# PROJECT – 1

|  |  |
| --- | --- |
| PROJECT 1 | DETAIL  Create a serverless image processing application that automatically resizes and optimizes images uploaded to an Amazon S3 bucket  Janhvi Mishra |

INTRODUCTION

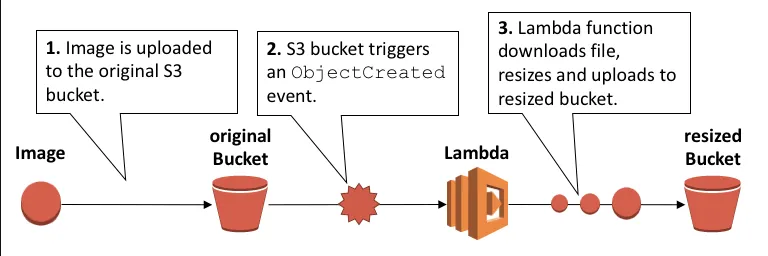
**AMAZON S3** - Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

**AWS LAMBDA** - AWS Lambda is an Amazon serverless computing system that runs code and automatically manages the underlying computing resources like (EC2). It is an event-driven computing service. It lets a person automatically run code in response to many types of events, such as HTTP requests from the Amazon API gateway, table updates in Amazon DynamoDB, and state transitions. It also enables the person to extend to other AWS services with custom logic and even creates its own back-end services. For example, just write the code and then upload it as a .zip file or any container image. The service works by running code on high-availability computer infrastructure. It then performs all the administrative duties of that compute resource.

# OVERVIEW

**Serverless image processing flow**

1. User uploads a file to the source S3 bucket (which is used for storing uploaded images).
2. 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.
3. Processed image is stored in the destination S3 bucket.
4. The processed image is requested by the user.



# **STEPS INVOLVED IN SERVERLESS IMAGE PROCESSING**

Step 1 : Creating two S3 buckets.

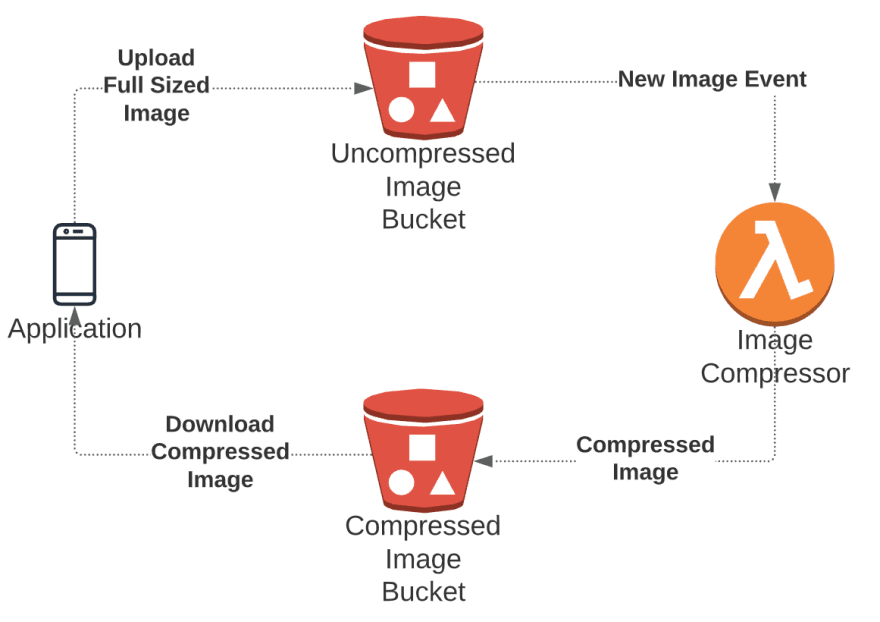
Step 2 : Upload image to be resized.

Step 3 : Create IAM policy and role.

Step 4 : Creating Lambda function.

Step 5 : Creating S3 trigger.

Step 6 : Testing the application.

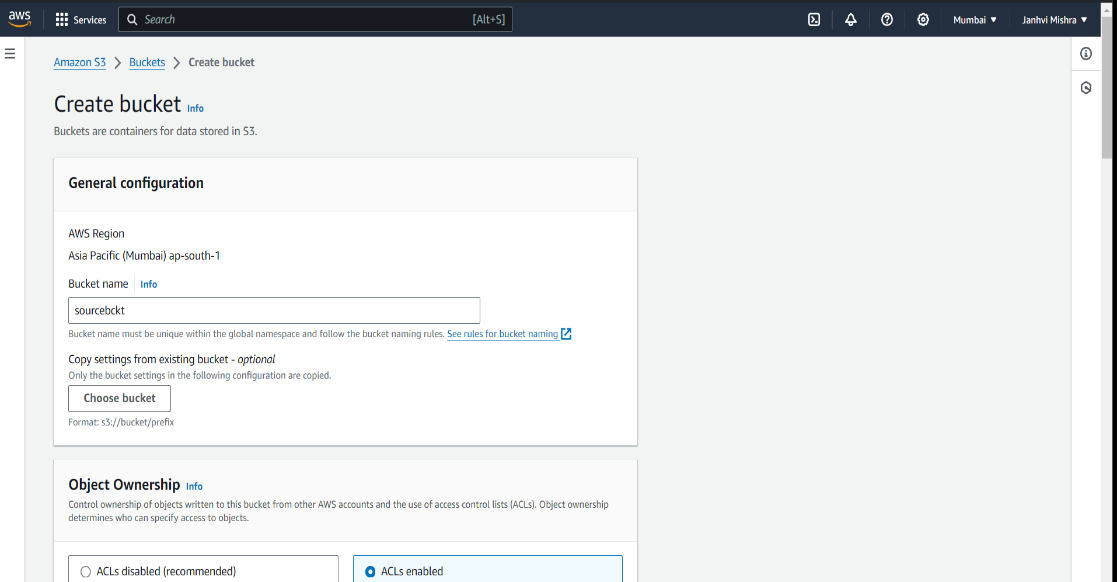


# **STEPS**

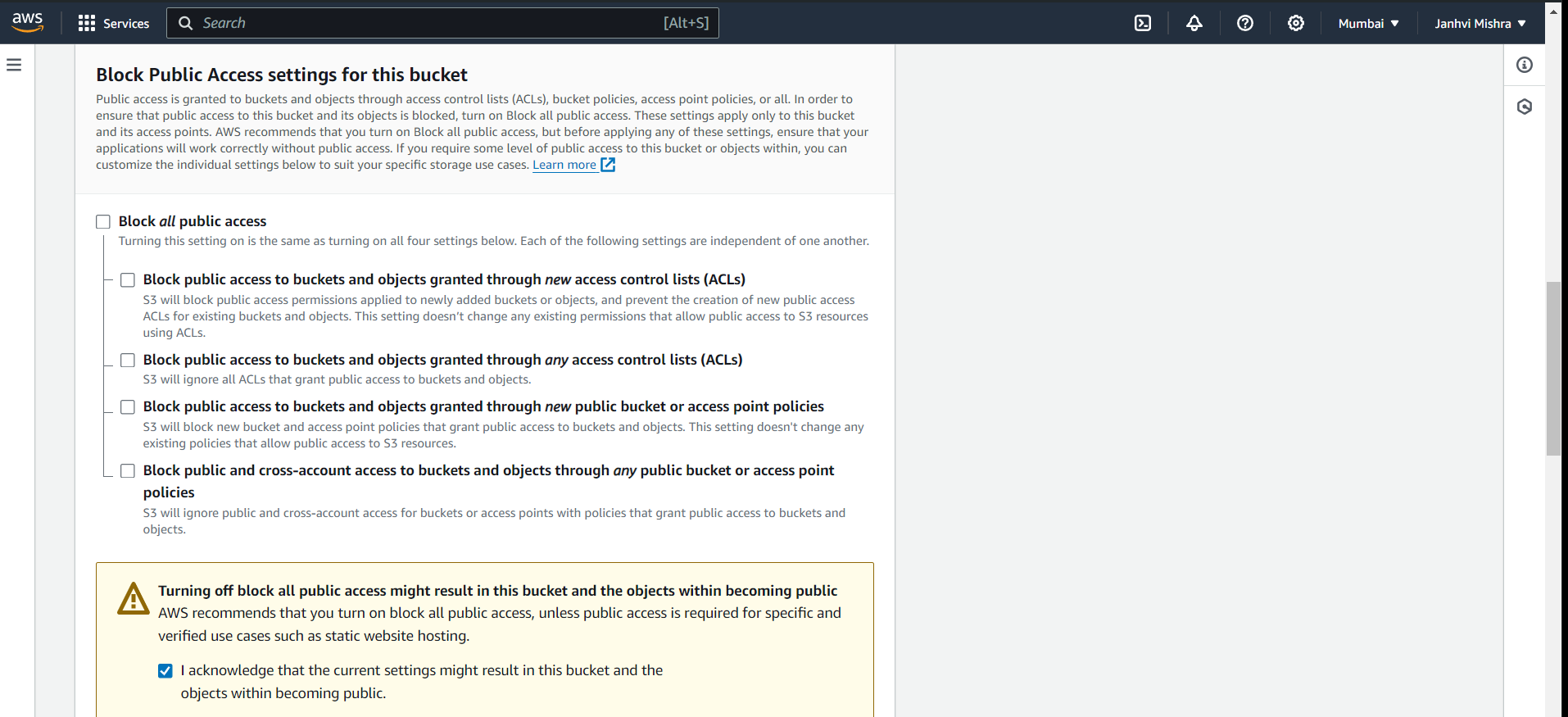
* Sign in to AWS management console.

**Step 1 : Creating two S3 buckets.**

* Navigate to service menu in the top and search for s3 service for creating s3 bucket.
* Then click on the option create bucket.
* We have to create two buckets so enter two buckets named sourcebckt and mydestinationbckt.
* AWS Region : Select Asia Pacific (Mumbai) ap-south-1.
* In object ownership select ACLs enabled.



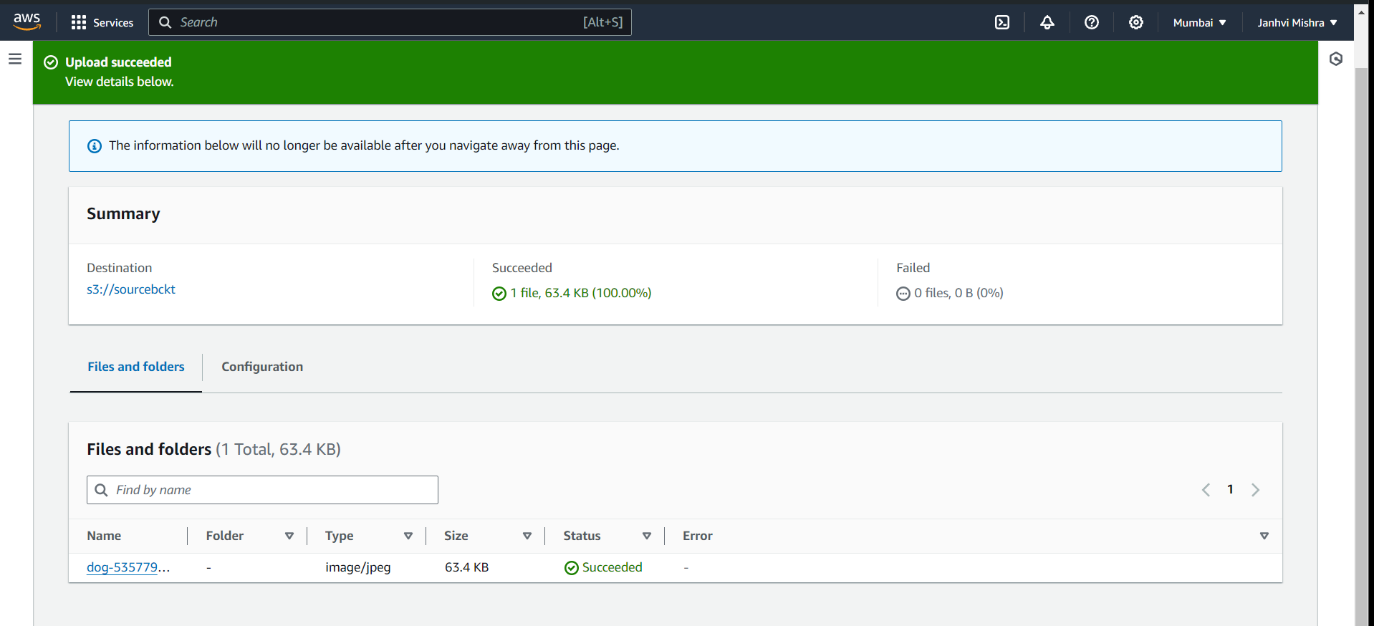
* Then uncheck block all public access and check the box of I acknowledge.



* Now keep all setting as default and click on create bucket.
* Create another bucket which will contain resized image .
* Above process will be same for creating destination bucket named mydestinationbckt.

**Step 2 : Upload image to be resized.**

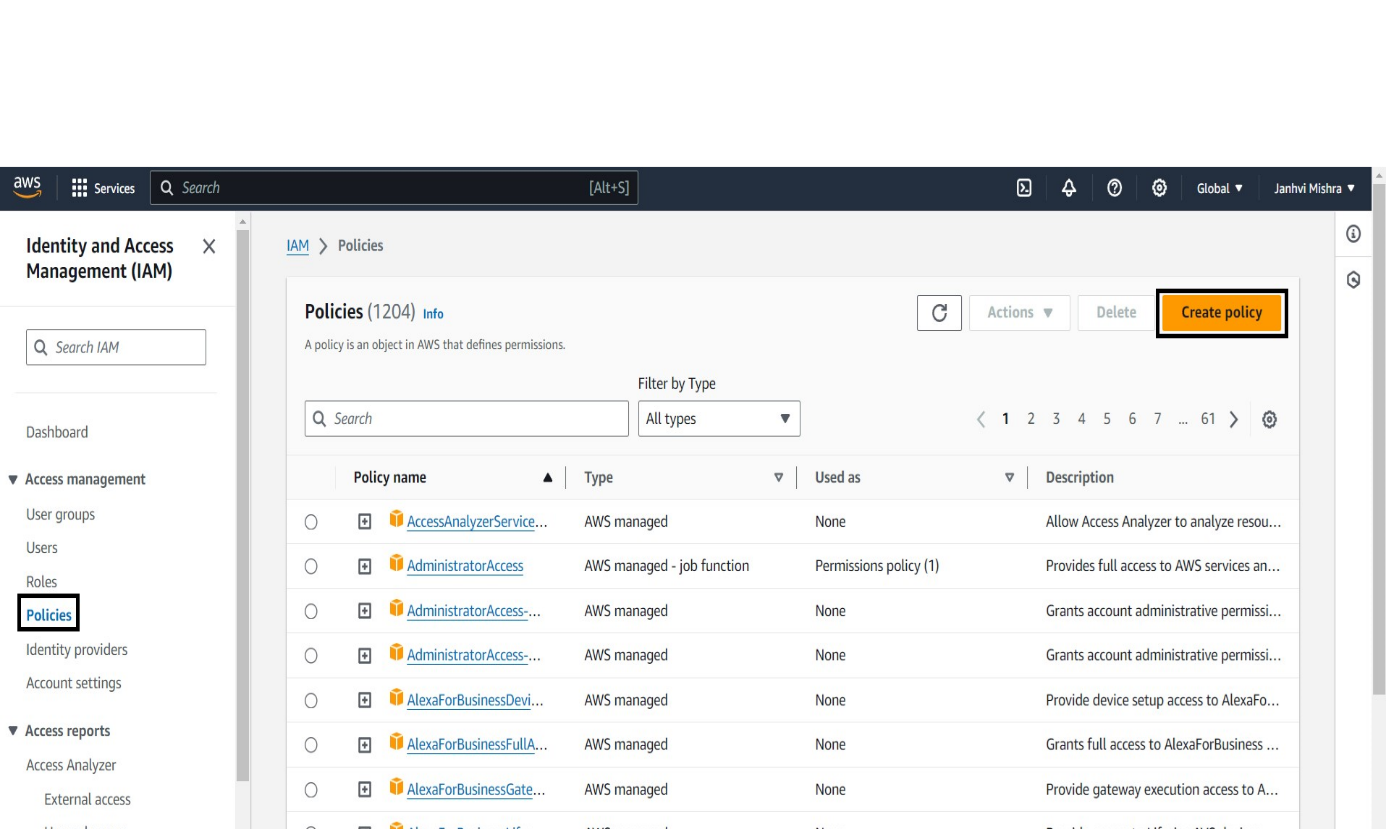
**Now after creating both the buckets we will upload image in the source bucket.**



**Now we will make use of our AWS Lambda function to copy the content from source bucket to destination bucket.**

**Step 3 : Create IAM policy and role.**

* Create an IAM policy by going to services and select IAM.
* Click on Policies in the left navigation bar and click on create policy button.



* Now go to JSON and remove the code and copy paste the below code replacing source bucket and destination bucket name.

{

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

"Statement": [

{

"Effect": "Allow",

"Action": [

"logs:PutLogEvents",

"logs:CreateLogGroup",

"logs:CreateLogStream"

],

"Resource": "arn:aws:logs:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": ["s3:GetObject"],

"Resource": "arn:aws:s3:::BUCKET\_NAME/\*"

},

{

"Effect": "Allow",

"Action": ["s3:PutObject"],

"Resource": "arn:aws:s3:::DEST\_BUCKET/\*"

}

]

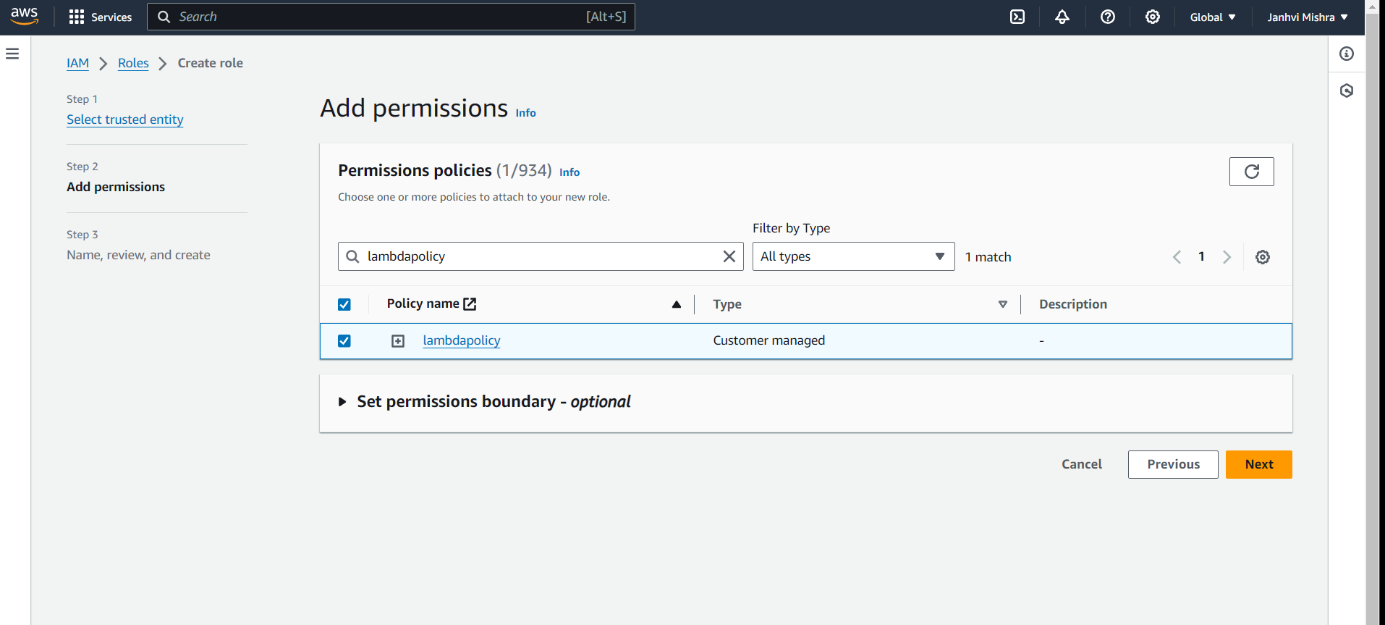
}



* Now click on next button.
* Then fill policy name as lambdapolicy and rest all setting will be default. Click on create policy button.

**Now after creating IAM policy we need to create IAM role by going to left panel and selecting role option.**

* Click on create role. In trusted entity type select AWS service and use case as lambda. Click on next.
* Now add permission by searching the IAM policy which you have created.

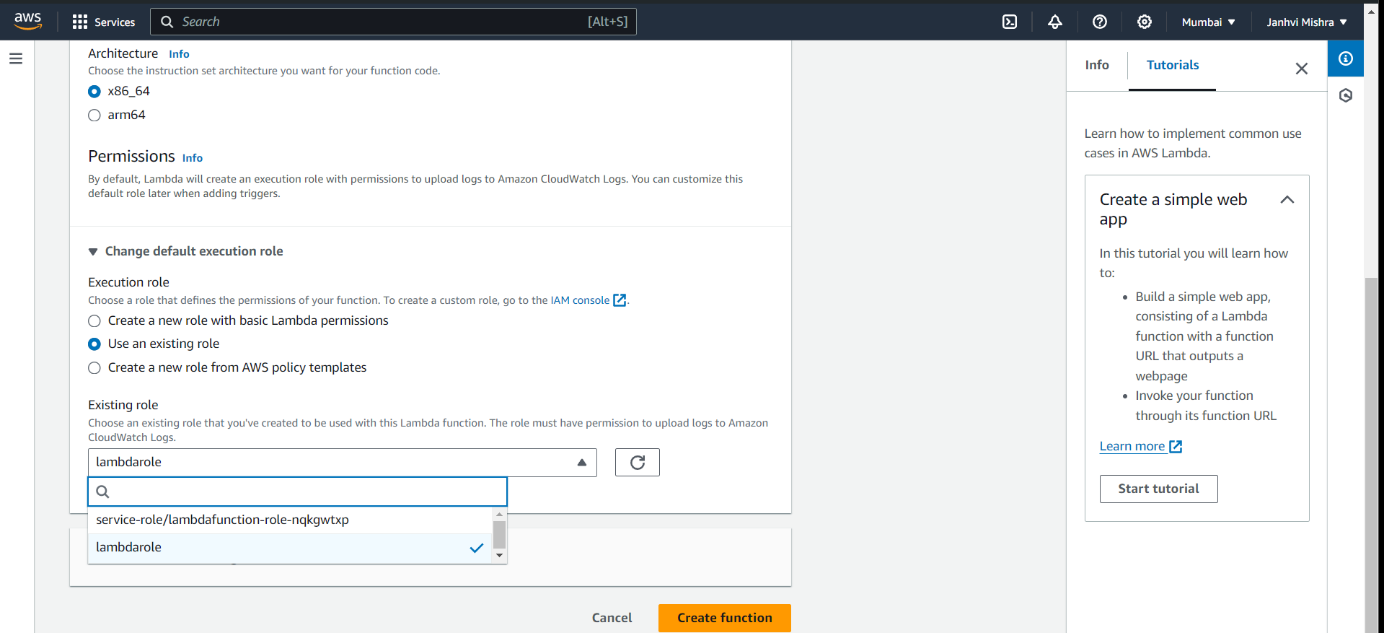


* Click on next button and add IAM role name as lambdarole.
* All setting will be default.Click on create role.

**Step 4 : Creating Lambda function.**

**For creating lambda function**

* Search lambda service in the search bar and click on create lambda function.
* Write name of function as lambda function. In runtime add node.js 18x.
* Then click on change default execution role and click on use existing role and then add your role.



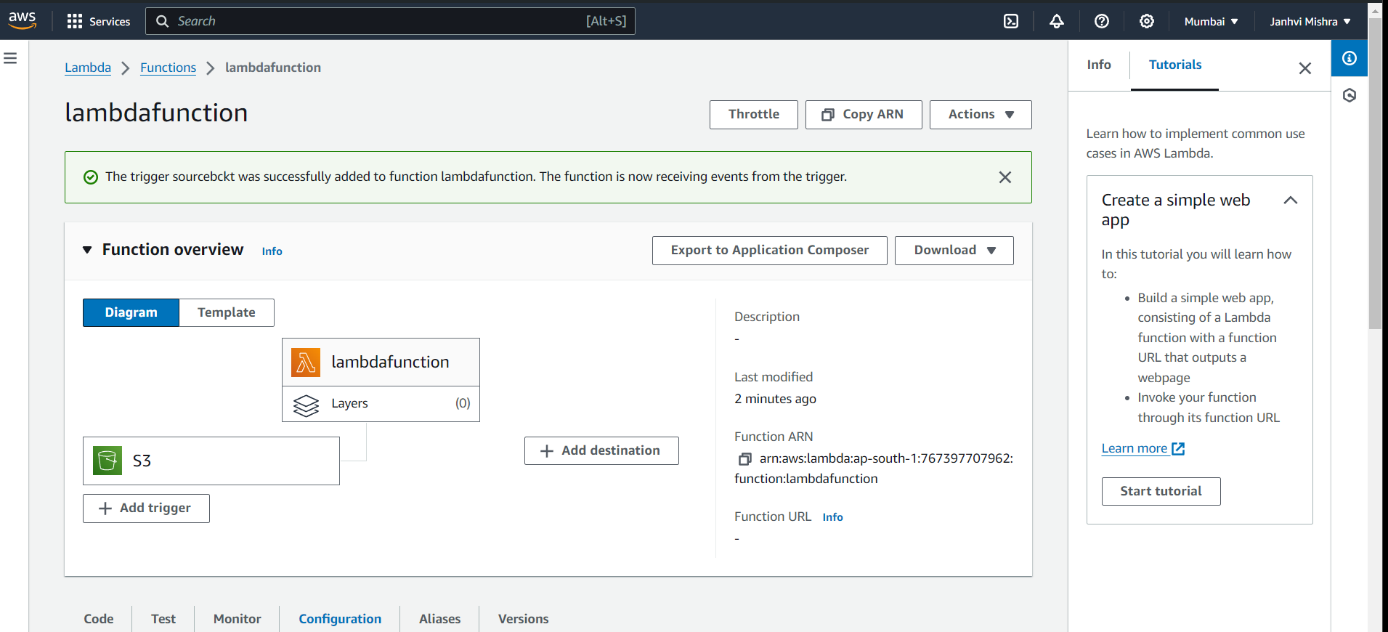
* Click on create function.

**Step 5 : Creating S3 trigger.**

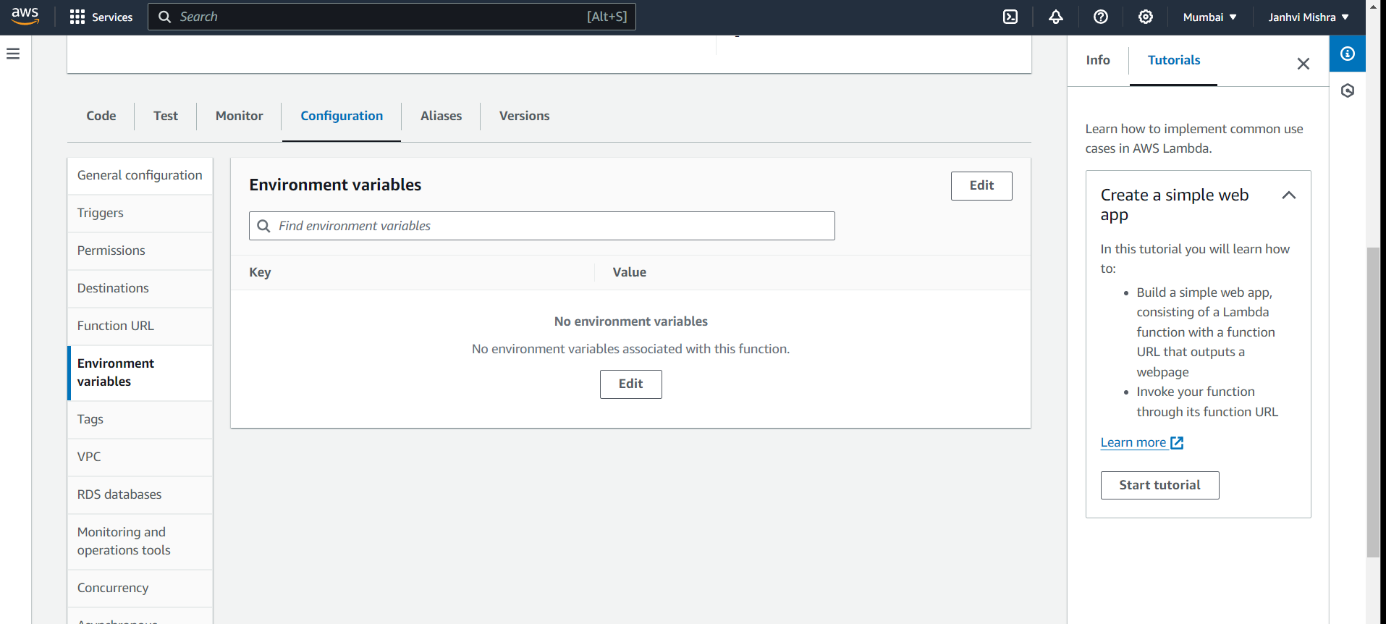
* Now we have to add trigger.
* In trigger configuration add S3 as a source and now add your source bucket and then check I acknowledge box
* Click on add.



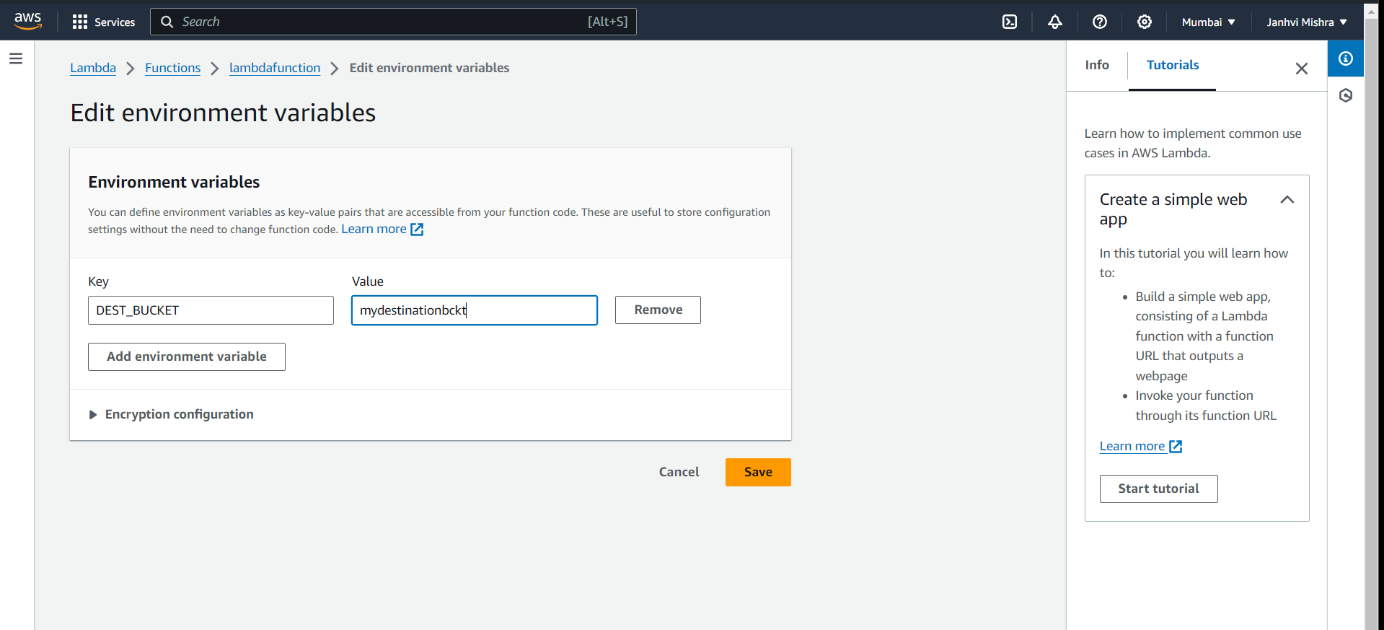
**Your trigger will be added**

****

* Now in the configuration section go to environment variables. And click on edit.

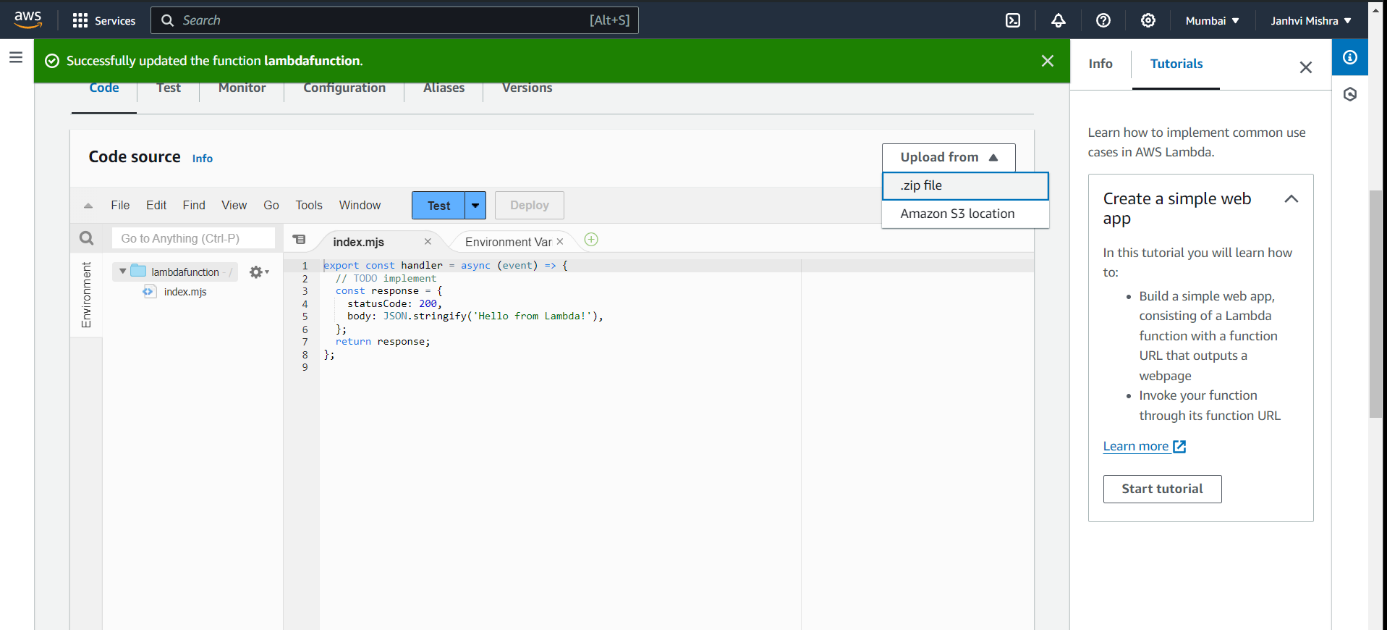


* Click on add environment variable and then write down key as DEST\_BUCKET and values as your destination bucket name mydestinationbckt.

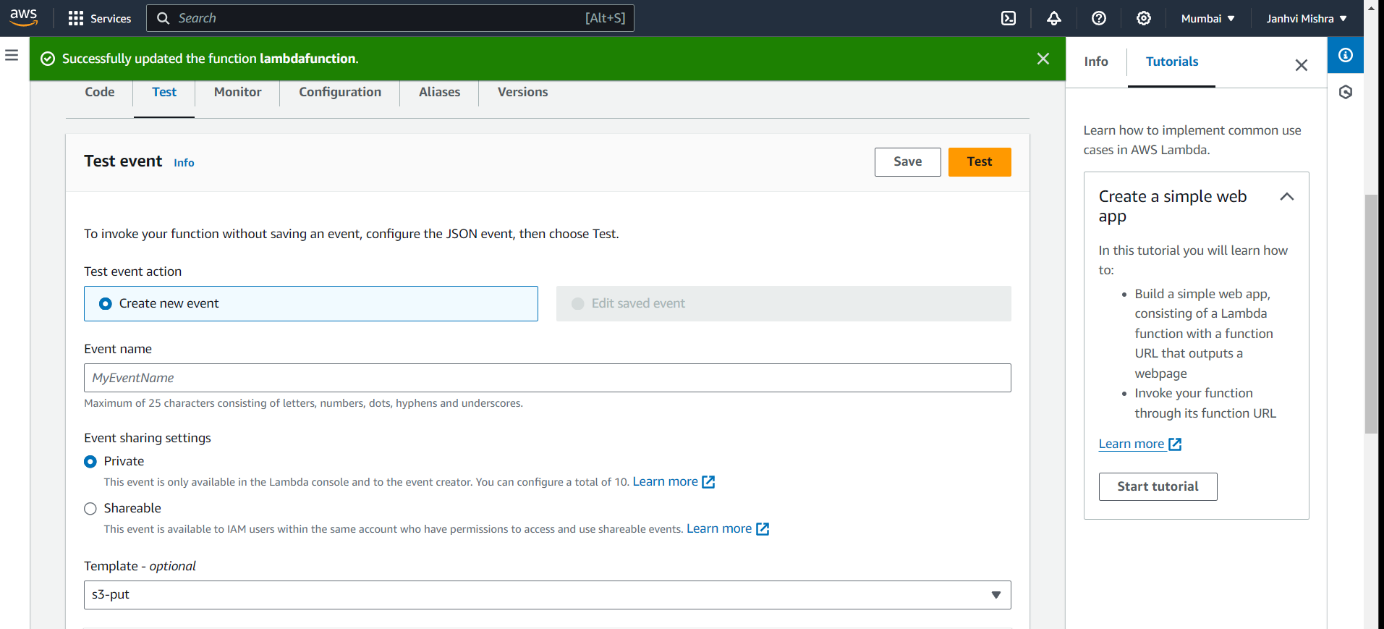


* Click on save button.
* Then click on code button and upload from a .zip file given below.

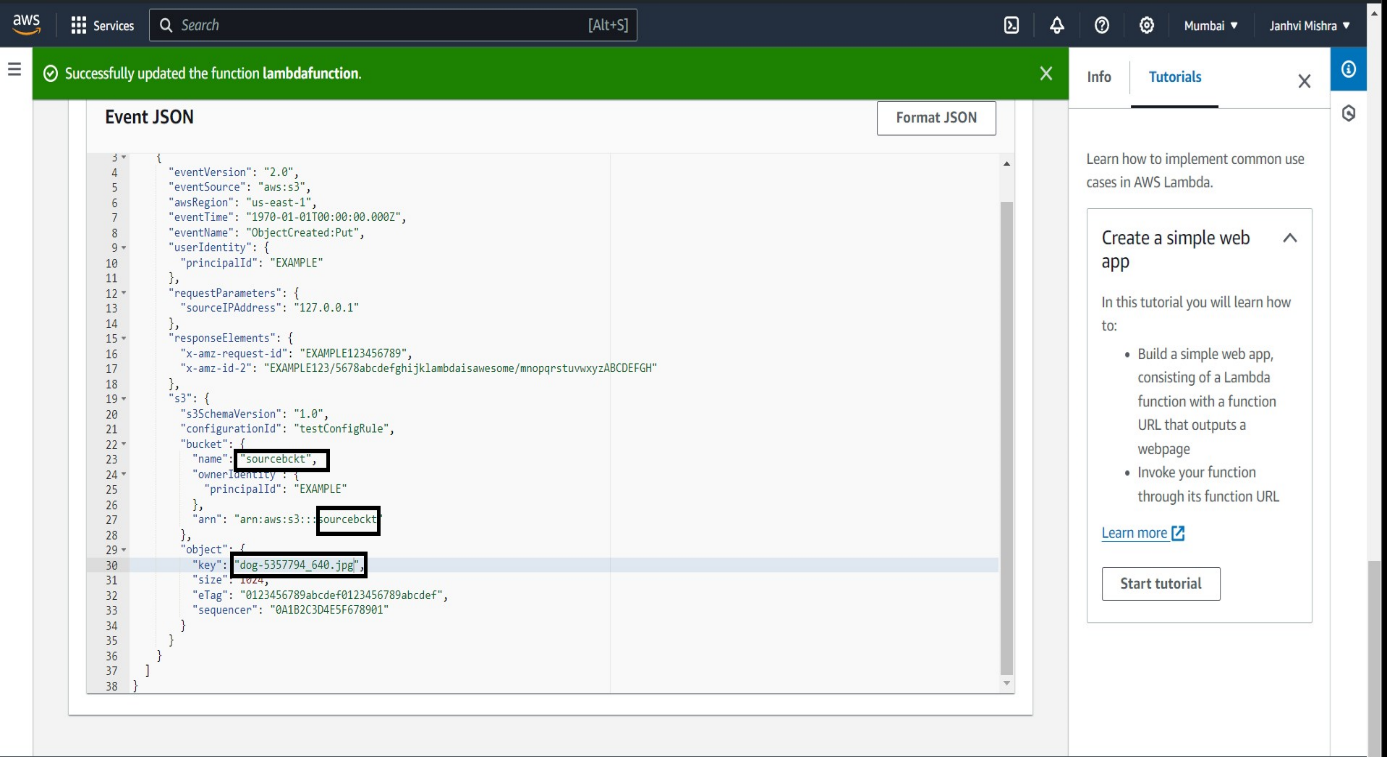




* Click on save button.
* Then click on Test button and in template add S3 Put.



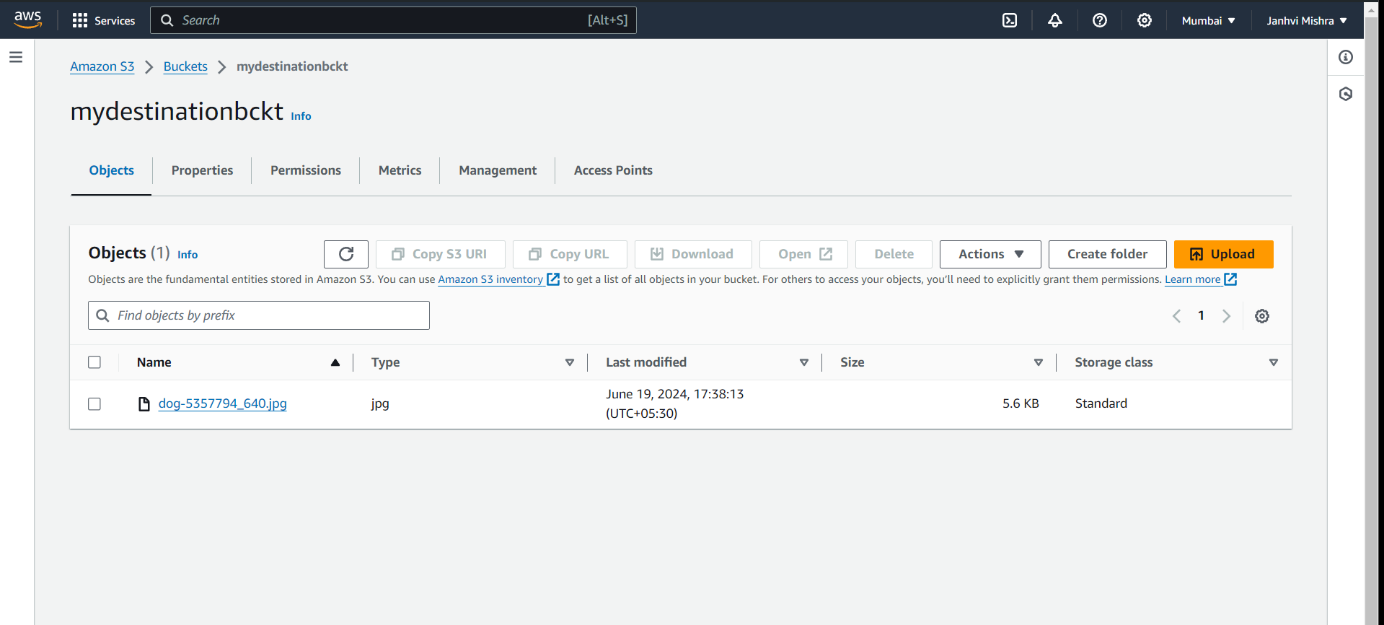
* In Event JSON a code will be generated.
* In line 23 and 27 write source bucket name as sourcebckt and in line 30 write the name of the image uploaded in source bucket.



* Then click on test. Check the details under executing function: succeeded.

**Step 6 : Testing the application.**

**If there is no error under details then go to destination bucket. You will se that bucket which was uploaded in source bucket is now visible in your destination bucket.**

****

* Then click on the link of the image and click on open. You will see your image which will be in smaller size.

