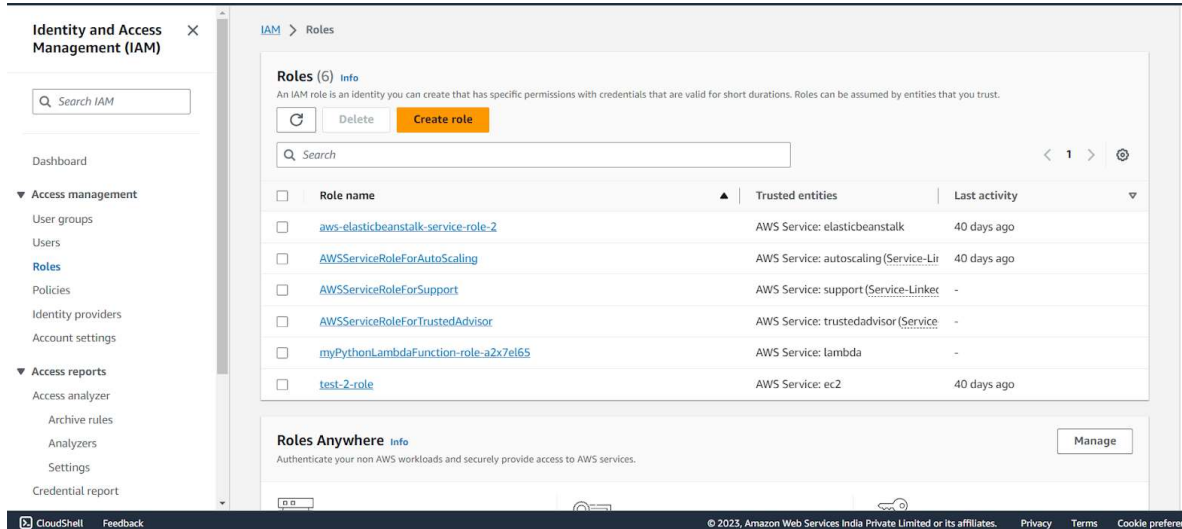


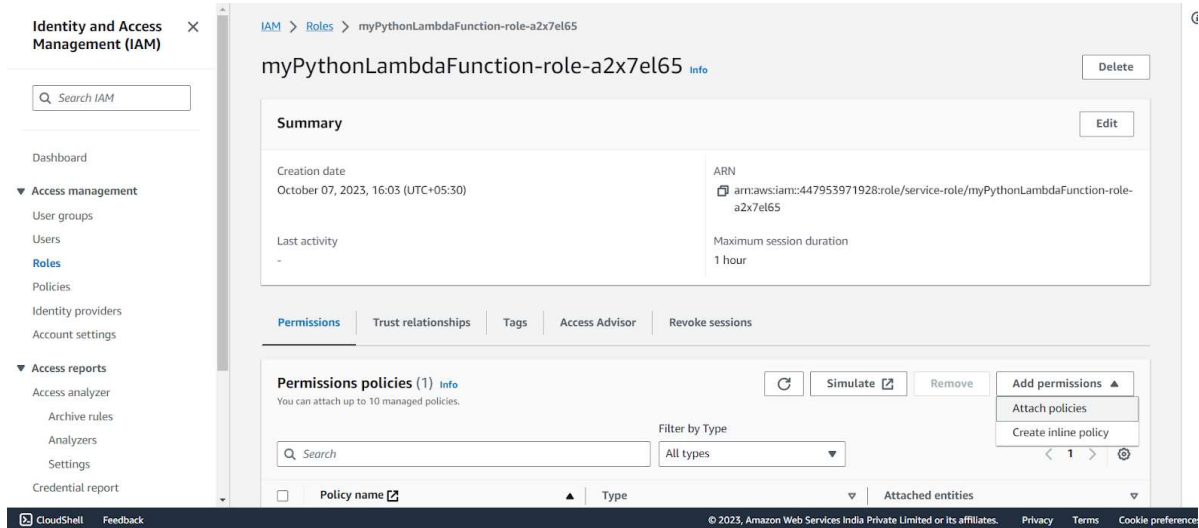
Experiment No 12

Prathamesh Palve
D15A 32

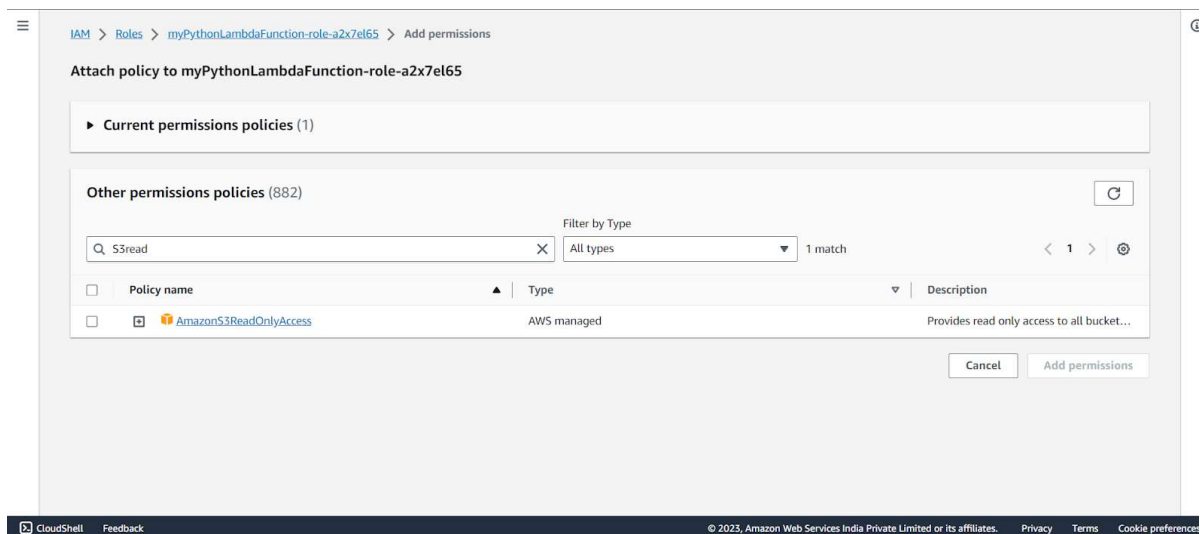
Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



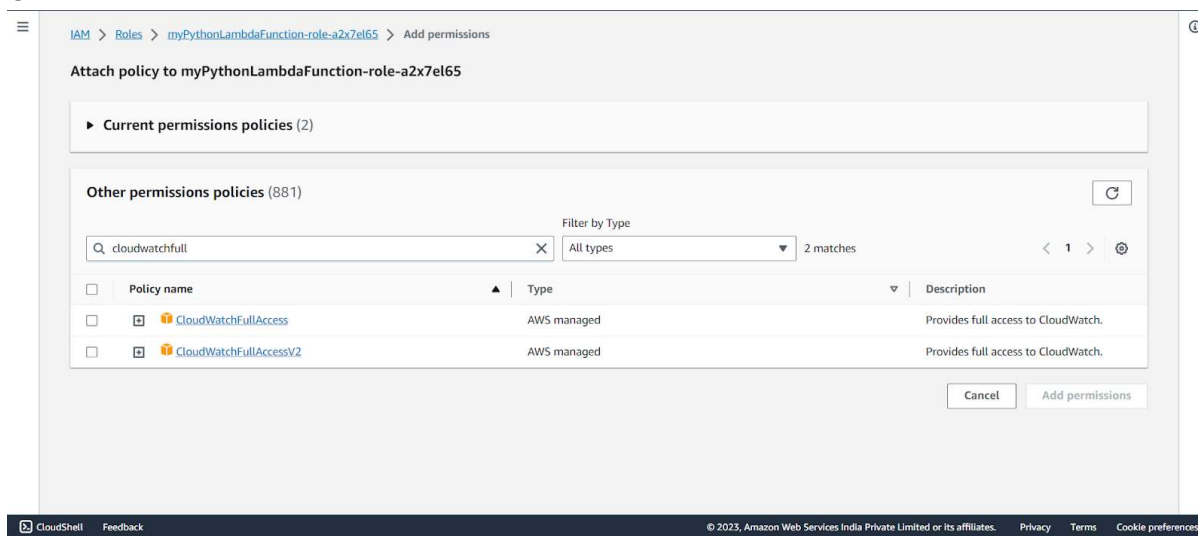
Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.



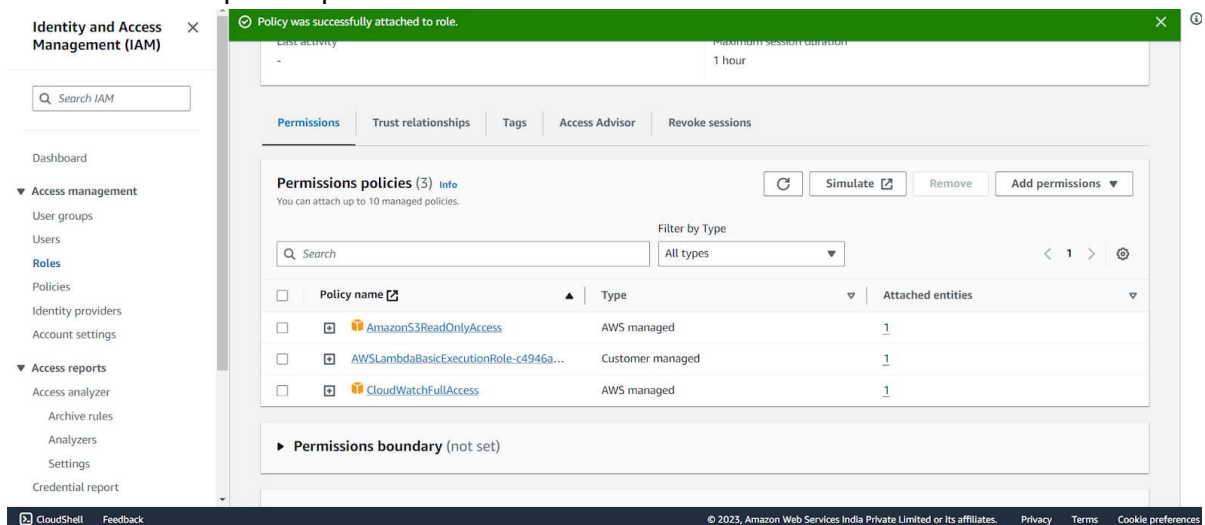
S3-ReadOnly



CloudWatchFull



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



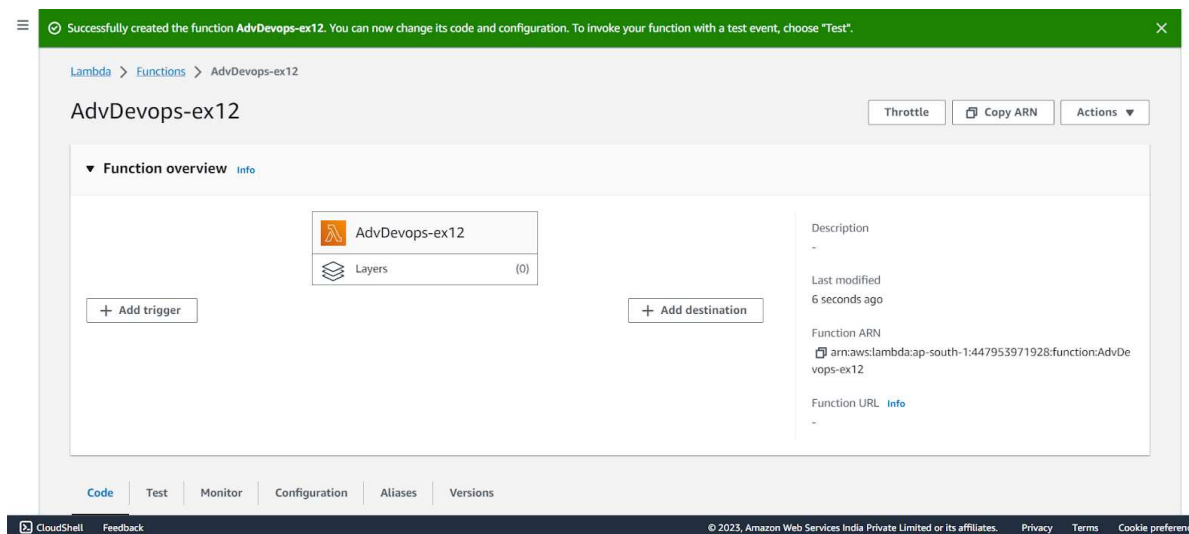
Step 3: Open up AWS Lambda and create a new Python function.

The screenshot shows the 'Create function' page in the AWS Lambda console. At the top, there are three tabs: 'Author from scratch' (selected), 'Use a blueprint', and 'Container image'. Below these, the 'Basic information' section is visible. It includes a 'Function name' field with the value 'AdvDevOps-ex12', a 'Runtime' dropdown set to 'Python 3.11', and an 'Architecture' dropdown set to 'x86_64'. The 'Permissions' section is partially visible at the bottom.

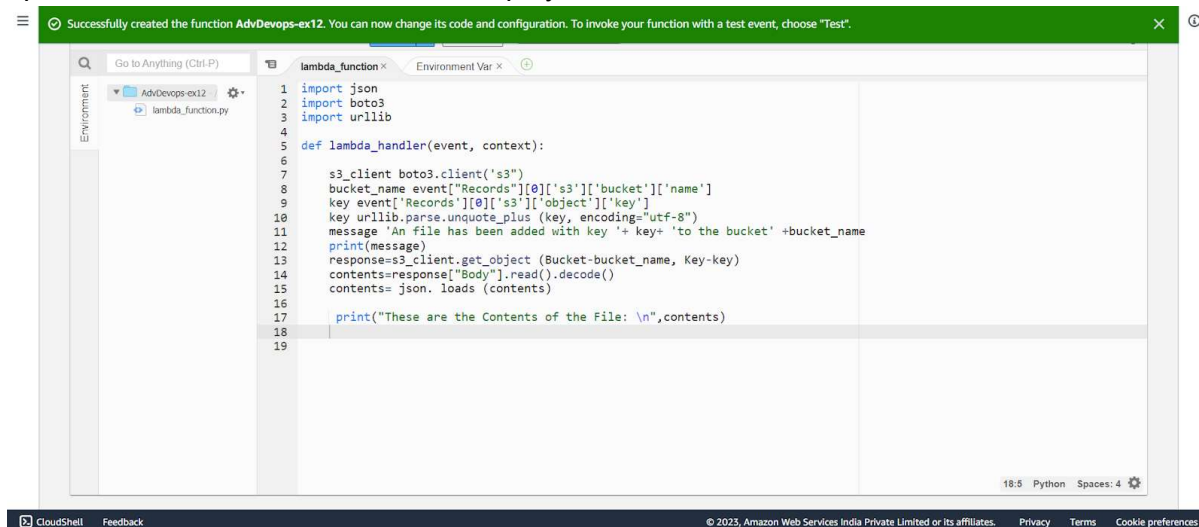
Under Execution Role, choose the existing role, then select the one which was previously created and to which we just added permissions.

This screenshot shows the 'Permissions' section of the 'Create function' page. It includes a 'Change default execution role' section with three radio buttons: 'Create a new role with basic Lambda permissions', 'Use an existing role' (selected), and 'Create a new role from AWS policy templates'. Below this, the 'Existing role' dropdown is set to 'service-role/myPythonLambdaFunction-role-a2x7el65'. At the bottom right, there are 'Cancel' and 'Create function' buttons.

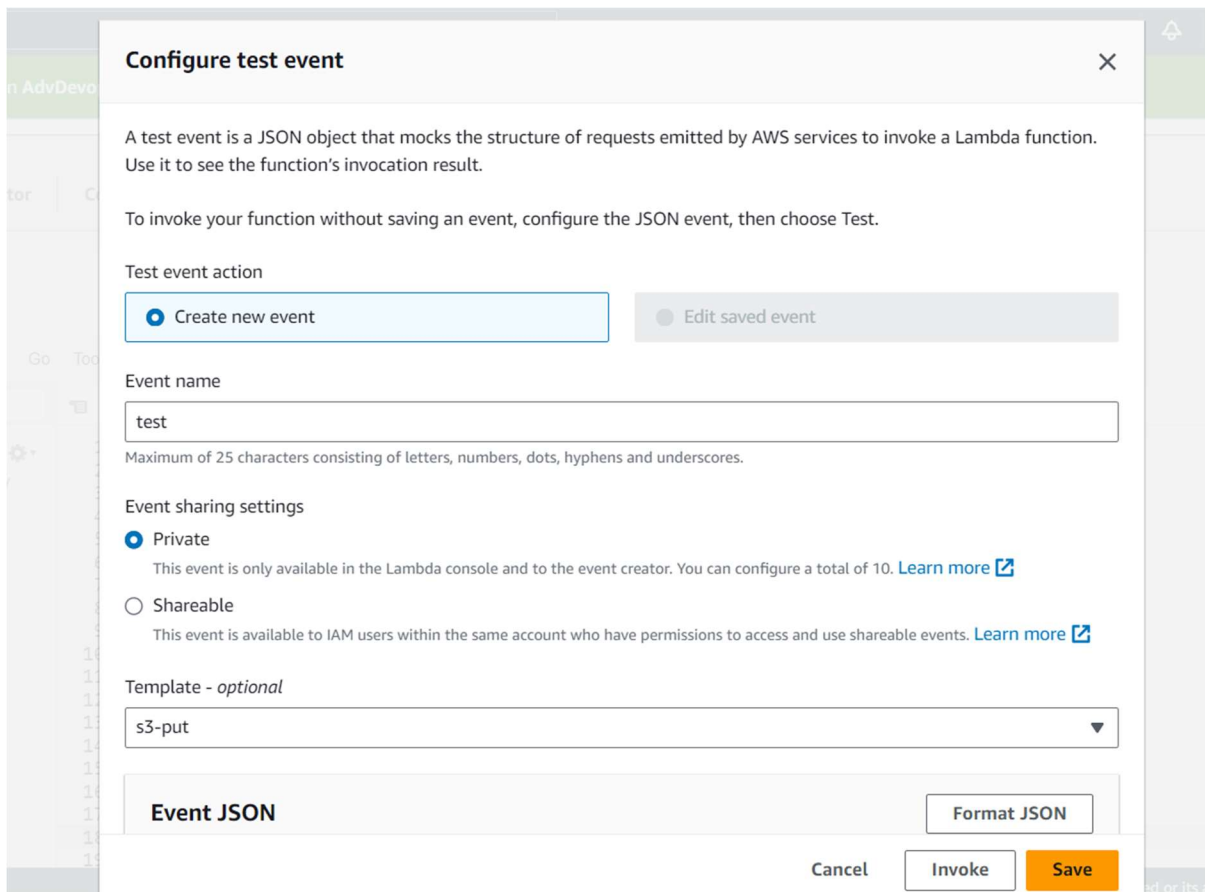
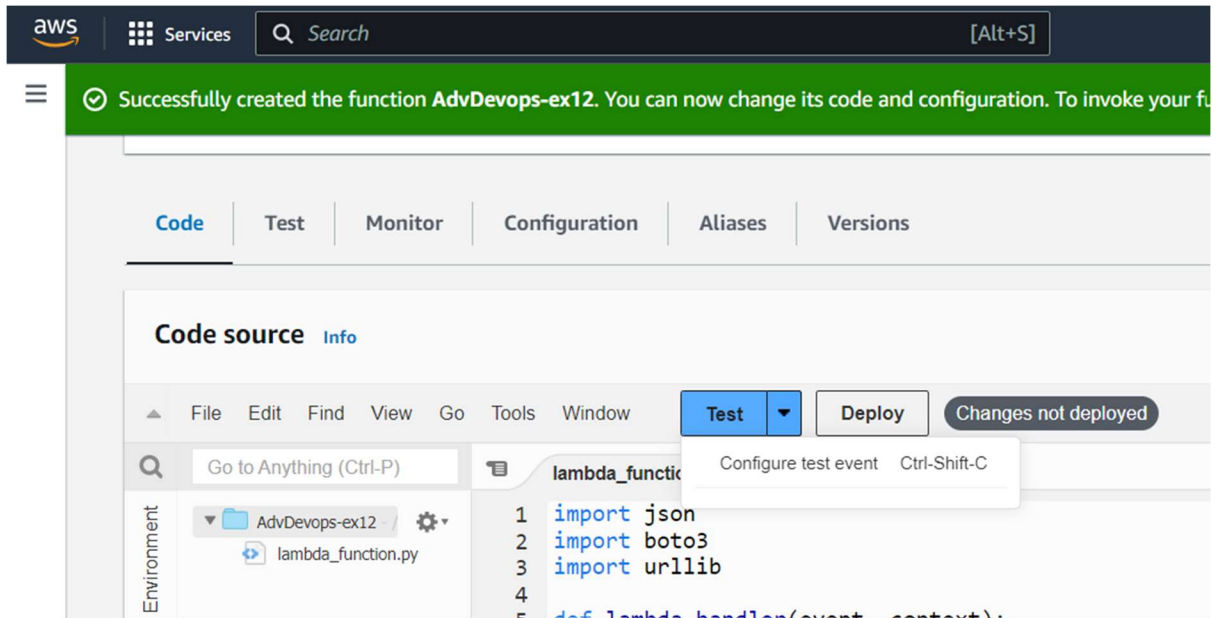
Step 4: The function is up and running.



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.

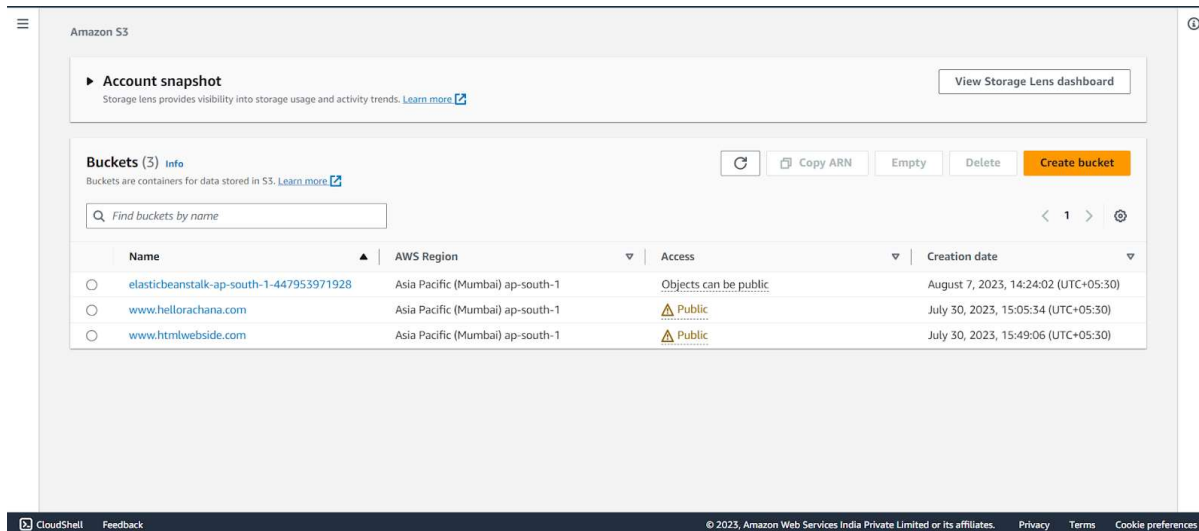


Step 6: Click on Test and choose the 'S3 Put' Template.

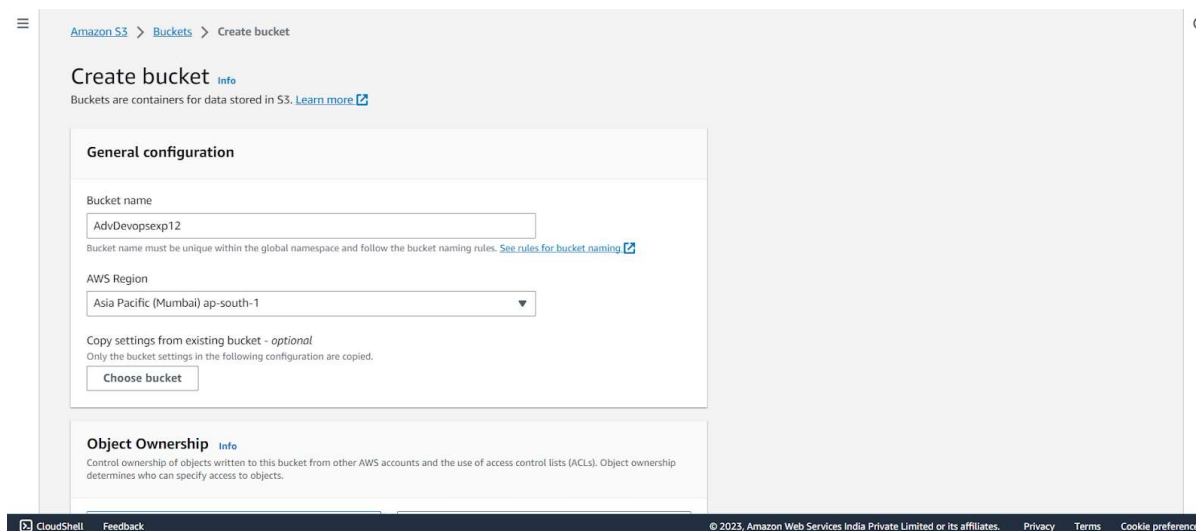


And Save it.

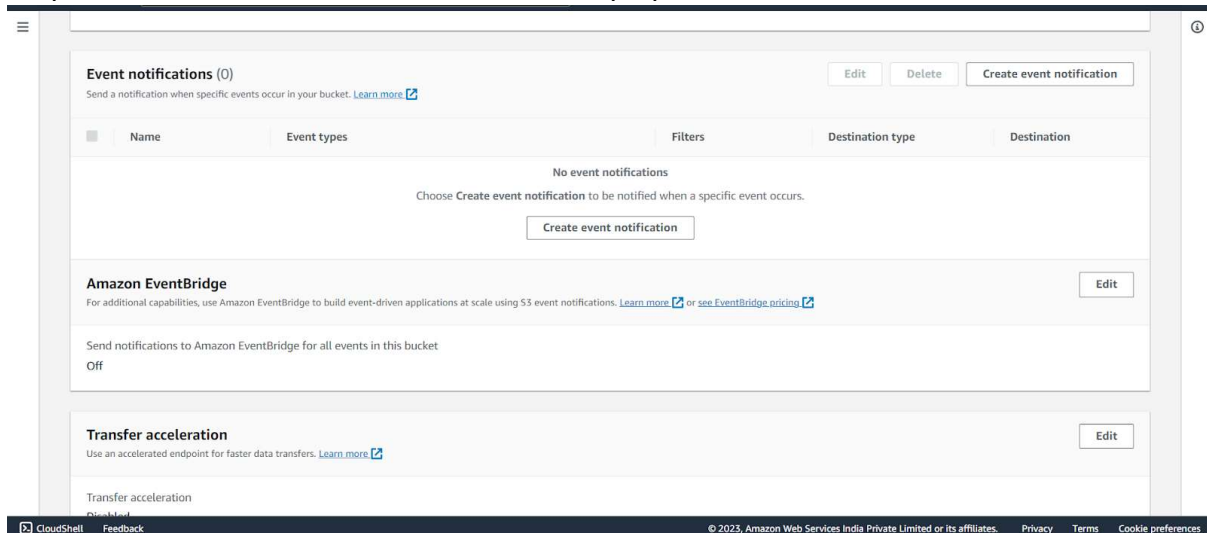
Step 7: Open up the S3 Console and create a new bucket.



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.

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

The screenshot shows the 'General configuration' section of the AWS S3 Event Notifications console. The 'Event name' field contains 'S3putrequest'. The 'Prefix - optional' field contains 'images/'. The 'Suffix - optional' field contains '.jpg'. Under the 'Event types' section, 'Object creation' is expanded, showing 'Put' selected with 's3:ObjectCreated:Put' and 'Post' with 's3:ObjectCreated:Post'.

General configuration

Event name
S3putrequest
Event name can contain up to 255 characters.

Prefix - *optional*
Limit the notifications to objects with key starting with specified characters.
images/

Suffix - *optional*
Limit the notifications to objects with key ending with specified characters.
.jpg

Event types
Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.

Object creation

☐ All object create events
s3:ObjectCreated:*

☒ Put
s3:ObjectCreated:Put

☐ Post
s3:ObjectCreated:Post

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

The screenshot shows the 'Destination' section of the AWS S3 Event Notifications console. A blue information box states that permissions must be granted to the Amazon S3 principal. The 'Destination' section has 'Lambda function' selected. Under 'Specify Lambda function', 'Choose from your Lambda functions' is selected. The 'Lambda function' dropdown menu shows 'AdvDevops-ex12'.

Destination

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](#)

Destination
Choose a destination to publish the event. [Learn more](#)

☒ **Lambda function**
Run a Lambda function script based on S3 events.

☐ **SNS topic**
Fanout messages to systems for parallel processing or directly to people.

☐ **SQS queue**
Send notifications to an SQS queue to be read by a server.

Specify Lambda function

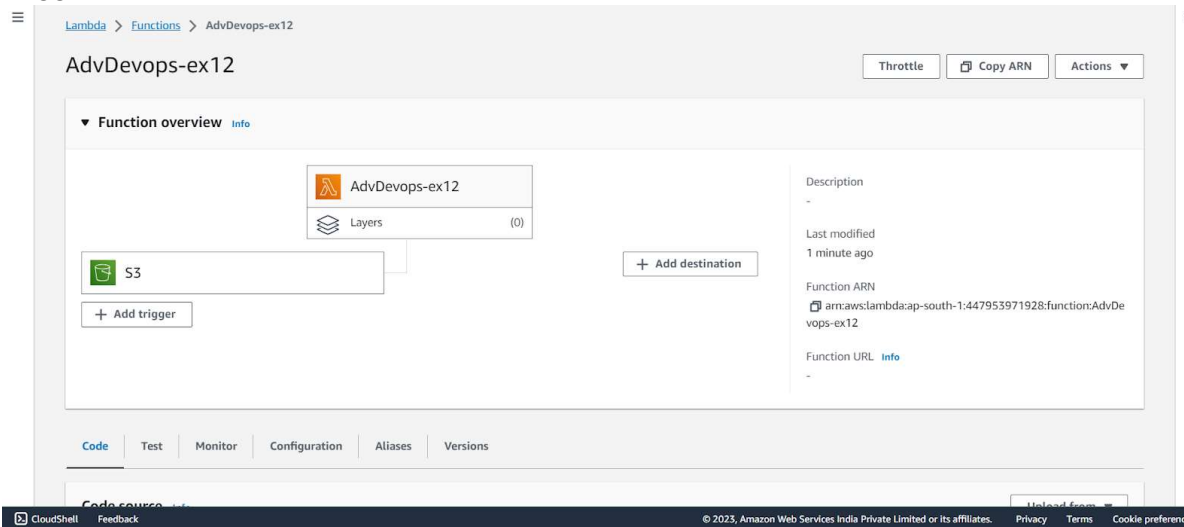
☒ **Choose from your Lambda functions**

☐ Enter Lambda function ARN

Lambda function
AdvDevops-ex12

Cancel Save changes

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

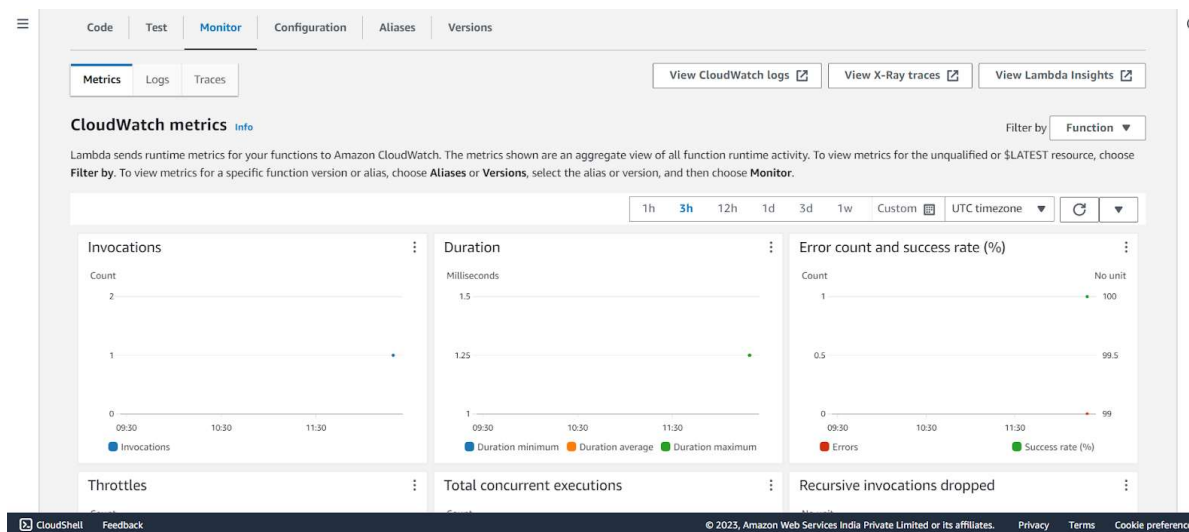


Step 12: Now, create a dummy JSON file locally.

```
{ } dummy.json X
{ } dummy.json > ...
1  {
2    "firstname" : "Shashwat",
3    "lastname"  : "Tripathi",
4    "gender"   : "Male",
5    "age": 19
6  }
```

Step 13: Go back to your S3 Bucket and click on Add Files to upload a new file.

Step 14: Select the dummy data file from your computer and click Upload.



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

The screenshot shows the AWS CloudWatch Log group details page. The Log group details are as follows:

- ARN:** arn:aws:logs:ap-south-1:447953971928:log-group:/aws/lambda/AdvDevops-ex12:*
- Stored bytes:** -
- Metric filters:** 0
- Subscription filters:** 0
- Creation time:** 10 minutes ago
- Retention:** Never expire
- Contributor Insights rules:** -
- KMS key ID:** -
- Data protection:** -
- Sensitive data count:** -

The Log streams tab is selected, showing a list of log streams. The last event time is 2023-10-07 17:49:20 (UTC+05:30).

Step 17: Click on this log Stream that was created to view what was logged by your function.

The screenshot shows the AWS CloudWatch Log events page. The Log events are as follows:

- Timestamp:** 2023-10-07T17:49:20.002+05:30
- Message:** INIT_START Runtime Version: python:3.11.v14 Runtime Version ARN: arn:aws:lambda:ap-south-1:runtime:9c87c21u94b293e1a306aad2c23c...
- Timestamp:** 2023-10-07T17:49:20.110+05:30
- Message:** START RequestId: a189471e-867f-4db4-824f-2b602f956879 Version: \$LATEST
- Timestamp:** 2023-10-07T17:49:20.111+05:30
- Message:** END RequestId: a189471e-867f-4db4-824f-2b602f956879
- Timestamp:** 2023-10-07T17:49:20.111+05:30
- Message:** REPORT RequestId: a189471e-867f-4db4-824f-2b602f956879 Duration: 1.25 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Us...

Conclusion: Thus, we have created a Lambda function which logs “An Image has been added” once you add an object to a specific bucket in S3.