

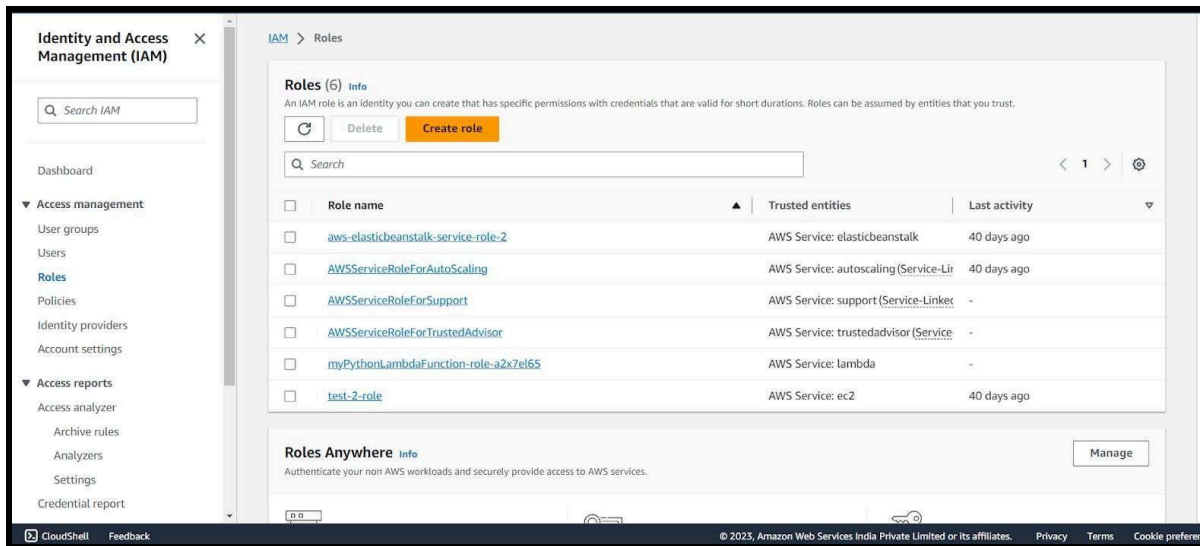
Adv. DevOps Exp. 12

Vaishnal Mali

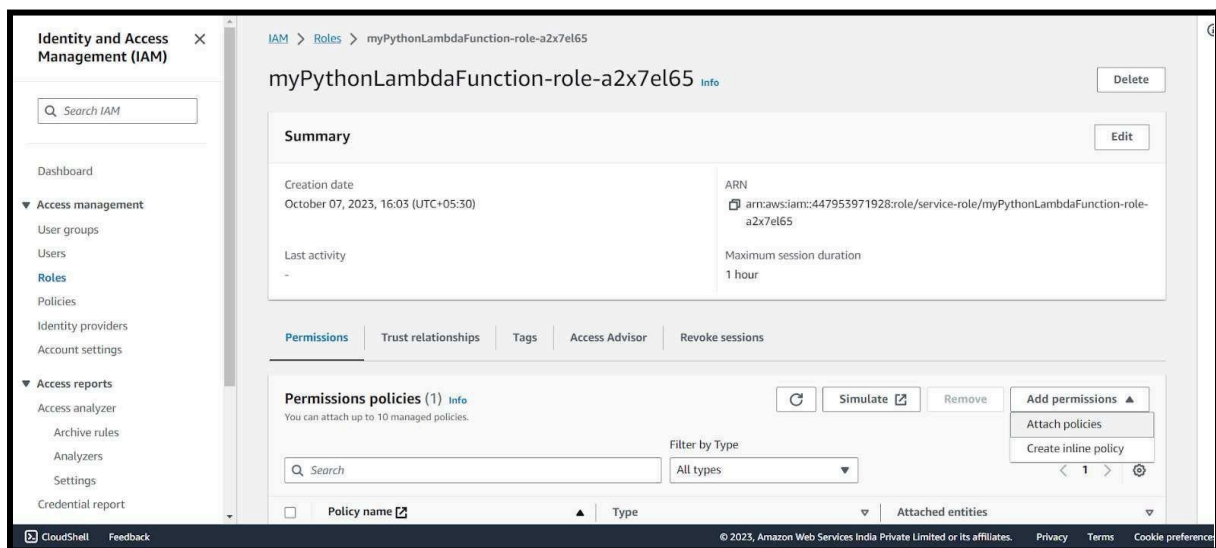
D15A - 27

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

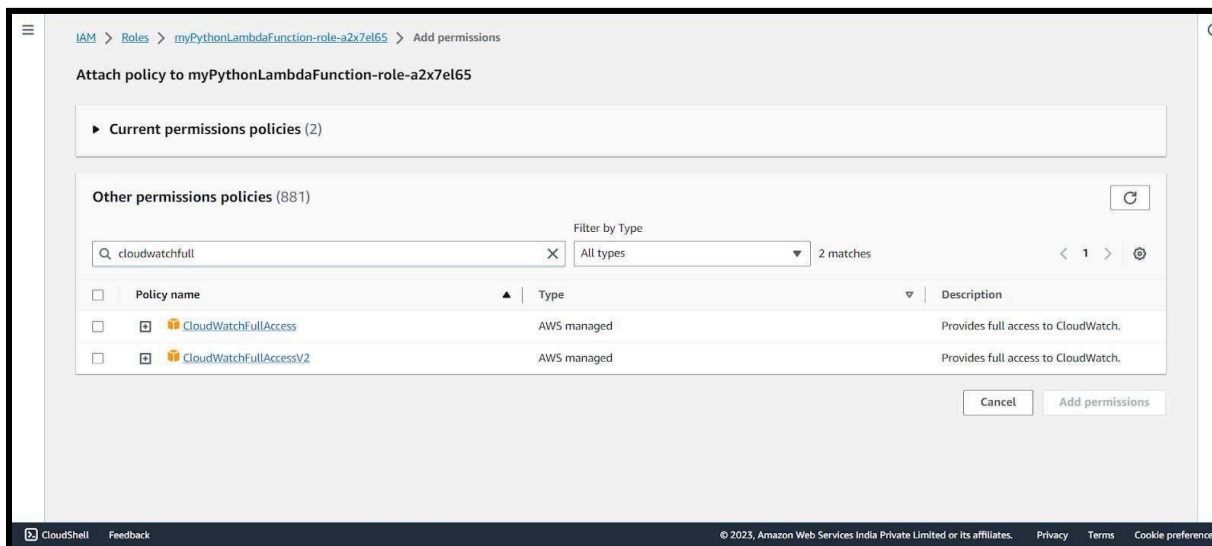
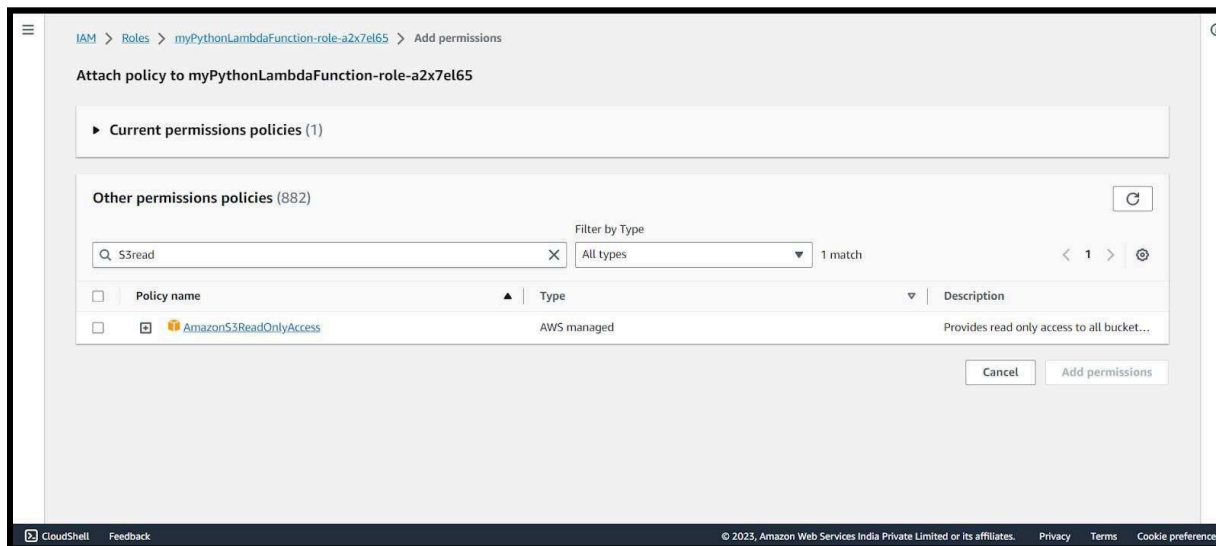
Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.

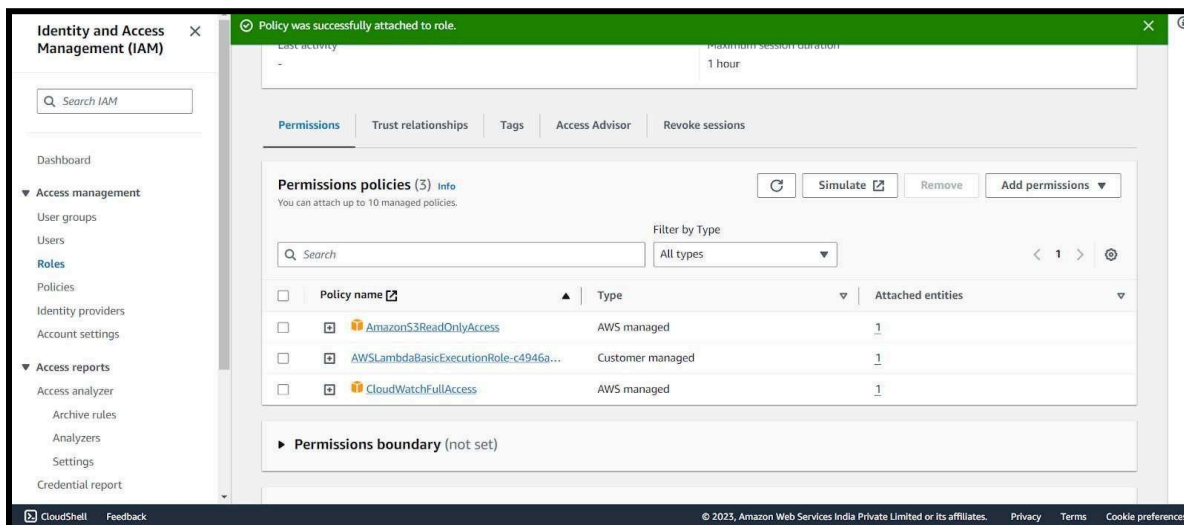


S3-ReadOnly



CloudWatchFull

After successful attachment of policy you will see something like this you will be able to see the updated policies.



Create function [Info](#)

AWS Serverless Application Repository applications have moved to Create application.

☒ **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.
AdvDevops-ex12
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.11

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.

Step 3: Open up AWS Lambda and create a new Python function.

Under Execution Role, choose the existing role, then select the one which was previously created and to which we just added permissions.

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

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☐ Create a new role with basic Lambda permissions

☒ **Use an existing role**

☐ Create a new role from AWS policy templates

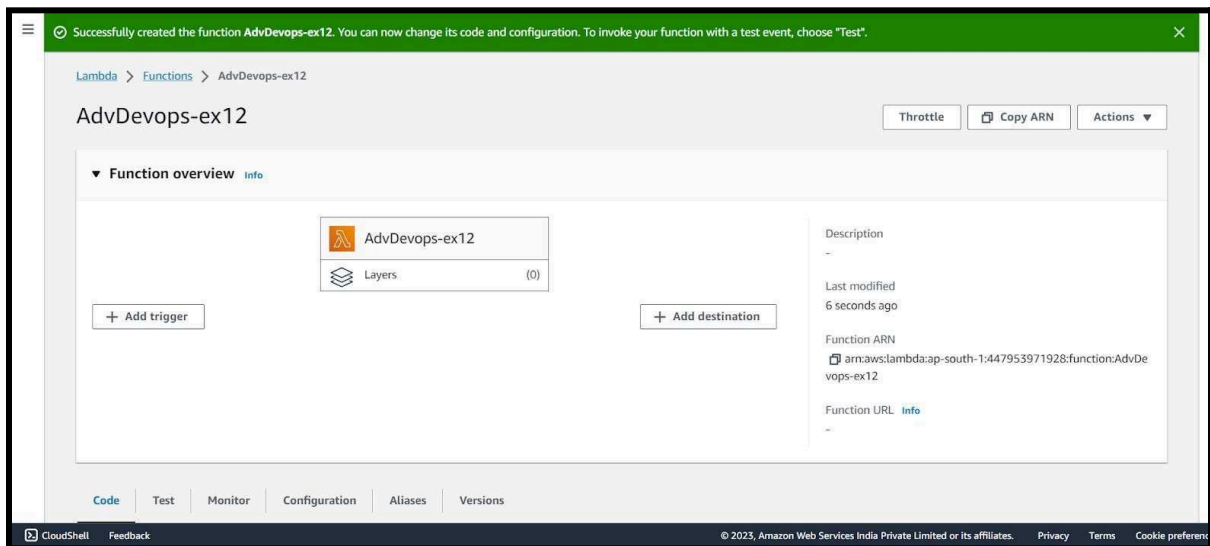
Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
service-role/myPythonLambdaFunction-role-a2x7el65

[View the myPythonLambdaFunction-role-a2x7el65 role](#) on the IAM console.

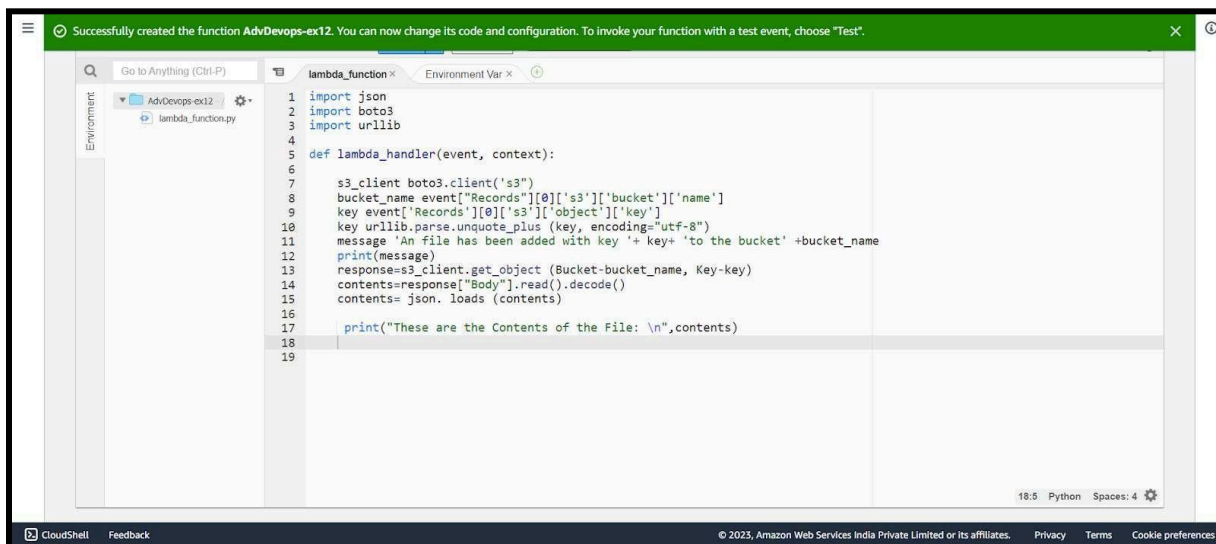
Advanced settings

[Cancel](#) [Create function](#)

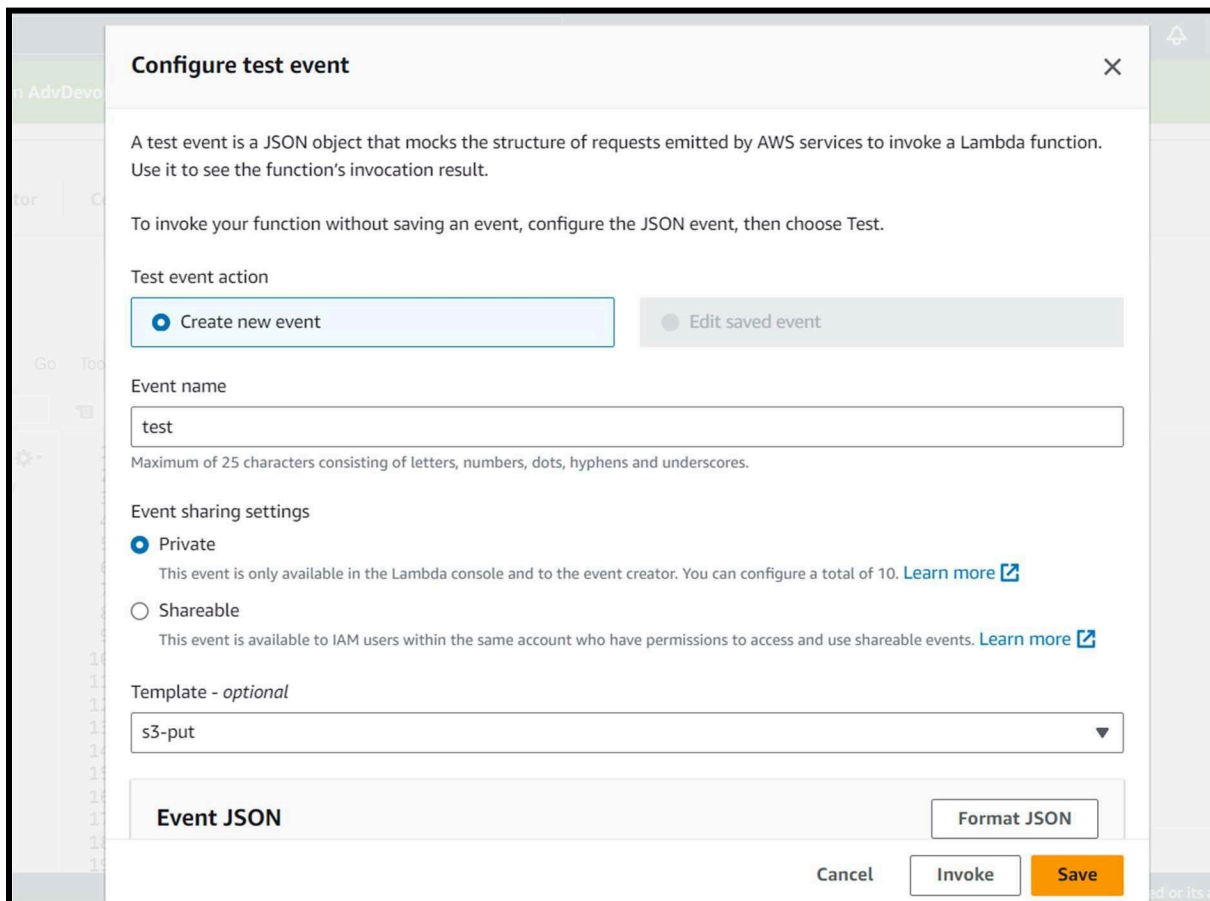
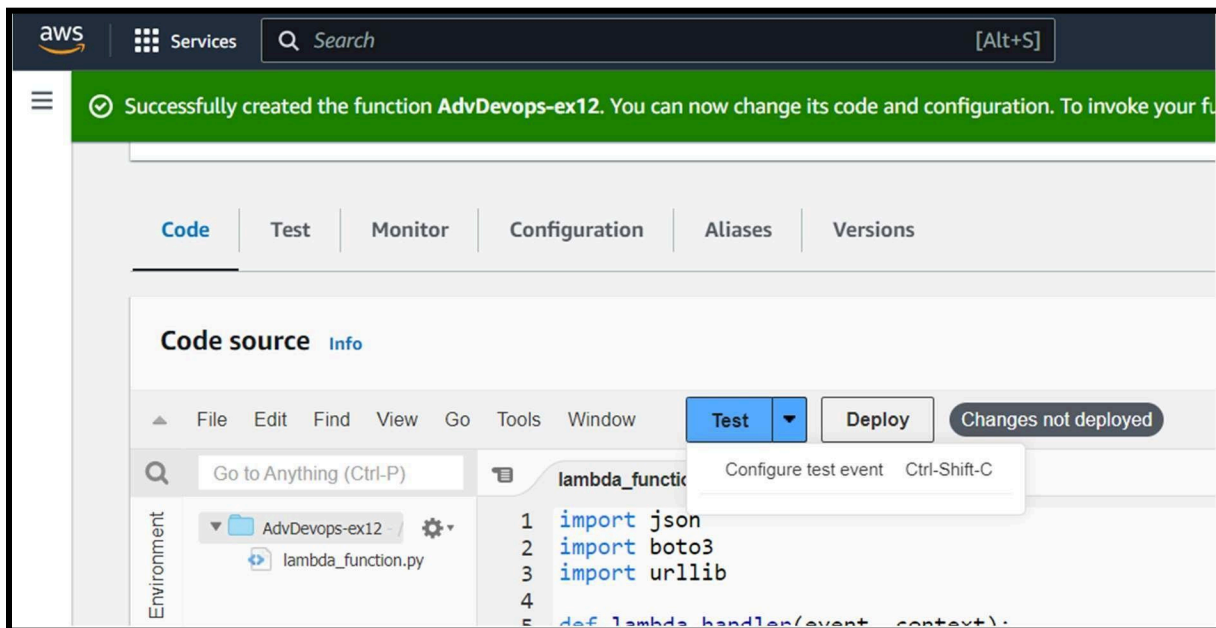
Step 4: The function is up and running.



Step 5: Make the following changes to the function and click on the deploy button.
This code basically logs a message and logs the contents of a JSON file which is uploaded to an S3 Bucket and then deploy the code.

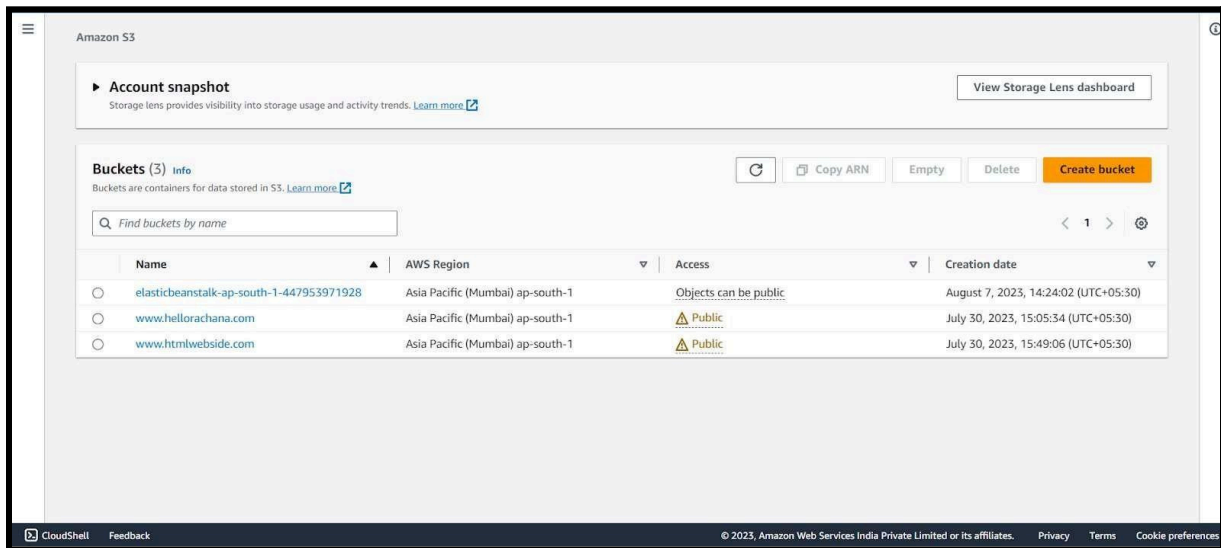


Step 6: Click on Test and choose the 'S3 Put' Template.

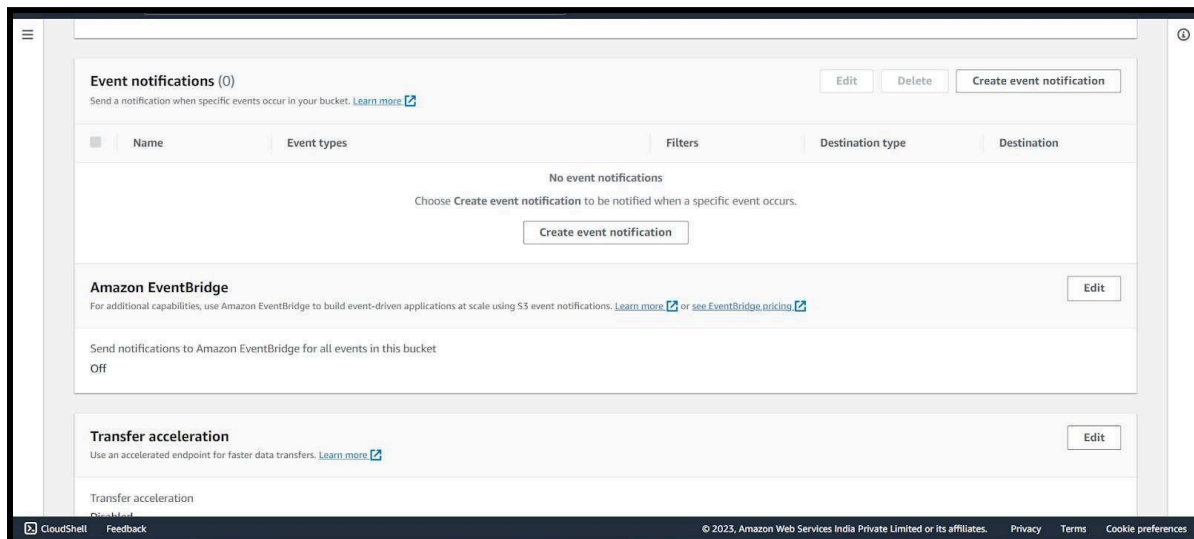
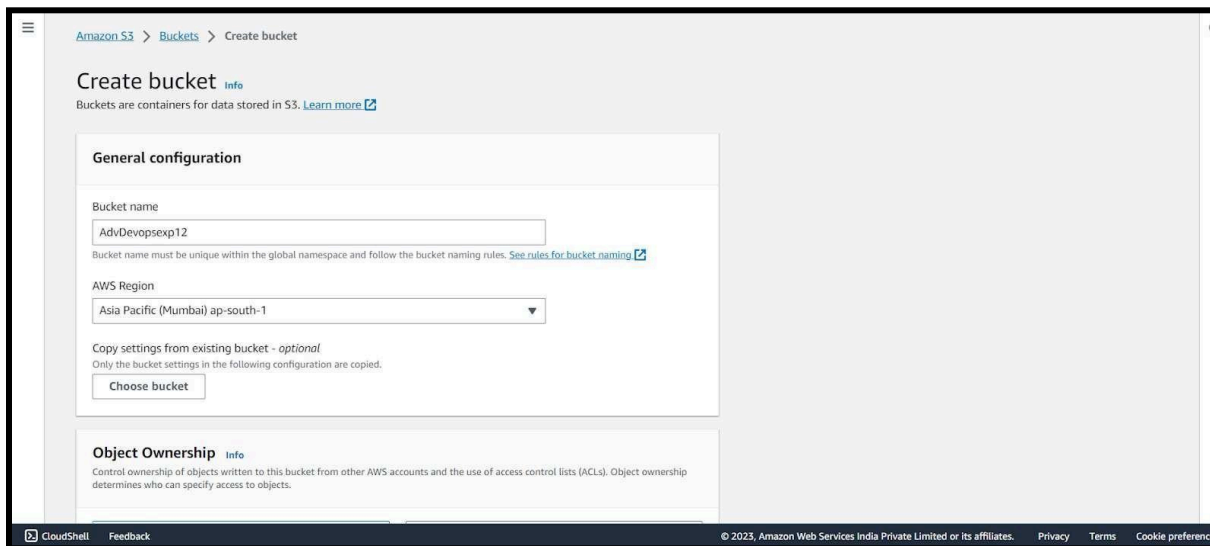


And Save it.

Step 7: Open up the S3 Console and create a new bucket.



Step 8: With all general settings, create the bucket in the same region as the function.



Step 9: Click on the created bucket and under properties, look for events.

Click on Create Event Notification.

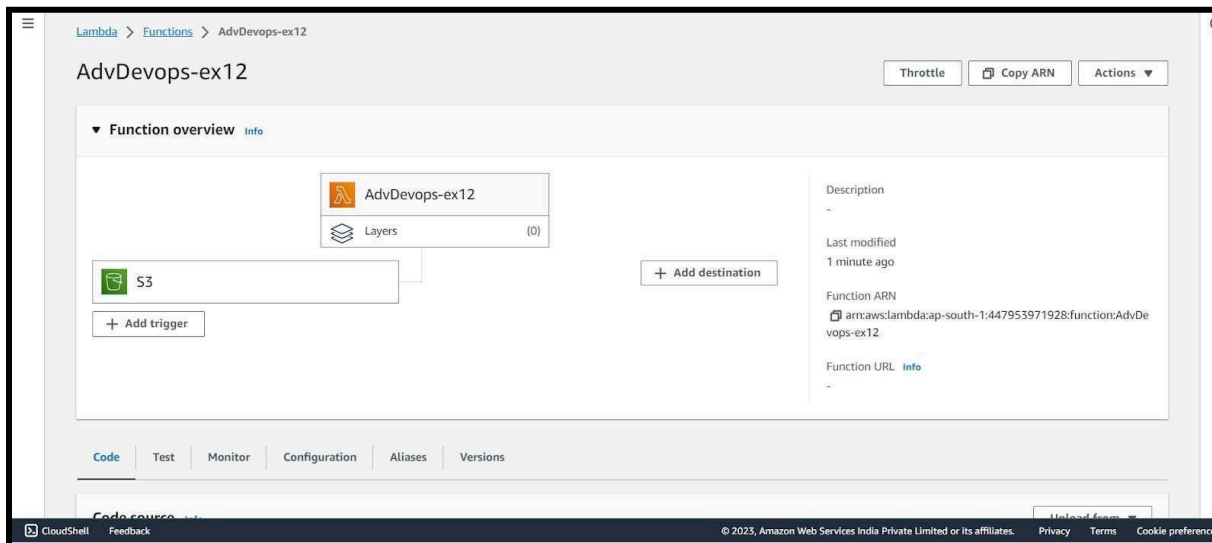
The screenshot shows the 'General configuration' page in the AWS Event Notifications console. The 'Event name' field contains 'S3putrequest'. The 'Prefix - optional' field contains 'images/'. The 'Suffix - optional' field contains '.jpg'. Under 'Event types', the 'Object creation' section has 'Put' checked, with 's3:ObjectCreated:Put' listed below it. The 'Post' option is unchecked. The footer shows 'CloudShell', 'Feedback', and '© 2023, Amazon Web Services India Private Limited'.

Step 10: Mention an event name and check Put under event types.

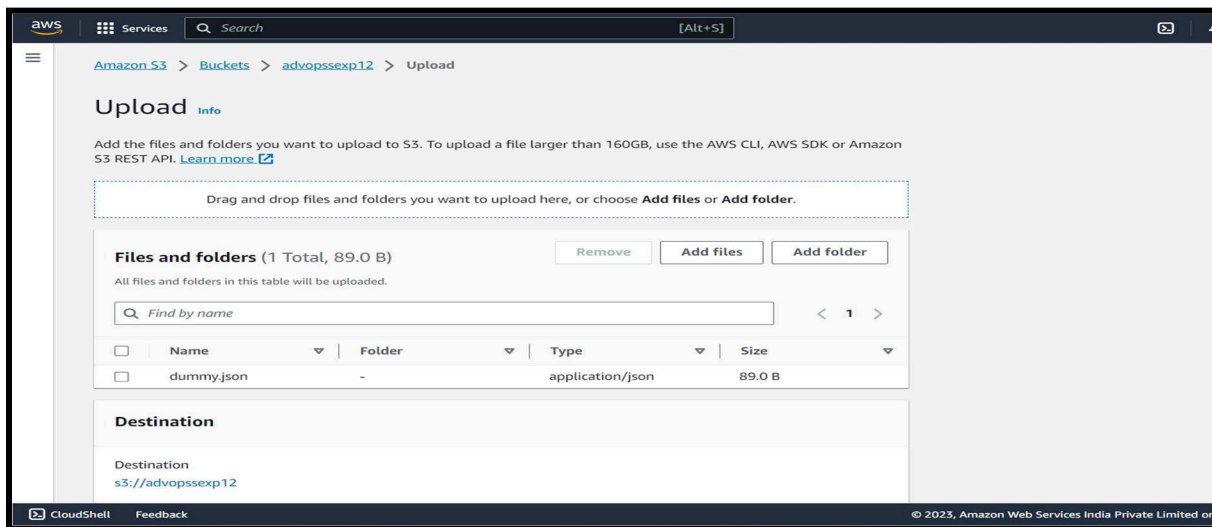
Choose Lambda function as destination and choose your lambda function and save the changes.

The screenshot shows the 'Destination' page in the AWS Event Notifications console. A blue information box at the top states: 'Before Amazon S3 can publish messages to a destination, you must grant the Amazon S3 principal the necessary permissions to call the relevant API to publish messages to an SNS topic, an SQS queue, or a Lambda function. [Learn more](#)'. Under 'Destination', 'Lambda function' is selected with a radio button. Below it, 'Specify Lambda function' has 'Choose from your Lambda functions' selected. The 'Lambda function' dropdown menu shows 'AdvDevops-ex12'. At the bottom right are 'Cancel' and 'Save changes' buttons. The footer shows 'CloudShell', 'Feedback', and '© 2023, Amazon Web Services India Private Limited'.

Step 11: Refresh the Lambda function console and you should be able to see an S3 Trigger in the overview.



Step 12: Go back to your S3 Bucket and click on Add Files to upload a new file. Select the dummy data file from



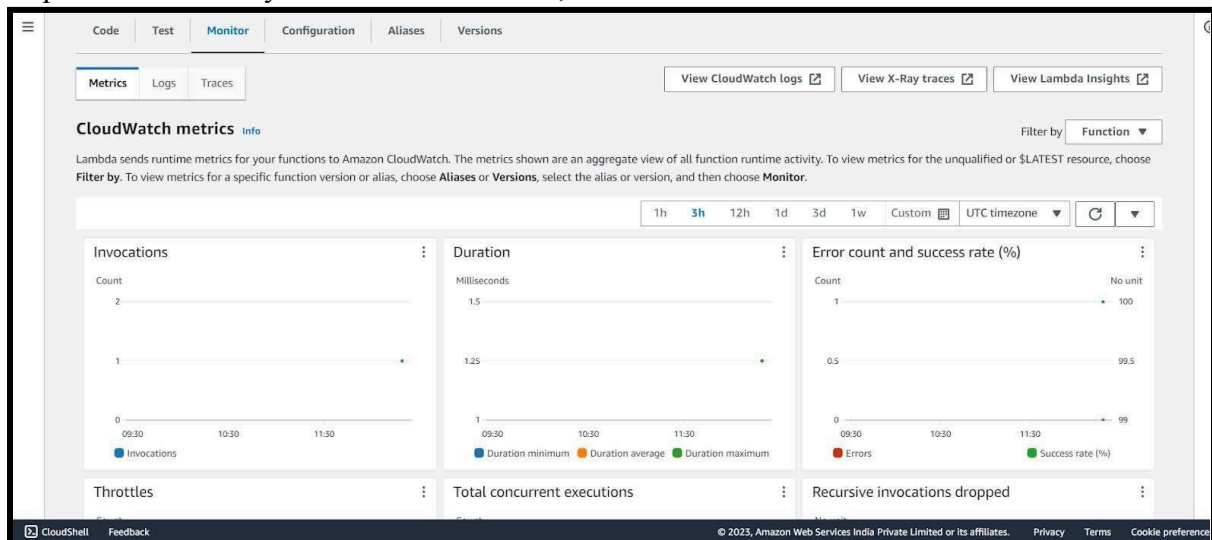
on your computer and click Upload.

Step 13: After this make the necessary changes in the Test configuration file which we created it previously by replacing the Bucket Name and the ARN of Bucket.

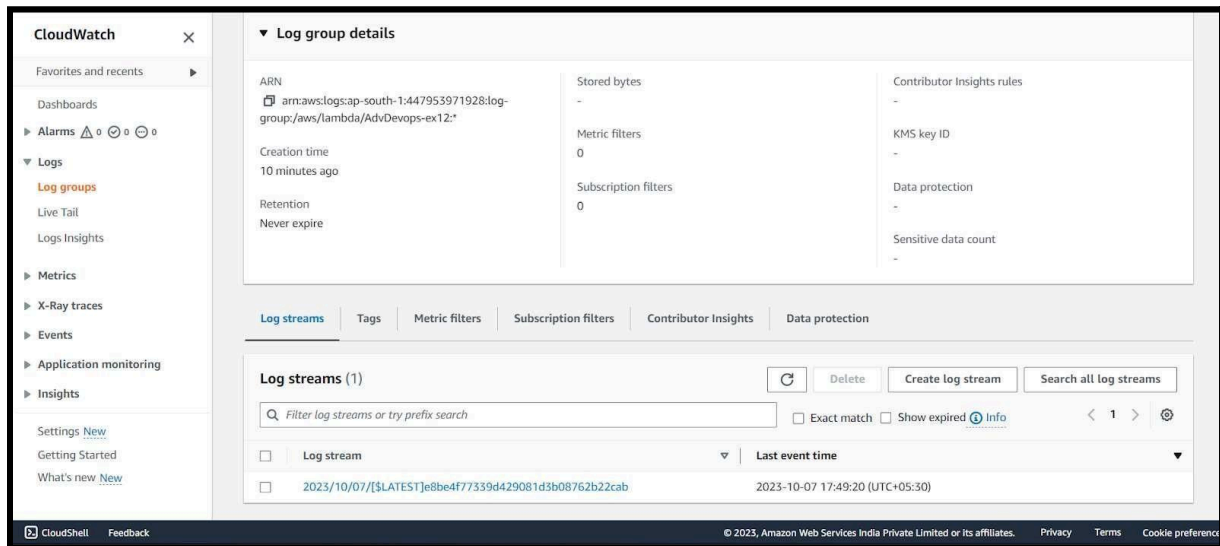
```
Event JSON
Format JSON

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": "EXAMPLE123/5678abcdefgghijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
18  },
19  "s3": {
20    "s3SchemaVersion": "1.0",
21    "configurationId": "testConfigRule",
22    "bucket": {
23      "name": "advopssexp12",
24      "ownerIdentity": {
25        "principalId": "EXAMPLE"
26      },
27      "arn": "arn:aws:s3:::advopssexp12"
28    },
29    "object": {
30      "key": "test%2Fkey",
31      "size": 1024,
32      "eTag": "0123456789abcdef0123456789abcdef",
33      "sequencer": "0A1B2C3D4E5F678901"
34    }
35  }
36 }
37 ]
38 }
```

Step 14: Go back to your Lambda function , Refresh it and ch



click the Monitor tab.



Under Log streams, click on View logs in Cloudwatch to check the Function logs.

Step 15: Click on this log Stream that was created to view what was logged by your function.

