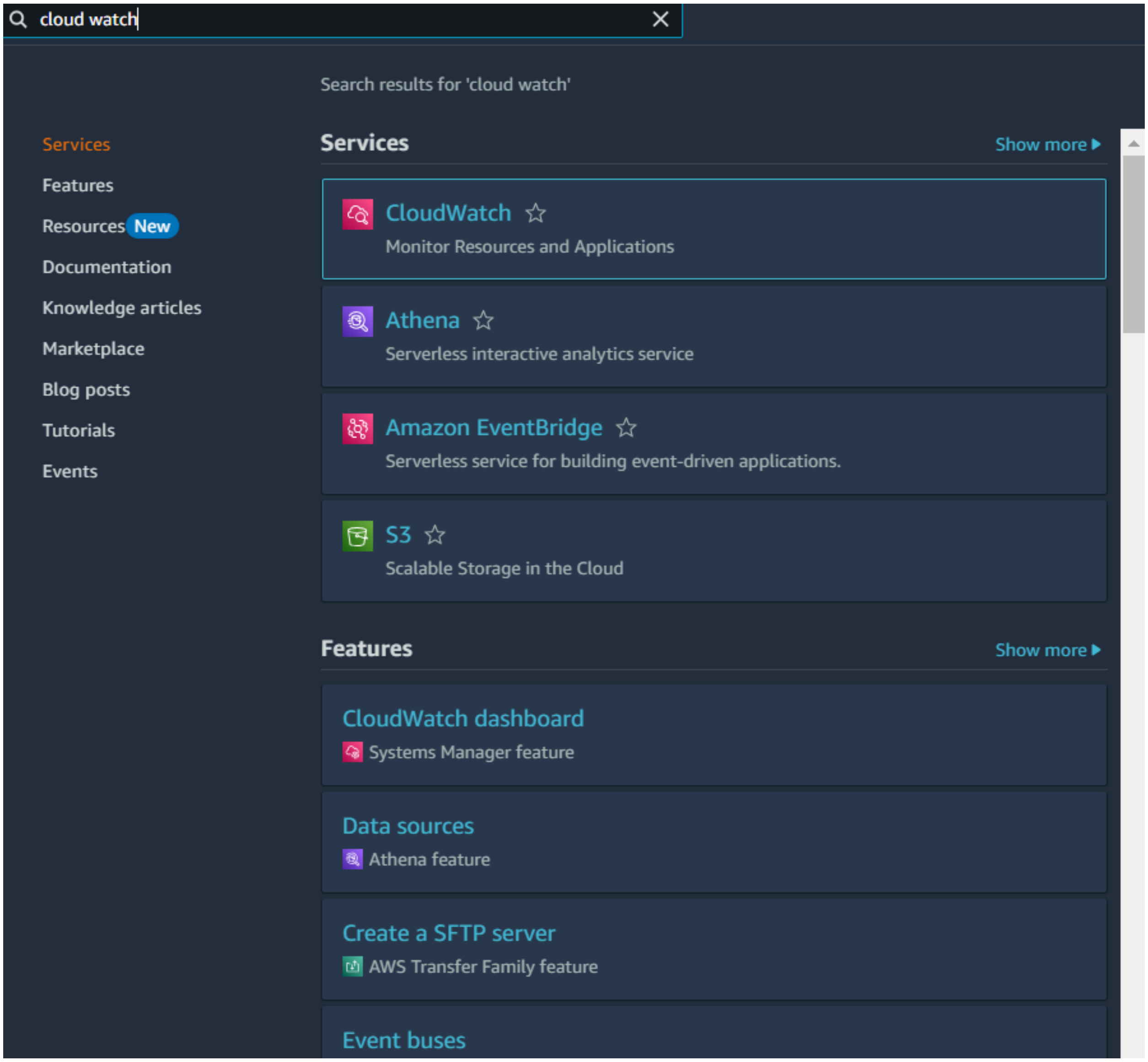
```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "example-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::example-bucket"
        },
        "object": {
          "key": "test%2Fkey",
```



The test event **myevent1** was successfully saved.

Step 3: Check the logs

1) To check the logs explicitly, search foe CloudWatch on services and open it in a new tab



cloud watch

Search results for 'cloud watch'

Services Show more ▸

CloudWatch ☆
Monitor Resources and Applications

Athena ☆
Serverless interactive analytics service

Amazon EventBridge ☆
Serverless service for building event-driven applications.

S3 ☆
Scalable Storage in the Cloud

Features Show more ▸

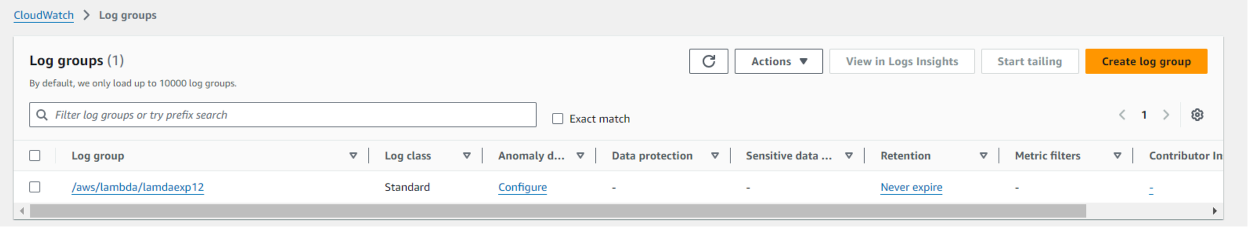
CloudWatch dashboard
Systems Manager feature

Data sources
Athena feature

Create a SFTP server
AWS Transfer Family feature

Event buses

2) Here, Click on Logs → Log Groups. Select the log that has the lambda function name you just ran.



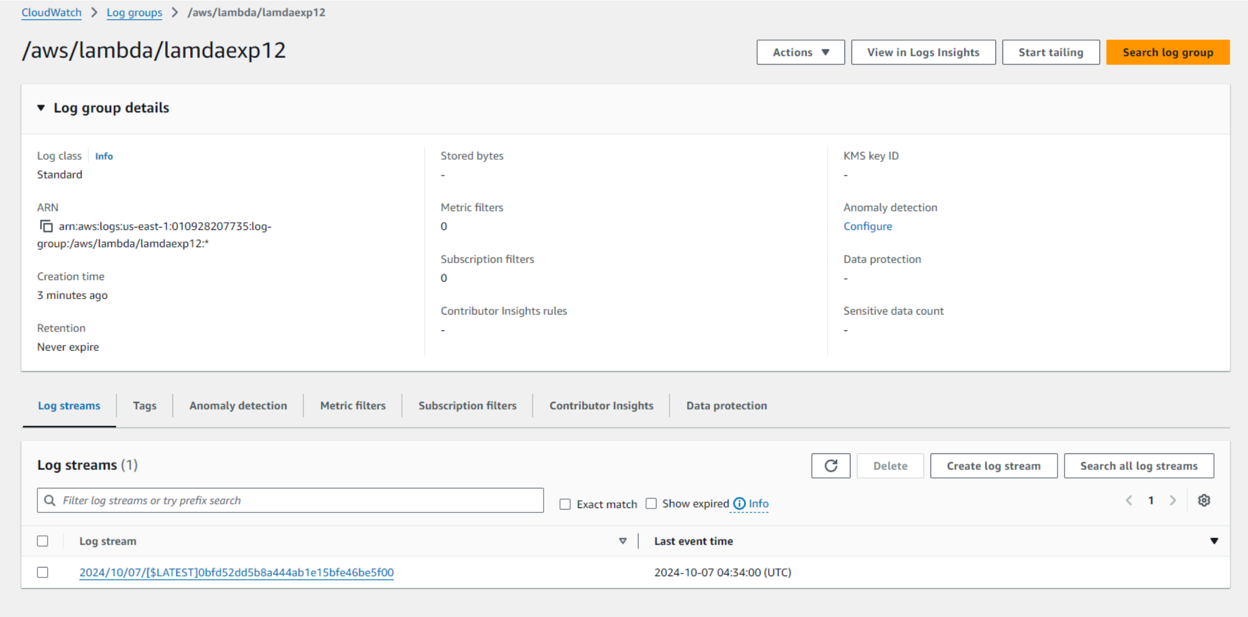
CloudWatch > Log groups

Log groups (1)

By default, we only load up to 10000 log groups.

□	Log group	Log class	Anomaly d...	Data protection	Sensitive data ...	Retention	Metric filters	Contributor In
□	/aws/lambda/lamdaexp12	Standard	Configure	-	-	Never expire	-	-

3) Here, under Log streams, select the log stream you want to check.



/aws/lambda/lamdaexp12

▼ Log group details

Log class Info
Standard

ARN
arn:aws:logs:us-east-1:010928207735:log-group:/aws/lambda/lamdaexp12:*

Creation time
3 minutes ago

Retention
Never expire

Stored bytes
-

Metric filters
0

Subscription filters
0

Contributor Insights rules
-

KMS key ID
-

Anomaly detection
Configure

Data protection
-

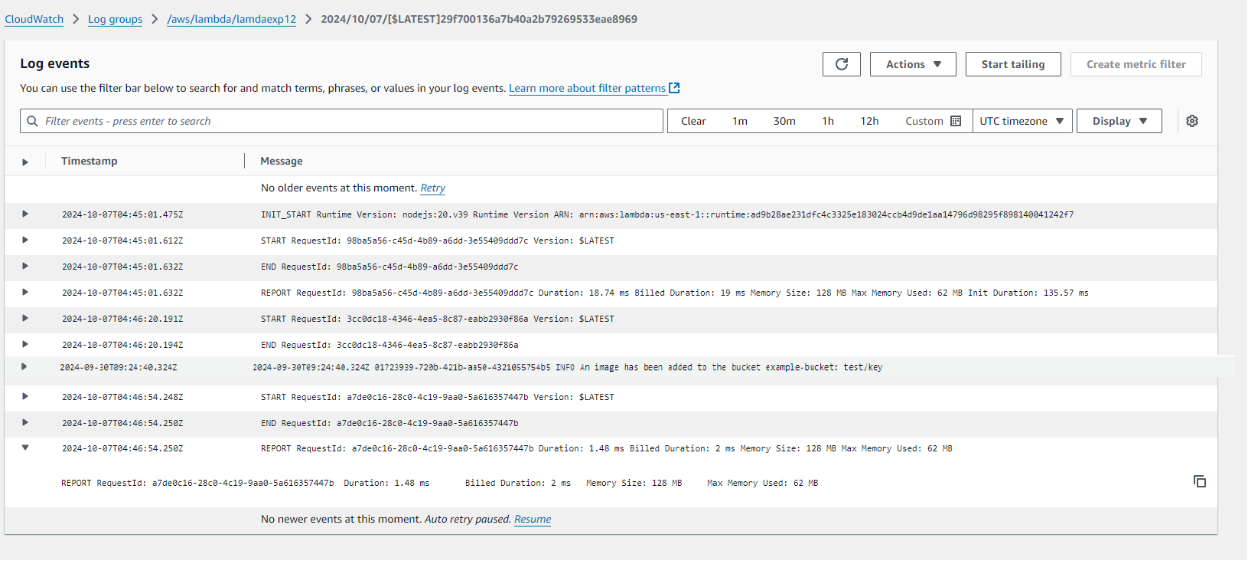
Sensitive data count
-

Log streams (1)

□ Exact match □ Show expired Info

□	Log stream	Last event time
□	2024/10/07/[$LATEST]0bfd52dd5b8a444ab1e15bfe46be5f00	2024-10-07 04:34:00 (UTC)

4) Here again, we can see that ‘An image has been added to the bucket’.


CloudWatch > Log groups > /aws/lambda/lamdaexp12 > 2024/10/07/[$LATEST]29f700136a7b40a2b79269533eae8969
Actions ▼
Start tailing
Create metric filter
Filter events - press enter to search
Clear
1m
30m
1h
12h
Custom
UTC timezone ▼
Display ▼

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

```
Timestamp                   Message
                            No older events at this moment. Retry
2024-10-07T04:45:01.475Z    INIT_START Runtime Version: nodejs:20.v39 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:ad9b28ae231dfc4c3325e183024ccb4d9de1aa14796d98295f898140041242f7
2024-10-07T04:45:01.612Z    START RequestId: 98ba5a56-c45d-4b89-a6dd-3e55409ddd7c Version: $LATEST
2024-10-07T04:45:01.632Z    END RequestId: 98ba5a56-c45d-4b89-a6dd-3e55409ddd7c
2024-10-07T04:45:01.632Z    REPORT RequestId: 98ba5a56-c45d-4b89-a6dd-3e55409ddd7c Duration: 18.74 ms Billed Duration: 19 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 135.57 ms
2024-10-07T04:46:20.191Z    START RequestId: 3cc0dc18-4346-4ea5-8c87-eabb2930f86a Version: $LATEST
2024-10-07T04:46:20.194Z    END RequestId: 3cc0dc18-4346-4ea5-8c87-eabb2930f86a
2024-09-30T09:24:40.324Z   2024-09-30T09:24:40.324Z 01723939-720b-421b-aa50-4321055754b5 INFO An image has been added to the bucket example-bucket: test/key
2024-10-07T04:46:54.248Z    START RequestId: a7de0c16-28c0-4c19-9aa0-5a616357447b Version: $LATEST
2024-10-07T04:46:54.250Z    END RequestId: a7de0c16-28c0-4c19-9aa0-5a616357447b
2024-10-07T04:46:54.250Z    REPORT RequestId: a7de0c16-28c0-4c19-9aa0-5a616357447b Duration: 1.48 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 62 MB

REPORT RequestId: a7de0c16-28c0-4c19-9aa0-5a616357447b  Duration: 1.48 ms      Billed Duration: 2 ms   Memory Size: 128 MB     Max Memory Used: 62 MB

                            No newer events at this moment. Auto retry paused. Resume
```

Conclusion: In this experiment, we successfully created a Lambda function that logs a message when an image is added to a specific S3 bucket. By configuring an S3 bucket trigger for the Lambda function, we demonstrated how AWS services can work together to automate tasks. The function logged important details about the event, including the bucket name and object key. After deploying the function and testing with a sample event, we verified the logs in CloudWatch, confirming that the Lambda function correctly detected and logged the addition of an image to the bucket. This experiment highlighted the seamless integration between AWS Lambda and S3 for event-driven processes.