**PROJECT**

**SERVERLESS IMAGE**

**PROCESSING**

**A diagram of a computer program

Description automatically generated**

**TASK**

**Create a serverless image processing application that automatically resizes and optimizes images uploaded to an**

**Amazon S3 Bucket.**

****

**PREREQUISITES**

1. **AWS Account**

* An Active AWS Account

1. **Understanding of AWS Services**

* Basic understanding of services like S3 Bucket and Lambda

1. **Familiarity of languages:**

* Basic knowledge of Node.js and Python as we will use any one of these language for lambda function.

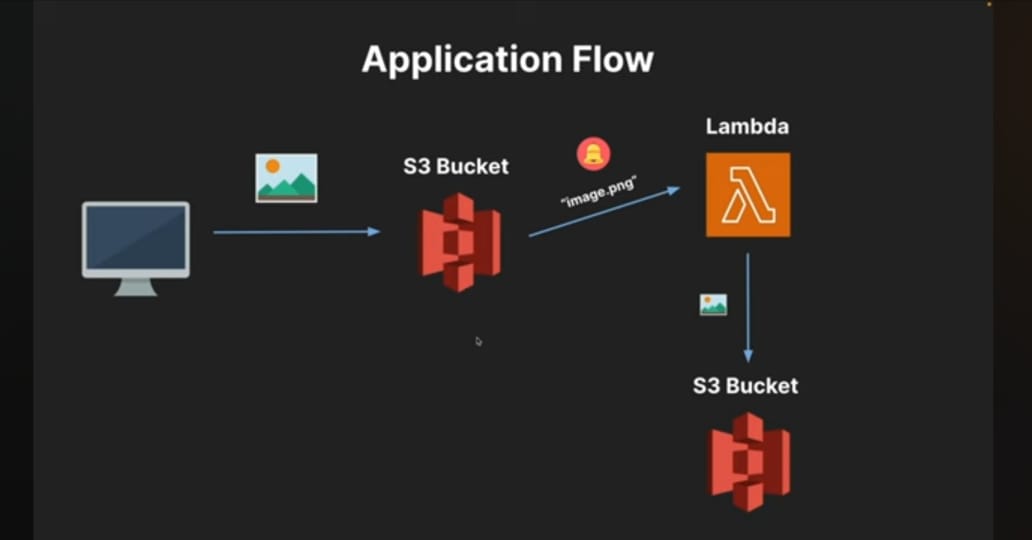
**SERVERLESS IMAGE PROCESSING**

**FLOW**

* Serverless Image Processing Flow.
* User uploads a file to the source S3 bucket (which is used for storing uploaded images).
* When the image is uploaded to a source S3 bucket, it triggers an event which invokes the Lambda function. The lambda function processes the image.

* Processed image is stored in the destination S3 bucket.
* The processed image is requested by the user

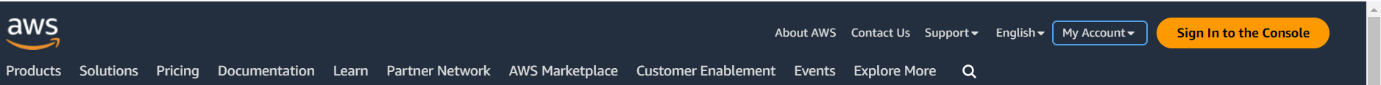
**PROCESS ARCHITECTURE**

****

**STEP 1 :**

**Sign-In to AWS Management Console**

* Click on the**Open Console** .



* Sign- in to your AWS account by using **Account ID,**

**Username** and **Password**.

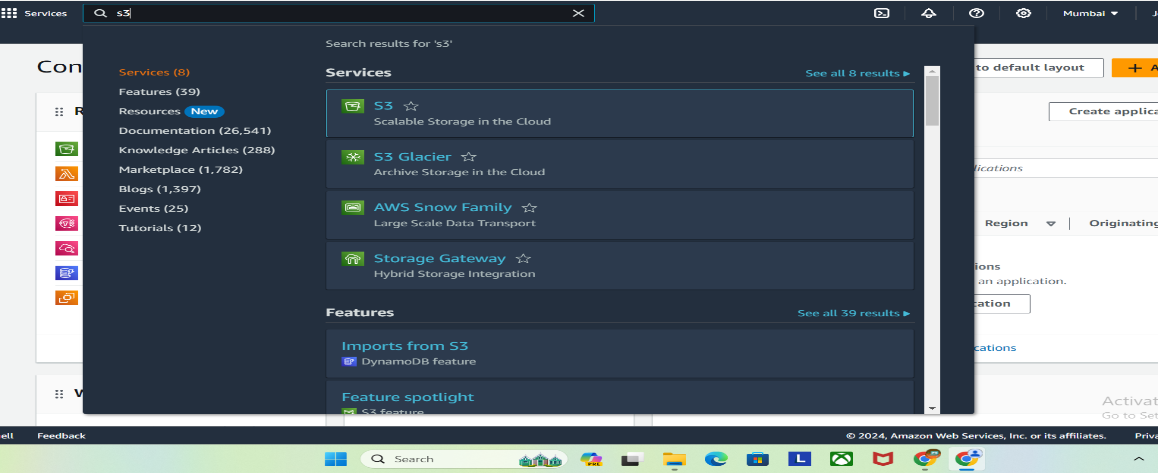
* Once you complete sign-in , you will directed to

AWS Management Console.

**STEP 2 :**

**Create Two Amazon S3 Buckets**

* Navigate to the **Services** menu in the top, then click on **S3** in the Storage section.



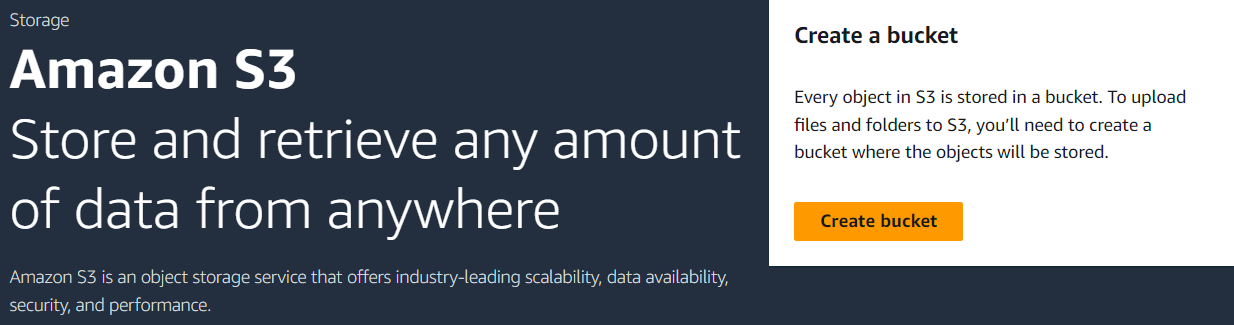
* We will create two S3 buckets :

1. **source Bucket:** For storing uploaded images.

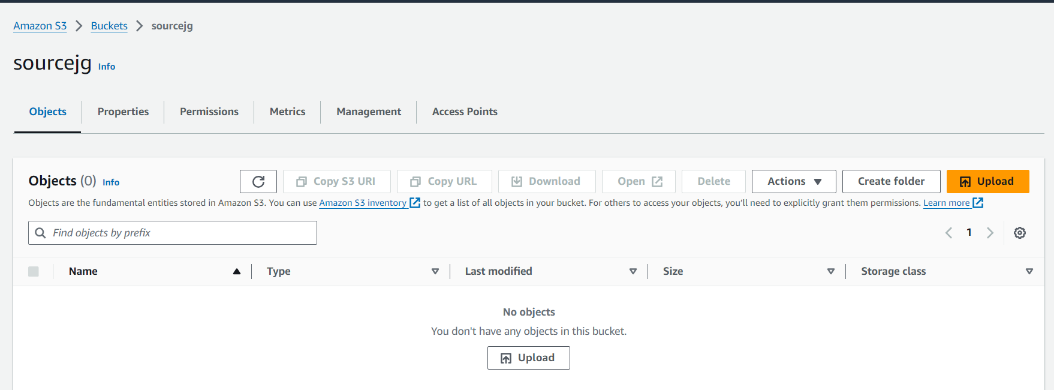
2. **destination Bucket:** For storing processed images.

**2.1 .Creating Source Bucket**

* Go to S3 console and click Create bucket

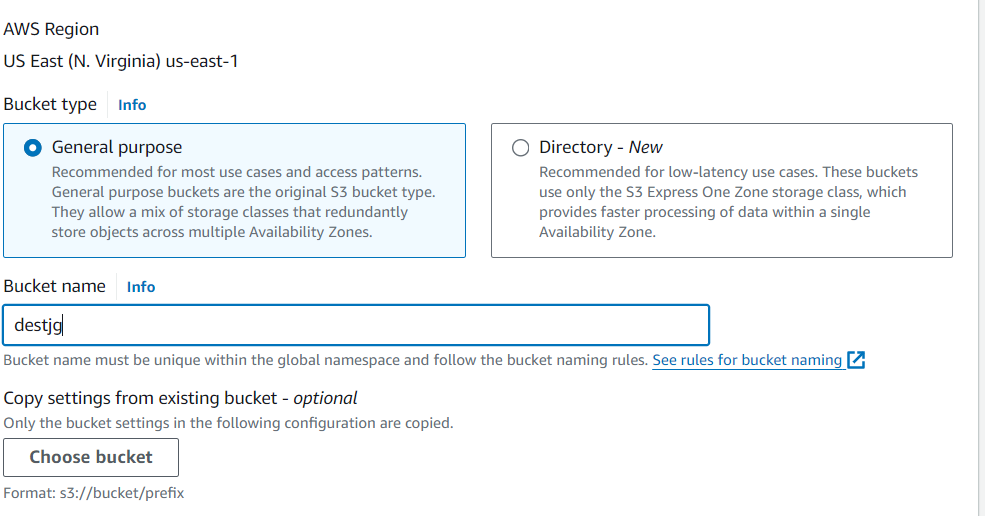


* Bucket Name: Enter **sourcejg**
* AWS Region: Select **US East (N. Virginia) us-east-1**
* Leave other settings as default and click on the **Create bucket** button.
* Select your created bucket ,Click on the **Copy ARN** button to copy the ARN.
* Save the source bucket ARN in a text file for later use.



**2.2 .Create Destination Bucket**

* Click on the **Create bucket** button.
* Bucket Name: Enter **destjg**
* AWS Region: Select **US East (N. Virginia) us-east-1**

****

* Leave other settings as default and click on the **Create bucket** button.
* Once the bucket is created successfully, Select your S3 bucket.
* Click on the Copy ARN button to copy the ARN
* Save the destination bucket ARN and save it in a text file for further use.

**STEP 3 :**

**Upload object to the source bucket**

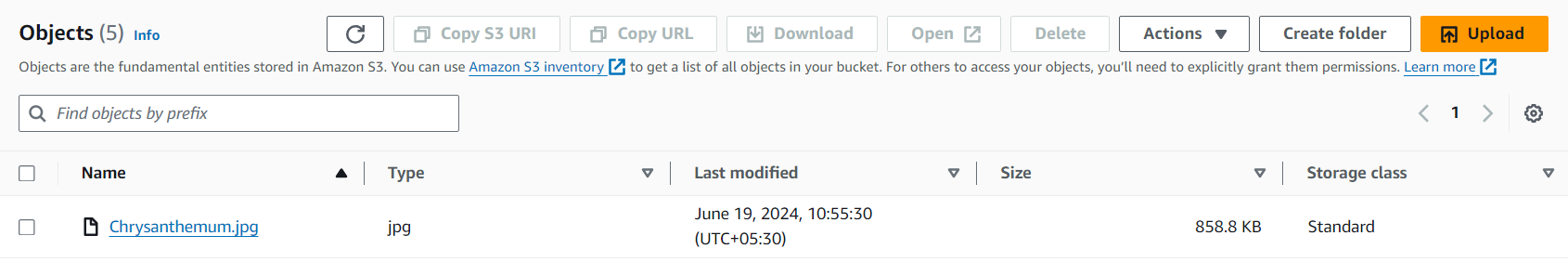
* Go to **S3 Bucket** list and click on source bucket –

sorcejg.

* Upload image to source S3 bucket. To do that:
* Click on the Upload button.
* Click on Add files button to add the files

in .jpg format.

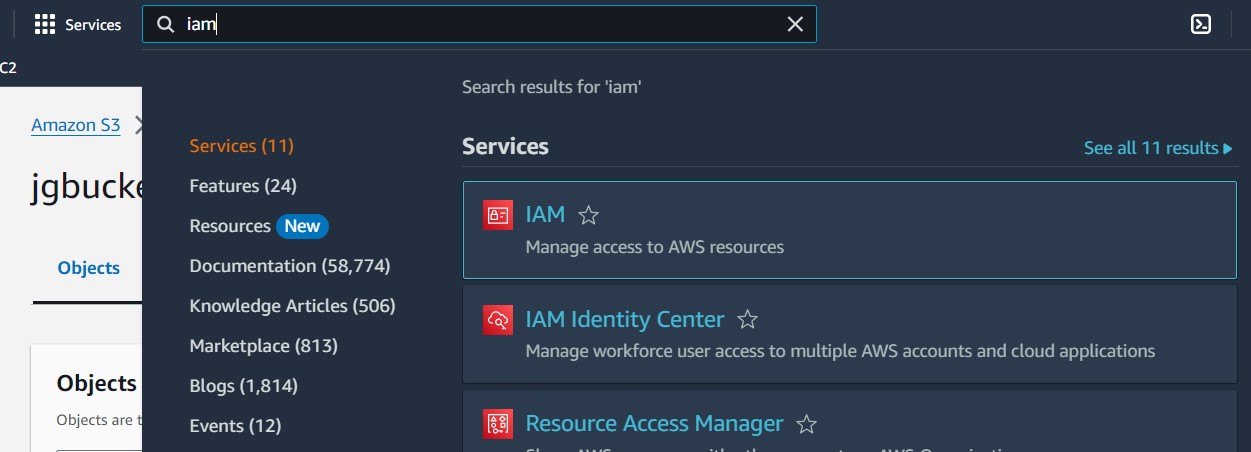
* Choose the file to be uploaded.
* Then, click on upload.
* The object is sucessfully uploaded.



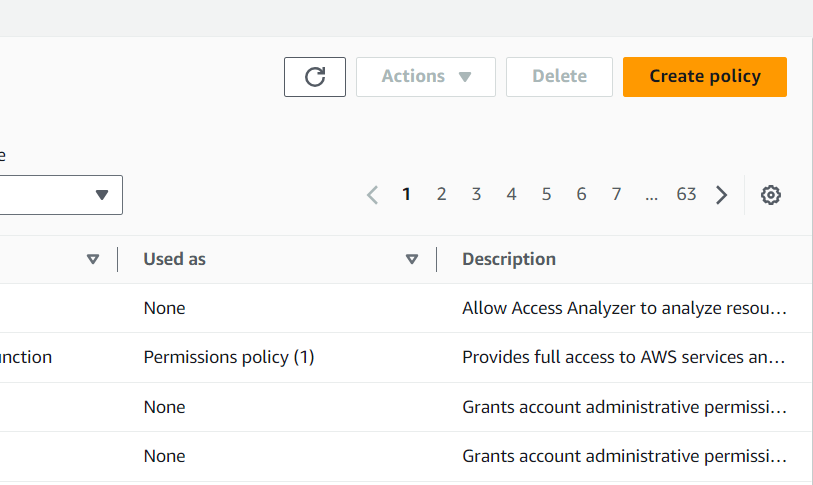
**STEP4 :**

**Create an IAM Policy**

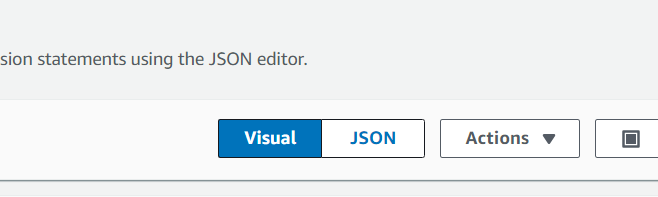
* Go to **Services** and Select IAM.



* Click on **Policies** in the left navigation bar and click on the **Create Policy** button.



* Choose the **JSON** tab.



* Remove the existing one and add the given

Statement in the editor.

* Policy JSON:

**{**

**"Version":"2012-10-17",**

**"Statement":[**

**{**

**"Effect":"Allow",**

**"Action":[**

**"s3:GetObject"**

**],**

**"Resource":[**

**"arn:aws:s3:::mysourcebucket12345/\*"**

**]**

**},**

**{**

**"Effect":"Allow",**

**"Action":[**

**"s3:PutObject"**

**],**

**"Resource":[**

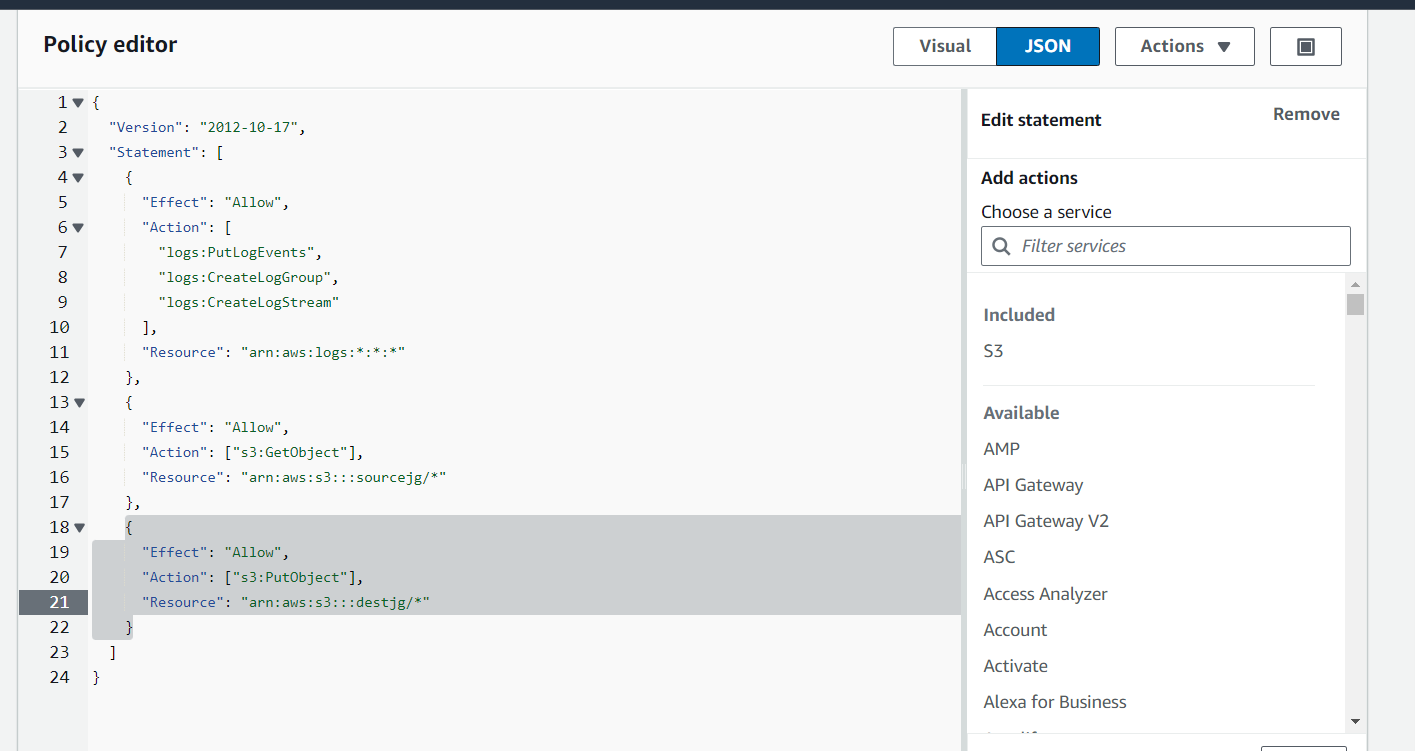
**"arn:aws:s3:::mydestinationbucket12345/\*"**

**]**

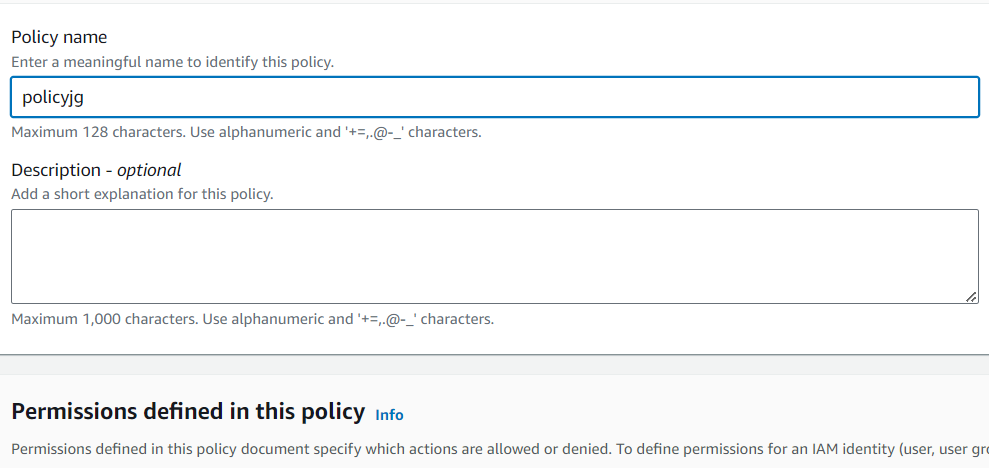
**}**

**]**

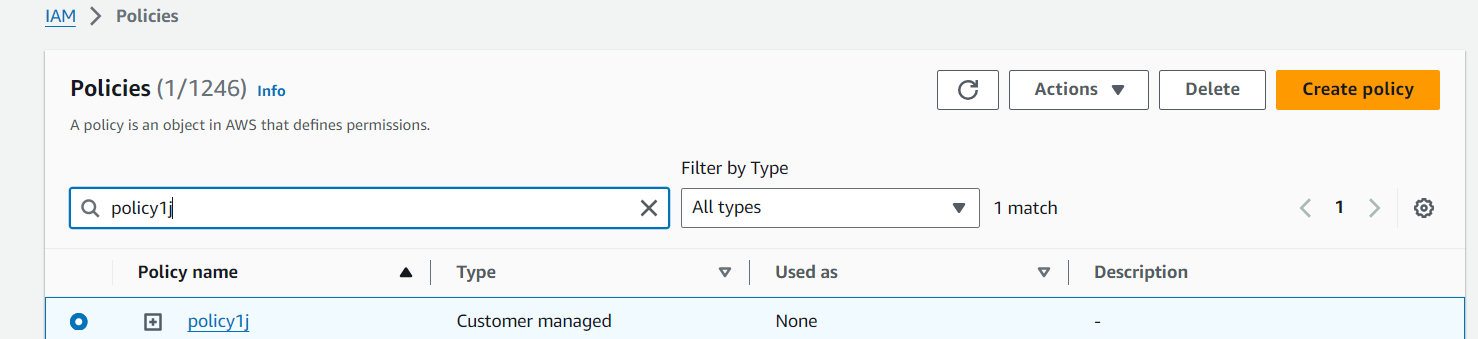
**}**

****

* Leave everything as default, click **Next**.
* On Policy Page:
* Enter Policy Name.
* Click **Create policy.**



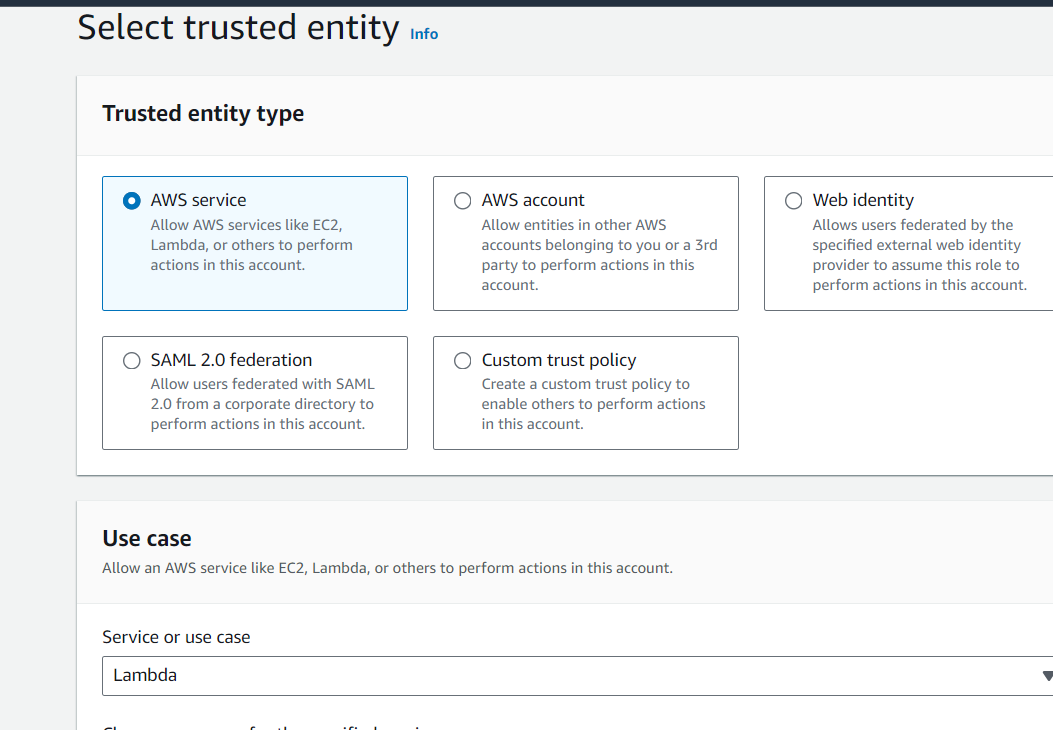
* An IAM Policy is created.



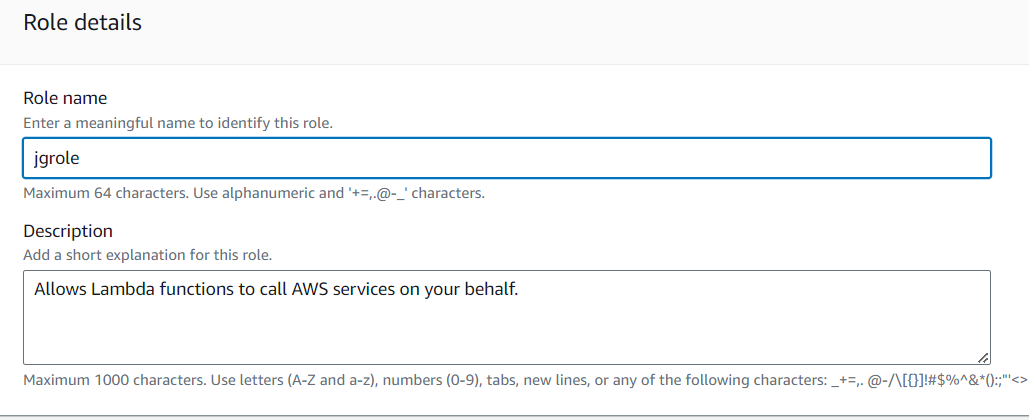
**STEP 5:**

**Create an IAM Role**

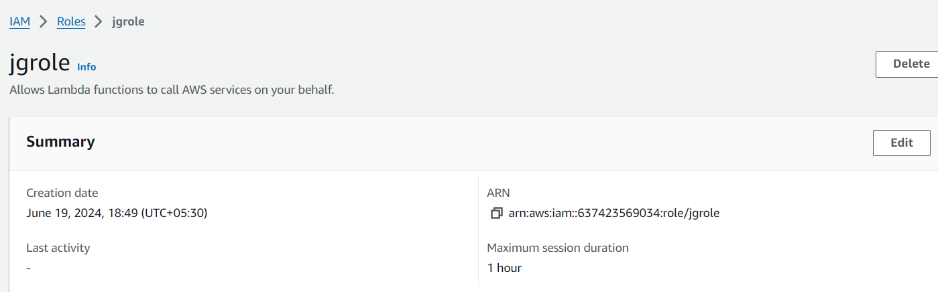
* Click on **Roles** from menu displaying on left side.
* Click on **Create role** button.
* Search **Lambda** from services list.
* From **Trusted Entity Type :** Select **AWS Service.**
* From **Use case:** Select **Lambda.**
* Click on**Next .**



* Select the created policy and click on **Next.**
* **Role Name:** Enter **jgrole**.



* Click on the **Create Role.**

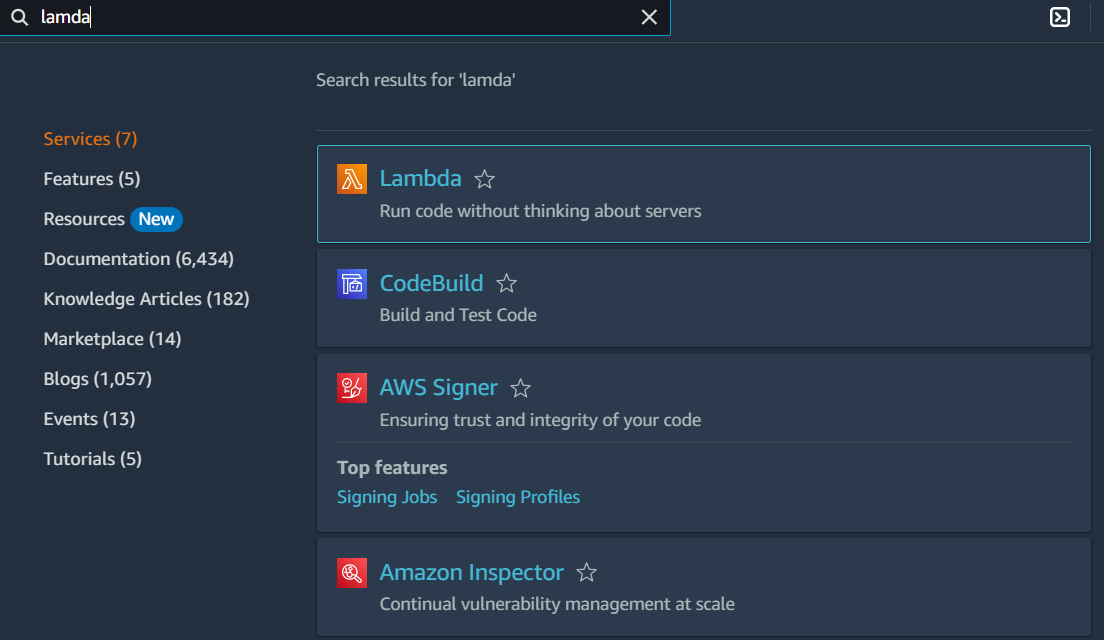


* IAM Role is created successfully.

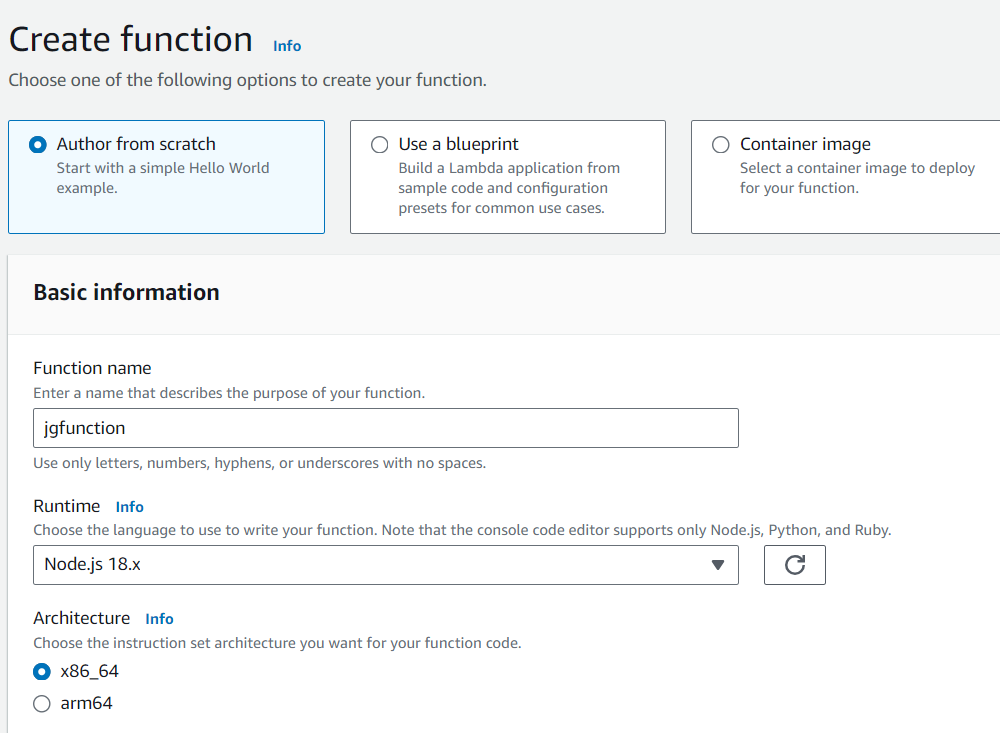
**STEP 6:**

**Create A Lambda Function**

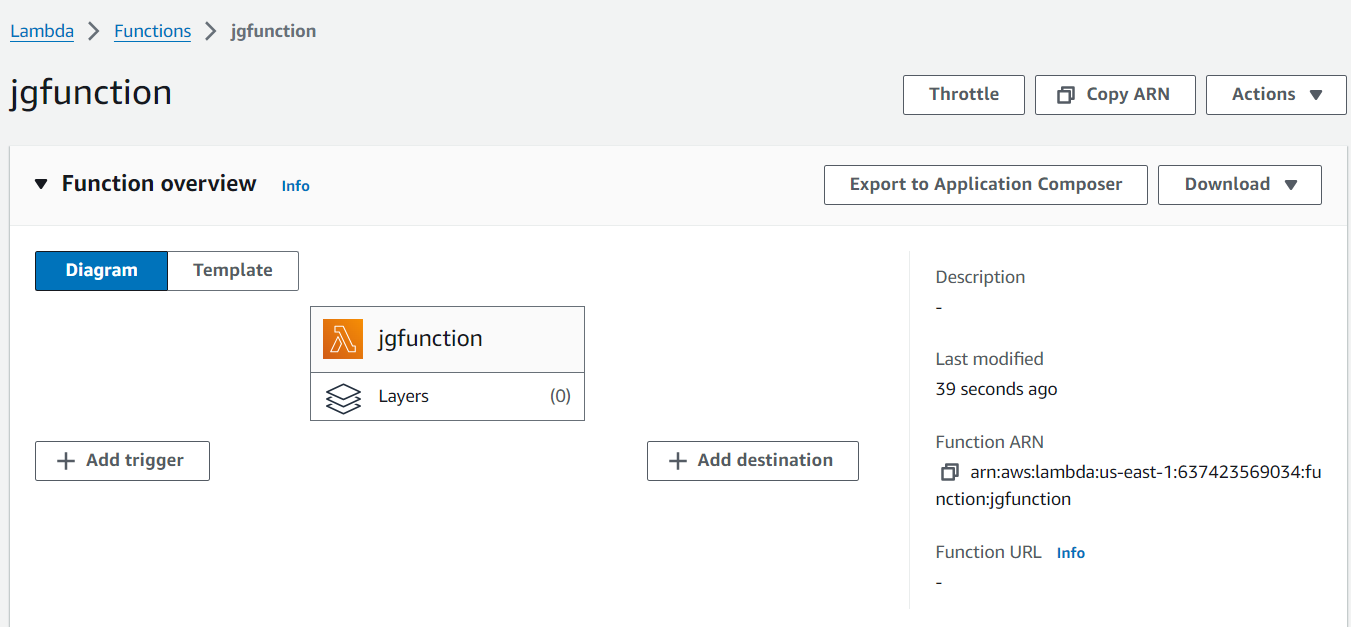
* You should be in the **US East (N. Virginia)** region.
* Go to the **Services** menu and click on **Lambda .**



* Click on the **Create a function** button.



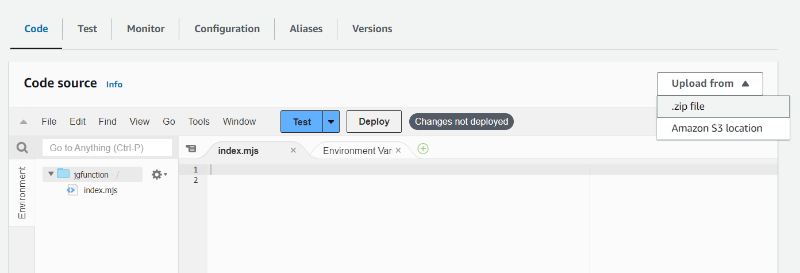
* Choose **Author from scratch**
* **Function name :** Enter mylambdafunction
* **Runtime:** Select Node.js 16x
* Click on the **Change default execution role**.
* Select an existing execution role.
* Select your existing role from the list.
* Lambda Function created successfully.

****

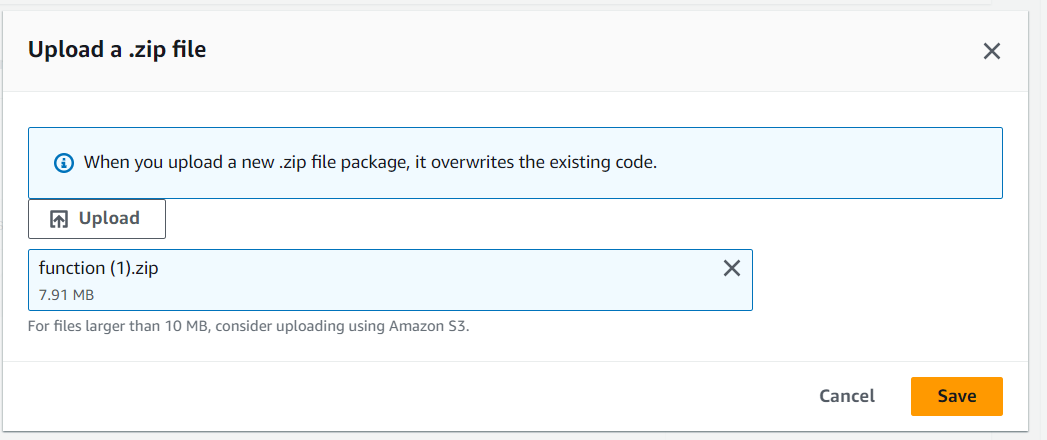
**STEP 7:**

**Adding Code to the Function**

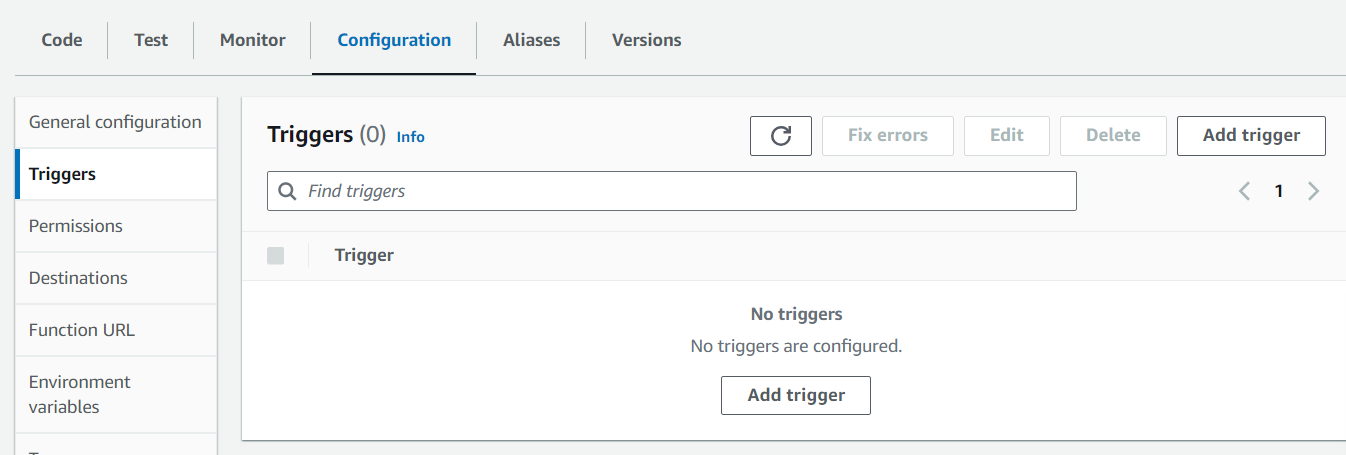
* Go the **Code** option.



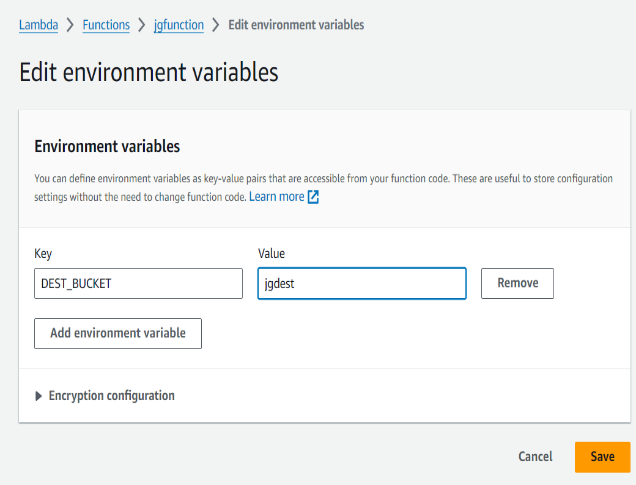
* Then click on the **Upload** option and select **.zip file.**
* Upload the file **function.zip** file there.



* After this click **Configuration** from the panel.



* Chooses Environmental Variable.

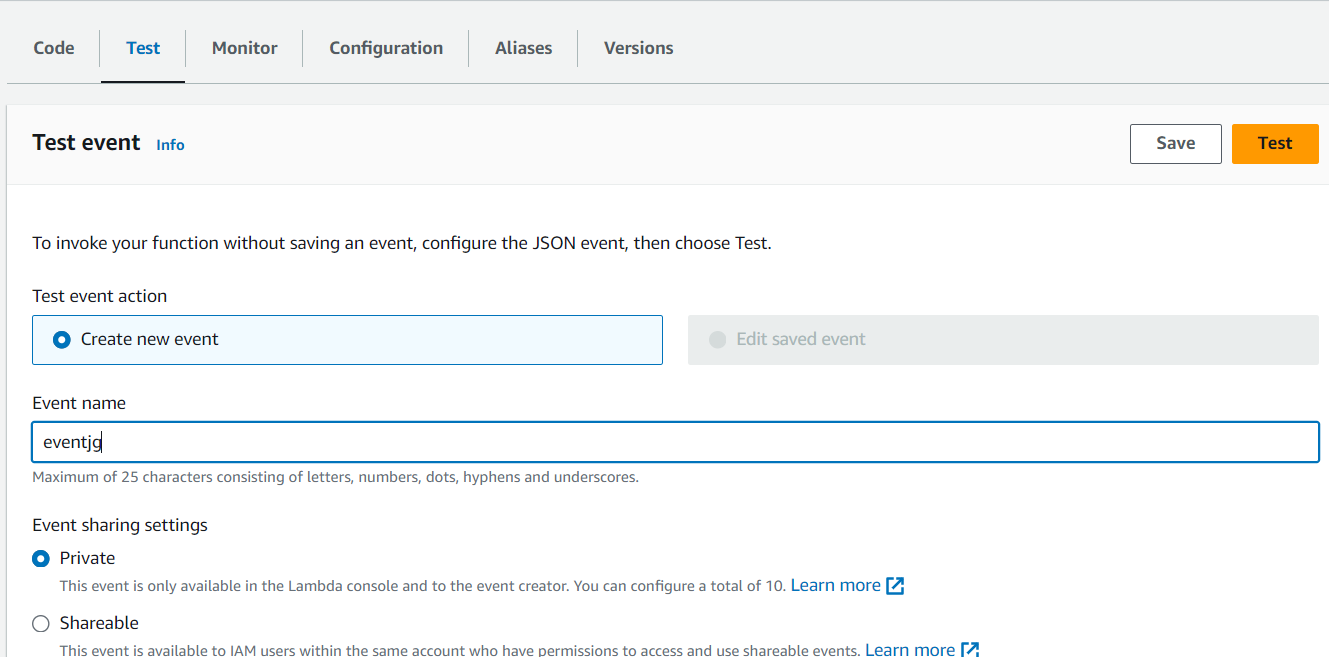


* In key:**DEST\_BUCKET**
* Value :**jgdest** (name of the destination bucket).

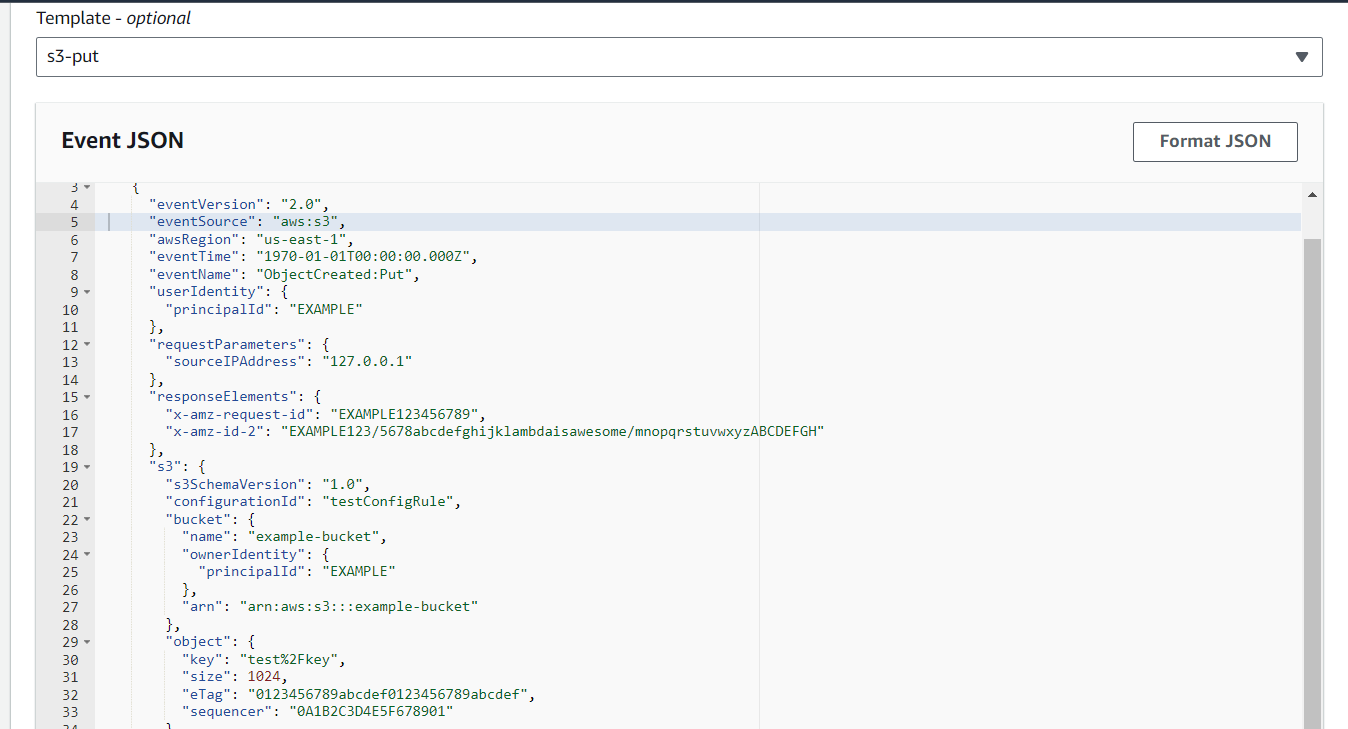
**STEP 8:**

**Test the Code**

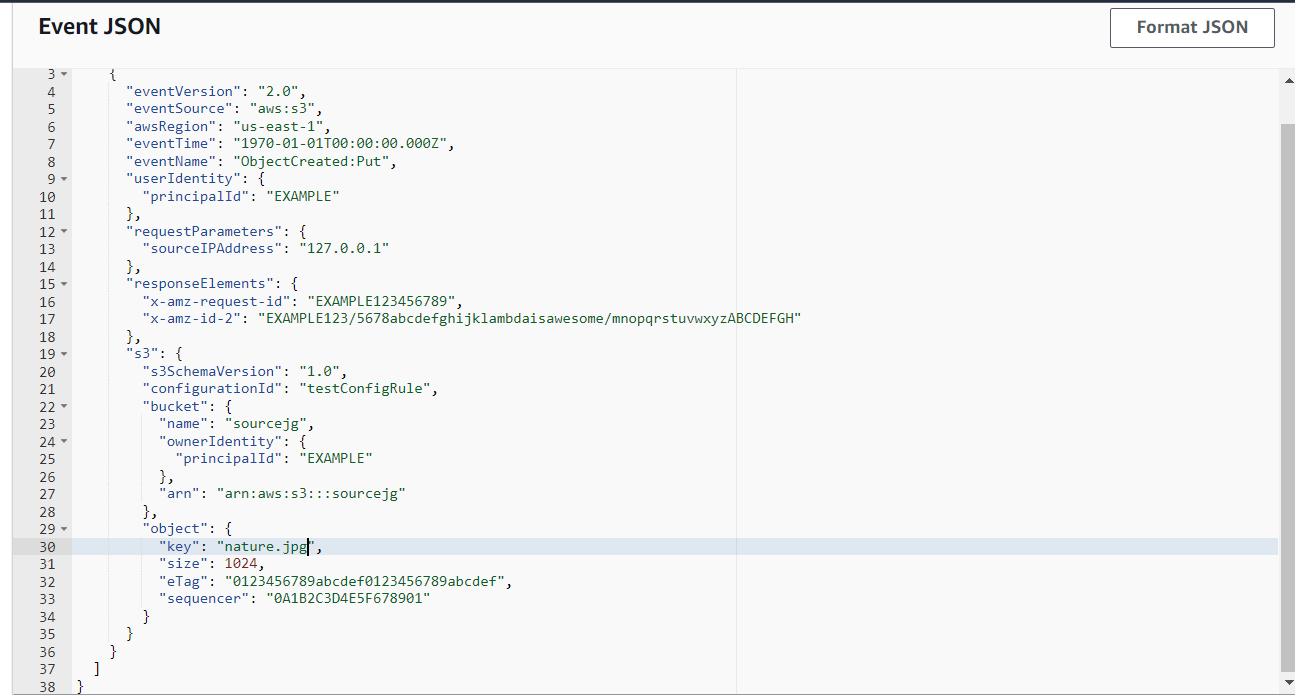
* Click on the **Test**option.
* Choose **Create new event .**
* Give a name of the event.



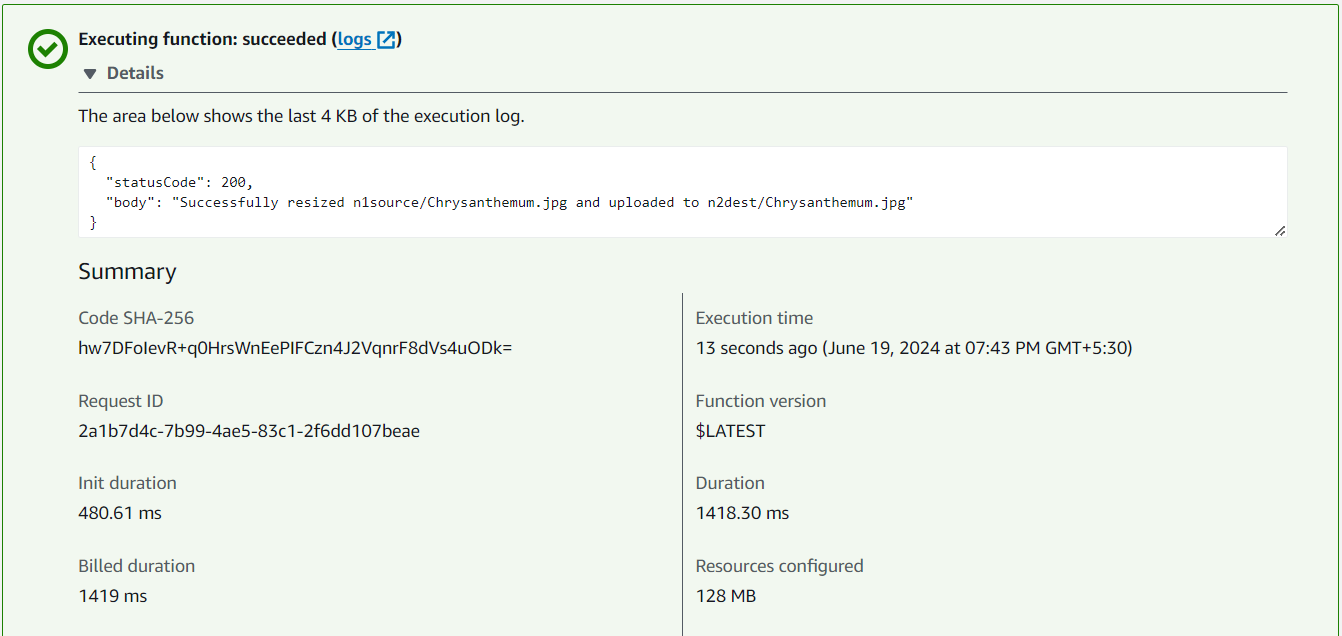
* Template : Choose **s3-put**.



* A JSON Code appears on the screen.
* Edit that code accordingly.
* **Example Bucket:** Name of the source bucket.
* **ARN:** ARN of the source bucket.
* **Key:** It is the name of the object uploaded in the source.
* Edited Policy :



* Click on **Test** option.
* If you the image given below:



* Then, your code is implemented successfully.

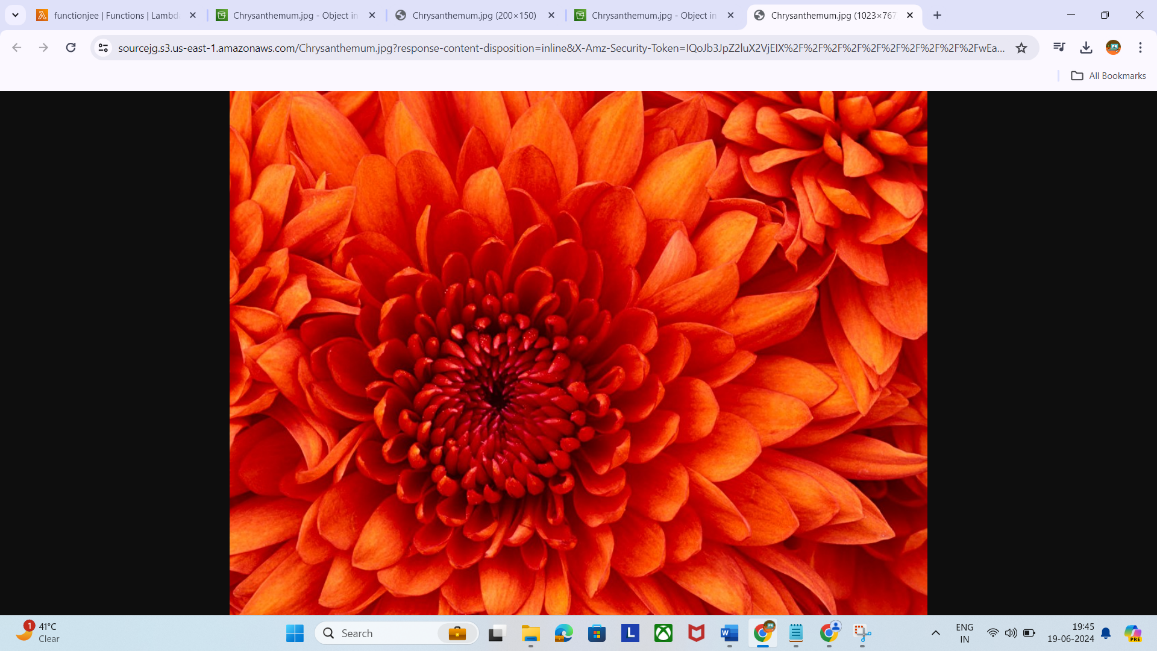
**STEP 9:**

**Test Lambda function**

* Go to **S3 Bucket** list and click on **source bucket**–

sourcejg.

* In Object ,go the uploaded one.
* Open and see it’s size.



* Now, go to the **destination bucket**, now you see the same object uploaded there.
* See the size and open it.



* **Successfully, we had implemented serverless resizing and optimization of image.**