



PROJECT :1

SERVERLESS IMAGE PROCESSING

CREATE A SERVERLESS IMAGE PROCESSING THAT AUTOMATICALLY RESIZE AND OPTIMIZE AN IMAGE UPLOADED TO AMAZON S3 BUCKET





TABLE OF CONTENT

- OVERVIEW
- INTRODUCTION TO SERVERLESS FUNCTION
- STEPS TO CREATE SERVERLESS IMAGE PROCESSING
- RESULT
- CONCLUSION

OVERVIEW

- CREATE TWO S3 BUCKET
- CREATE IAM POLICY
- CREATE LAMBDA FUNCTION
- ADD TRIGGER
- TEST EVENT



SERVERLESS FUNCTION

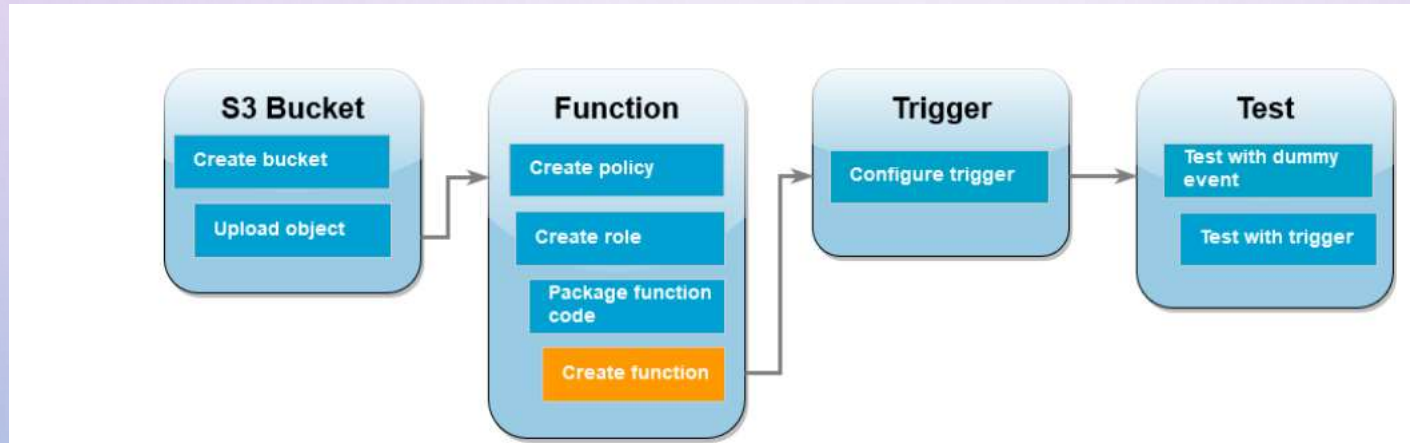
In the context of AWS (Amazon Web Services), lambda functions refer to AWS Lambda, which is a serverless computing service provided by Amazon. Here's a detailed explanation of what AWS Lambda is and how lambda functions relate to it.

AWS Lambda allows you to run code without provisioning or managing servers. It automatically scales your application by running code in response to triggers or events. Lambda functions are the units of code that you deploy and execute on AWS Lambda.

Key Feature of Lambda Function

- Serverless Execution
- Event-driven
- Pay-per-use
- Integration with AWS Services

HOW LAMBDA WORK



STEPS TO CREATE SERVERLESS IMAGE PROCESSING

STEP 1: create s3 buckets

- Go to AWS management console and sign in with your credentials
- Once logged in, you'll be in the AWS Management Console dashboard. Find and click on **"S3"** under **"Storage"**, or you can type "S3" in the search bar and select it
- In the S3 dashboard, click on the **"Create bucket"** button.
- Enter a unique name for your bucket. Bucket names must be globally unique across all of AWS, so if your desired name is taken, you'll need to choose another.
- In the option **Block Public Access Setting**, uncheck the option
- Click on I acknowledge the current setting might result in this bucket and the object it becoming public checkbox
- Keep everything default and click on **Create Bucket button**
- Create two buckets source and destination by above steps.

us-east-1.console.aws.amazon.com/s3/home?region=us-east-1

aws

Services

Search

[Alt+S]

N. Virginia

Kavya @ 3814-9214-6477

Amazon S3

Buckets

Access Grants

Access Points

Object Lambda Access Points

Multi-Region Access Points

Batch Operations

IAM Access Analyzer for S3

General purpose buckets (11)

Info

All AWS Regions

Copy ARN

Empty

Delete

Create bucket

Find buckets by name

< 1 >

Name	AWS Region	IAM Access Analyzer	Creation date
<input type="radio"/> kavyadestinationbucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 20, 2024, 15:53:40 (UTC+05:30)
<input type="radio"/> kavyasourcebucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 20, 2024, 15:52:57 (UTC+05:30)
<input type="radio"/> kavyawebsite	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 19, 2024, 23:04:21 (UTC+05:30)

CREATING IAM POLICY AND ROLE

- Navigate to the IAM console.
- Click on "Policies" in the left-hand menu.
- Click on "Create policy".
- Select the "JSON" tab to create your policy from scratch.
- Paste the JSON policy document given below


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::your-bucket-
name/*"
    }
  ]
}
```

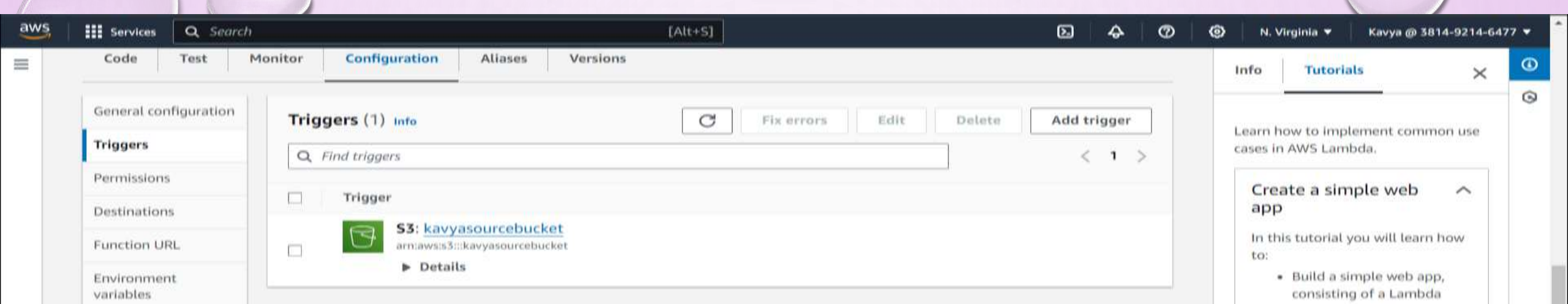
- Make sure that you update arn of your source bucket.
- Give policy name resizer.
- Click on "create policy" to save and create the IAM policy.
- Now on left panel select **Roles**.
- Click on create role and click on next.
- Select **Lambda** in **Use Case** dialog box.
- **Click on next**
- Select policy we created.
- Give role name and clickmon **Create Role**

Create Lambda Function

- Navigation to lambda.
- Create a function. Set function name Lambda function.
- Set runtime to Node.js 18.x
- Change default execution role to select existing one.
- Select image-resize-policy.
- Create function . lambda function is created

Add trigger

- Click on add trigger.
- Select **s3** in dialog box.
- Select **source bucket name**.
- Click on I acknowledge..... check box. And leave all as default.
- Click on **ADD** button




Create environmental variable

- Click on environmental variable in left panel.
- Click on edit
- IN dialog box
- Key : dest-buct
- Value : destination bucket.
- Click on **save** button

Edit environment variables

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. [Learn more](#) 

Key

Value

dest_buct

kavya destination bucket

Remove

Add environment variable

► Encryption configuration

Cancel

Save

Upload the zip file by clicking on code then upload.

Test the function

- Click on test tabs on the tabs
- On **Template** : select s3 Put

☑ Successfully updated the function kavalambdafunction.

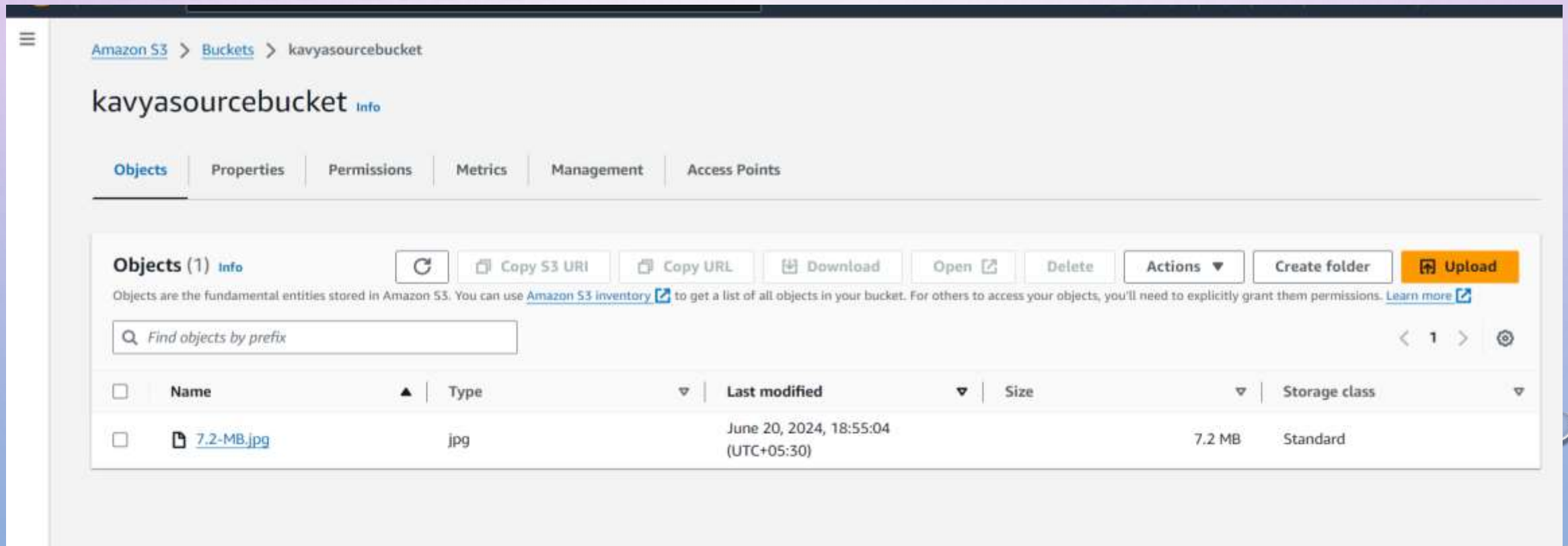
Event JSON

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": "kavyasourcebucket",
24          "ownerIdentity": {
25            "principalId": "EXAMPLE"
26          },
27          "arn": "arn:aws:s3:::kavyasourcebucket"
28        },
29        "object": {
30          "key": "7.2-MB.jpg",
31          "size": 1024,
32          "eTag": "0123456789abcdef0123456789abcdef",
33          "sequencer": "0A1B2C3D4E5F678901"
34        }
35      }
36    ]
37  }
```

RESULT

Source Bucket



The screenshot displays the Amazon S3 console interface for a bucket named 'kavyasourcebucket'. The breadcrumb navigation shows 'Amazon S3 > Buckets > kavyasourcebucket'. The bucket name 'kavyasourcebucket' is prominently displayed with an 'Info' link. Below this, a series of tabs allows navigation between 'Objects', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. The 'Objects' tab is active, showing a list of objects. At the top of the object list, there are several action buttons: a refresh icon, 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', an 'Actions' dropdown, 'Create folder', and an 'Upload' button. A search bar with the placeholder 'Find objects by prefix' is located below the buttons. The object list table has columns for 'Name', 'Type', 'Last modified', 'Size', and 'Storage class'. One object is listed: '7.2-MB.jpg' with a type of 'jpg', a last modified date of 'June 20, 2024, 18:55:04 (UTC+05:30)', a size of '7.2 MB', and a storage class of 'Standard'.

Amazon S3 > Buckets > kavyasourcebucket


kavyasourcebucket [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (1) [Info](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

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](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	 7.2-MB.jpg	jpg	June 20, 2024, 18:55:04 (UTC+05:30)	7.2 MB	Standard

Destination Bucket

Amazon S3 > Buckets > kavyadestinationbucket

kavyadestinationbucket [Info](#)

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (1) [Info](#)

Copy S3 URI

Copy URL

Download

Open

Delete

Actions ▾

Create folder

Upload

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](#)

Find objects by prefix

☐ Show versions

< 1 >

<input type="checkbox"/>	Name ▲	Type ▼	Last modified ▼	Size ▼	Storage class ▼
<input type="checkbox"/>	7.2-MB (1).jpg	jpg	June 20, 2024, 19:13:50 (UTC+05:30)	1.9 MB	Standard

CONCLUSION

lambda functions offer a concise and efficient way to resize images programmatically. Their ability to be defined inline and used directly where needed reduces the need for separate named functions or methods, streamlining code and improving readability. By leveraging lambda functions for image resizing, developers can achieve faster implementation times and maintain cleaner code bases, ultimately enhancing the overall efficiency and maintainability of image processing tasks in their applications.