

Problem Statement: -

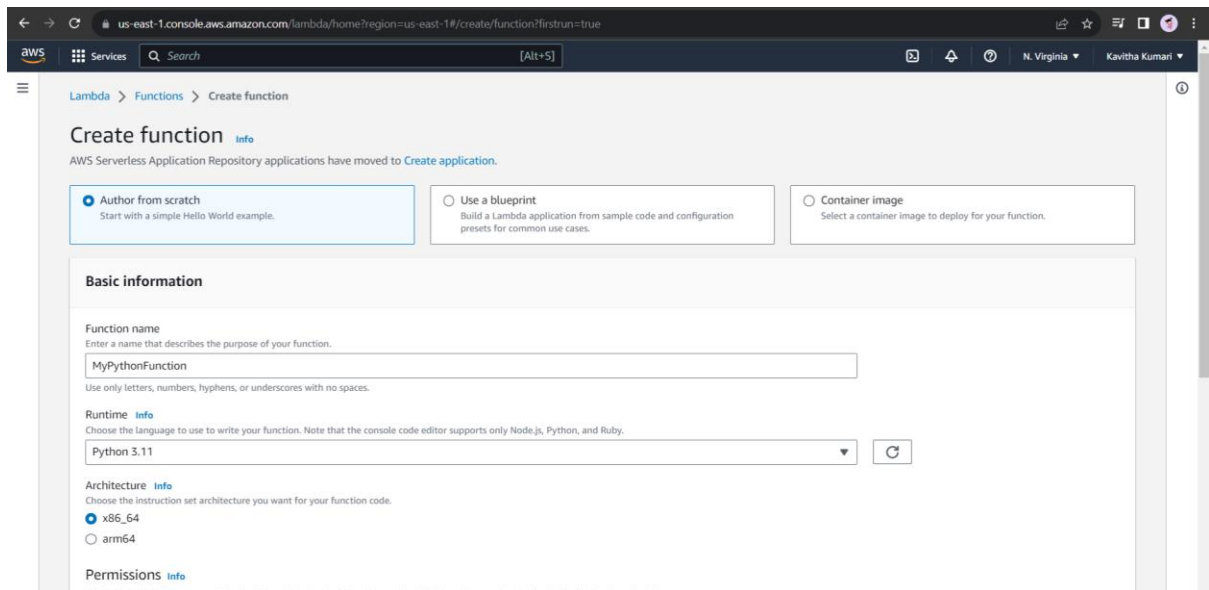
You work for XYZ Corporation. Your corporation wants to launch a new web-based application and they do not want their servers to be running all the time. It should also be managed by AWS. Implement suitable solutions.

Tasks To Be Performed:

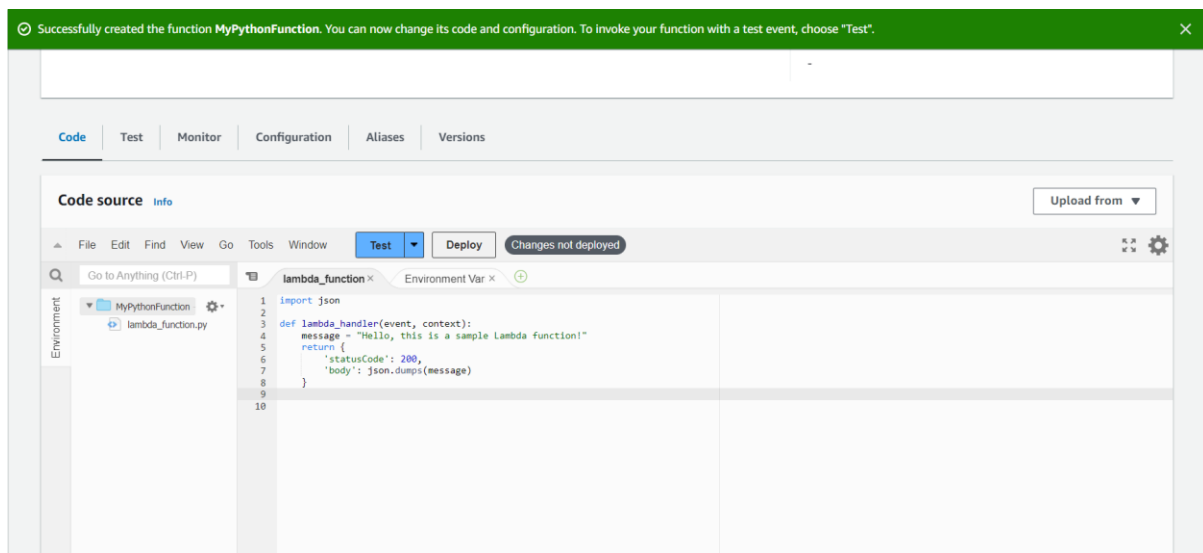
1. Create a sample Python Lambda function.
2. Set the Lambda Trigger as SQS and send a message to test invocations

Procedure: -

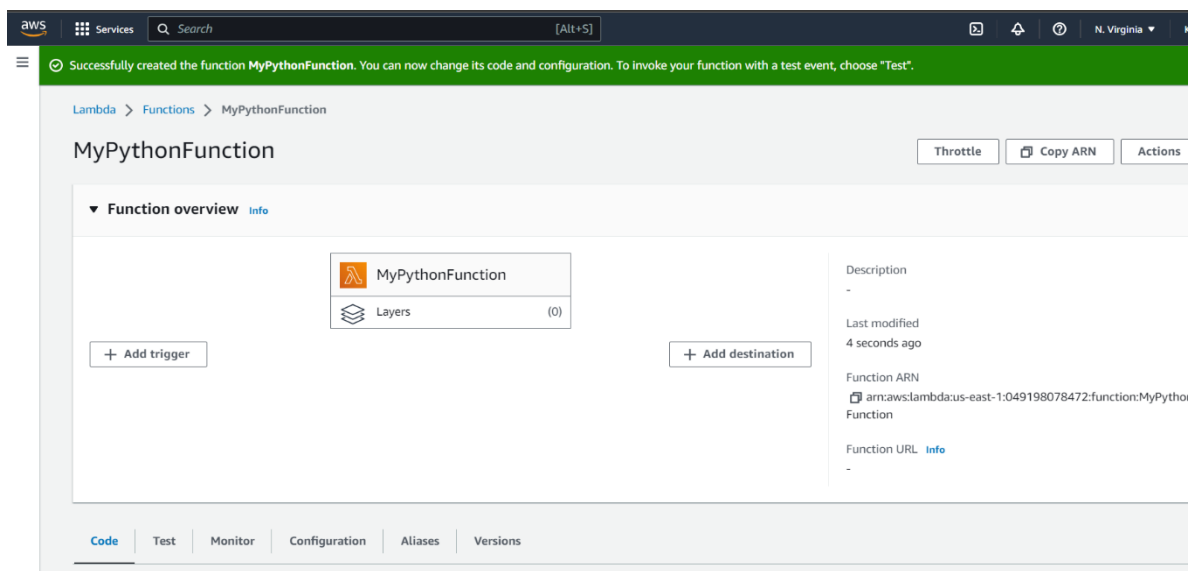
- Let us first create the Lambda function. Go to the AWS Management Console. Open the Lambda service.
- Click on "Create function."
- Choose "Author from scratch."
- Enter a name for your function, like "MyPythonFunction."
- For Runtime, select "Python" and choose the latest supported version.
- Under "Permissions," you can either choose an existing role with Lambda execution permissions or create a new role with basic Lambda permissions and also the basic permission of cloudWatch and SQS.
- Click on "Create function."



- In the code editor, you can write a simple Python function, such as:



- Click on "deploy."
- Now we need to Set up the Lambda Trigger as SQS.
- For this let us create an SQS service.
- Click on "create queue."



- Enter a name for your queue, like "MyQPython."
- Leave other settings as default and click on "Create queue."

Queue MyQPython created successfully
You can now send and receive messages.

Amazon SQS > Queues > MyQPython

MyQPython

Edit Delete Purge Send and receive messages Start DLQ redrive

Details info

Name MyQPython	Type Standard	ARN arn:aws:sqs:us-east-1:049198078472:MyQPython
Encryption Amazon SQS key (SSE-SQS)	URL https://sqs.us-east-1.amazonaws.com/049198078472/MyQPython	Dead-letter queue -

► More

SNS subscriptions Lambda triggers Dead-letter queue Monitoring Tagging Access policy Encryption Dead-letter queue redrive tasks

Subscription region
us-east-1

- Next, we need to set up the SQS trigger for our Lambda function:
- Go back to the Lambda service.
- Open your Lambda function (MyPythonFunction) that you created earlier.
- In the function configuration page, click on "Add trigger."
- Choose "SQS" as the trigger type.
- From the "Queue" dropdown, select the SQS queue you created earlier (MyQPython).
- Leave other settings as default, and click on "Add."

Lambda > Add trigger

Add trigger

Trigger configuration info

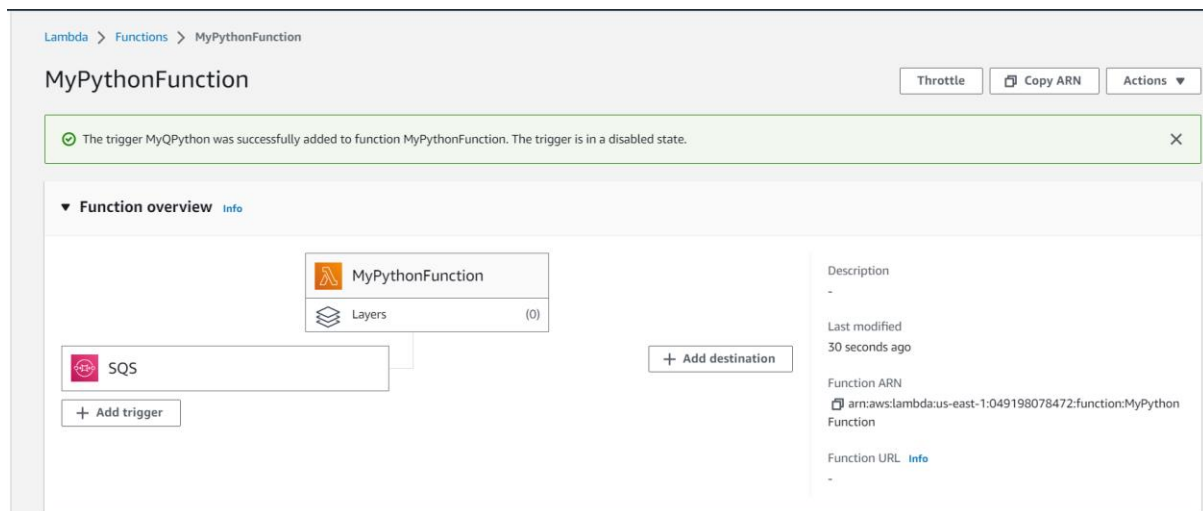
SQS
aws event-source-mapping polling queue

SQS queue
Choose or enter the ARN of an SQS queue.
arn:aws:sqs:us-east-1:049198078472:MyQPython

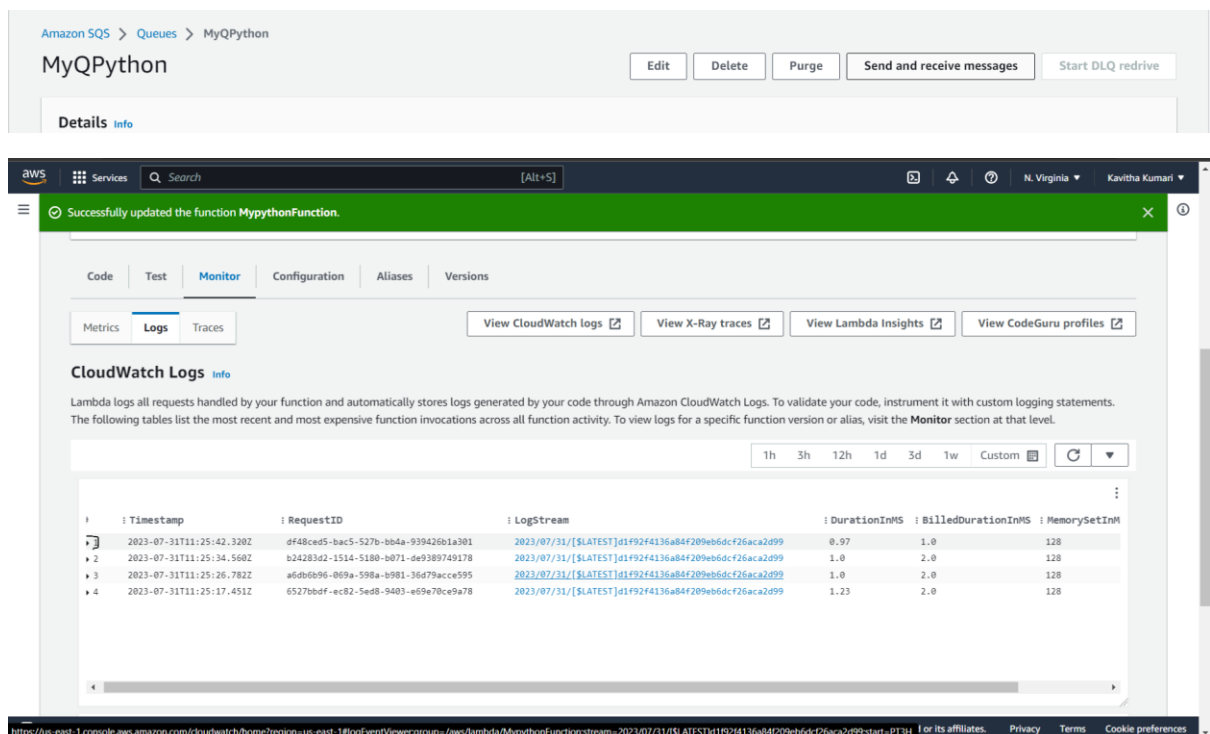
☒ **Activate trigger**
Select to activate the trigger now. Keep unchecked to create the trigger in a deactivated state for testing (recommended).

Batch size - optional
The number of records in each batch to send to the function.
10
The maximum is 10,000 for standard queues and 10 for FIFO queues.

Batch window - optional
The maximum amount of time to gather records before invoking the function, in seconds.
0
When the batch size is greater than 10, set the batch window to at least 1 second.



- Now, when a message is sent to the SQS queue (MyQPython), it will trigger the Lambda function (MyPythonFunction) and invoke it.
- Go to the AWS Management Console.
- Open the Amazon SQS service.
- Select your SQS queue (MyQpython) from the list.
- Click on "Send and receive messages" and then "Send a message."
- In the message body, you can enter any test message (e.g., "Test message for Lambda").
- Click on "Send message."
- After a moment, the Lambda function will be triggered and will process the message. You can check the function's logs in the AWS Lambda console to see the output.
- That is, it! You have successfully created a sample Python Lambda function and set up an SQS trigger to invoke it when messages are sent to the SQS queue.



- You can also click on **view CloudWatch logs** to view the detailed information.

The screenshot displays the AWS CloudWatch console interface. The left sidebar contains navigation links for Dashboards, Alarms, Logs, Metrics, X-Ray traces, Events, Application monitoring, and Insights. The main content area shows the 'Log events' page for a specific Lambda function. It includes a search bar, a filter bar, and a table of log events. The table has two columns: 'Timestamp' and 'Message'. The log events are listed in chronological order, showing the function's startup, request processing, and completion.

Timestamp	Message
2023-07-31T16:55:10.410+05:30	INIT_START Runtime Version: python:3.11.v8 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:a96d37d47210bbc80cb1c500...
2023-07-31T16:55:10.516+05:30	START RequestId: 75a4a99-634d-5b34-8edf-f9d8ff01072b Version: \$LATEST
2023-07-31T16:55:10.517+05:30	END RequestId: 75a4a99-634d-5b34-8edf-f9d8ff01072b
2023-07-31T16:55:10.517+05:30	REPORT RequestId: 75a4a99-634d-5b34-8edf-f9d8ff01072b Duration: 1.26 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memo...
2023-07-31T16:55:17.450+05:30	START RequestId: 6527bbdf-ec82-5ed8-9403-e69e70ce9a78 Version: \$LATEST
2023-07-31T16:55:17.451+05:30	END RequestId: 6527bbdf-ec82-5ed8-9403-e69e70ce9a78
2023-07-31T16:55:17.451+05:30	REPORT RequestId: 6527bbdf-ec82-5ed8-9403-e69e70ce9a78 Duration: 1.23 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memo...
2023-07-31T16:55:26.781+05:30	START RequestId: a6db6b96-069a-598a-b981-36d79acce595 Version: \$LATEST
2023-07-31T16:55:26.782+05:30	END RequestId: a6db6b96-069a-598a-b981-36d79acce595
2023-07-31T16:55:26.782+05:30	REPORT RequestId: a6db6b96-069a-598a-b981-36d79acce595 Duration: 1.00 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memo...

-----END-----