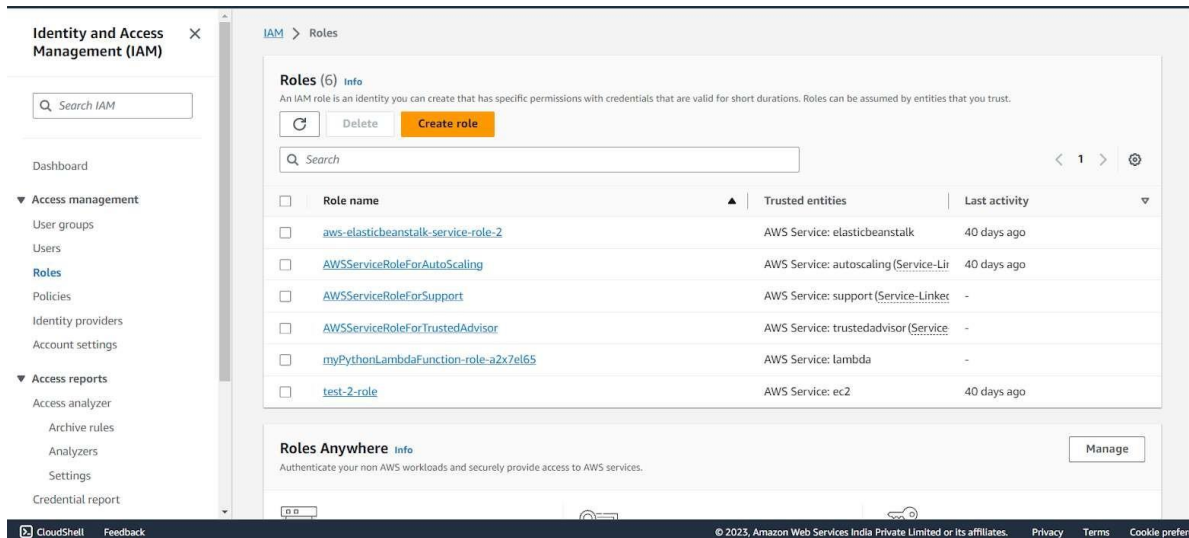


Adv. DevOps Exp. 12

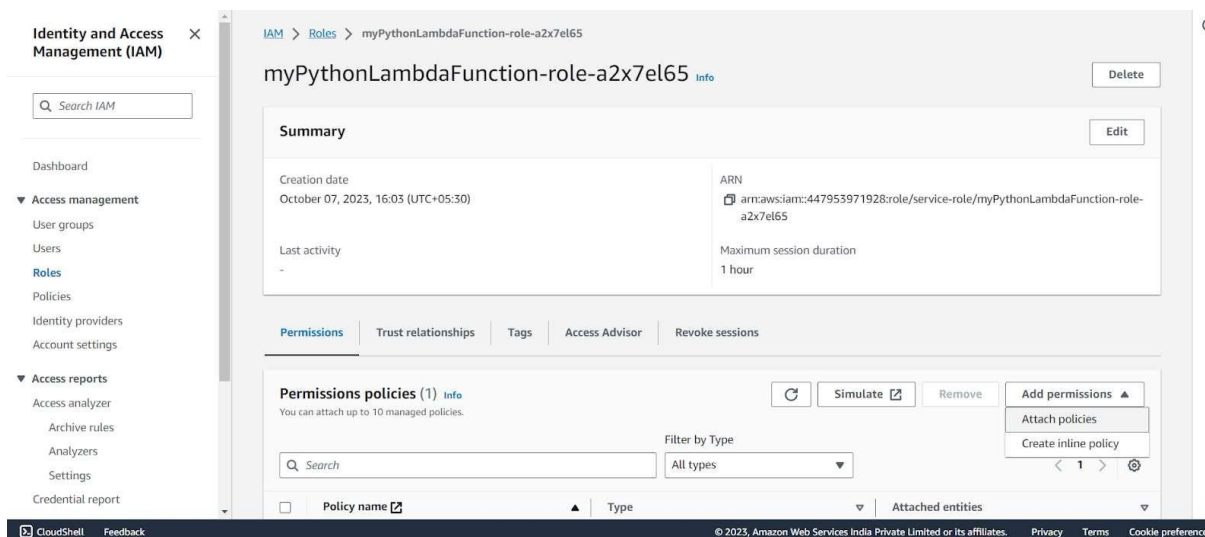
Niraj S. Kothawade

D15A - 24

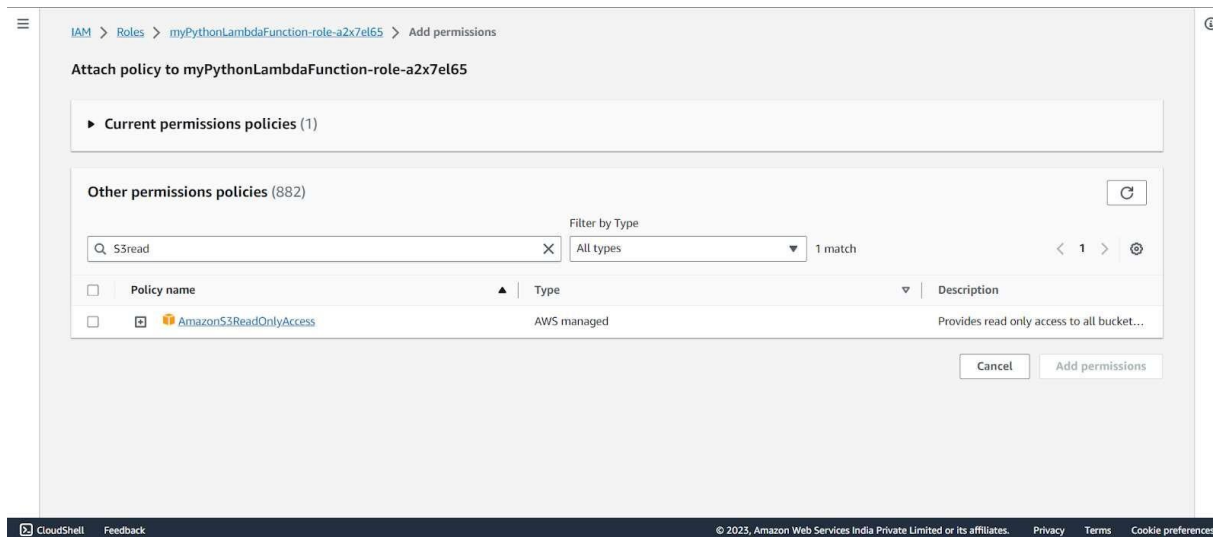
Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



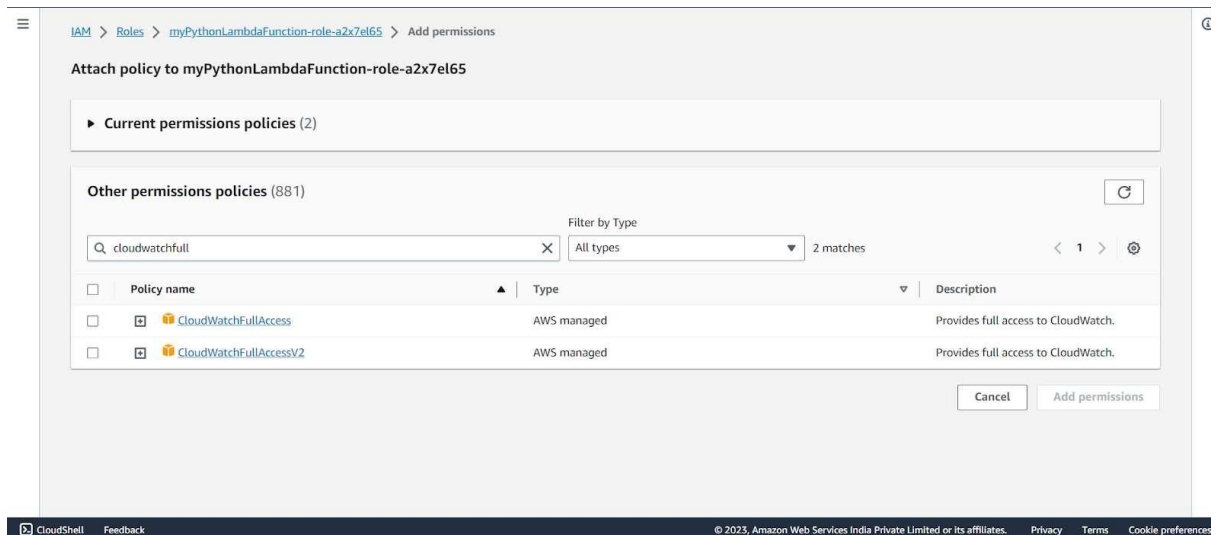
Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.



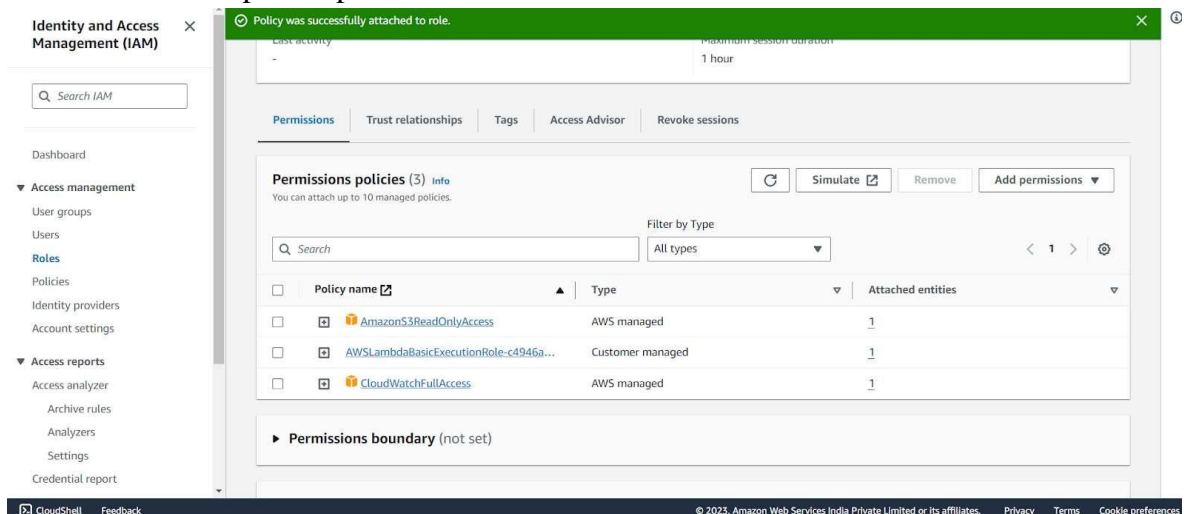
S3-ReadOnly



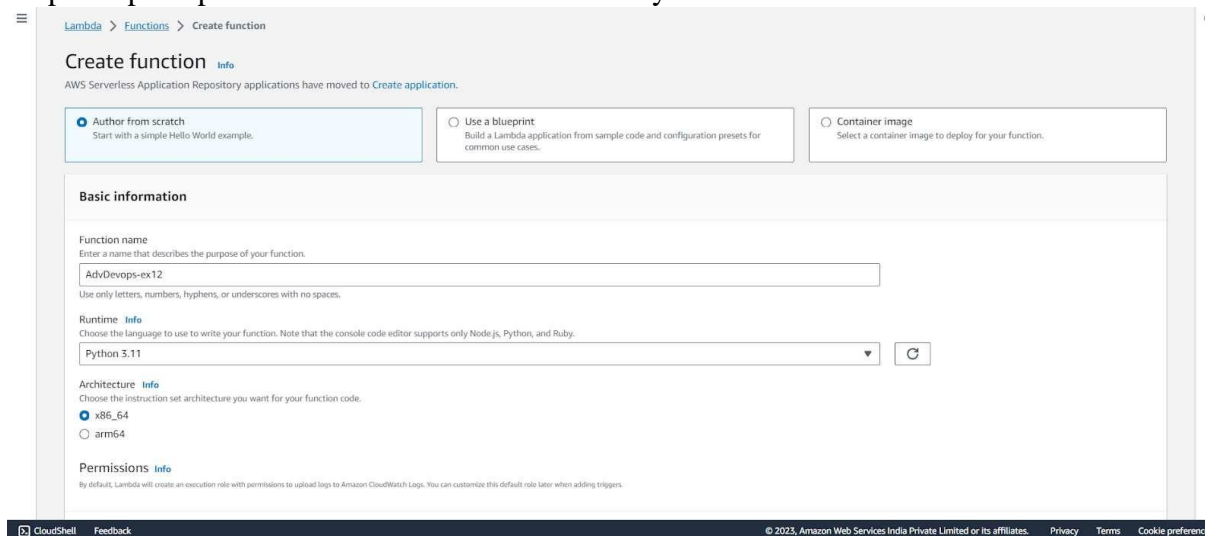
CloudWatchFull



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

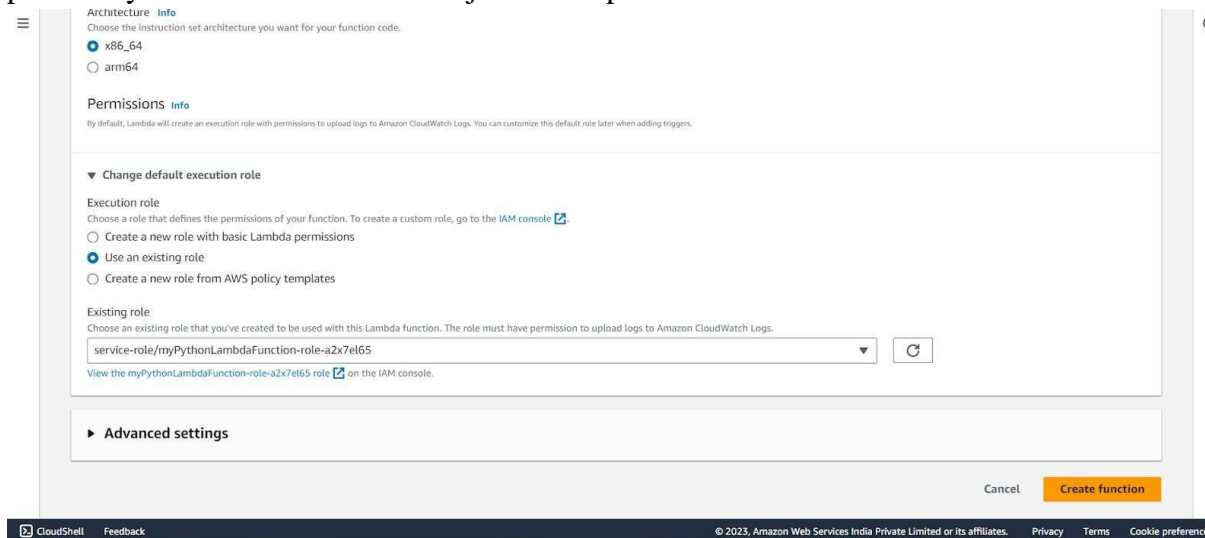


Step 3: Open up AWS Lambda and create a new Python function.



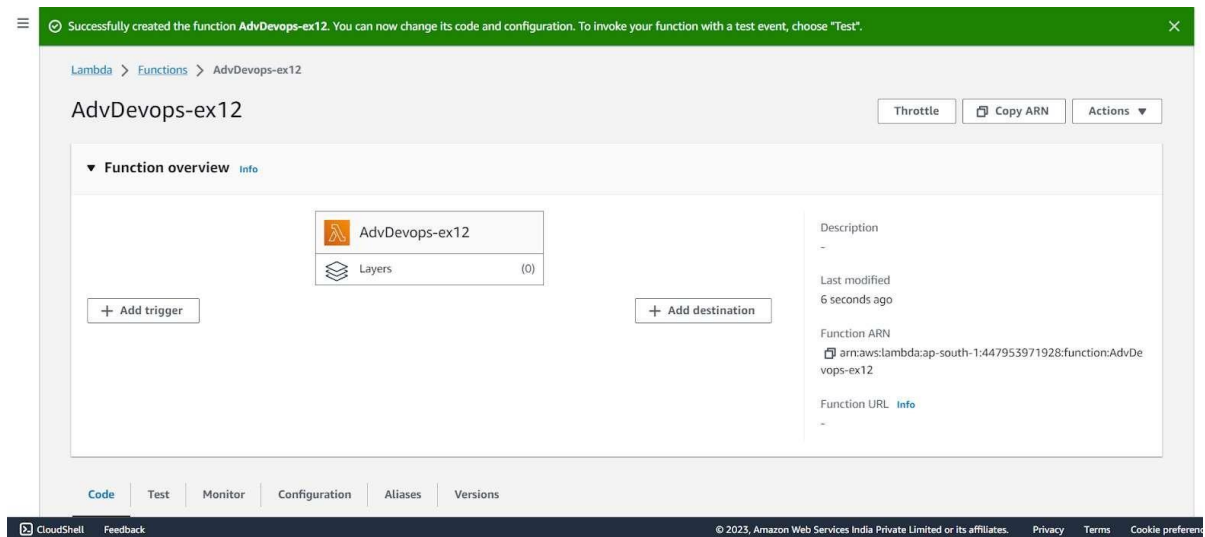
The screenshot shows the 'Create function' page in the AWS Lambda console. The breadcrumb navigation at the top indicates the path: Lambda > Functions > Create function. The page title is 'Create function' with an 'Info' link. A note states: 'AWS Serverless Application Repository applications have moved to Create application.' There are three main options for creating a function: 'Author from scratch' (selected, with a sub-note 'Start with a simple Hello World example.'), 'Use a blueprint' (with a sub-note 'Build a Lambda application from sample code and configuration presets for common use cases.'), and 'Container image' (with a sub-note 'Select a container image to deploy for your function.'). Below these is the 'Basic information' section. It contains a 'Function name' field with the value 'AdvDevoops-ex12' and a note 'Enter a name that describes the purpose of your function. Use only letters, numbers, hyphens, or underscores with no spaces.' The 'Runtime' is set to 'Python 3.11' with a note 'Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.' The 'Architecture' is set to 'x86_64' with a note 'Choose the instruction set architecture you want for your function code.' and a sub-note '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.' At the bottom of the page, there is a footer with 'CloudShell', 'Feedback', and copyright information for Amazon Web Services India Private Limited.

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

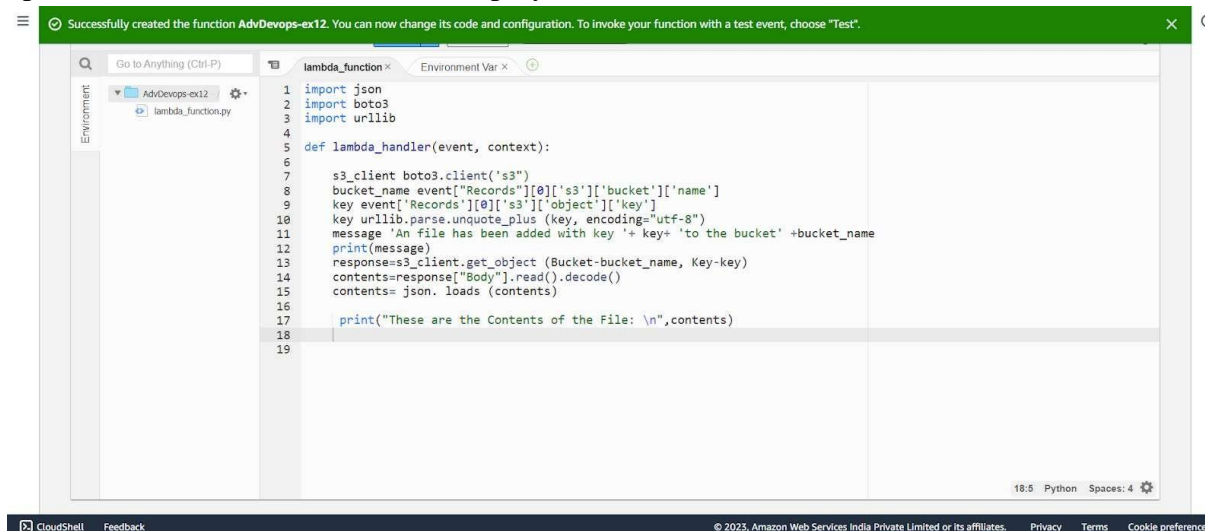


This screenshot shows the 'Permissions' section of the 'Create function' page. The 'Architecture' section is visible at the top, with 'x86_64' selected. The 'Permissions' section has a note: 'By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.' Below this is a section titled 'Change default execution role'. It includes an 'Execution role' section with three options: 'Create a new role with basic Lambda permissions', 'Use an existing role' (selected), and 'Create a new role from AWS policy templates'. A note says 'Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console'. The 'Existing role' section has a dropdown menu showing 'service-role/myPythonLambdaFunction-role-a2x7el65' and a note: '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. View the myPythonLambdaFunction-role-a2x7el65 role on the IAM console.' At the bottom right, there are 'Cancel' and 'Create function' buttons. The footer is identical to the previous screenshot.

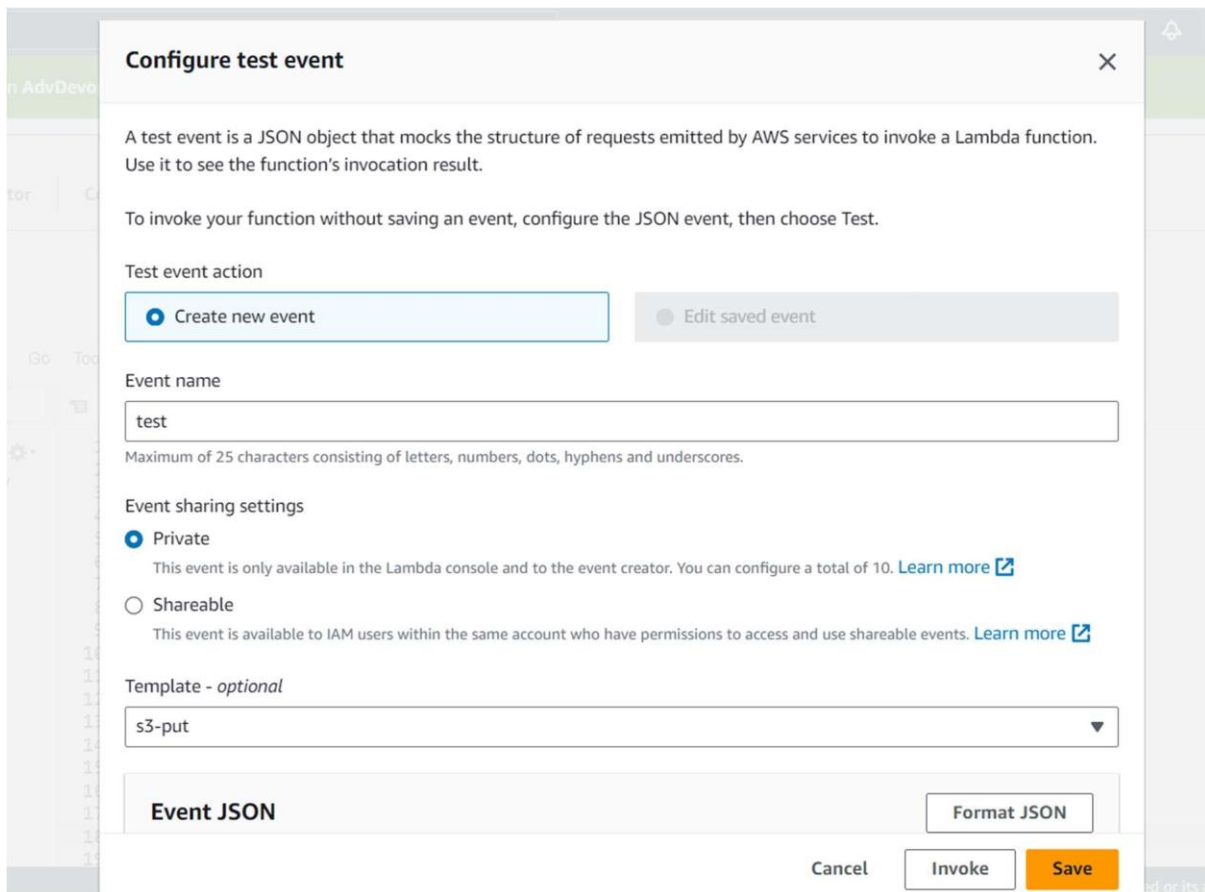
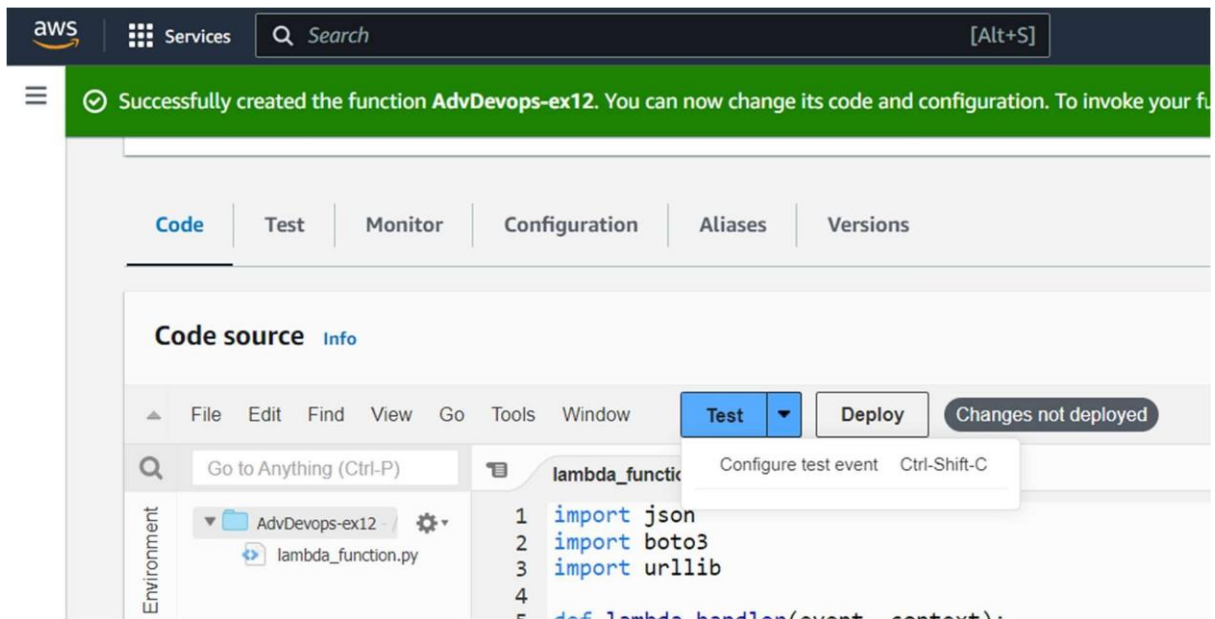
Step 4: The function is up and running.



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.

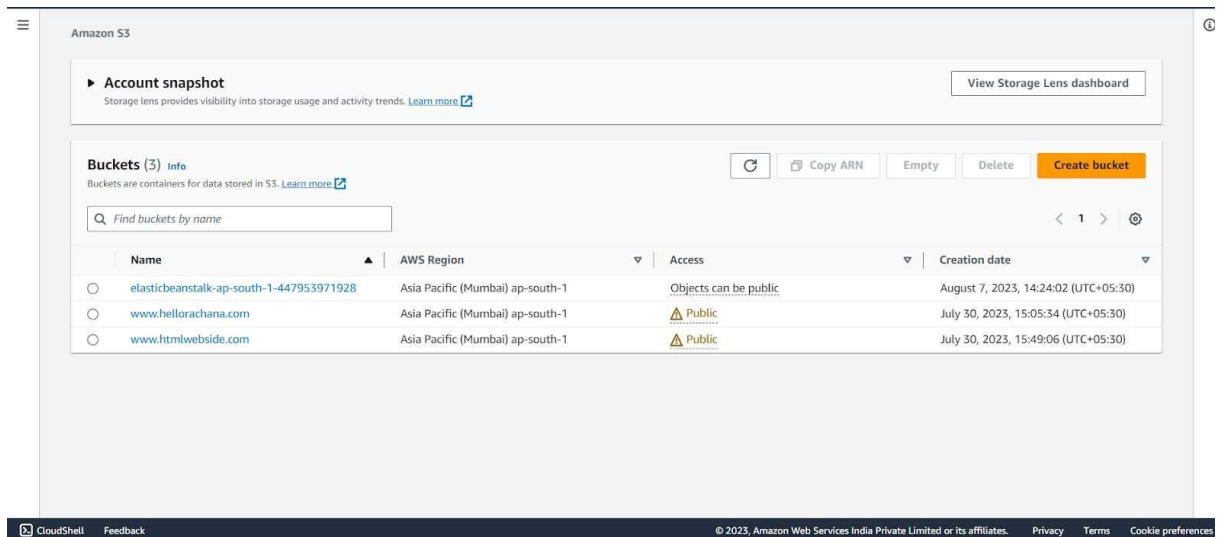


Step 6: Click on Test and choose the 'S3 Put' Template.

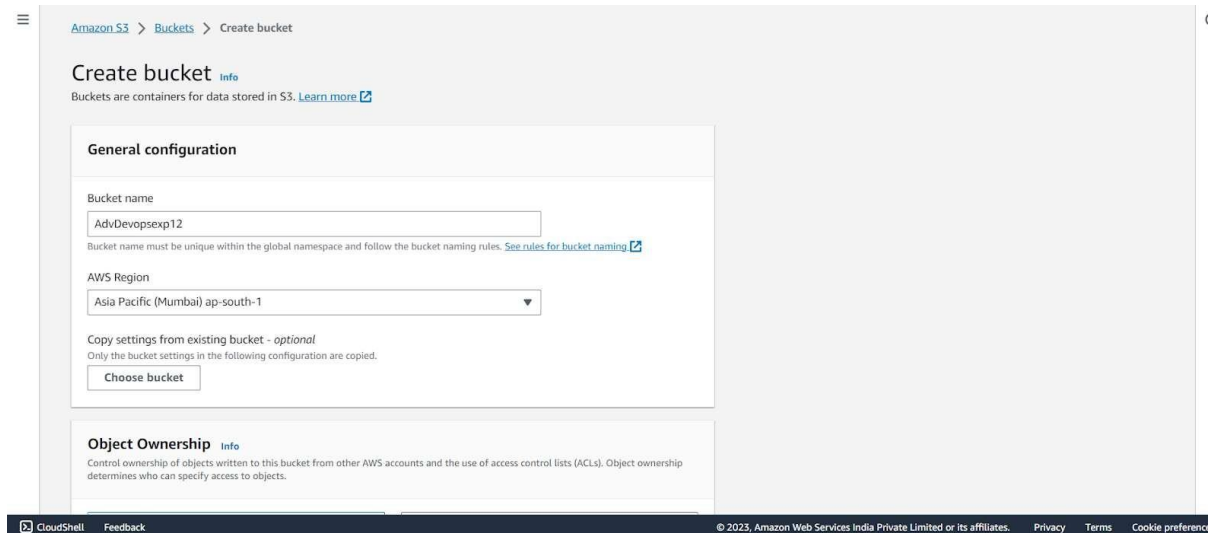


And Save it.

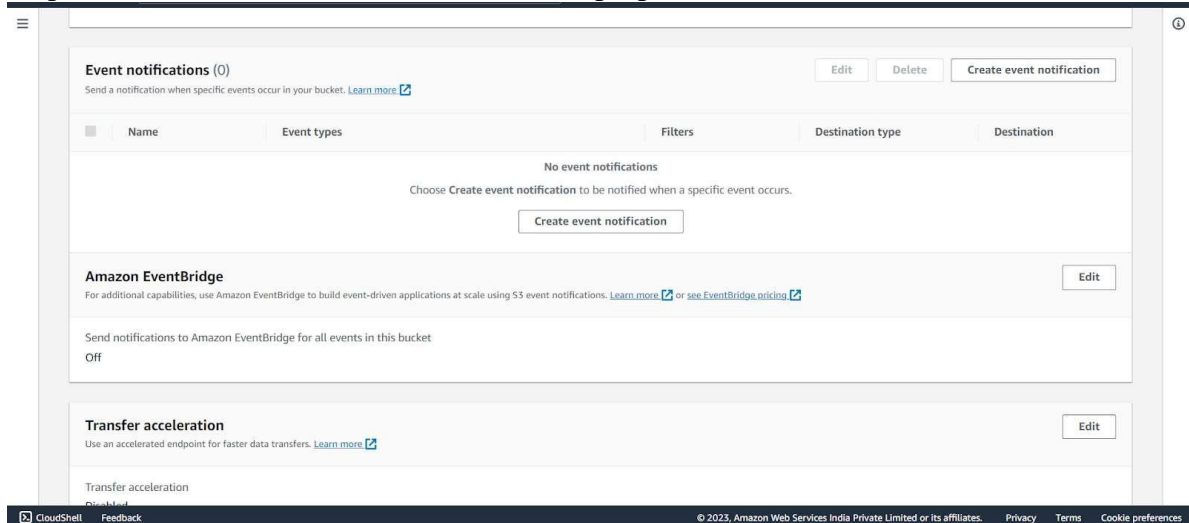
Step 7: Open up the S3 Console and create a new bucket.



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. At the top, there's a header with the AWS logo, 'Services', a search bar, and a keyboard shortcut '[Alt+S]'. A hamburger menu is on the left. The main content area has a title 'General configuration'. Below it, there's a section for 'Event name' with a text input field containing 'S3putrequest' and a note 'Event name can contain up to 255 characters.' Below that is a 'Prefix - optional' section with a text input field containing 'images/' and a note 'Limit the notifications to objects with key starting with specified characters.' Below that is a 'Suffix - optional' section with a text input field containing '.jpg' and a note 'Limit the notifications to objects with key ending with specified characters.' The next section is 'Event types' with a note '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.' Below this is a section for 'Object creation' with two columns. The left column has a checkbox 'All object create events' with the text 's3:ObjectCreated:*' below it. The right column has a checked checkbox 'Put' with the text 's3:ObjectCreated:Put' below it, and an unchecked checkbox 'Post' with the text 's3:ObjectCreated:Post' below it. At the bottom of the console, there's a footer with 'CloudShell', 'Feedback', and '© 2023, Amazon Web Services India Pvt'.

aws Services Search [Alt+S]

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

CloudShell Feedback © 2023, Amazon Web Services India Pvt

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. At the top, there's a header with the AWS logo, 'Services', a search bar, and a keyboard shortcut '[Alt+S]'. A hamburger menu is on the left. The main content area has a title 'Destination'. Below it, there's a blue information box with a note: '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'. Below this is a section for 'Destination' with a note 'Choose a destination to publish the event. Learn more'. There are three radio button options: 'Lambda function' (selected), 'SNS topic', and 'SQS queue'. Below 'Lambda function' is a note 'Run a Lambda function script based on S3 events.' Below 'SNS topic' is a note 'Fanout messages to systems for parallel processing or directly to people.' Below 'SQS queue' is a note 'Send notifications to an SQS queue to be read by a server.' Below these is a section for 'Specify Lambda function' with two radio button options: 'Choose from your Lambda functions' (selected) and 'Enter Lambda function ARN'. Below this is a section for 'Lambda function' with a dropdown menu showing 'AdvDevops-ex12'. At the bottom right, there are two buttons: 'Cancel' and 'Save changes'. At the bottom of the console, there's a footer with 'CloudShell', 'Feedback', and '© 2023, Amazon Web Serv'.

aws Services Search [Alt+S]

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

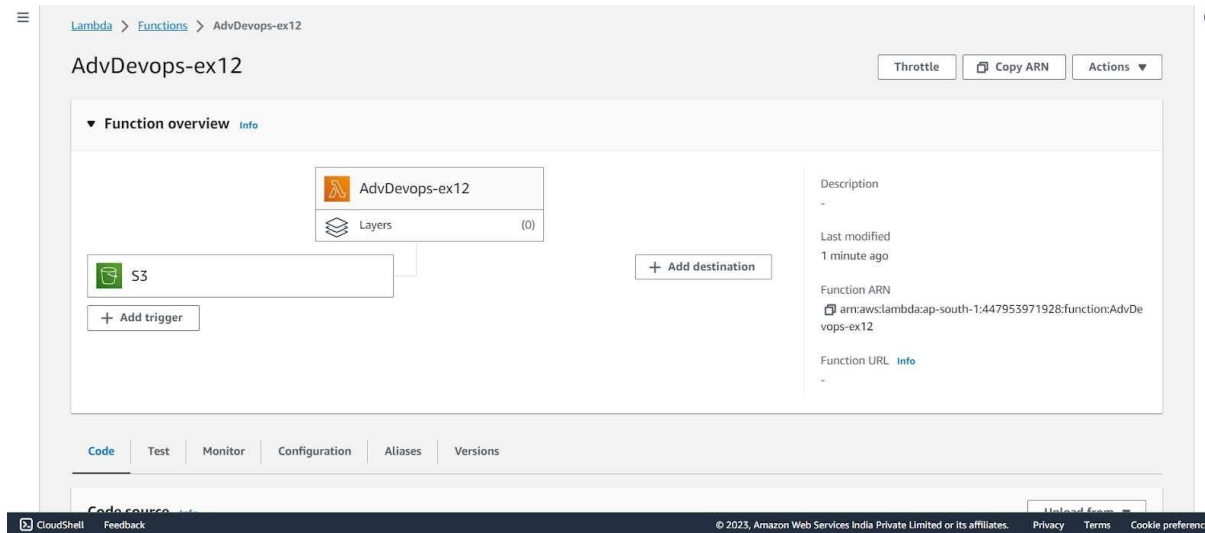
Lambda function

AdvDevops-ex12

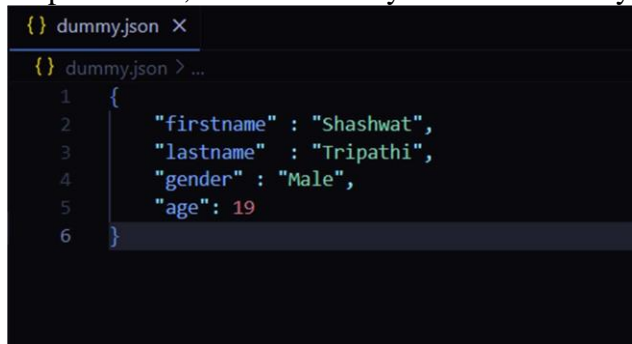
Cancel Save changes

CloudShell Feedback © 2023, Amazon Web Serv

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

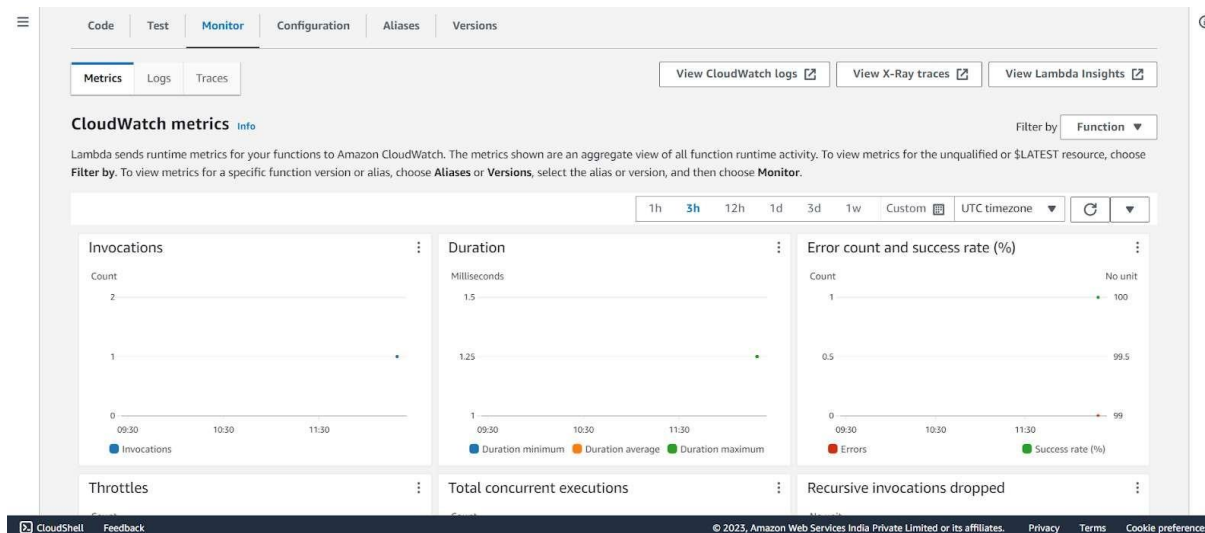


Step 12: Now, create a dummy JSON file locally.

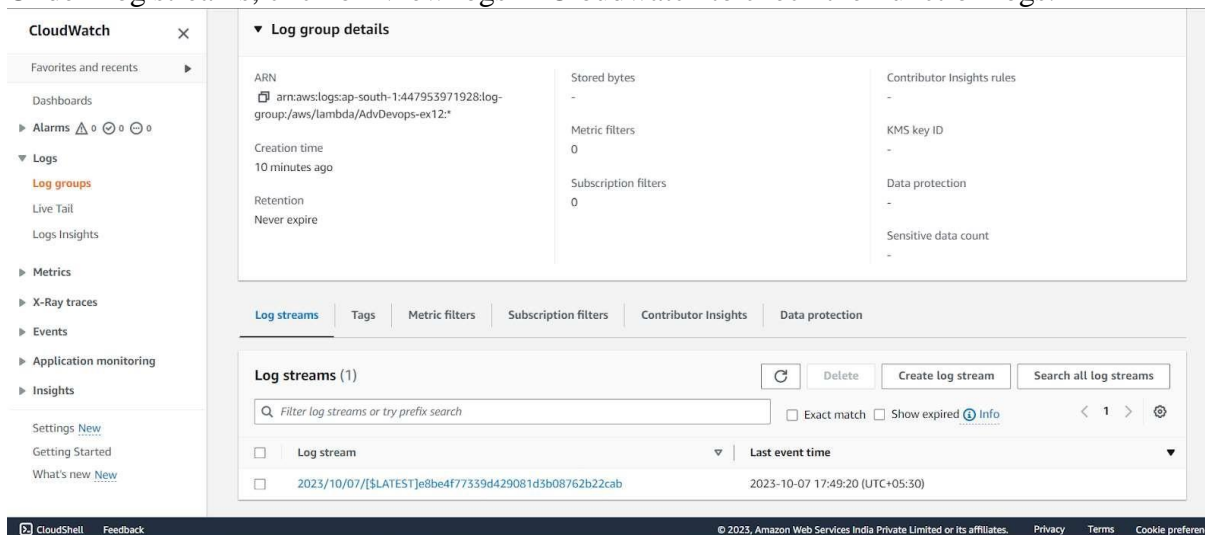


Step 13: Go back to your S3 Bucket and click on Add Files to upload a new file.

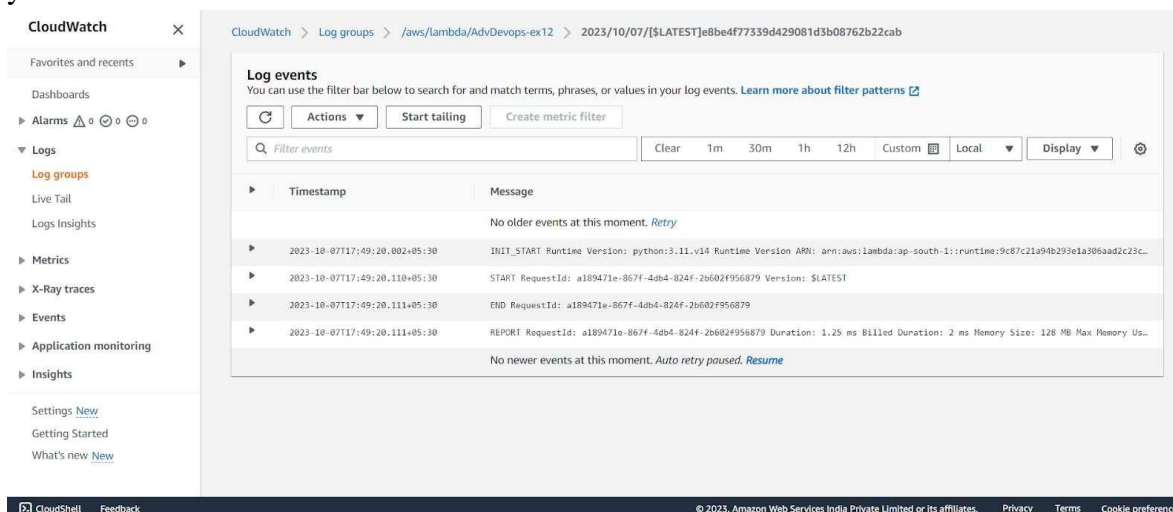
Step 14: Select the dummy data file from your computer and click Upload.



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



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



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.