

Lambda assignment

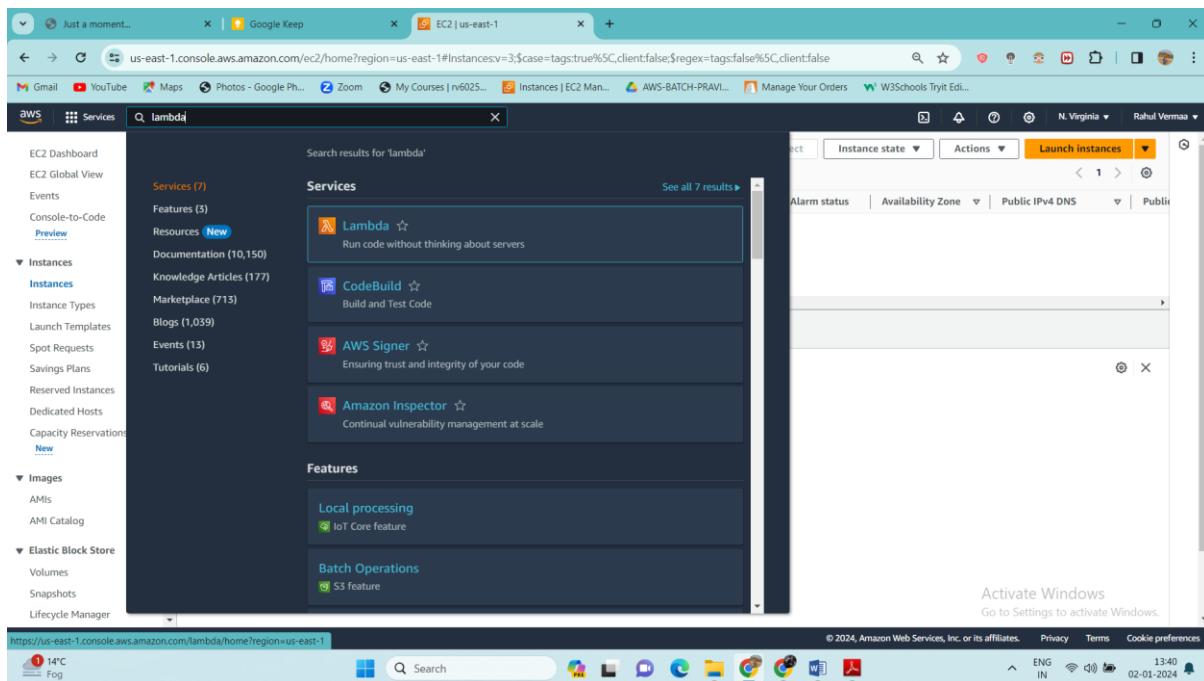
Problem Statement:

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.

First search for Lambda in your aws account



The screenshot shows a browser window with the AWS Lambda search results. The search term 'lambda' is entered in the search bar. The results are categorized into 'Services' and 'Features'.

Services (7):

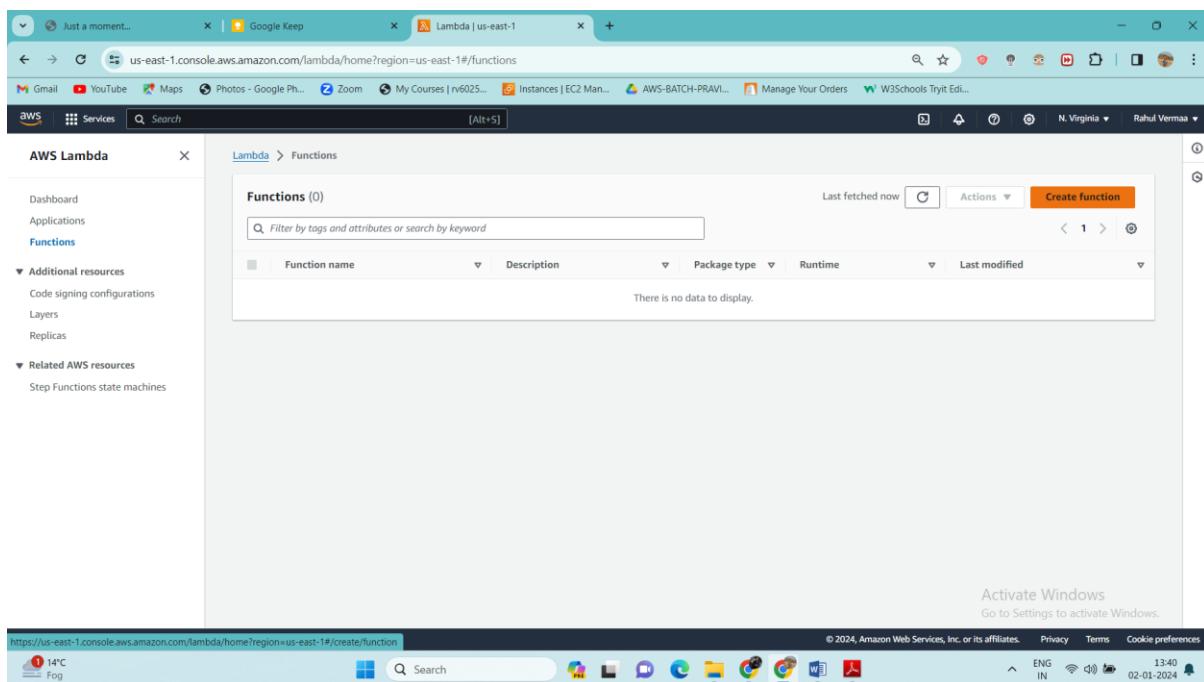
- Lambda: Run code without thinking about servers
- CodeBuild: Build and Test Code
- AWS Signer: Ensuring trust and integrity of your code
- Amazon Inspector: Continual vulnerability management at scale

Features:

- Local processing: IoT Core feature
- Batch Operations: S3 feature

The left sidebar shows the AWS navigation menu with 'Lambda' selected. The top navigation bar shows the region as 'us-east-1'.

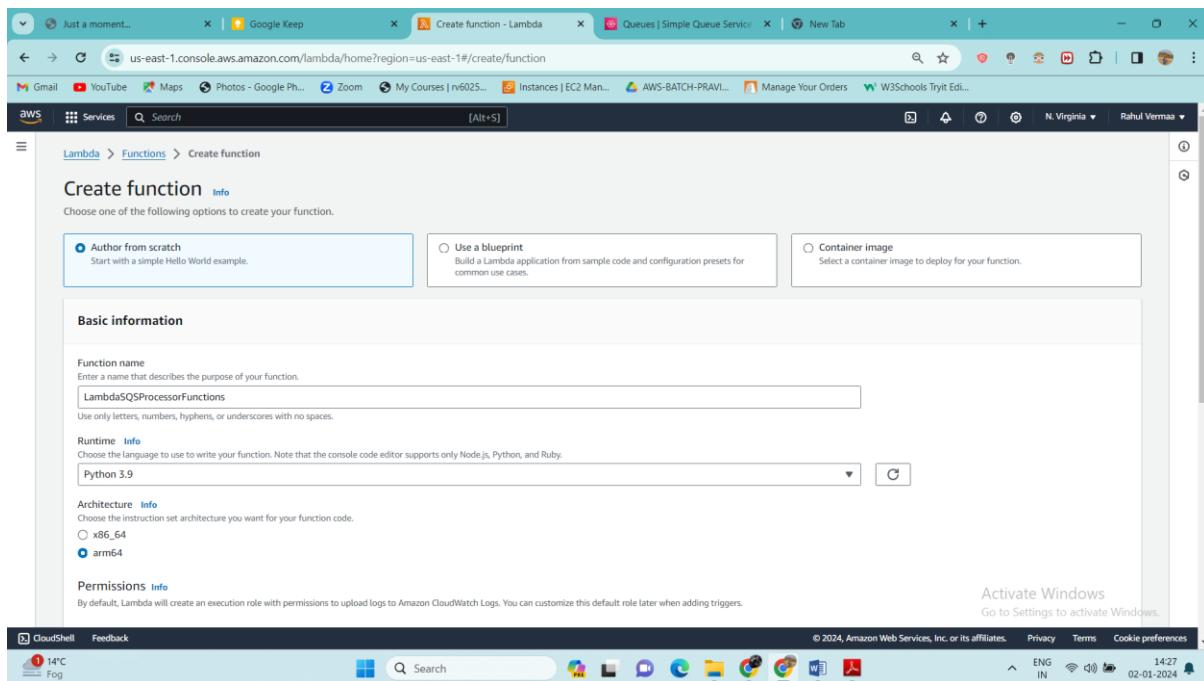
Now create one lambda function



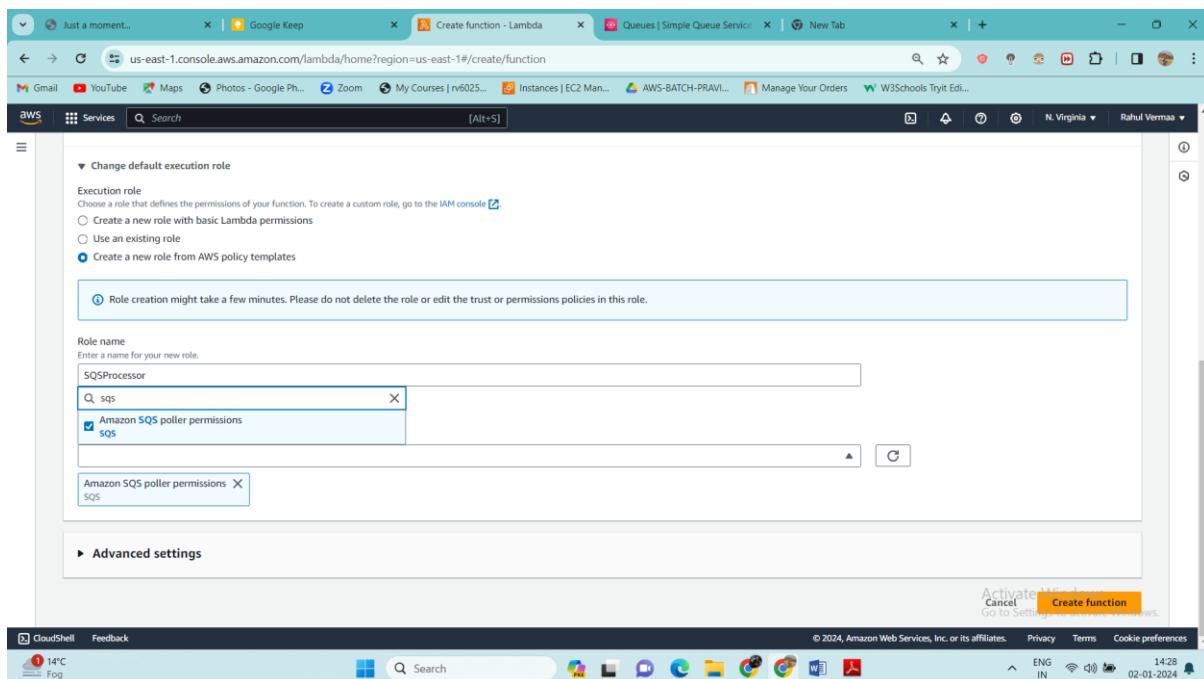
The screenshot shows the AWS Lambda Functions page. The sidebar on the left is titled 'AWS Lambda' and includes 'Dashboard', 'Applications', 'Functions' (selected), 'Additional resources', 'Related AWS resources', and 'Step Functions state machines'. The main content area shows a table titled 'Functions (0)' with a search bar and a 'Create function' button. The table has columns for 'Function name', 'Description', 'Package type', 'Runtime', and 'Last modified'. A message at the bottom of the table says 'There is no data to display.'

The top navigation bar shows the region as 'us-east-1'.

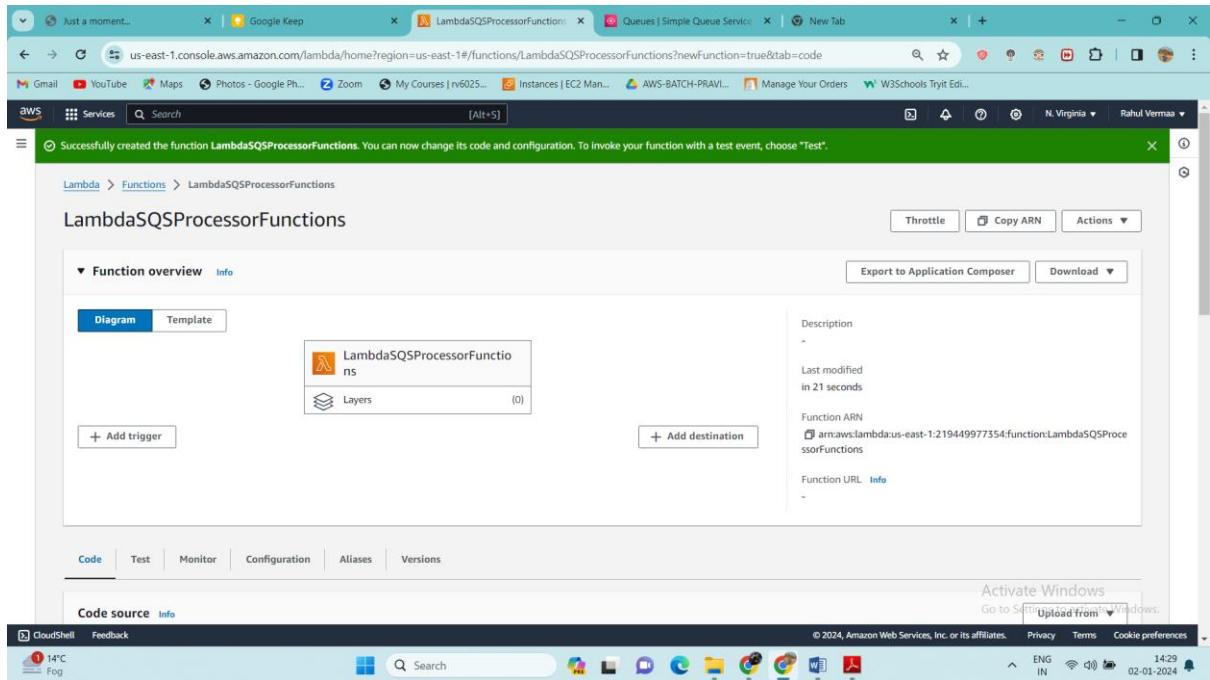
Select configurations and options



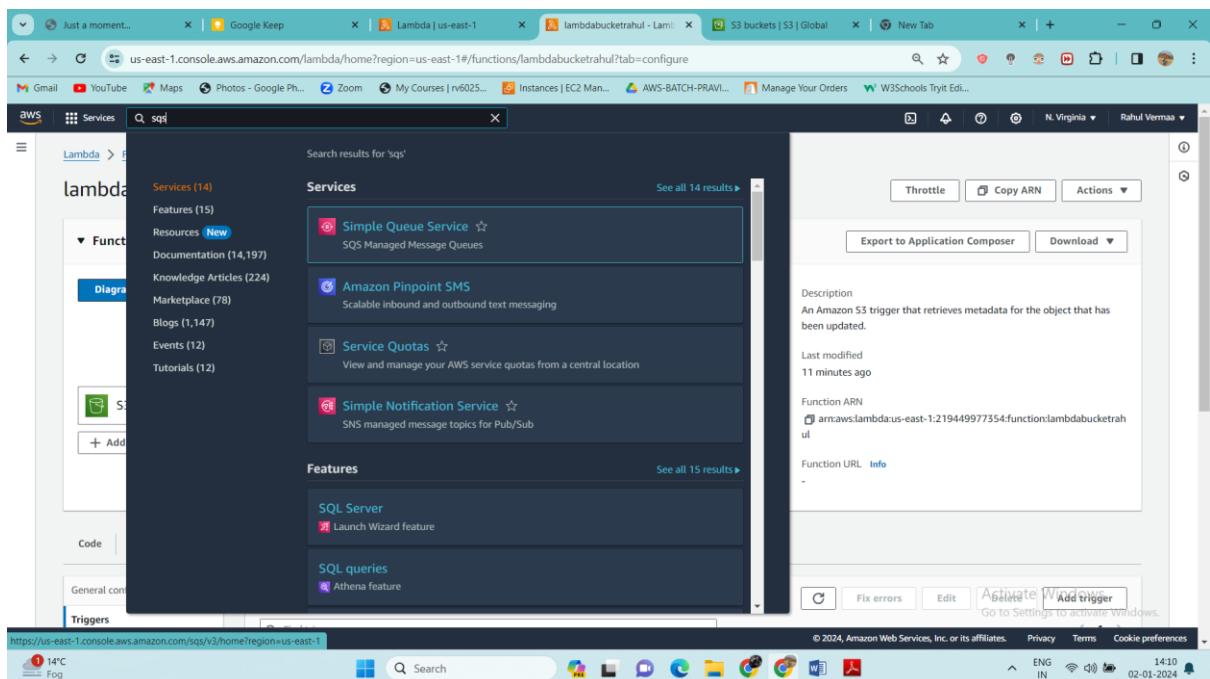
Now we have to add permissions to your lambda function.



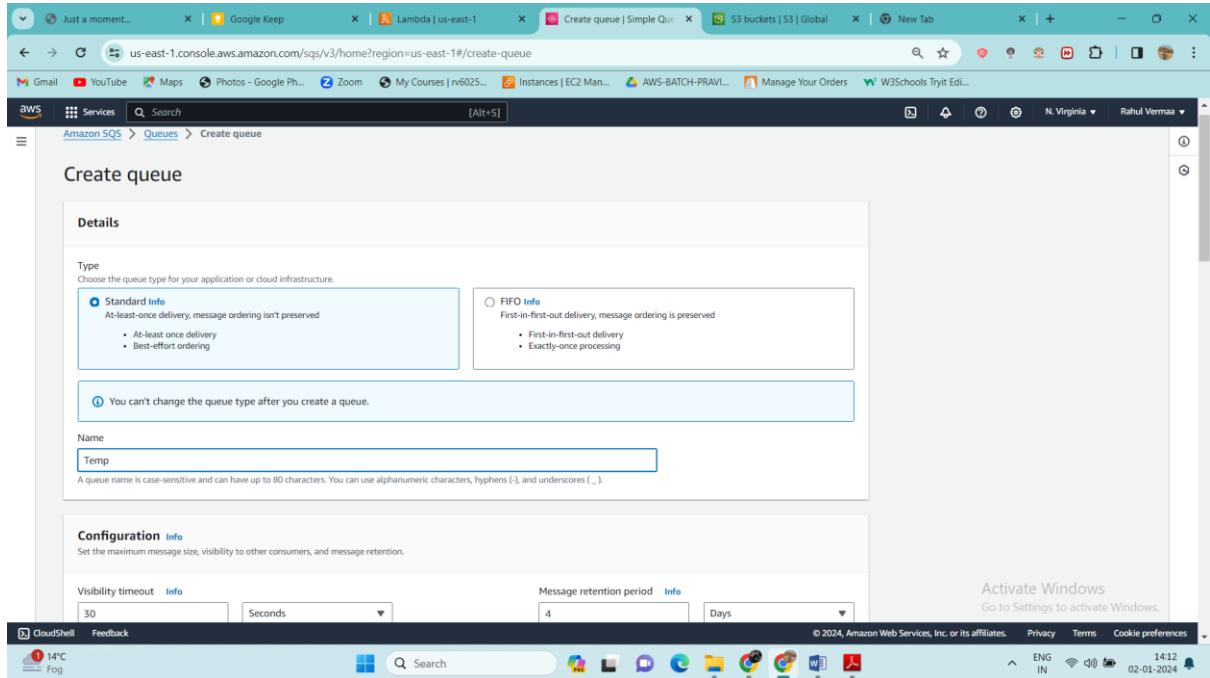
Our lambda function is created



We have to add trigger as SQS, so first im gonna create one SQS.



Select options and define your SQS



Just a moment... x | Google Keep x | Lambda | us-east-1 x | Create queue | Simple Que... x | S3 buckets | S3 | Global x | New Tab x | + - | ○ : |

us-east-1.console.aws.amazon.com/sqs/v3/home/?region=us-east-1#/create-queue

Gmail YouTube Maps Photos - Google Ph... Zoom My Courses | rv6025... Instances | EC2 Man... AWS-BATCH-PRAVI... Manage Your Orders W3Schools Tryit Edi... N. Virginia Rahul Vermaa

aws Services Search [Alt+S]

Amazon SQS > Queues > Create queue

Create queue

Details

Type
Choose the queue type for your application or cloud infrastructure.

Standard Info
At-least-once delivery, message ordering isn't preserved
• At-least once delivery
• Best-effort ordering

FIFO Info
First-in-first-out delivery, message ordering is preserved
• First-in-first-out delivery
• Exactly-once processing

Info You can't change the queue type after you create a queue.

Name
Temp

A queue name is case-sensitive and can have up to 80 characters. You can use alphanumeric characters, hyphens (-), and underscores (_).

Configuration **Info**
Set the maximum message size, visibility to other consumers, and message retention.

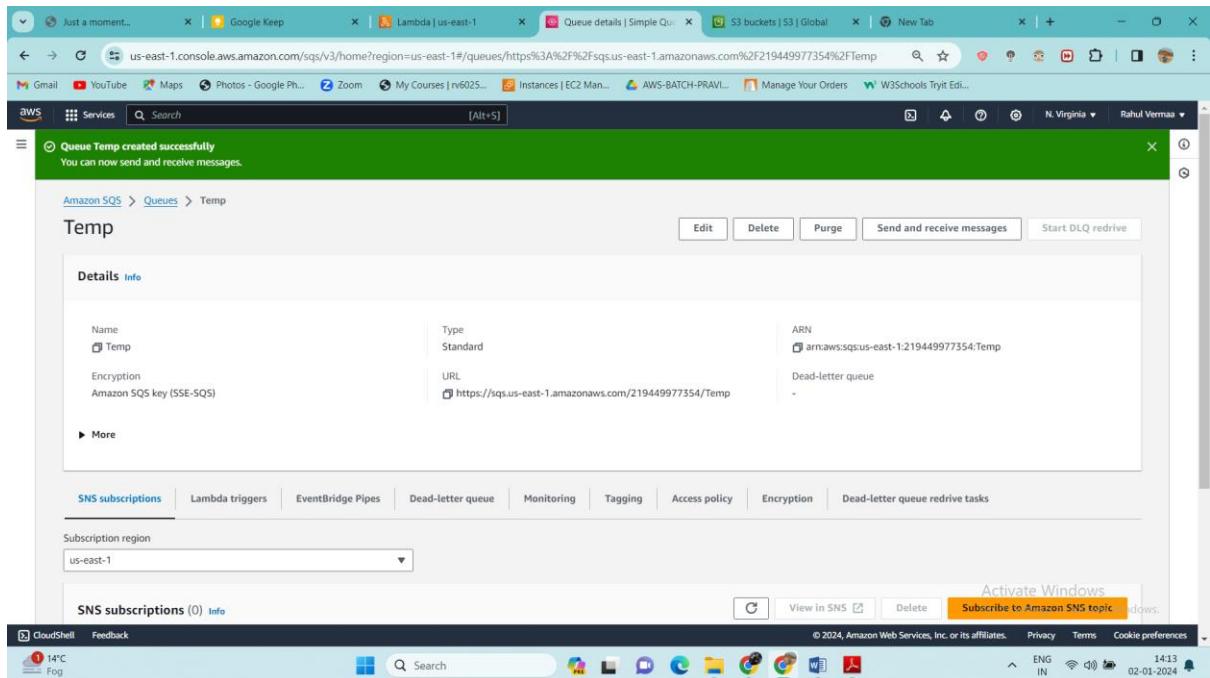
Visibility timeout **Info**
30 Seconds

Message retention period **Info**
4 Days

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 14:12 02-01-2024

Our SQS is created



Just a moment... x | Google Keep x | Lambda | us-east-1 x | Queue details | Simple Que... x | S3 buckets | S3 | Global x | New Tab x | + - | ○ : |

us-east-1.console.aws.amazon.com/sqs/v3/home/?region=us-east-1#/queues/https%3A%2F%2Fsq.us-east-1.amazonaws.com%2F219449977354%2FTemp

Gmail YouTube Maps Photos - Google Ph... Zoom My Courses | rv6025... Instances | EC2 Man... AWS-BATCH-PRAVI... Manage Your Orders W3Schools Tryit Edi... N. Virginia Rahul Vermaa

aws Services Search [Alt+S]

Amazon SQS > Queues > Temp

Temp

Details **Info**

Name
Temp

Type
Standard

ARN
arn:aws:sqs:us-east-1:219449977354:Temp

Encryption
Amazon SQS key (SSE-SQS)

URL
https://sq.us-east-1.amazonaws.com/219449977354/Temp

Dead-letter queue
-

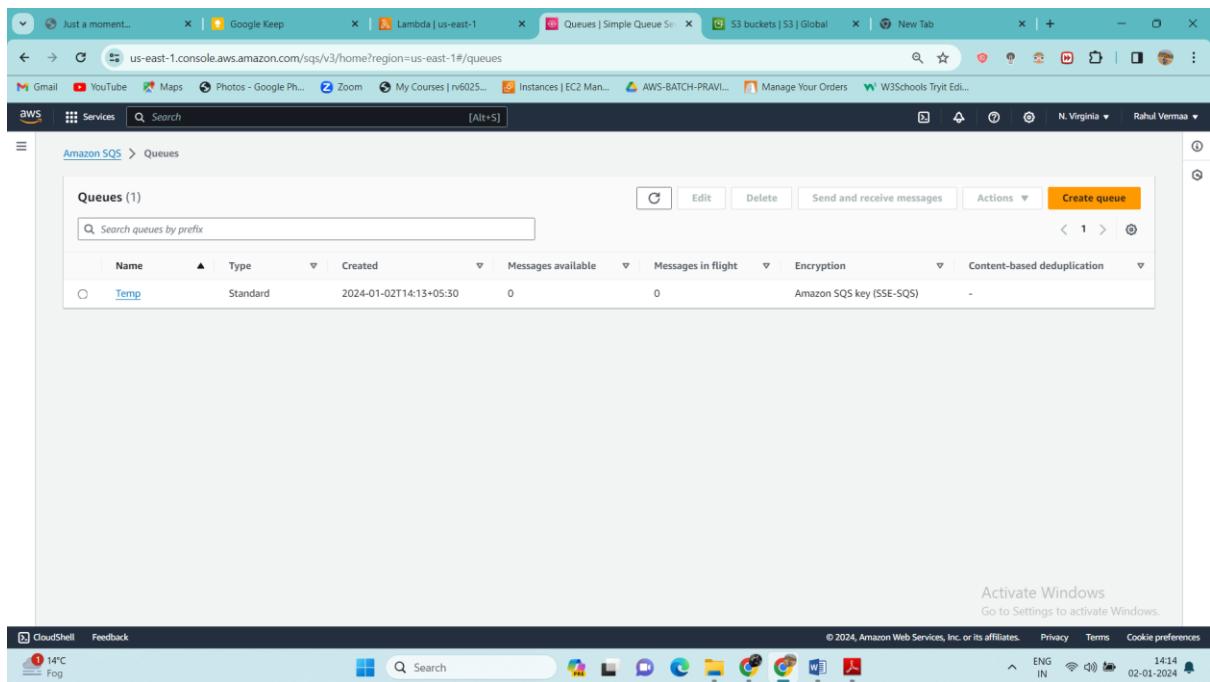
SNS subscriptions **Info**

Subscription region
us-east-1

SNS subscriptions (0) **Info**

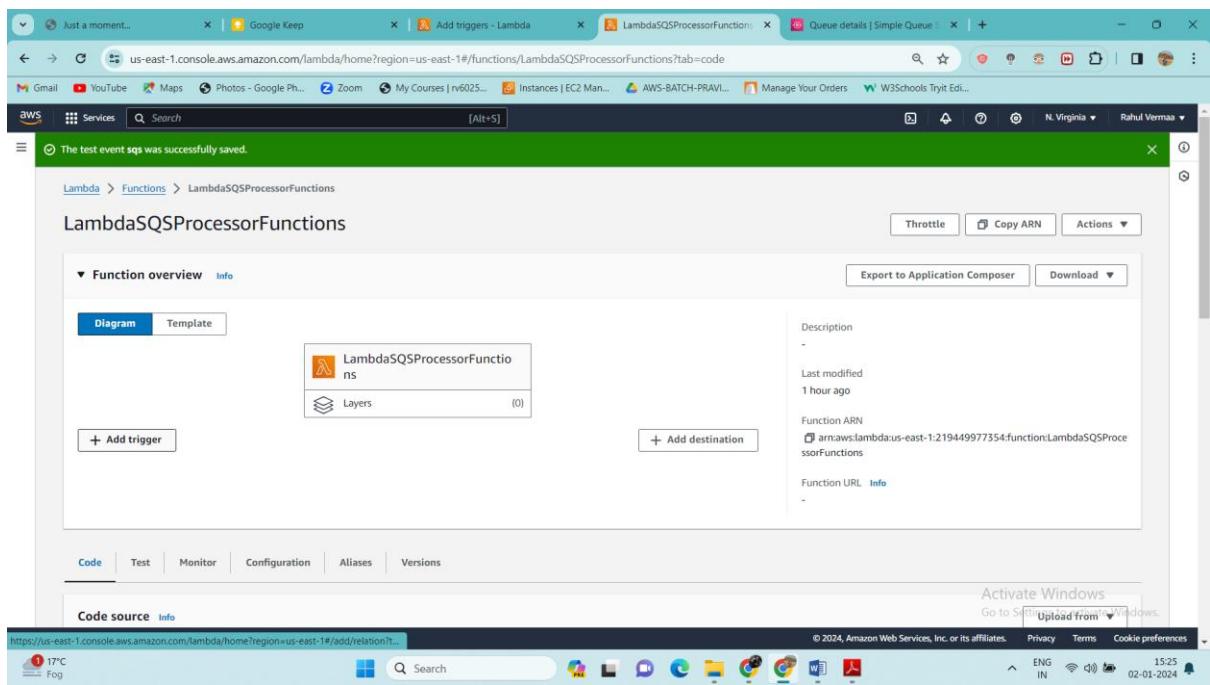
View in SNS Delete Subscribe to Amazon SNS topic

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 14:13 02-01-2024



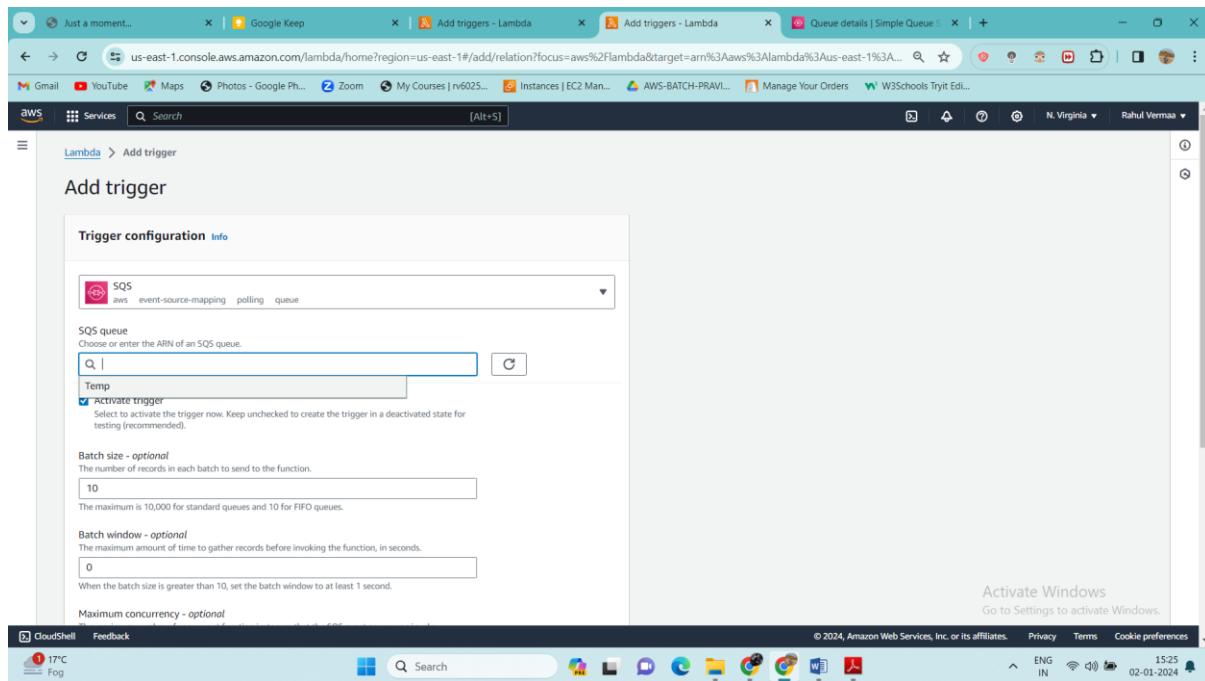
The screenshot shows the AWS Lambda console with the 'Add triggers' step selected. The 'SQS' trigger is chosen, and the 'SQS' dropdown shows a list of available queues, including 'Temp'.

Now we have to add this SQS with our lambda, we just go to lambda function and click it on add trigger option and select our SQS- Temp

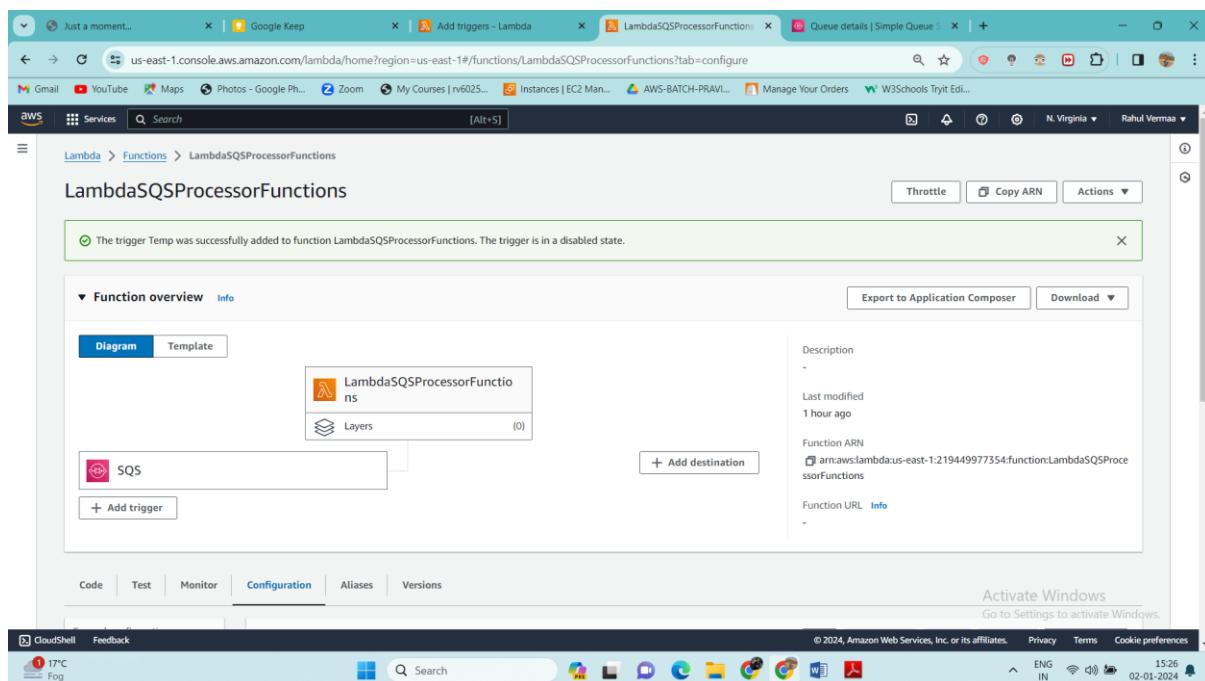


The screenshot shows the LambdaSQSPProcessorFunctions function details. The 'Add trigger' button is visible under the 'Function overview' tab.

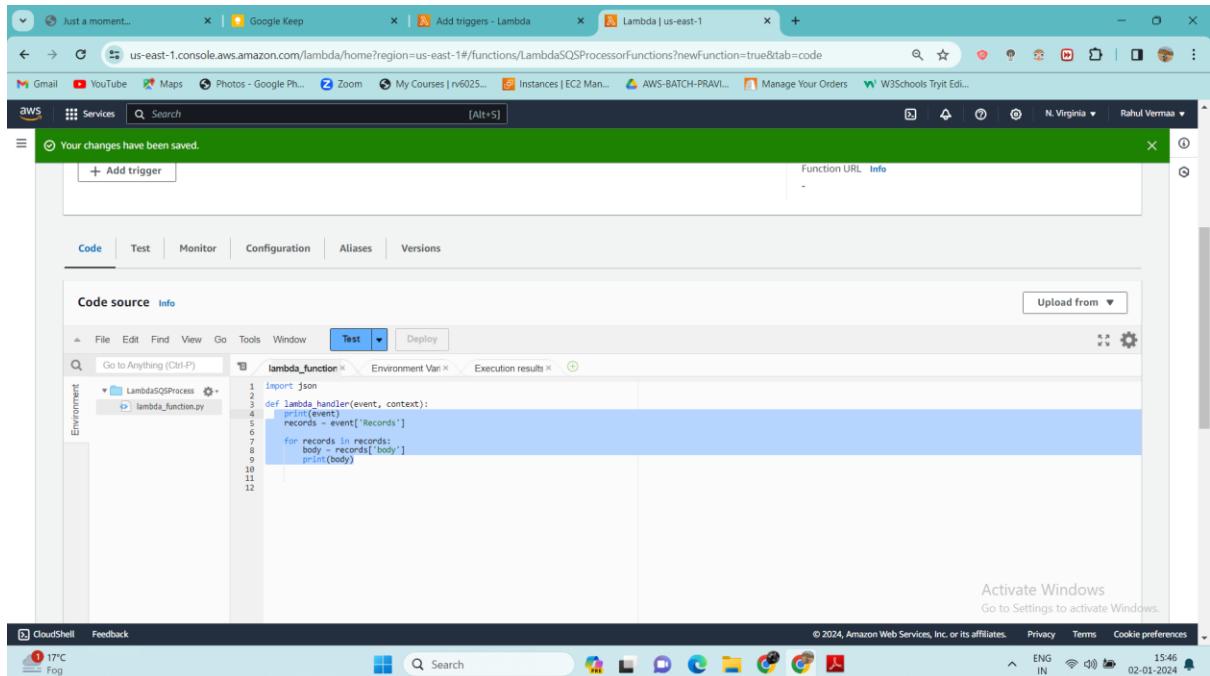
Select your SQS, We have disabled trigger as of now



And it's done



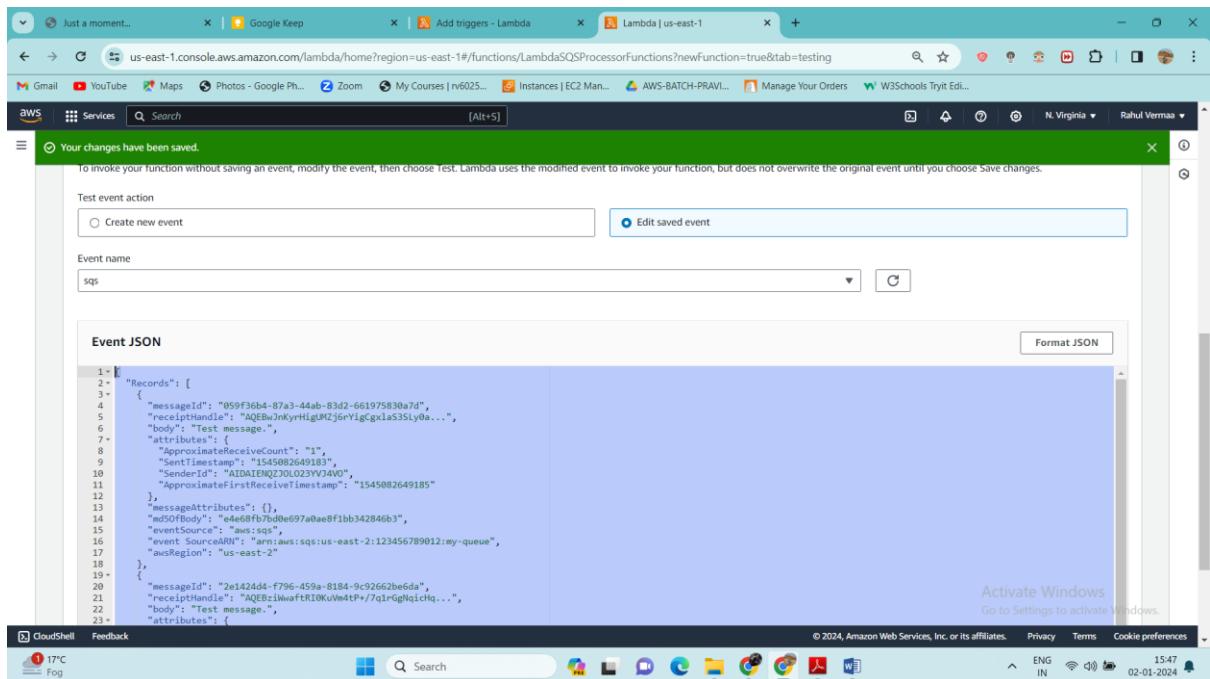
Now we have to edit code



The screenshot shows the AWS Lambda code editor for a function named 'lambda_function'. The code is as follows:

```
1 import json
2
3 def lambda_handler(event, context):
4     print(event)
5     records = event['Records']
6
7     for records in records:
8         body = records["body"]
9         print(body)
10
11
12
```

Now we have to create one test event and test it



The screenshot shows the 'Test event action' section of the AWS Lambda function configuration. It includes fields for 'Event name' (set to 'sq') and an 'Event JSON' editor. The 'Event JSON' editor contains a JSON object representing an SQS message:

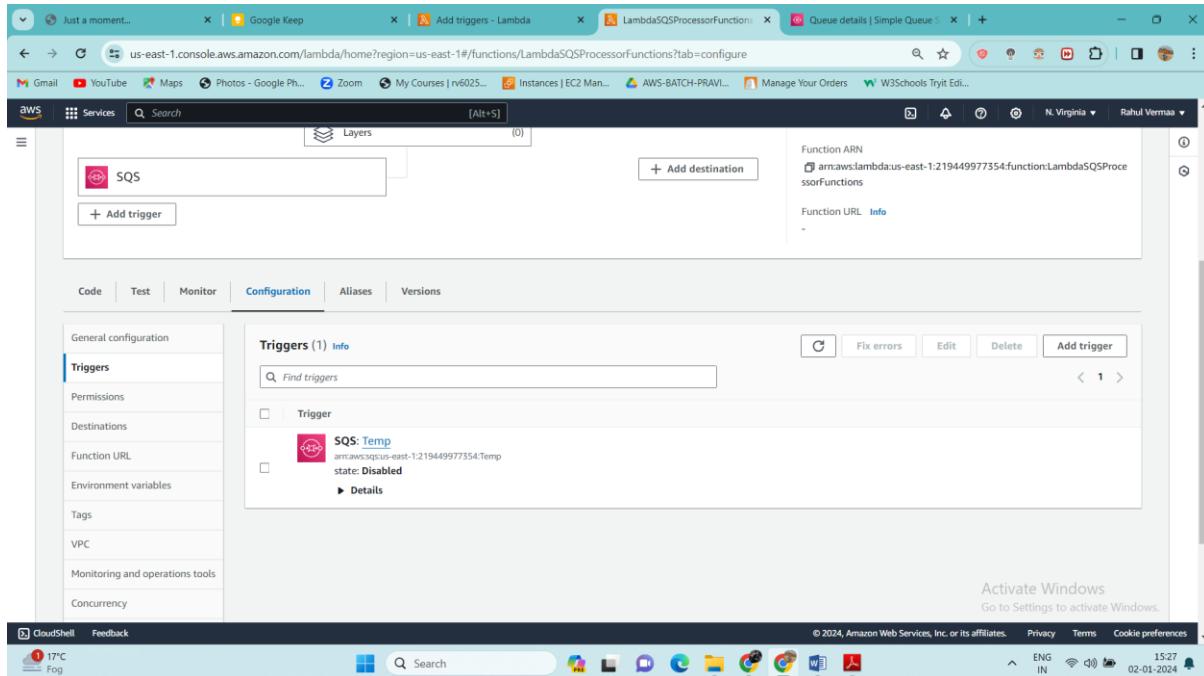
```
1 {
2     "Records": [
3         {
4             "messageId": "059f36b4-97a3-44ab-83d2-661975830a7d",
5             "receiptHandle": "AQEBuJnkyrHigUMZj6rYigCgxla53SLy0a...",
6             "body": "Test message",
7             "attributes": {
8                 "ApproximateReceiveCount": "1",
9                 "SentTimestamp": "1545082649183",
10                "SenderId": "AIDAIENQ230L023YV3AV0",
11                "ApproximateFirstReceiveTimestamp": "1545082649185"
12            },
13            "messageAttributes": {},
14            "md5OfBody": "ed509f1d4de697a0ae8fibb342846b3",
15            "eventSource": "aws:sqs",
16            "eventSourceARN": "arn:aws:sqs:us-east-1:123456789012:my-queue",
17            "awsRegion": "us-east-2"
18        },
19        {
20            "messageId": "2e1424d4-f796-459a-9184-9c92662be6da",
21            "receiptHandle": "AQEBz1mkwftR10KuVm4tP+7q1rGqicHQ...",
22            "body": "Test message",
23            "attributes": {}
24        }
25    ]
26}
```

We can get this from aws documentation:

```
{  
  "Records": [  
    {  
      "messageId": "059f36b4-87a3-44ab-83d2-  
      661975830a7d",  
      "receiptHandle":  
      "AQEBwJnKyrHigUMZj6rYigCgxlaS3SLy0a...",  
      "body": "Test message.",  
      "attributes": {  
        "ApproximateReceiveCount": "1",  
        "SentTimestamp": "1545082649183",  
        "SenderId": "AIDAIENQZJOLO23YVJ4VO",  
        "ApproximateFirstReceiveTimestamp":  
        "1545082649185"  
      },  
      "messageAttributes": {},  
      "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",  
      "eventSource": "aws:sqs",  
      "event SourceARN": "arn:aws:sqs:us-east-  
      2:123456789012:my-queue",  
      "awsRegion": "us-east-2"  
    },  
  ]}
```

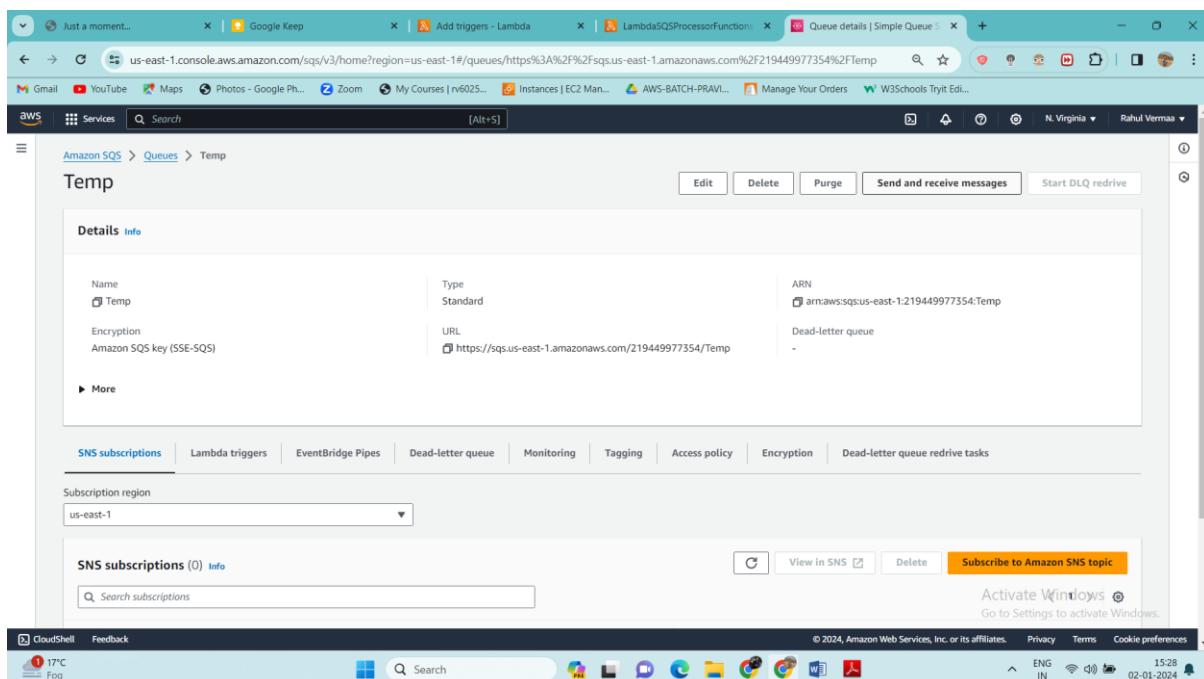
```
{  
  "messageId": "2e1424d4-f796-459a-8184-  
9c92662be6da",  
  "receiptHandle":  
  "AQEBziWwaftRIOKuVm4tP+/7q1rGgNqicHq...",  
  "body": "Test message.",  
  "attributes": {  
    "ApproximateReceiveCount": "1",  
    "SentTimestamp": "1545082650636",  
    "SenderId": "AIDAIENQZJOLO23YV34VO",  
    "ApproximateFirstReceiveTimestamp":  
    "1545082650649"  
  },  
  "messageAttributes": {},  
  "md5OfBody": "e4e68fb7bd0e697a0ae8f1bb342846b3",  
  "eventSource": "aws:sqs",  
  "eventSourceARN": "arn:aws:sqs:us-east-  
2:123456789012:my-queue",  
  "awsRegion": "us-east-2"  
}  
]  
}
```

Our trigger is disabled now as you can see



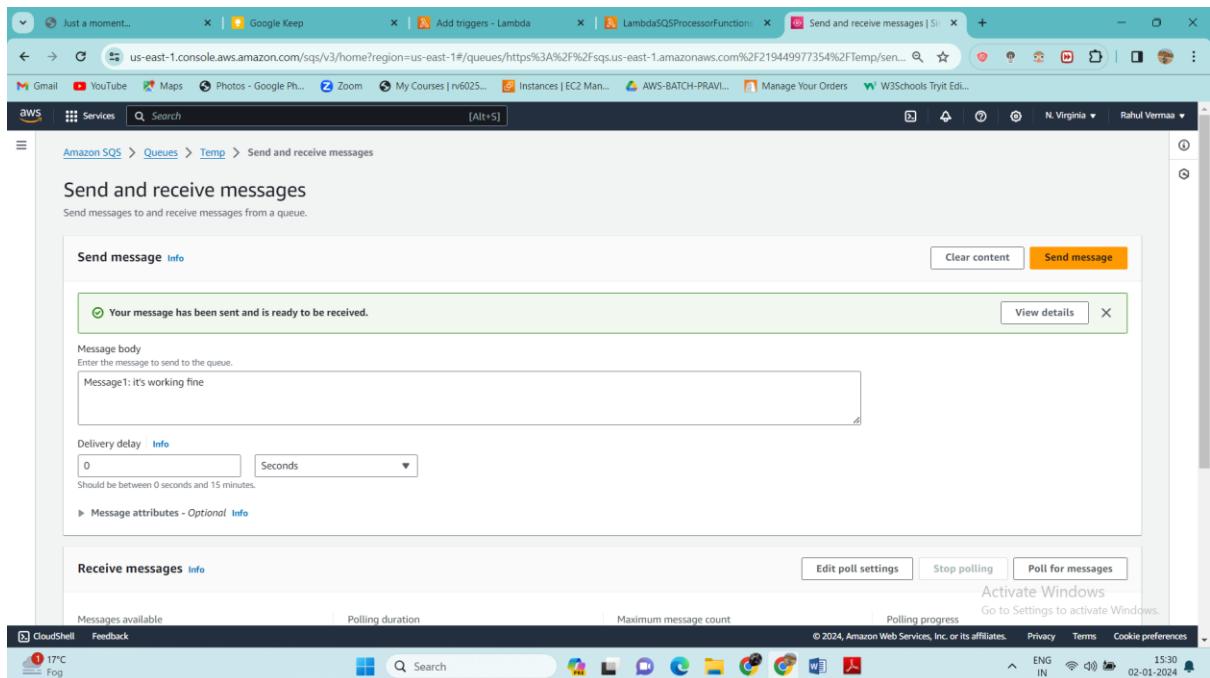
The screenshot shows the AWS Lambda console with the 'Configuration' tab selected. On the left, a sidebar lists options like General configuration, Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, VPC, and Monitoring and operations tools. The 'Triggers' option is currently selected. In the main pane, the 'Triggers (1) Info' section shows a single trigger named 'SQS: Temp'. The trigger is associated with the ARN `arn:aws:lambda:us-east-1:219449977354:function:LambdaSQSPProcessorFunctions` and the URL `https://lambda.us-east-1.amazonaws.com/2019-03-01/functions/arn:aws:lambda:us-east-1:219449977354:function:LambdaSQSPProcessorFunctions/invocations`. The status is listed as 'Disabled'.

Now go to SQS and send some messages



The screenshot shows the AWS SQS console. The top navigation bar includes links for CloudShell, Feedback, and various AWS services. The main content area shows a queue named 'Temp' under the 'Queues' section. The 'Details' tab is selected, displaying the queue's name ('Temp'), type ('Standard'), ARN ('arn:aws:sqs:us-east-1:219449977354:Temp'), and URL ('https://sqs.us-east-1.amazonaws.com/219449977354/Temp'). The 'Dead-letter queue' field is empty. Below the details, tabs for 'SNS subscriptions', 'Lambda triggers', 'EventBridge Pipes', 'Dead-letter queue', 'Monitoring', 'Tagging', 'Access policy', 'Encryption', and 'Dead-letter queue redrive tasks' are visible. The 'SNS subscriptions' tab shows a single subscription to an SNS topic with the ARN `arn:aws:sns:us-east-1:219449977354:Temp`. The 'Send and receive messages' button is located at the top right of the queue details page.

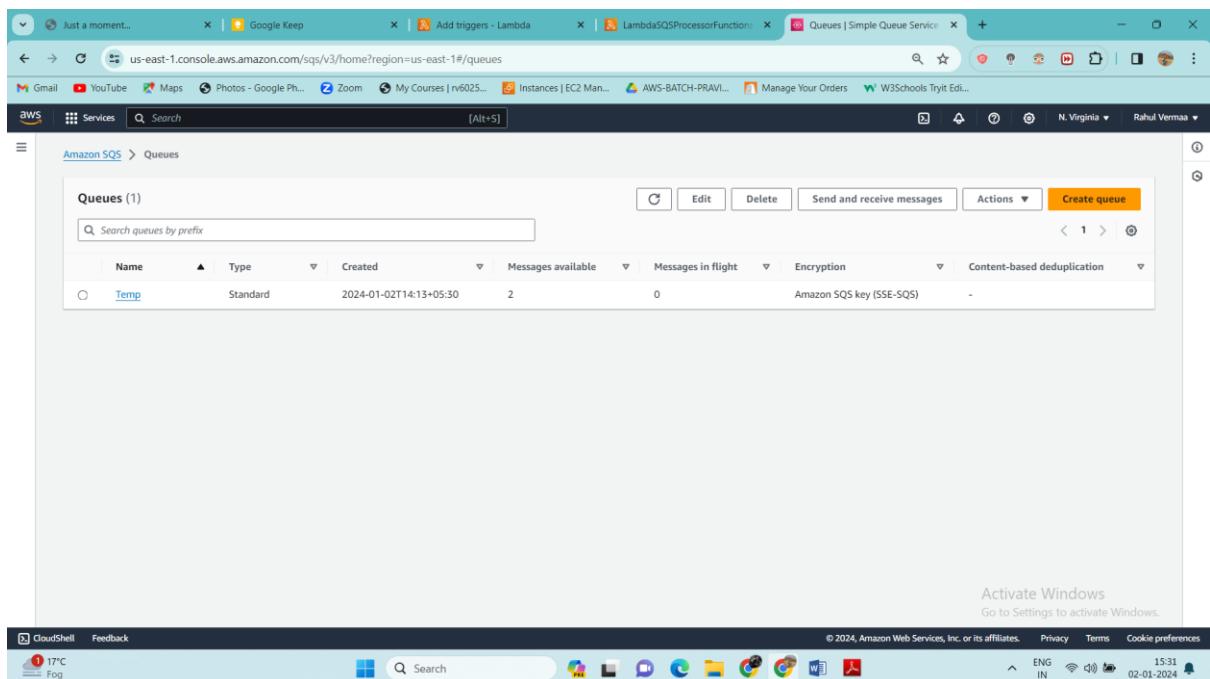
First message



The screenshot shows the AWS SQS 'Send and receive messages' interface. In the 'Send message' section, a message body 'Message1: it's working fine' has been entered. A success message 'Your message has been sent and is ready to be received.' is displayed. In the 'Receive messages' section, there are buttons for 'Edit poll settings', 'Stop polling', and 'Poll for messages'. The status bar at the bottom shows 'Activate Windows' and the date '02-01-2024'.

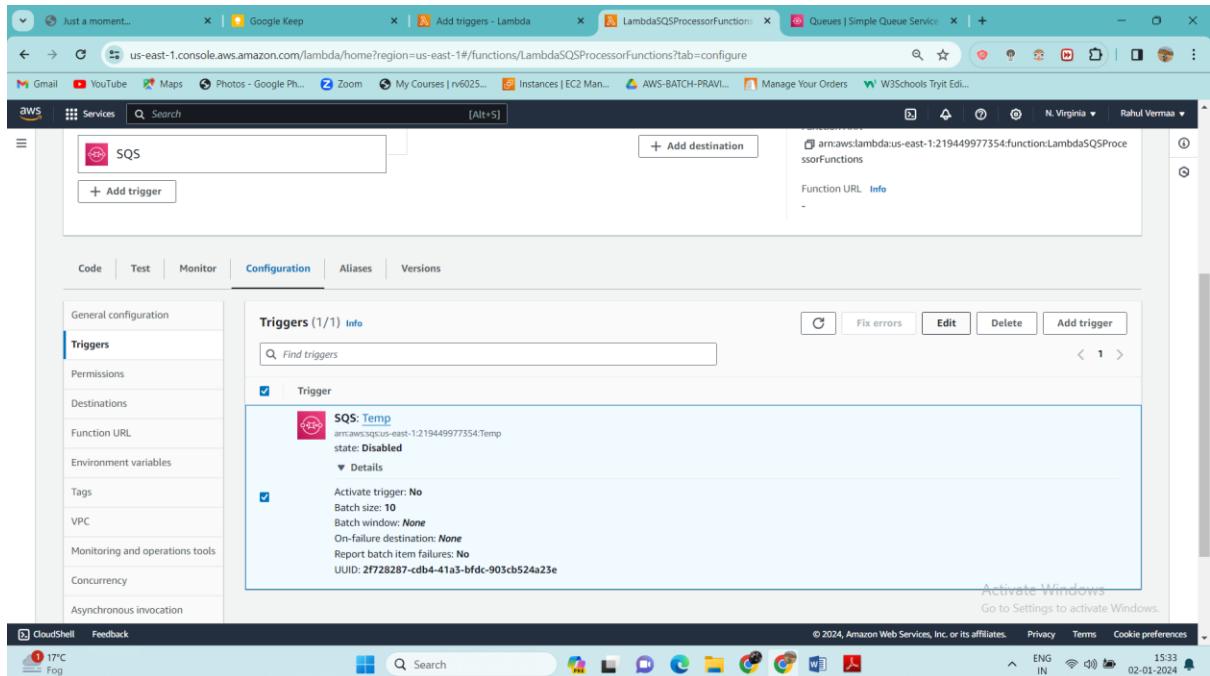
Second message- Message2 finally it's done

And after that go and check queue (it's showing message available 2)



The screenshot shows the AWS SQS 'Queues' interface. It displays a table with one queue named 'Temp'. The queue details are: Type: Standard, Created: 2024-01-02T14:13:05:30, Messages available: 2, Messages in flight: 0, Encryption: Amazon SQS key (SSE-SQS), and Content-based deduplication: -. The status bar at the bottom shows 'Activate Windows' and the date '02-01-2024'.

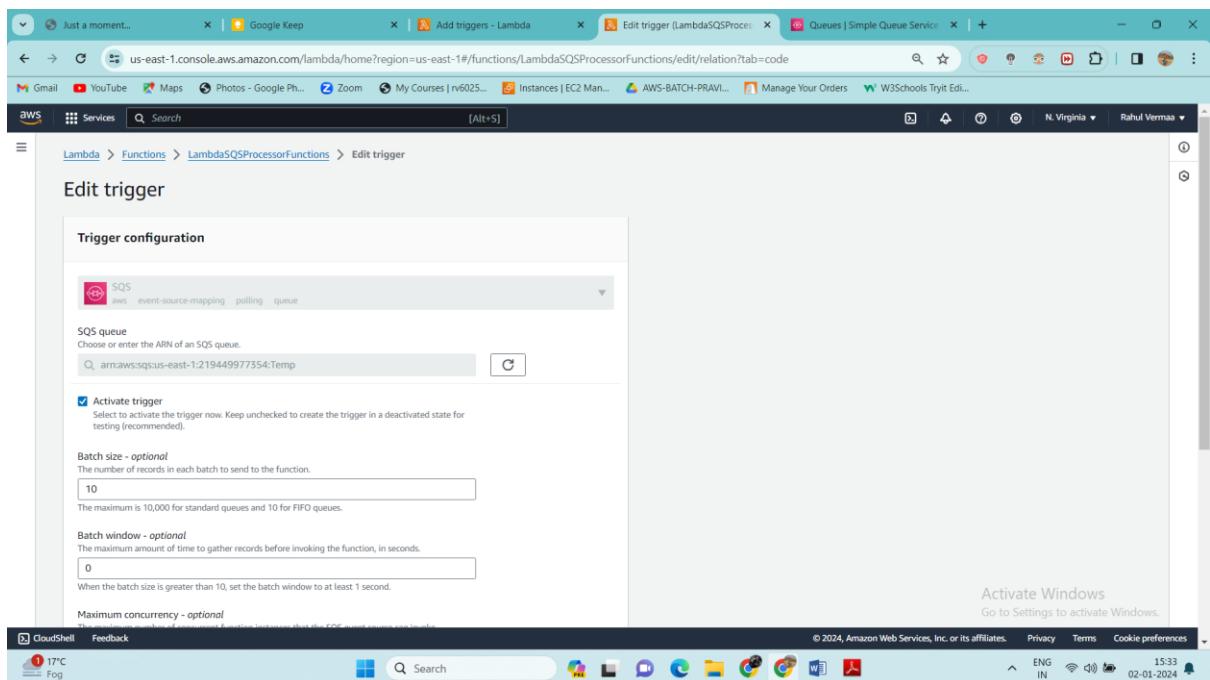
Now we will enable our SQS trigger



The screenshot shows the AWS Lambda triggers configuration page. The left sidebar lists triggers, permissions, destinations, function URL, environment variables, tags, VPC, monitoring and operations tools, concurrency, and asynchronous invocation. The main panel shows a table for triggers, with one entry named 'Temp'. The 'Temp' trigger is associated with the ARN `arn:aws:sqs:us-east-1:219449977354:Temp` and is currently **Disabled**. The configuration details for 'Temp' include:

- Activate trigger: **No**
- Batch size: **10**
- Batch window: **None**
- On-failure destination: **None**
- Report batch item failures: **No**
- UUID: **2f728287-cdb4-41a3-bfdc-903cb524a23e**

At the bottom right, there is a note: "Activate Windows" and "Go to Settings to activate Windows." The status bar at the bottom shows the date and time as 02-01-2024.

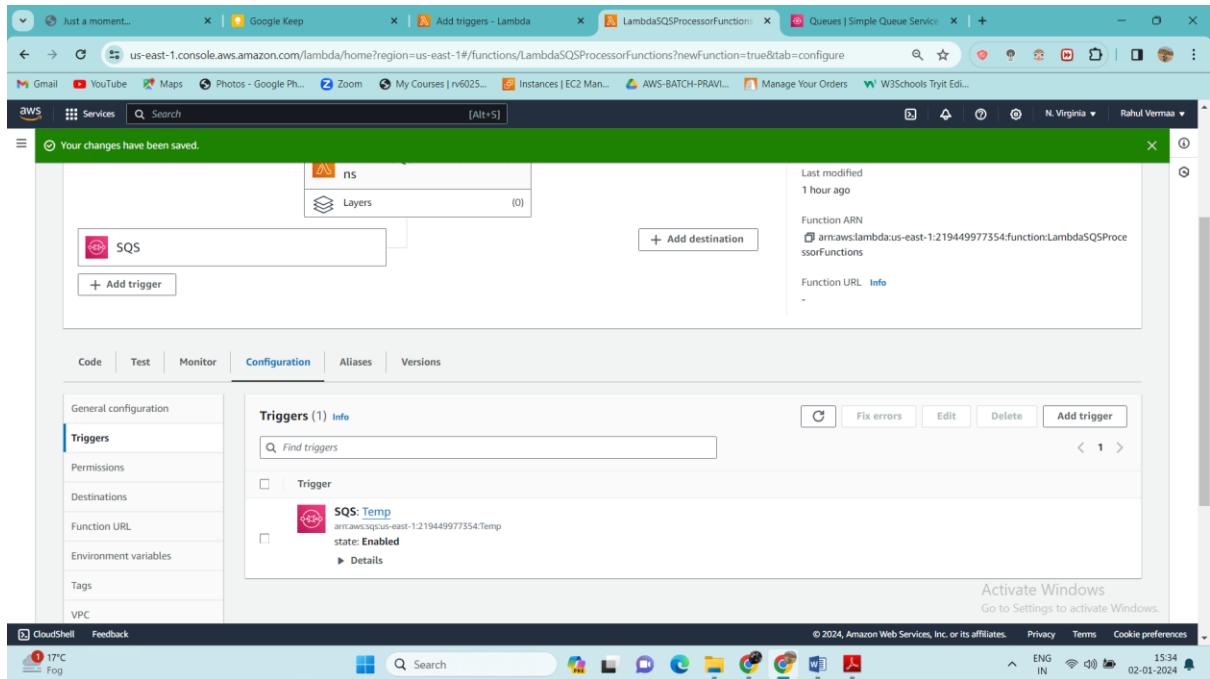


The screenshot shows the 'Edit trigger' interface for the 'Temp' trigger. The left sidebar shows the navigation path: Lambda > Functions > LambdaSQSProcessorFunctions > Edit trigger. The main panel is titled 'Edit trigger' and contains the 'Trigger configuration' section. The configuration fields are as follows:

- Trigger configuration:** SQS (aws.event-source-mapping.polling.queue)
- SQS queue:** Choose or enter the ARN of an SQS queue. The input field contains `arn:aws:sqs:us-east-1:219449977354:Temp`.
- 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. The input field contains `10`.
- Batch window - optional:** The maximum amount of time to gather records before invoking the function, in seconds. The input field contains `0`.
- Maximum concurrency - optional:** The maximum number of concurrent executions for this trigger. The input field contains `1`.

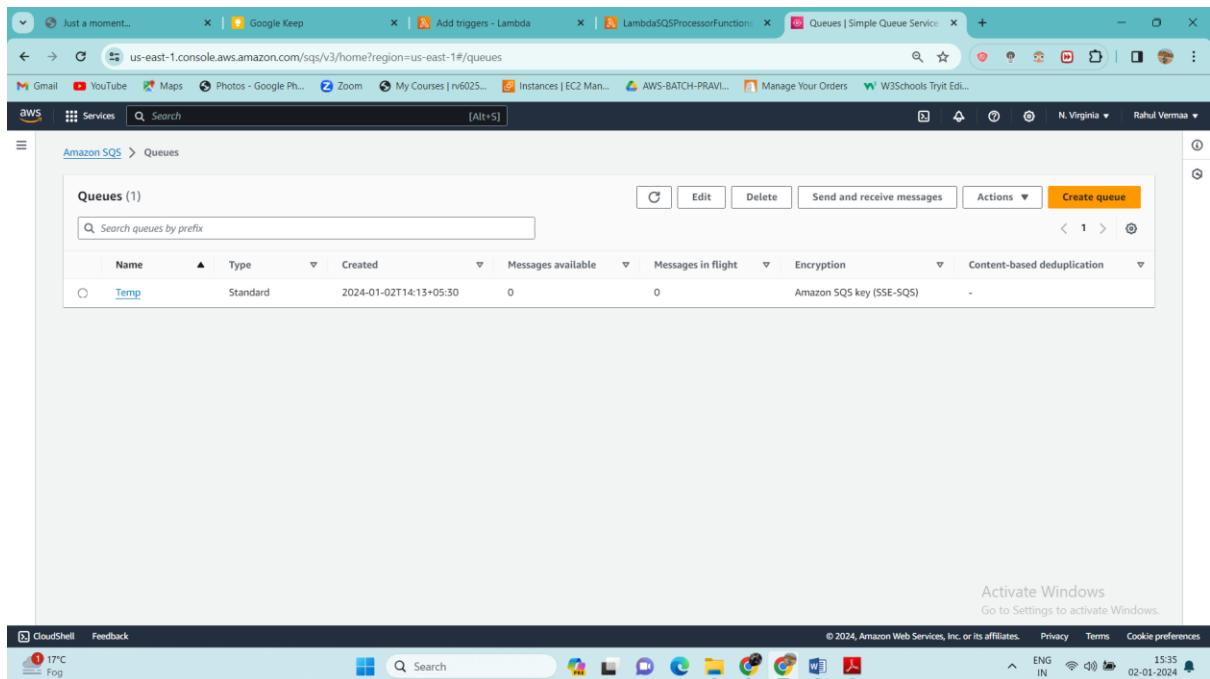
At the bottom right, there is a note: "Activate Windows" and "Go to Settings to activate Windows." The status bar at the bottom shows the date and time as 02-01-2024.

And it's done



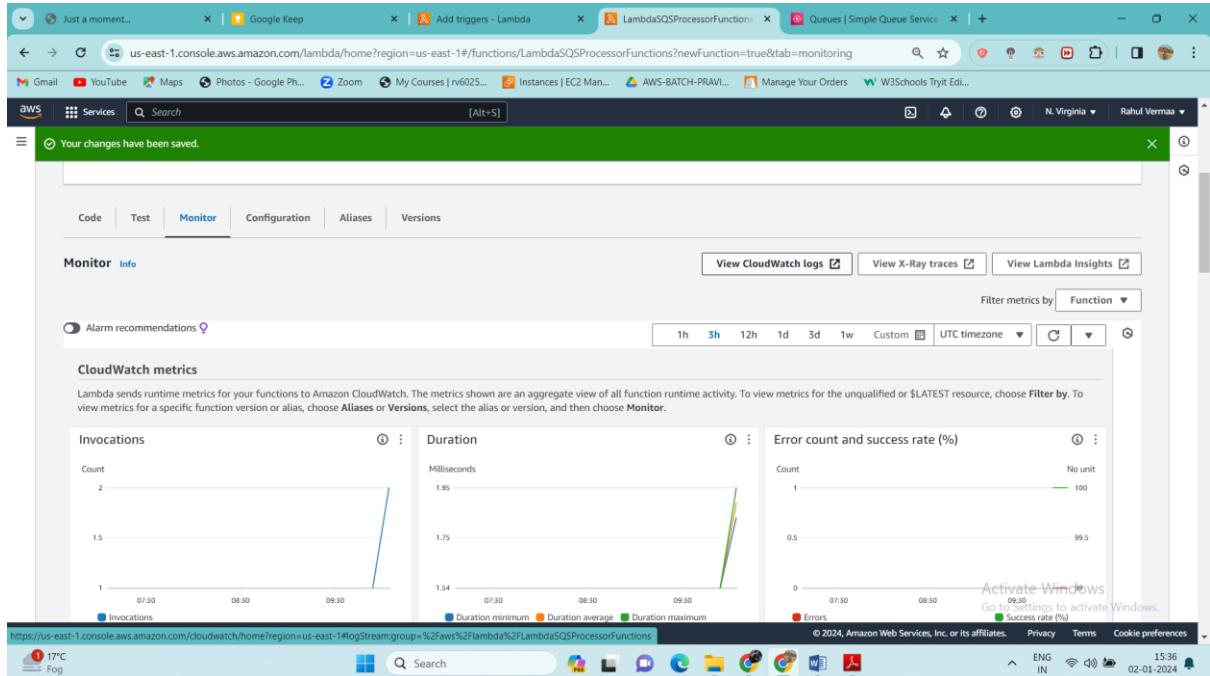
The screenshot shows the AWS Lambda console for the function 'LambdaSQSProcessorFunctions'. The 'Configuration' tab is selected. On the left, a sidebar lists 'General configuration', 'Triggers', 'Permissions', 'Destinations', 'Function URL', 'Environment variables', 'Tags', and 'VPC'. The 'Triggers' section is expanded, showing a table with one item: 'SQS: Temp' (arn:aws:sqs:us-east-1:123456789012:temp) with the state 'Enabled'. A message at the top says 'Your changes have been saved.'

Now let's check our queue. As we can see something pulled out those messages.



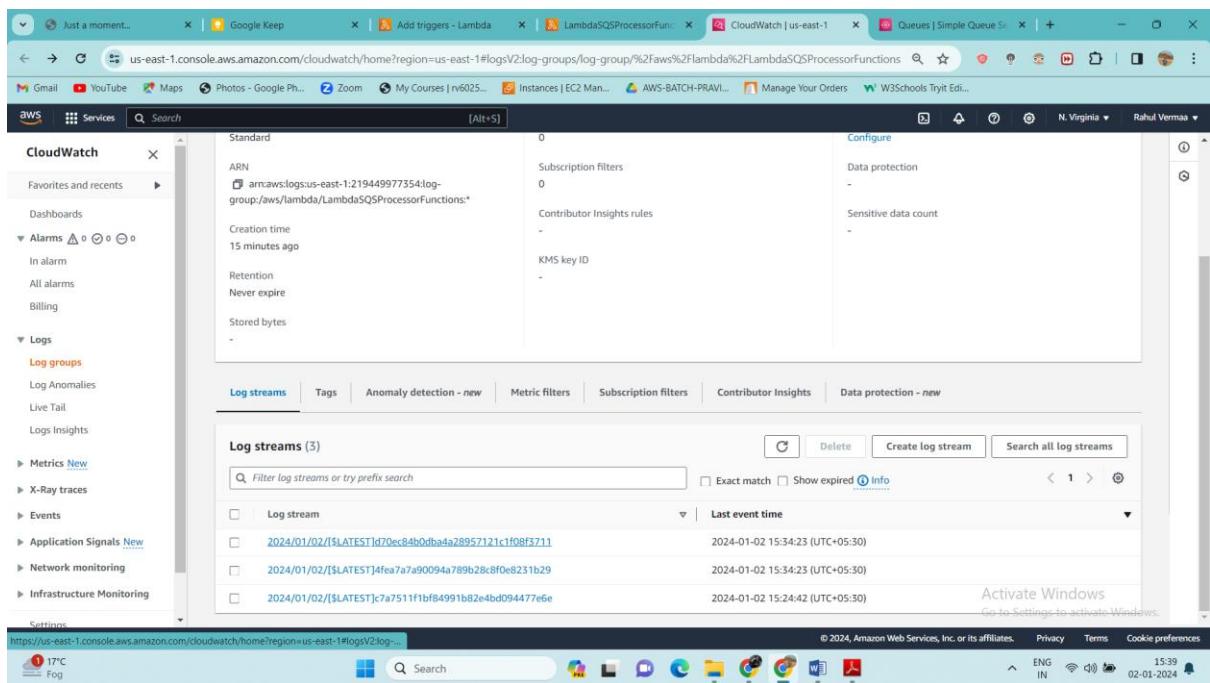
The screenshot shows the AWS SQS console for the queue 'Temp'. The 'Queues' table has one row: 'Temp' (Standard, 2024-01-02T14:13+05:30, 0 messages available, 0 messages in flight, Amazon SQS key (SSE-SQS)). The 'Actions' dropdown menu is open, showing options like 'Edit', 'Delete', 'Send and receive messages', and 'Create queue'. A message at the top says 'Activate Windows'.

Let's check in our lambda function now. Got to monitor- and click it on view CloudWatch logs



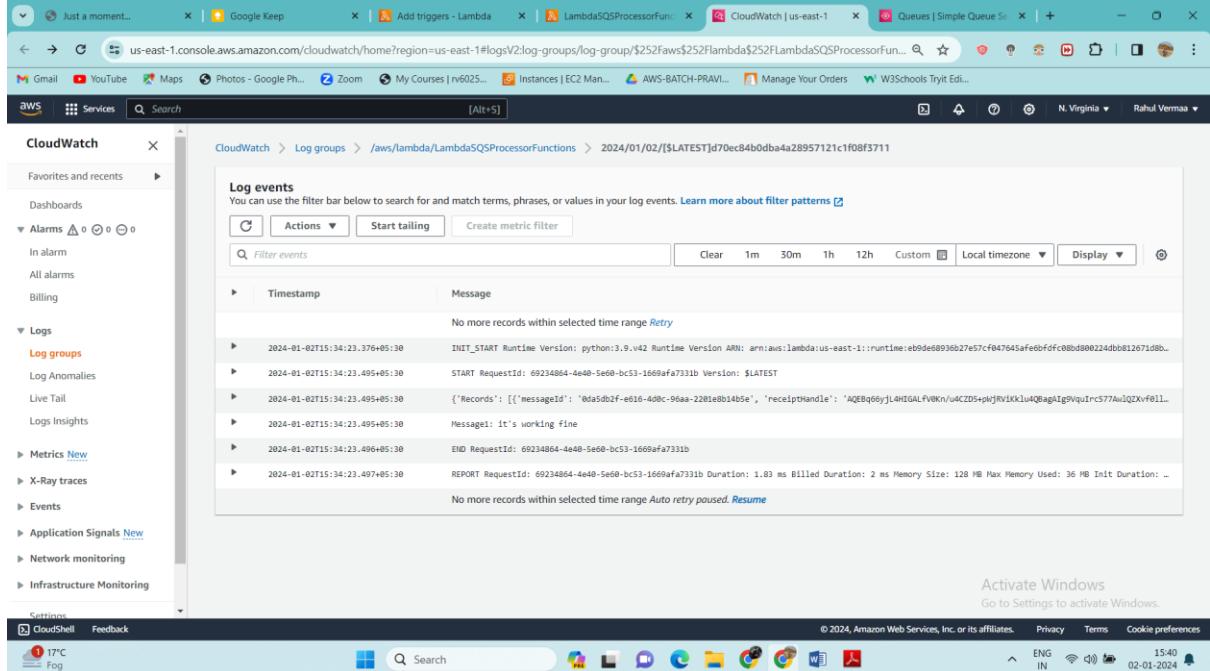
The screenshot shows the AWS Lambda function monitoring page for 'LambdaSQSPProcessorFunction'. The 'Monitor' tab is selected. The 'CloudWatch metrics' section displays three charts: 'Invocations' (Count: 2), 'Duration' (Milliseconds: 1.95), and 'Error count and success rate (%)' (Count: 1, Success rate: 99.5%). Below the charts, there are buttons for 'View CloudWatch logs', 'View X-Ray traces', and 'View Lambda Insights'.

And we just click it on first one



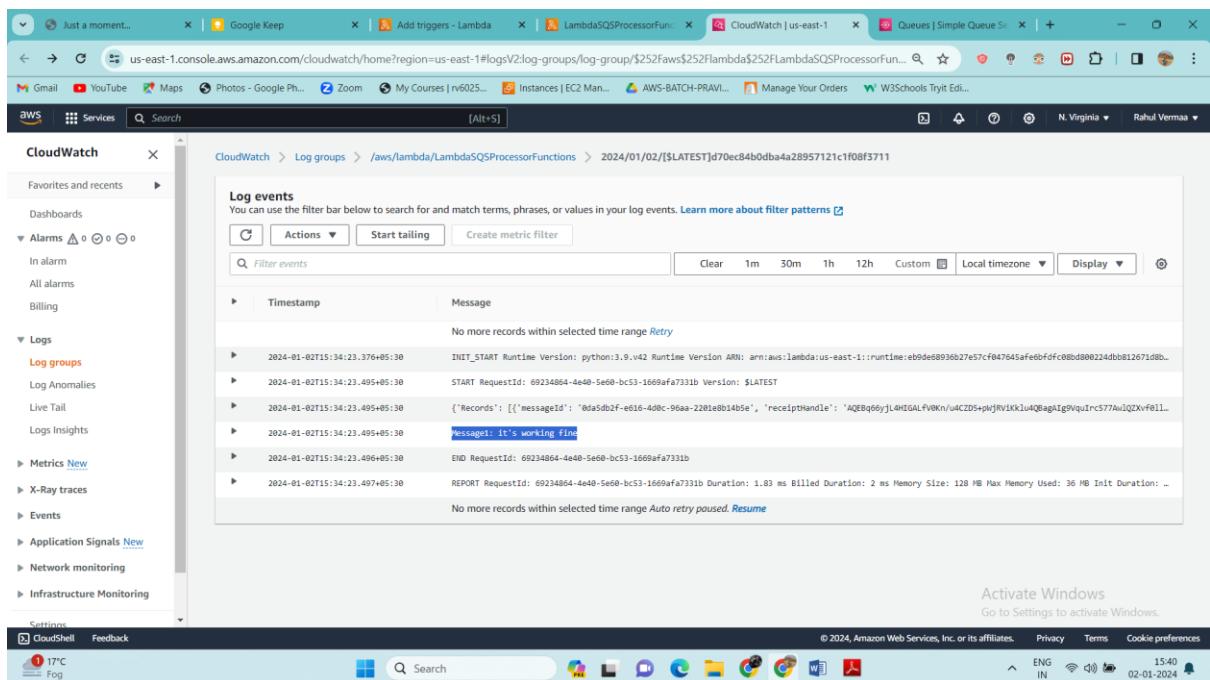
The screenshot shows the AWS CloudWatch logs page for the 'LambdaSQSPProcessorFunction' log group. The 'Log streams' section lists three log streams: '2024/01/02/[\$LATEST]ld70ec84b0dba4a28957121c1f08f3211' (Last event time: 2024-01-02 15:34:23 UTC+05:30), '2024/01/02/[\$LATEST]4fea7a7a90094a789b28c8f0e8231b29' (Last event time: 2024-01-02 15:34:23 UTC+05:30), and '2024/01/02/[\$LATEST]c7a7511fbf84991b82e4bd094477e6e' (Last event time: 2024-01-02 15:24:42 UTC+05:30). The left sidebar shows the CloudWatch navigation menu with 'Logs' selected.

And we can see our message 1 is there



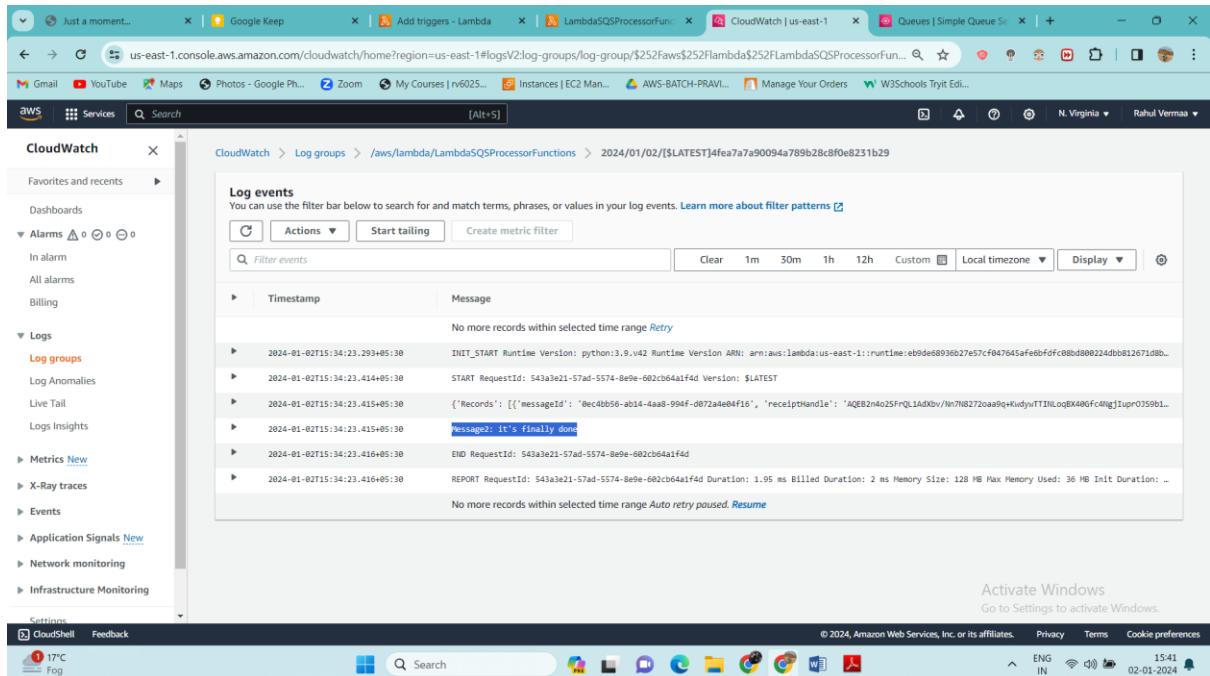
The screenshot shows the AWS CloudWatch Log Groups interface. The left sidebar shows navigation options like Dashboards, Alarms, Logs (Log groups, Log anomalies, Live tail, Logs insights), Metrics, X-Ray traces, Events, Application Signals, Network monitoring, and Infrastructure Monitoring. The main content area is titled 'Log events' and shows a list of log entries. The first entry is 'Message1: it's working fine'. The log entries are timestamped and include details like RequestID and Duration.

Timestamp	Message
2024-01-02T15:34:23.376+05:30	INIT_START Runtime Version: python:3.9.v42 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:eb9d68936b27e57cf04764safe6bfdfc08bd800224dbb812671db0...
2024-01-02T15:34:23.495+05:30	START RequestID: 69234864-4e40-5e08-bc53-1669afa7331b Version: \$LATEST
2024-01-02T15:34:23.495+05:30	{ "Records": [{ "messageId": "0645db2f-e016-40bc-96aa-2201e01a01b5e", "receiptHandle": "AQEBqd6yjL4HIGALPv0K/u4CZD5+pljRv1K2l4Q8agIg%QqUrc577Av1Q2XvF0ll..." }
2024-01-02T15:34:23.495+05:30	Message1: it's working fine
2024-01-02T15:34:23.496+05:30	END RequestID: 69234864-4e40-5e08-bc53-1669afa7331b
2024-01-02T15:34:23.497+05:30	REPORT RequestID: 69234864-4e40-5e08-bc53-1669afa7331b Duration: 1.83 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 36 MB Init Duration: -



This screenshot is identical to the one above, showing the AWS CloudWatch Log Groups interface. It displays the same log events for the Lambda function, including the message 'Message1: it's working fine'. The interface includes the same sidebar and main content area with the log events table.

Message 2 also there



CloudWatch > Log groups > /aws/lambda/LambdaSQSPProcessorFunctions > 2024/01/02/[\$LATEST]4fea7a7a90094a789b28c8f0e8231b29

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Timestamp	Message
2024-01-02T15:34:23.293+05:30	INIT_START Runtime Version: python:3.9.v42 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:eb9ded8936b27e57cf04764afe6bfdfc08bd800224dbb812671db0..
2024-01-02T15:34:23.414+05:30	START RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d Version: \$LATEST
2024-01-02T15:34:23.415+05:30	{ "Records": [{ "messageId": "8ec4bb56-ab14-4aa8-994f-d072a4e04f16", "receiptHandle": "AQEB2n4o25FrQ1Ad0bV/Nn7H8272oaa8q+KudyTTInLoqBX40FcWigJ1uprO399b1..
2024-01-02T15:34:23.415+05:30	Message2: it's finally done
2024-01-02T15:34:23.416+05:30	END RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d
2024-01-02T15:34:23.416+05:30	REPORT RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d Duration: 1.95 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 36 MB Init Duration: ..

No more records within selected time range [Retry](#)

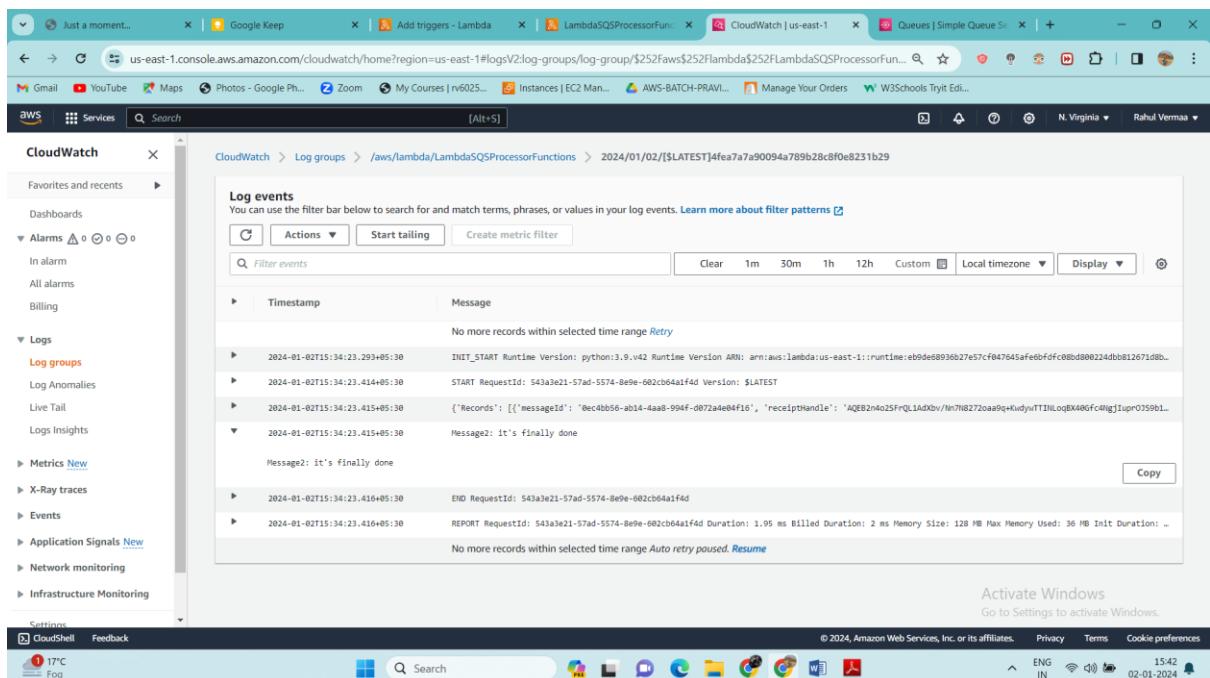
Activate Windows
Go to Settings to activate Windows.

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudShell Feedback

17°C Fog

15:41 02-01-2024



CloudWatch > Log groups > /aws/lambda/LambdaSQSPProcessorFunctions > 2024/01/02/[\$LATEST]4fea7a7a90094a789b28c8f0e8231b29

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Timestamp	Message
2024-01-02T15:34:23.293+05:30	INIT_START Runtime Version: python:3.9.v42 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:eb9ded8936b27e57cf04764afe6bfdfc08bd800224dbb812671db0..
2024-01-02T15:34:23.414+05:30	START RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d Version: \$LATEST
2024-01-02T15:34:23.415+05:30	{ "Records": [{ "messageId": "8ec4bb56-ab14-4aa8-994f-d072a4e04f16", "receiptHandle": "AQEB2n4o25FrQ1Ad0bV/Nn7H8272oaa8q+KudyTTInLoqBX40FcWigJ1uprO399b1..
2024-01-02T15:34:23.415+05:30	Message2: it's finally done
2024-01-02T15:34:23.415+05:30	Message2: it's finally done
2024-01-02T15:34:23.416+05:30	END RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d
2024-01-02T15:34:23.416+05:30	REPORT RequestId: 543a3e21-57ad-5574-8e9e-682cb64a1f4d Duration: 1.95 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 36 MB Init Duration: ..

No more records within selected time range [Retry](#)

Activate Windows
Go to Settings to activate Windows.

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudShell Feedback

17°C Fog

15:42 02-01-2024