

EXPERIMENT NO. 12

DATE -

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 -

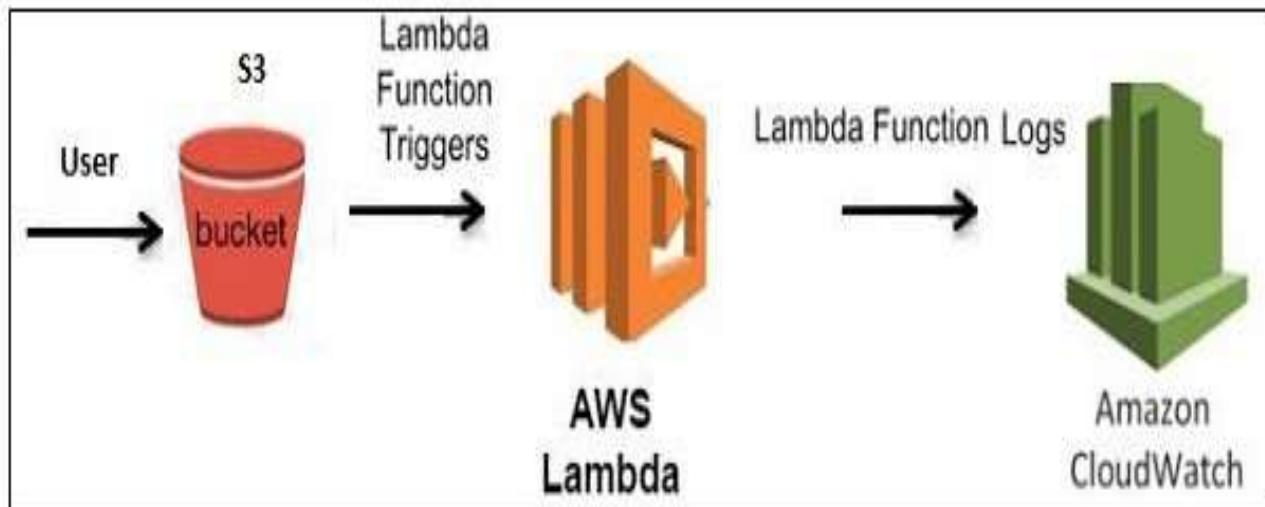
Amazon S3 service is used for file storage, where you can upload or remove files. We can trigger AWS Lambda on S3 when there are any file uploads in S3 buckets. AWS Lambda has a handler function which acts as a start point for AWS Lambda function. The handler has the details of the events. In this chapter, let us see how to use AWS S3 to trigger AWS Lambda function when we upload files in S3 bucket.

Steps for Using AWS Lambda Function with Amazon S3

To start using AWS Lambda with Amazon S3, we need the following –

Create S3 Bucket

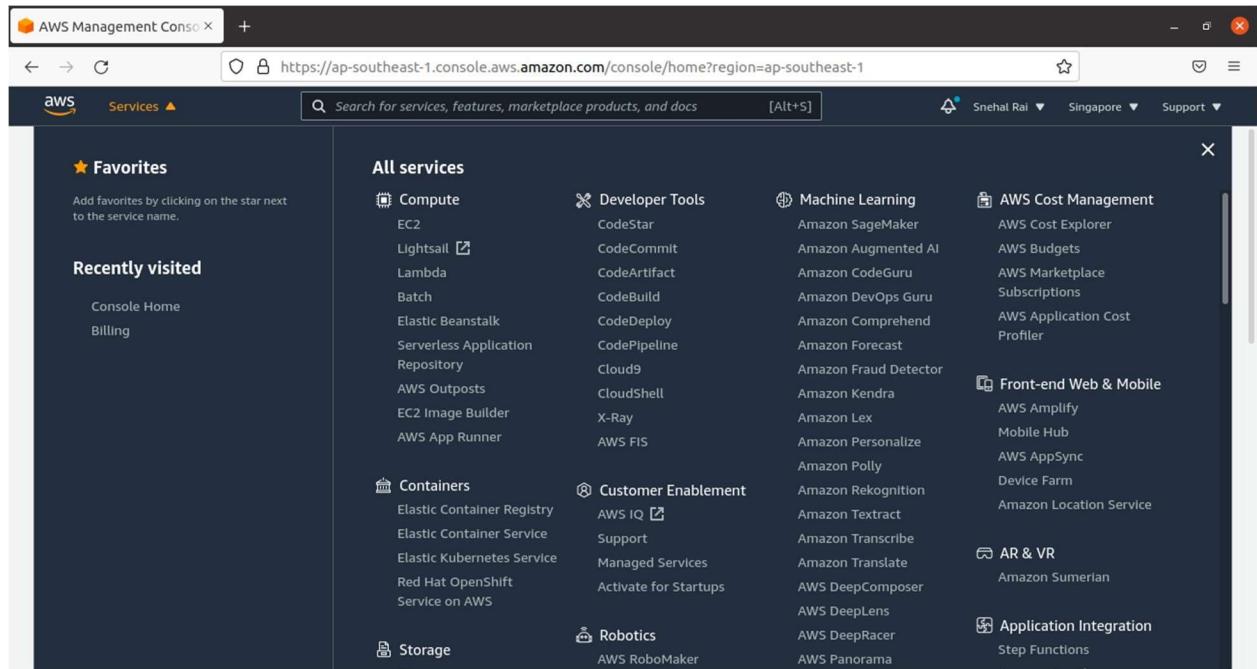
Create role which has permission to work with s3 and lambda
Create lambda function and add s3 as the trigger.



Creating S3 Bucket

Step 1

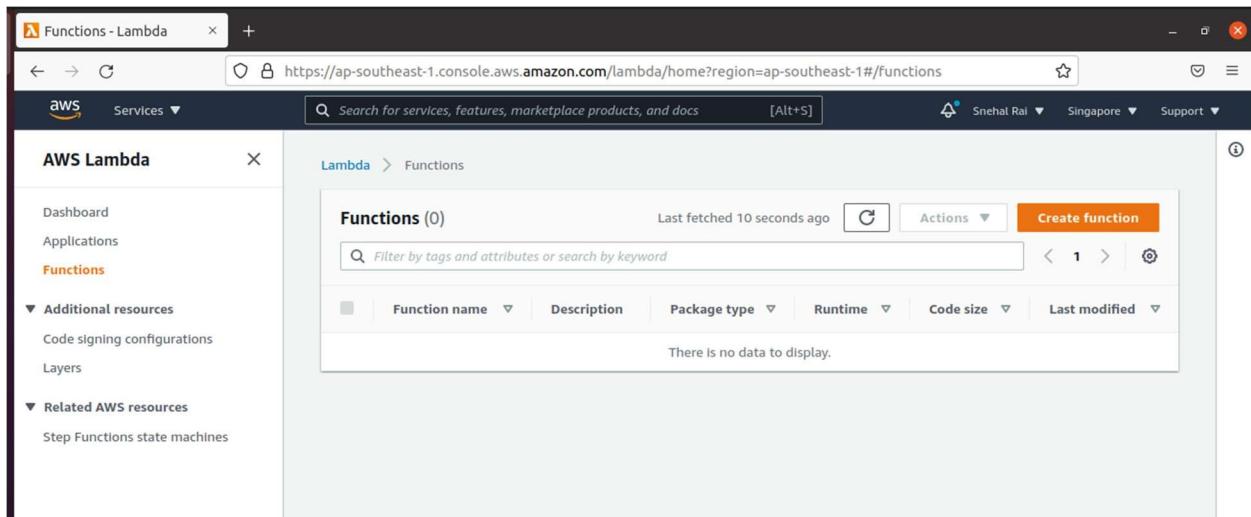
Go to Amazon services and click S3 in storage section as highlighted in the image given below –



The screenshot shows the AWS Management Console homepage. On the left, there's a sidebar with 'Favorites' and 'Recently visited' sections. The main area is titled 'All services' and contains a grid of service icons and names. The 'Storage' section, which includes S3, is located in the bottom-left quadrant of the grid.

Step 2

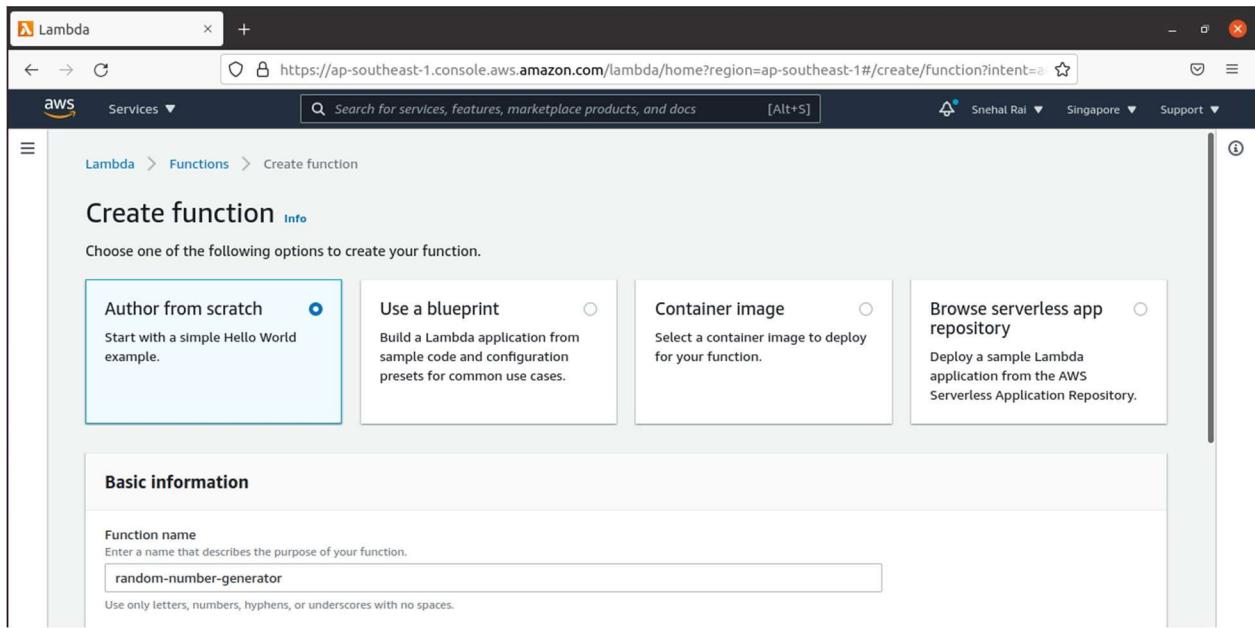
Click S3 storage and Create bucket which will store the files uploaded.



The screenshot shows the AWS Lambda console. The left sidebar has 'AWS Lambda' selected, with 'Functions' highlighted. The main area is titled 'Functions (0)' and shows a table with columns for Function name, Description, Package type, Runtime, Code size, and Last modified. A message at the bottom says 'There is no data to display.'

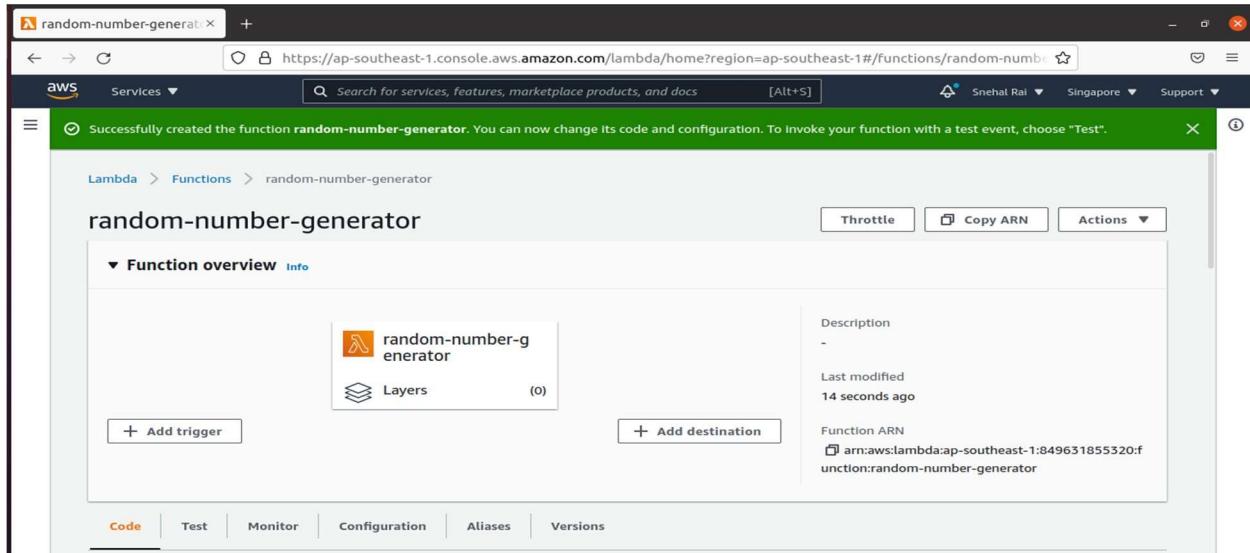
Step 3

Once you click Create bucket button, you can see a screen as follows –



Step 4

Enter the details Bucket name, Select the Region and click Create button at the bottom left side. Thus, we have created bucket with name : working with lambda and s3.



Step 5

Now, click the bucket name and it will ask you to upload files as shown below –

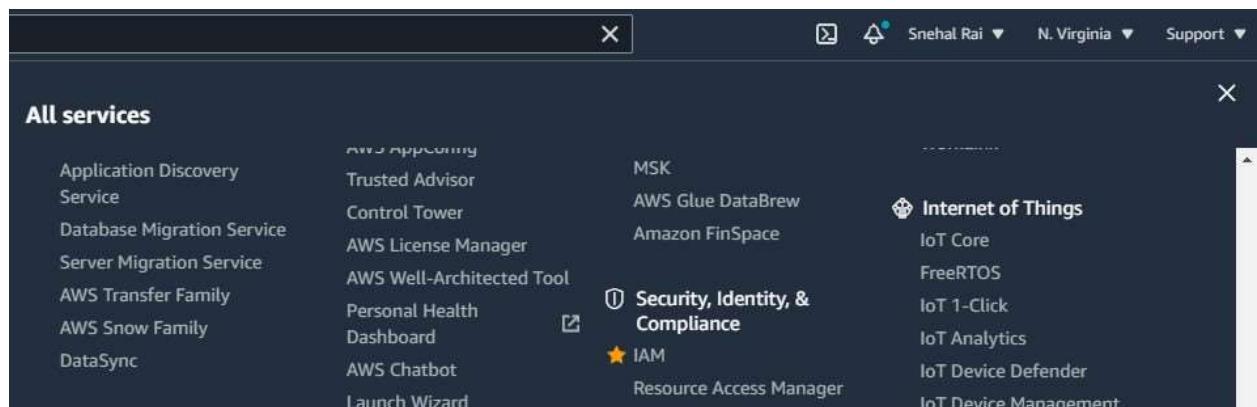


Thus, we are done with bucket creation in S3.

Create Role that Works with S3 and Lambda

To create role that works with S3 and Lambda, please follow the Steps given below – Step 1

Go to AWS services and select IAM as shown below –



Step 2

Now, click IAM -> Roles as shown below –

| Create role | | | Delete role | | | |
|--|--|---------------------|-------------------|--|--|--|
| <input type="text"/> Search | | | Showing 5 results | | | |
| Role name | Description | Trusted entities | | | | |
| <input type="checkbox"/> lambdaapipolicy | Allows Lambda functions to call AWS services | AWS service: lambda | | | | |

Step 3

Now, click Create role and choose the services that will use this role. Select Lambda and click Permission button.

| Choose the service that will use this role | | | | |
|---|-------------------|---------------------------|------------------|-----------------|
| EC2 Allows EC2 instances to call AWS services on your behalf. | | | | |
| Lambda Allows Lambda functions to call AWS services on your behalf. | | | | |
| API Gateway | Config | Elastic Beanstalk | Lambda | SNS |
| AppSync | DMS | Elastic Container Service | Lex | SWF |
| Application Auto Scaling | Data Pipeline | Elastic Transcoder | Machine Learning | SageMaker |
| Auto Scaling | DeepLens | Elastic Load Balancing | MediaConvert | Service Catalog |
| Batch | Directory Service | Glue | OpsWorks | Step Functions |
| CloudFormation | DynamoDB | Greengrass | RDS | Storage Gateway |
| CloudHSM | EC2 | GuardDuty | Redshift | |
| CloudWatch Events | EC2 - Fleet | Inspector | Rekognition | |

* Required

[Cancel](#)

[Next: Permissions](#)

Step 4

Add the permission from below and click Review.

| Create policy | | |
|---|-------------|---|
| <input type="button" value="Create policy"/> <input type="button" value="Refresh"/> | | |
| Filter: Policy type: <input type="button" value="Search"/> Showing 392 results | | |
| Policy name | Attachments | Description |
| <input type="checkbox"/> AdministratorAccess | 0 | Provides full access to AWS services and resources. |
| <input type="checkbox"/> AlexaForBusinessDeviceSetup | 0 | Provide device setup access to AlexaForBusiness services. |
| <input type="checkbox"/> AlexaForBusinessFullAccess | 0 | Grants full access to AlexaForBusiness resources and acc... |
| <input type="checkbox"/> AlexaForBusinessGatewayExecution | 0 | Provide gateway execution access to AlexaForBusiness s... |
| <input type="checkbox"/> AlexaForBusinessReadOnlyAccess | 0 | Provide read only access to AlexaForBusiness services. |
| <input type="checkbox"/> AmazonAPIGatewayAdministrator | 1 | Provides full access to create/edit/delete APIs in Amazon ... |
| <input type="checkbox"/> AmazonAPIGatewayInvokeFullAccess | 2 | Provides full access to invoke APIs in Amazon API Gateway. |
| <input type="checkbox"/> AmazonAPIGatewayPushToCloudWatchLogs | 0 | Allows API Gateway to push logs to user's account. |
| <input type="checkbox"/> AmazonAppStreamFullAccess | 0 | Provides full access to Amazon AppStream via the AWS ... |
| <input type="checkbox"/> AmazonAppStreamReadOnlyAccess | 0 | Provides read only access to Amazon AppStream via the ... |
| <input type="checkbox"/> AmazonAppStreamServiceAccess | 0 | Default policy for Amazon AppStream service role. |

* Required:

[Cancel](#)

[Previous](#)

[Next: Review](#)

Step 5

Observe that we have chosen the following permissions –

Create role

Review

Provide the required information below and review this role before you create it.

| | |
|--|---|
| Role name* | <input type="text"/> |
| Use alphanumeric and '-_, @-' characters. Maximum 64 characters. | |
| Role description | Allows Lambda functions to call AWS services on your behalf. |
| Maximum 1000 characters. Use alphanumeric and '-_, @-' characters. | |
| Trusted entities | AWS service: lambda.amazonaws.com |
| Policies | <input checked="" type="checkbox"/> AmazonS3FullAccess  <input checked="" type="checkbox"/> AWSLambdaFullAccess  <input checked="" type="checkbox"/> CloudWatchFullAccess  |

Observe that the Policies that we have selected are AmazonS3FullAccess, AWSLambdaFullAccess and CloudWatchFullAccess.

Step 6

Now, enter the Role name, Role description and click Create Role button at the bottom.

| Create role | | | Delete role |
|---|---|----------------------------|--------------------|
| <input type="text"/> Search | | | |
| Role name | Description | Trusted entities | |
| <input type="checkbox"/> lambdaapipolicy | Allows Lambda functions to call AWS services... | AWS service: lambda | |
| <input type="checkbox"/> lambdapolicyjava | Allows Lambda functions to call AWS services... | AWS service: lambda | |
| <input type="checkbox"/> lambdawithdynamodb | Allows Lambda functions to call AWS services... | AWS service: lambda | |
| <input type="checkbox"/> lambdawiths3 | Allows Lambda functions to call AWS services... | AWS service: lambda | |
| <input type="checkbox"/> lambdawiths3service | Allows Lambda functions to call AWS services... | AWS service: lambda | |
| <input type="checkbox"/> roleforlambdatesting | Allows Lambda functions to call AWS services... | AWS service: lambda | |

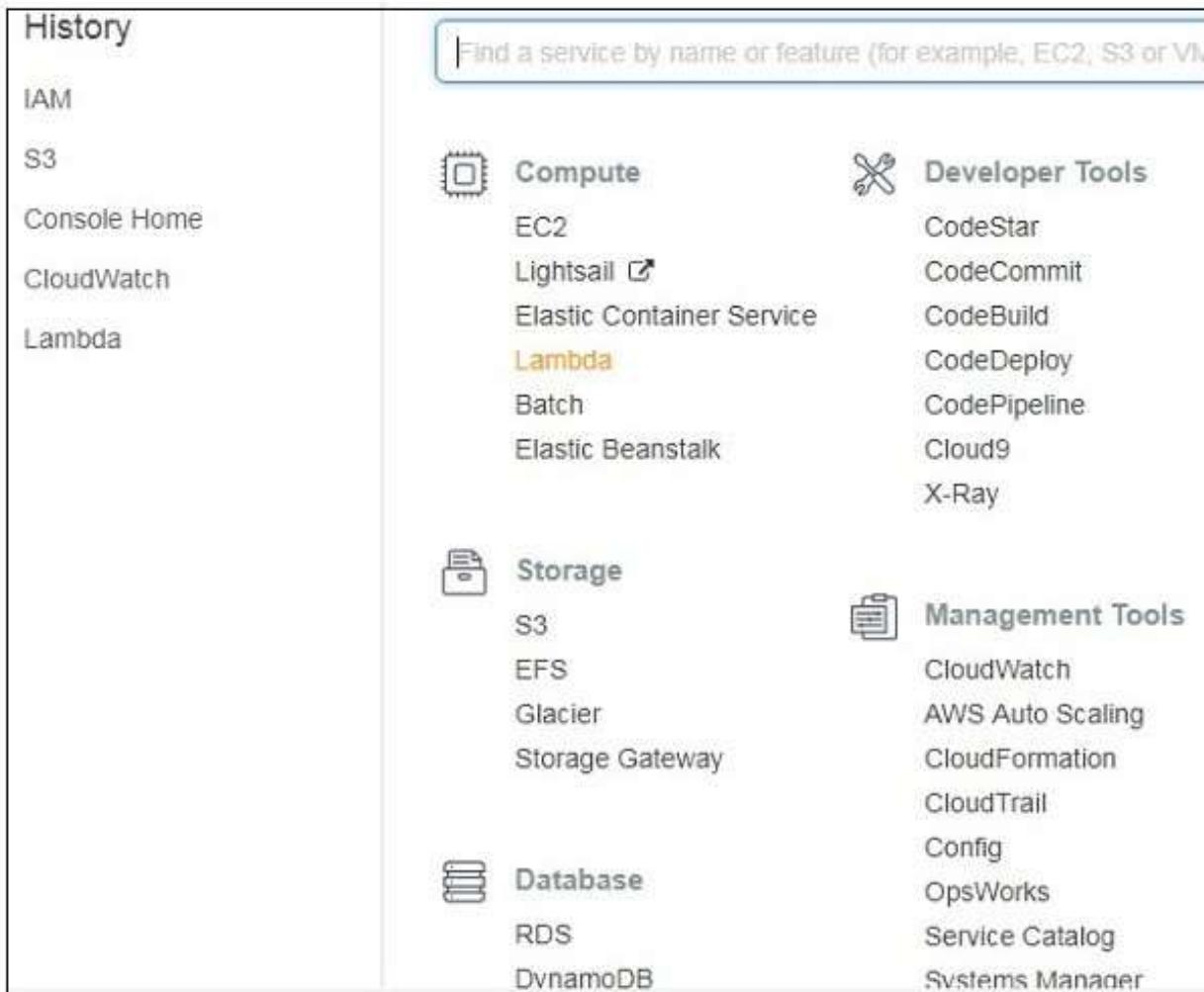
Thus, our role named lambdawiths3service is created.

Create Lambda function and Add S3 Trigger

In this section, let us see how to create a Lambda function and add a S3 trigger to it. For this purpose, you will have to follow the Steps given below –

Step 1

Go to AWS Services and select Lambda as shown below –



Step 2

Click Lambda and follow the process for adding Name. Choose the Runtime, Role etc. and create the function. The Lambda function that we have created is shown in the screenshot below –



Step 3

Now let us add the S3 trigger.

Step 4

Choose the trigger from above and add the details as shown below –

Configure triggers

Bucket

Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

workingwithlambdaands3



Event type

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.

Object Created (All)



Prefix

Enter an optional prefix to limit the notifications to objects with keys that start with matching characters.

e.g. images/

Filter pattern

Enter an optional filter pattern.

e.g. jpg

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Enable trigger

Enable the trigger now, or create it in a disabled state for testing (recommended).

Cancel

Add

Step 5

Select the bucket created from bucket dropdown. The event type has following details

Event type

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.

Object Created (All)



Object Created (All)



Object Created (All)

PUT

POST

COPY

Complete Multipart Upload

Object Removed (All)

Object Removed (All)

DELETE

Delete Marker Created

Reduced Redundancy Lost Object

Select Object Created (All), as we need AWS Lambda trigger when file is uploaded, removed etc.

Step 6

You can add Prefix and File pattern which are used to filter the files added. For Example, to trigger lambda only for .jpg images. Let us keep it blank for now as we need to trigger Lambda for all files uploaded. Click Add button to add the trigger.

Configure triggers

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

Event type
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.

Prefix
Enter an optional prefix to limit the notifications to objects with keys that start with matching characters.

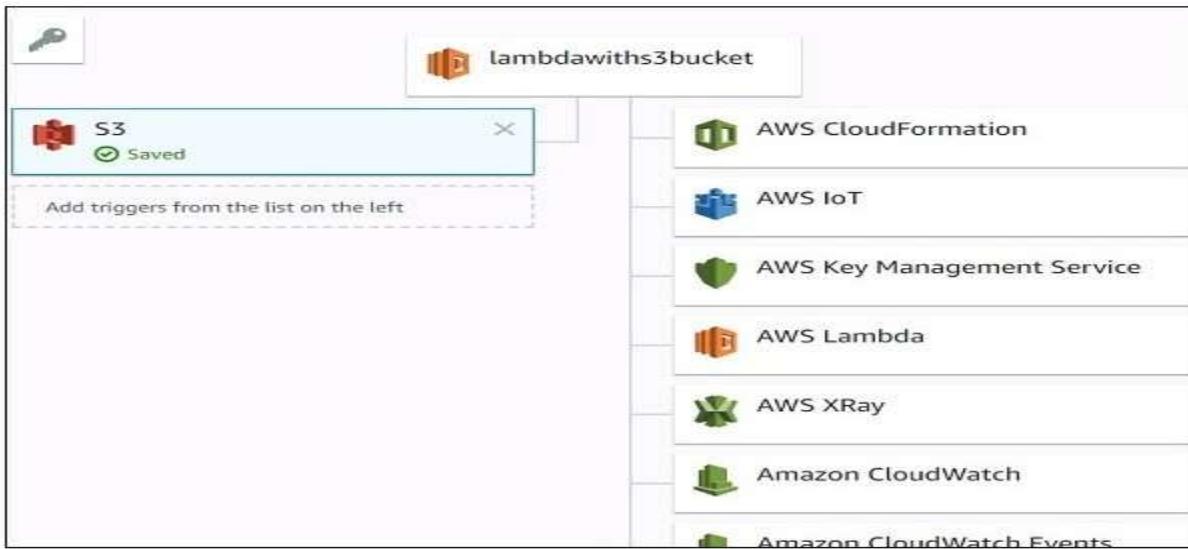
Filter pattern
Enter an optional filter pattern.

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Enable trigger
Enable the trigger now, or create it in a disabled state for testing (recommended).

Step 7

You can find the the trigger display for the Lambda function as shown below –



Let's add the details for the aws lambda function. Here, we will use the online editor to add our code and use nodejs as the runtime environment.

Step 8

To trigger S3 with AWS Lambda, we will have to use S3 event in the code as shown below –

```

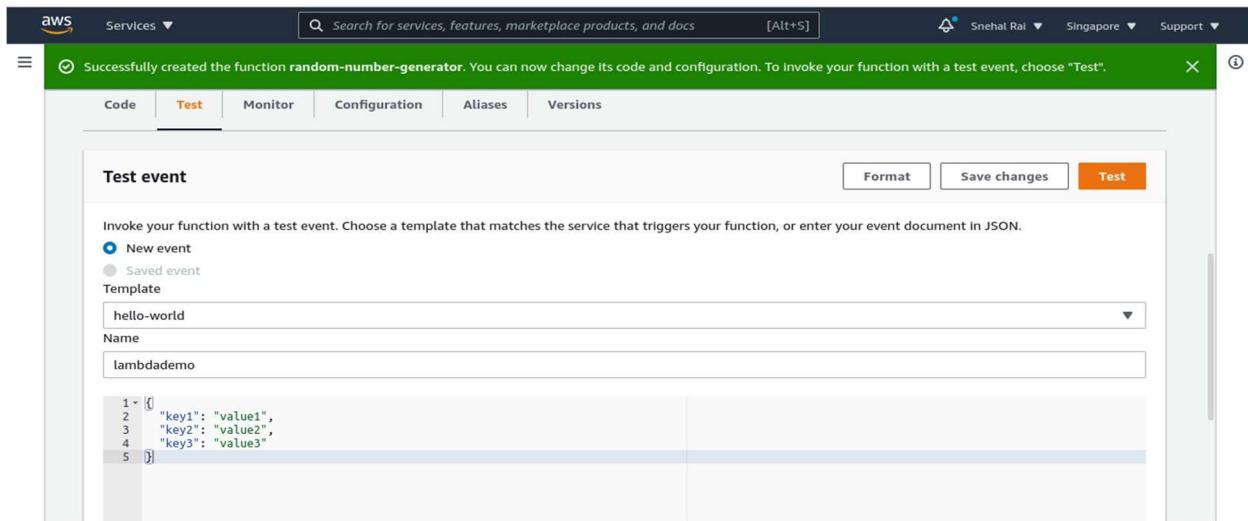
exports.handler = async (event) => {
  // TODO implement
  const response = {
    statusCode: 200,
    body: JSON.stringify('Hello from Lambda!'),
  };
  return response;
};

```

Note that the event param has the details of the S3event. We have consoled the bucket name and the file name which will get logged when you upload image in S3bucket.

Step 9

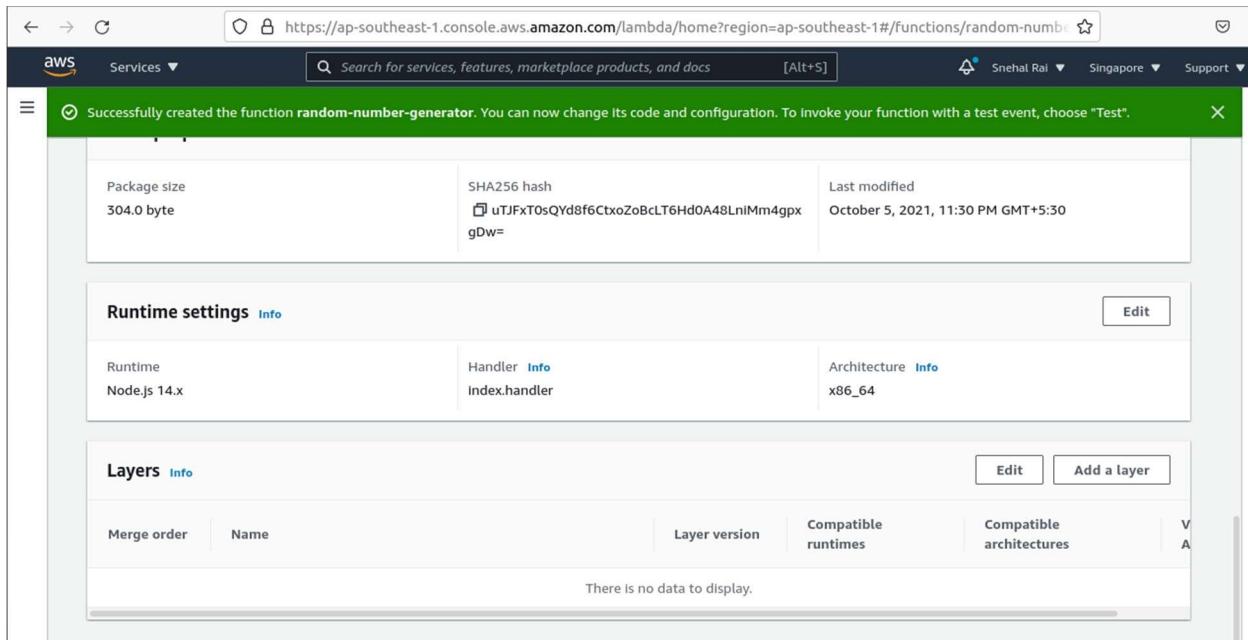
Now, let us save the changes and test the lambda function with S3upload. The following are the code details added in AWS Lambda –



Step 10

Now, save the Lambda function. Open S3 from Amazon services and open the bucket we created earlier namely workingwithlambdaands3.

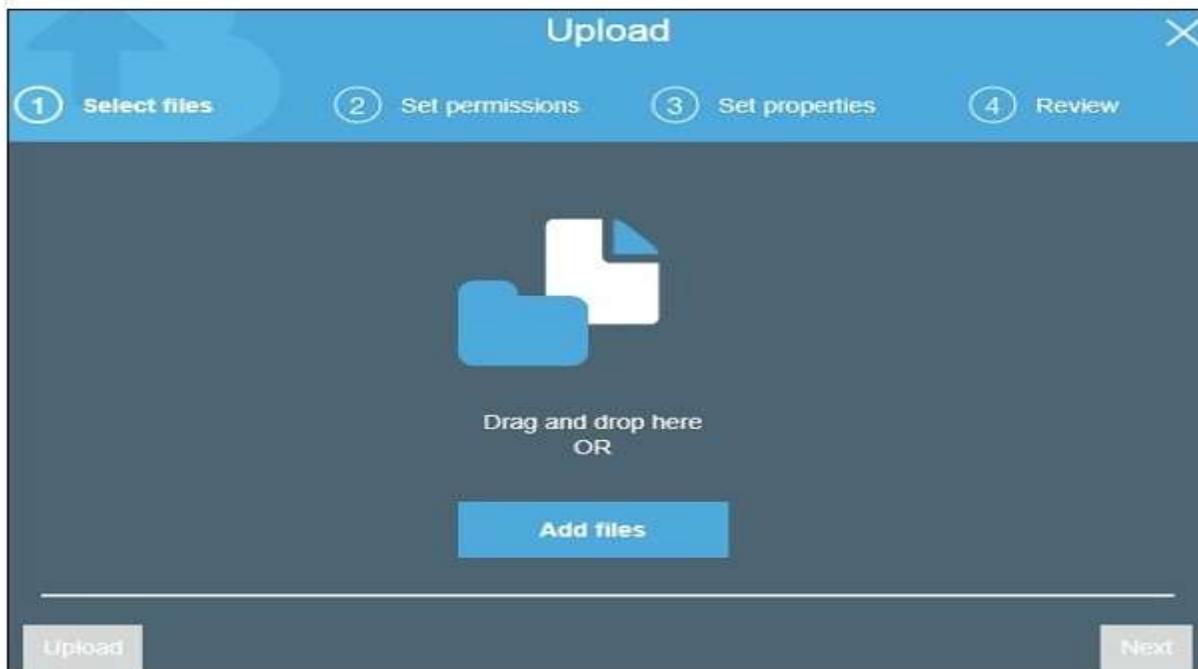
Upload the image in it as shown below –





Step 11

Click Upload button to add files as shown –



Step 12

Click Add files to add files. You can also drag and drop the files. Now, click Upload button.

The screenshot shows the Amazon S3 console interface. At the top, there's a navigation bar with 'Amazon S3 > workingwithlambdaands3'. Below it is a tab bar with 'Overview' (selected), 'Properties', 'Permissions', and 'Management'. A search bar says 'Type a prefix and press Enter to search. Press ESC to clear.' Underneath are buttons for 'Upload', 'Create folder', and 'More'. The region is set to 'US East (N. Virginia)'. A message 'Viewing 1 to 1' is displayed. A table lists one item: 'image3.jpg'. The table has columns for checkbox, Name, Last modified, Size, and Storage class. The file was last modified on May 31, 2016, at 11:38:13 AM (GMT+0530), is 10.3 KB, and is in the Standard storage class. Another 'Viewing 1 to 1' message is at the bottom.

Thus, we have uploaded one image in our S3 bucket.

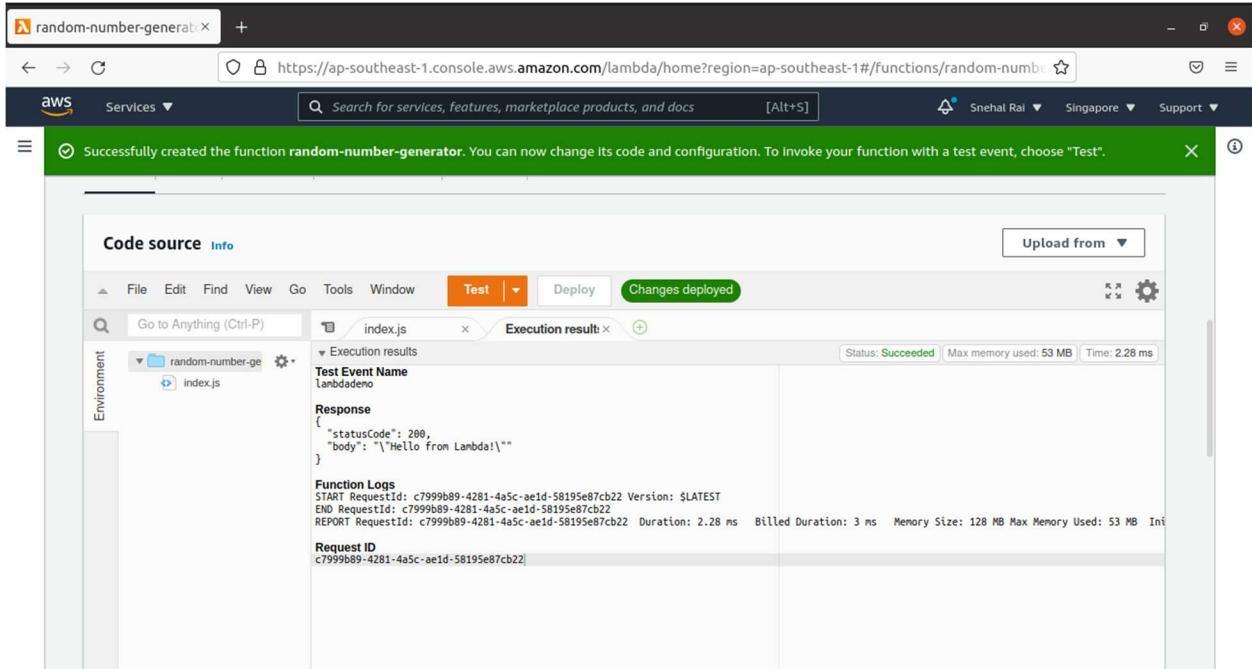
Step 13

To see the trigger details, go to AWS service and select CloudWatch. Open the logs for the Lambda function and use the following code –

```
exports.handler = function(event, context, callback) {
  console.log("Incoming Event: ", event);
  const bucket = event.Records[0].s3.bucket.name;
  const filename = decodeURIComponent(event.Records[0].s3.object.key.replace(/\n/g));
  const message = `File is uploaded in - ${bucket} -> ${filename}`;
  console.log(message);
  callback(null, message);
};
```

The output you can observe in Cloudwatch is as shown –

AWS Lambda function gets triggered when file is uploaded in S3 bucket and the details are logged in Cloudwatch as shown below –



Conclusion -

AWS Lambda is a service which takes care of computing your code without any server. It is said to be serverless compute. The code is executed based on the response of events in AWS services like adding /removing files in S3 bucket.