	Marwadi University **Faculty of Engineering and Technology** **Department of Information and Communication Technology**	
Subject: Cloud Developing (01CT0720)	**Aim: Implement the messaging system using Amazon SNS and Amazon SQS.**	
Experiment No: 11	**Date:**	**Enrolment No: 92200133030**

Aim :- Implement the messaging system using Amazon SNS and Amazon SQS.

Lab overview and objectives

In this lab, you will use Amazon Simple Queue Service (Amazon SQS) and Amazon Simple Notification Service (Amazon SNS) to set up a system to receive, queue, and send data for an application to process. You will use a Python publisher to send messages to a notification topic. You will also review the Node.js consumer that a web application will use to retrieve and process the data from a queue.

After completing this lab, you should be able to:

- Configure SNS topics and SQS queues to support programmatic receipt of messages
- Develop an Amazon SNS publisher to send messages to an SNS topic
- Develop an Amazon SQS consumer to read messages from an SQS queue

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

Scenario

Customers love being able to buy coffee beans from the café, but maintaining the bean inventory is time-consuming. Café employees must call each coffee supplier and then use the coffee suppliers web application to manually update each record. Frank has asked Sofía if she can find a better way to keep the coffee inventory current. He would like café staff to spend more time helping customers and less time doing data entry.

Sofía has researched setting up a messaging system to automatically receive and process inventory updates. Mateo, a café regular and AWS consultant, suggested using an SNS topic to receive messages from suppliers, and an SQS queue to store the messages until the application is ready to process them. This way, the café application won't lose any messages if the application or database happens to be unavailable. He also highly recommended that she create a *dead-letter queue* to handle messages that cannot be processed. Mateo also explained that suppliers will need to run a script called a *producer* to publish their updates to the SNS topic. The coffee suppliers application code will need to include a *consumer* to poll and retrieve messages from the SQS queue.

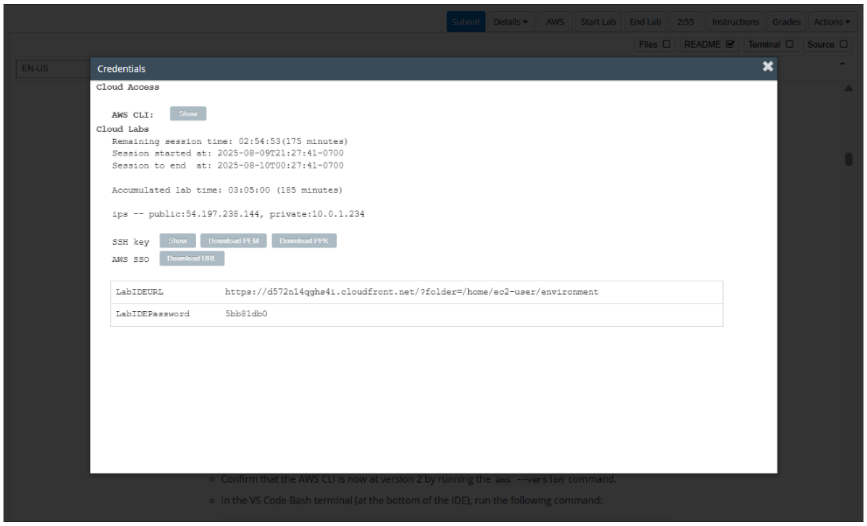
In this lab, you will again play the role of Sofía as you develop the automated inventory processing for the café's coffee suppliers application.

Task 1: Preparing the lab

Connect to the VS Code IDE.

1. At the top of these instructions, choose Details followed by **AWS: Show**

2. Copy values from the table **similar** to the following and paste it into an editor of your choice for use later.
    a. **LabIDEURL**
    b. **LabIDEPassword**

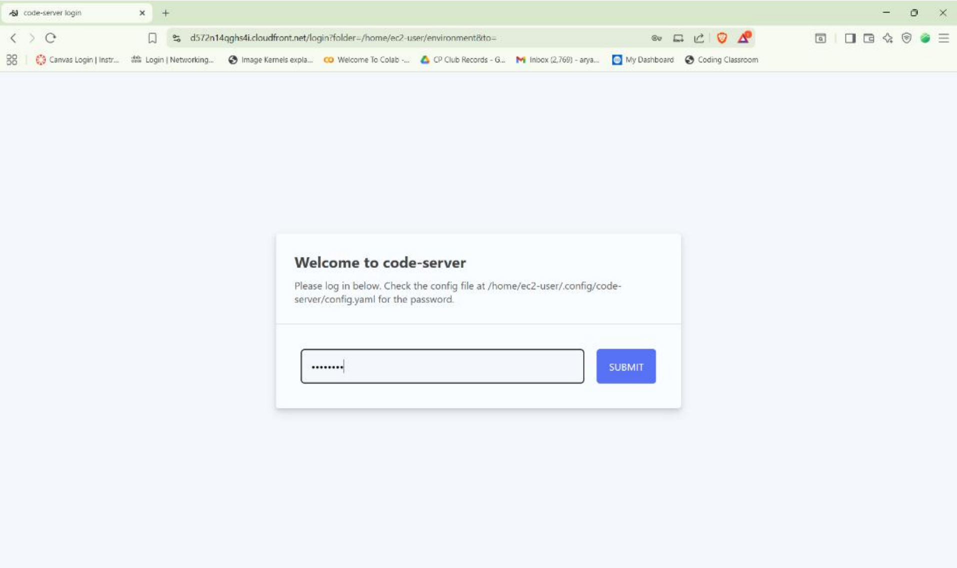


```
Cloud Access

  AWS CLI:
Cloud Labs
  Remaining session time: 02:54:53(175 minutes)
  Session started at: 2025-08-09T21:27:41-0700
  Session to end  at: 2025-08-10T00:27:41-0700

  Accumulated lab time: 03:05:00 (185 minutes)

  ips -- public:54.197.238.144, private:10.0.1.234

  SSH key
  AWS SSO
```

LabIDEURL	https://d572n14qghs4i.cloudfront.net/?folder=/home/ec2-user/environment
LabIDEPassword	5bb81db0

- Confirm that the AWS CLI is now at version 2 by running the `aws --version` command.
- In the VS Code Bash terminal (at the bottom of the IDE), run the following command:

3. In a new browser tab, paste the value for **LabIDEURL** to open the VS Code IDE.
4. On the prompt window **Welcome to code-server**, enter the value for **LabIDEPassword** you copied to the editor earlier, choose **Submit** to open the VS Code IDE.



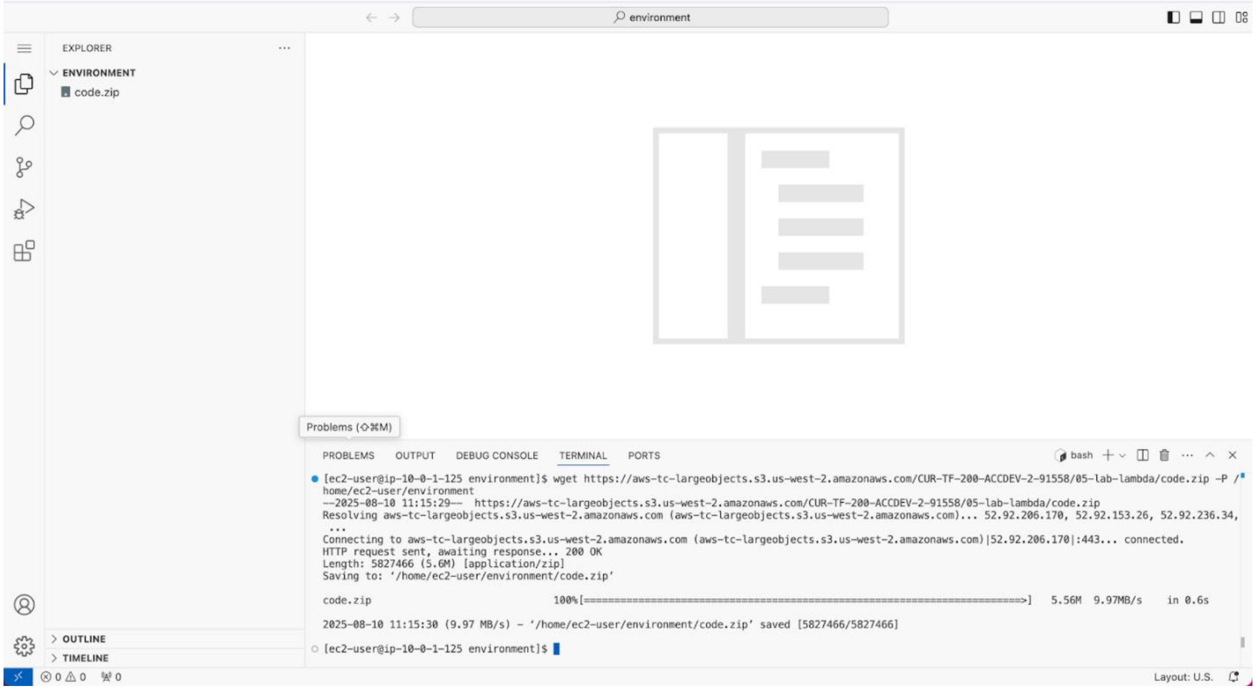
Welcome to code-server

Please log in below. Check the config file at /home/ec2-user/.config/code-server/config.yaml for the password.

•••••••••

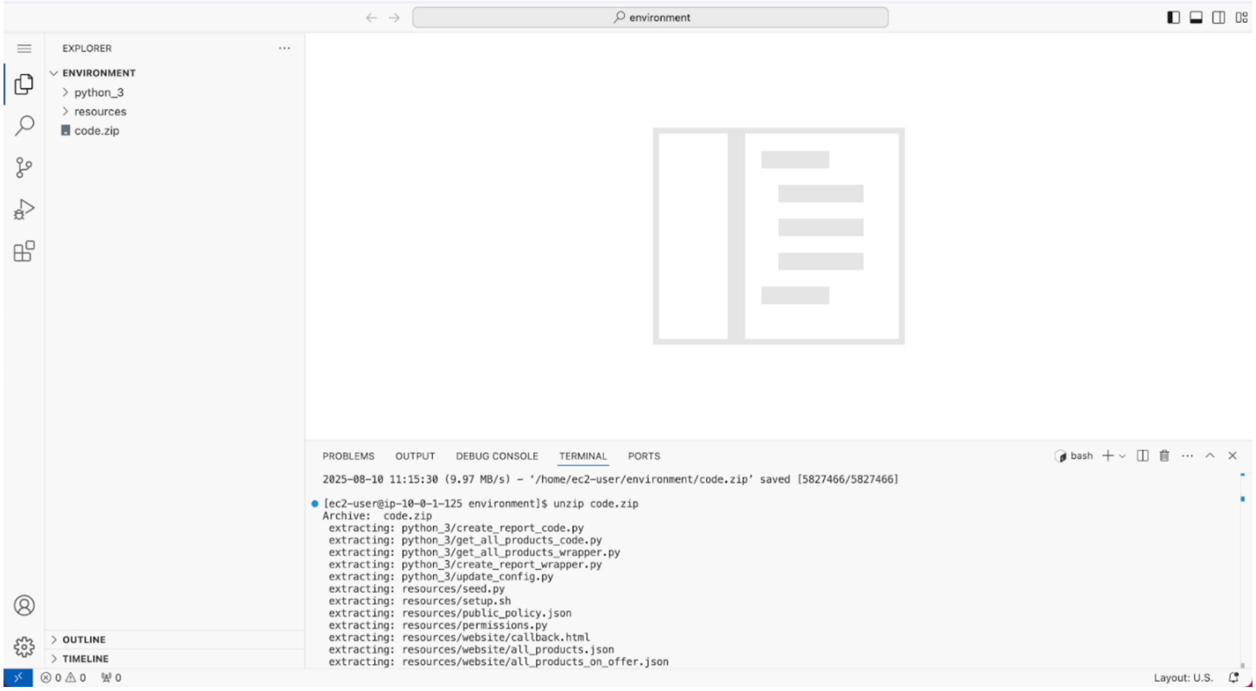
5. Download and extract the files that you need for this lab.
    - In the VS Code bash terminal (located at the bottom of the IDE), run the following commands:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/10-lab-sqs/code.zip -P /home/ec2-user/environment
```



```
[ec2-user@ip-10-0-1-125 environment]$ wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/05-lab-lambda/code.zip -P /
home/ec2-user/environment
--2025-08-10 11:15:29--  https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/05-lab-lambda/code.zip
Resolving aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)... 52.92.206.170, 52.92.153.26, 52.92.236.34,
 ...
Connecting to aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)|52.92.206.170|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5827466 (5.6M) [application/zip]
Saving to: '/home/ec2-user/environment/code.zip'

code.zip                    100%[=========================================================================>]   5.56M  9.97MB/s    in 0.6s

2025-08-10 11:15:30 (9.97 MB/s) - '/home/ec2-user/environment/code.zip' saved [5827466/5827466]

[ec2-user@ip-10-0-1-125 environment]$ 
```

6. You should see that the **code.zip** file was downloaded to the VS Code IDE and is now in the left navigation pane.
    - Extract the file by running the following command:
      `unzip code.zip`

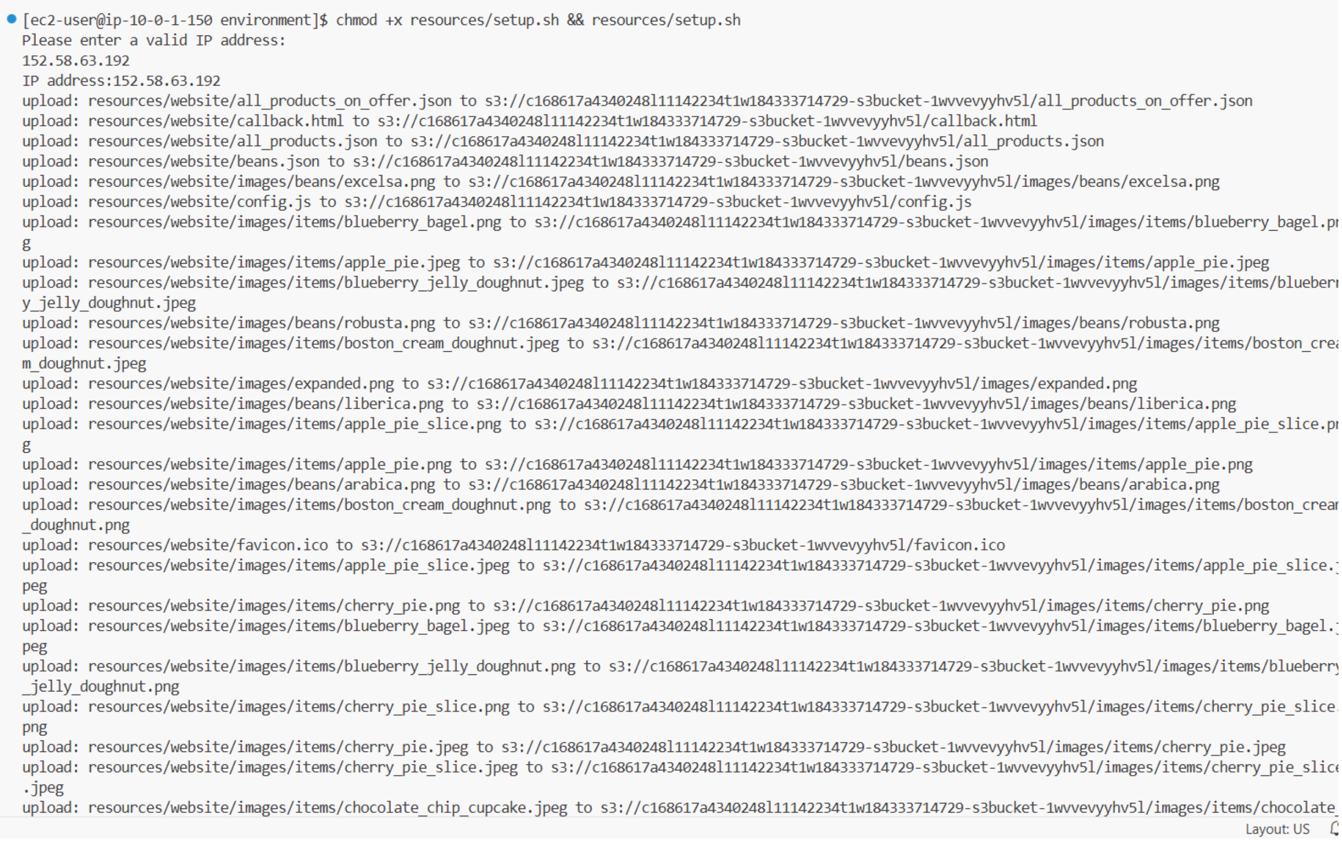


```
2025-08-10 11:15:30 (9.97 MB/s) - '/home/ec2-user/environment/code.zip' saved [5827466/5827466]

[ec2-user@ip-10-0-1-125 environment]$ unzip code.zip
Archive:  code.zip
  extracting: python_3/create_report_code.py
  extracting: python_3/get_all_products_code.py
  extracting: python_3/get_all_products_wrapper.py
  extracting: python_3/create_report_wrapper.py
  extracting: python_3/update_config.py
  extracting: resources/seed.py
  extracting: resources/setup.sh
  extracting: resources/public_policy.json
  extracting: resources/permissions.py
  extracting: resources/website/callback.html
  extracting: resources/website/all_products.json
  extracting: resources/website/all_products_on_offer.json
```

7. Run a script that upgrades the version of the AWS CLI installed on the VS Code IDE.
   - To set permissions on the script and then run it, run the following commands in the Bash terminal:

```
chmod +x ./resources/setup.sh && ./resources/setup.sh
```

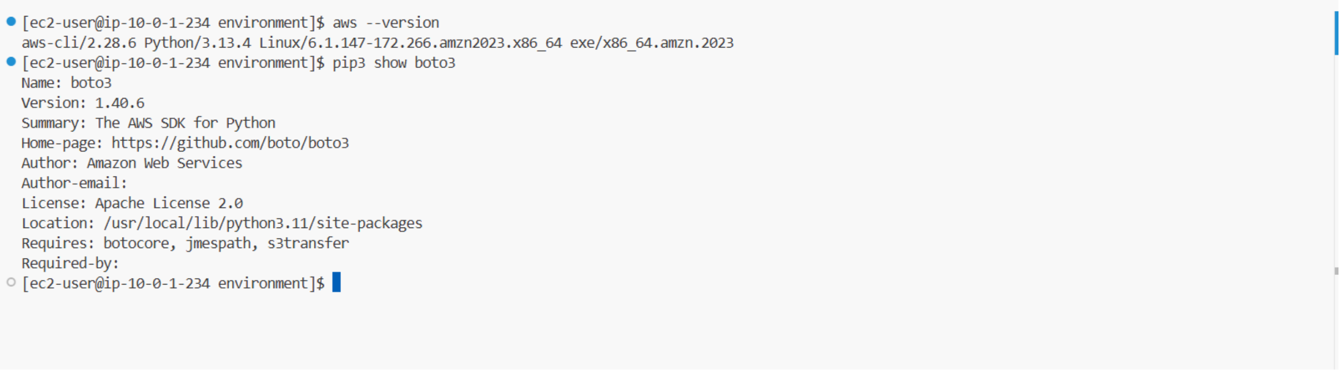
The script will prompt you for the **IP address** by which your computer is known to the internet. Use www.whatismyip.com to discover this address and then paste the IPv4 address into the command prompt and finish running the script.



```
● [ec2-user@ip-10-0-1-150 environment]$ chmod +x resources/setup.sh && resources/setup.sh
  Please enter a valid IP address:
  152.58.63.192
  IP address:152.58.63.192
  upload: resources/website/all_products_on_offer.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/all_products_on_offer.json
  upload: resources/website/callback.html to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/callback.html
  upload: resources/website/all_products.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/all_products.json
  upload: resources/website/beans.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/beans.json
  upload: resources/website/images/beans/excelsa.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/excelsa.png
  upload: resources/website/config.js to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/config.js
  upload: resources/website/images/items/blueberry_bagel.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry_bagel.pn
  g
  upload: resources/website/images/items/apple_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie.jpeg
  upload: resources/website/images/items/blueberry_jelly_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberr
  y_jelly_doughnut.jpeg
  upload: resources/website/images/beans/robusta.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/robusta.png
  upload: resources/website/images/items/boston_cream_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/boston_crea
  m_doughnut.jpeg
  upload: resources/website/images/expanded.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/expanded.png
  upload: resources/website/images/beans/liberica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/liberica.png
  upload: resources/website/images/items/apple_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie_slice.pn
  g
  upload: resources/website/images/items/apple_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie.png
  upload: resources/website/images/beans/arabica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/beans/arabica.png
  upload: resources/website/images/items/boston_cream_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/boston_crean
  _doughnut.png
  upload: resources/website/favicon.ico to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/favicon.ico
  upload: resources/website/images/items/apple_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/apple_pie_slice.j
  peg
  upload: resources/website/images/items/cherry_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie.png
  upload: resources/website/images/items/blueberry_bagel.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry_bagel.j
  peg
  upload: resources/website/images/items/blueberry_jelly_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/blueberry
  _jelly_doughnut.png
  upload: resources/website/images/items/cherry_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie_slice.
  png
  upload: resources/website/images/items/cherry_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie.jpeg
  upload: resources/website/images/items/cherry_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/cherry_pie_slice
  .jpeg
  upload: resources/website/images/items/chocolate_chip_cupcake.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvevyyhv5l/images/items/chocolate_
```

8. Verify the AWS CLI version and also verify that the SDK for Python is installed.
   - Confirm that the AWS CLI is now at version 2 by running the **`aws --version`** command.
   - In the VS Code Bash terminal (at the bottom of the IDE), run the following command:

```
pip3 show boto3
```



```
● [ec2-user@ip-10-0-1-234 environment]$ aws --version
  aws-cli/2.28.6 Python/3.13.4 Linux/6.1.147-172.266.amzn2023.x86_64 exe/x86_64.amzn.2023
● [ec2-user@ip-10-0-1-234 environment]$ pip3 show boto3
  Name: boto3
  Version: 1.40.6
  Summary: The AWS SDK for Python
  Home-page: https://github.com/boto/boto3
  Author: Amazon Web Services
  Author-email:
  License: Apache License 2.0
  Location: /usr/local/lib/python3.11/site-packages
  Requires: botocore, jmespath, s3transfer
  Required-by:
○ [ec2-user@ip-10-0-1-234 environment]$
```

Task 2: Configuring the Amazon SQS dead-letter queue

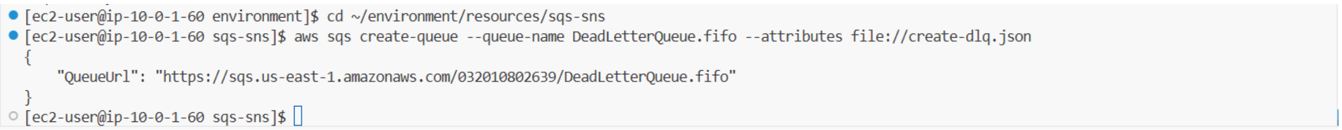
9. Create an Amazon SQS dead-letter queue.

- In the VS Code IDE bash terminal, change to the *sns-sqs* directory:

  `cd ~/environment/resources/sqs-sns`

- Create a dead-letter queue named *DeadLetterQueue.fifo*:

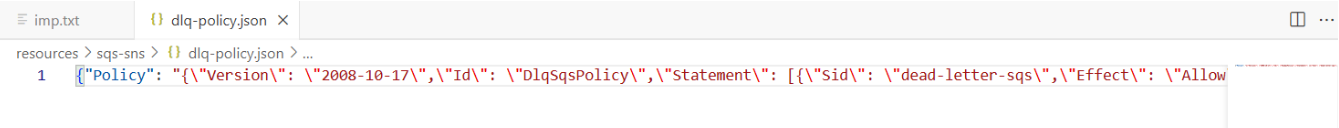
  `aws sqs create-queue --queue-name DeadLetterQueue.fifo --attributes file://create-dlq.json`



```
● [ec2-user@ip-10-0-1-60 environment]$ cd ~/environment/resources/sqs-sns
● [ec2-user@ip-10-0-1-60 sqs-sns]$ aws sqs create-queue --queue-name DeadLetterQueue.fifo --attributes file://create-dlq.json
  {
      "QueueUrl": "https://sqs.us-east-1.amazonaws.com/032010802639/DeadLetterQueue.fifo"
  }
○ [ec2-user@ip-10-0-1-60 sqs-sns]$
```

10. Update the policy statement with your AWS account ID.

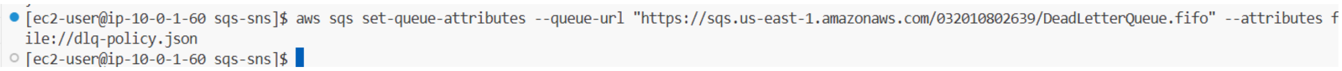
- In the environment window, under *resources/sqs-sns*, open the *dlq-policy.json* file.
- The *Policy* attribute's value is the same as the policy that you just reviewed. However, to work with the AWS Command Line Interface (AWS CLI), the entire policy must be formatted on a single line. In addition, the double quotes (") that are part of the policy definition must be escaped with a backslash (\) for the AWS CLI to correctly interpret the policy definition.
- In the *dlq-policy.json* file, replace the two *<FMI_1>* placeholders with your AWS account ID.



```
{"Policy": "{\"Version\": \"2008-10-17\",\"Id\": \"DlqSqsPolicy\",\"Statement\": [{\"Sid\": \"dead-letter-sqs\",\"Effect\": \"Allow
```

11. In the terminal window, run the following command to update the dead-letter queue's access policy. Replace the <FMI_1> placeholder with the DeadLetterQueue QueueURL from your text file:

    `aws sqs set-queue-attributes --queue-url "<FMI_1>" --attributes file://dlq-policy.json`



```
● [ec2-user@ip-10-0-1-60 sqs-sns]$ aws sqs set-queue-attributes --queue-url "https://sqs.us-east-1.amazonaws.com/032010802639/DeadLetterQueue.fifo" --attributes f
  ile://dlq-policy.json
○ [ec2-user@ip-10-0-1-60 sqs-sns]$ 
```

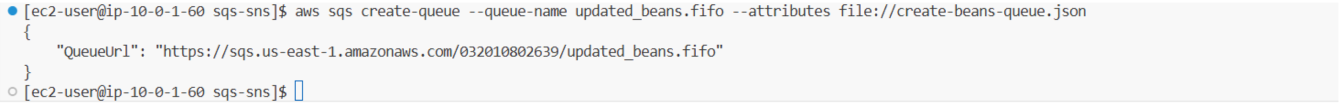
Task 3: Configuring the SQS queue

12. Review the attributes that will define your SQS queue.

- In the environment window, under *resources/sqs-sns*, open the *create-beans-queue.json* file.
- In the *create-beans-queue.json* file, replace the <*FMI_1*> placeholder with your AWS account ID.

13. In the terminal window, run the following command to create a queue named *updated_beans.fifo*:

```
aws sqs create-queue --queue-name updated_beans.fifo --attributes
file://create-beans-queue.json
```



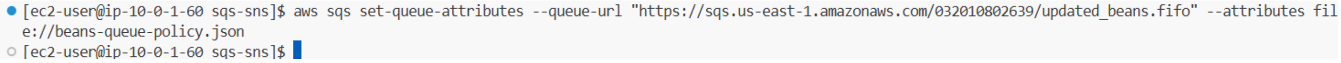
```
● [ec2-user@ip-10-0-1-60 sqs-sns]$ aws sqs create-queue --queue-name updated_beans.fifo --attributes file://create-beans-queue.json
  {
      "QueueUrl": "https://sqs.us-east-1.amazonaws.com/032010802639/updated_beans.fifo"
  }
○ [ec2-user@ip-10-0-1-60 sqs-sns]$ 
```

14. Update the policy statement with your AWS account ID.

- In the environment window, under *resources/sqs-sns*, open the *beans-queue-policy.json* file.
- Replace all eight <*FMI_1*> placeholders with your AWS account ID.

15. In the terminal window, run the following command to update the updated_beans.fifo queue's policy. Replace the <FMI_1> placeholder with the updated_beans.fifo QueueURL that you saved earlier:

```
aws sqs set-queue-attributes --queue-url "<FMI_1>" --attributes
file://beans-queue-policy.json
```

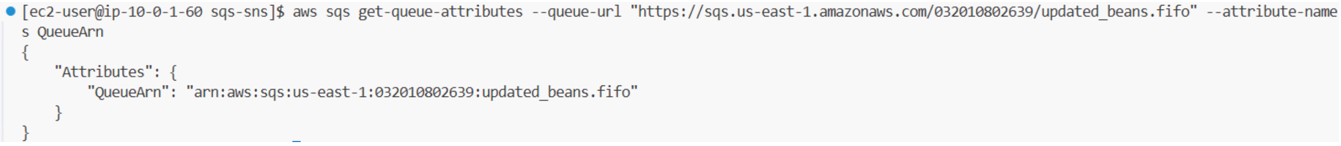


```
● [ec2-user@ip-10-0-1-60 sqs-sns]$ aws sqs set-queue-attributes --queue-url "https://sqs.us-east-1.amazonaws.com/032010802639/updated_beans.fifo" --attributes fil
  e://beans-queue-policy.json
○ [ec2-user@ip-10-0-1-60 sqs-sns]$ 
```

16. Retrieve the ARN for the *updated_beans.fifo* queue to use later.

- Run the following command. Replace the <*FMI_1*> placeholder with your AWS account ID.

```
aws sqs get-queue-attributes --queue-url "https://sqs.us-east-
1.amazonaws.com/<FMI_1>/updated_beans.fifo" --attribute-names QueueArn
```

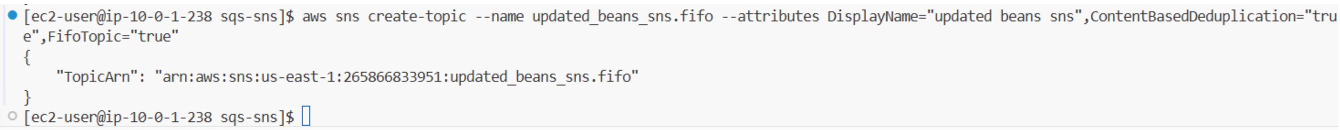


```
● [ec2-user@ip-10-0-1-60 sqs-sns]$ aws sqs get-queue-attributes --queue-url "https://sqs.us-east-1.amazonaws.com/032010802639/updated_beans.fifo" --attribute-name
  s QueueArn
  {
      "Attributes": {
          "QueueArn": "arn:aws:sqs:us-east-1:032010802639:updated_beans.fifo"
      }
  }
```

Task 4: Configuring the SNS topic

17. To create the SNS topic, run the following command:

```
aws sns create-topic --name updated_beans_sns.fifo --attributes
DisplayName="updated beans
sns",ContentBasedDeduplication="true",FifoTopic="true"
```



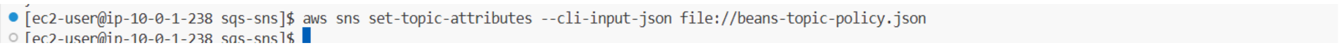
```
● [ec2-user@ip-10-0-1-238 sqs-sns]$ aws sns create-topic --name updated_beans_sns.fifo --attributes DisplayName="updated beans sns",ContentBasedDeduplication="tru
  e",FifoTopic="true"
  {
      "TopicArn": "arn:aws:sns:us-east-1:265866833951:updated_beans_sns.fifo"
  }
○ [ec2-user@ip-10-0-1-238 sqs-sns]$ 
```

18. Update the policy statement with your AWS account ID.

- In the environment window, under *resources/sqs-sns*, open the *beans-topic-policy.json* file.
- Update the placeholders:
    - Replace the *<FMI_1>* placeholder with the *updated_beans_sns.fifo TopicArn* that you saved earlier.
    - Replace the five *<FMI_2>* placeholders with your AWS account ID.

19. To apply the policy to your queue, run the following command.

```
aws sns set-topic-attributes --cli-input-json file://beans-topic-policy.json
```



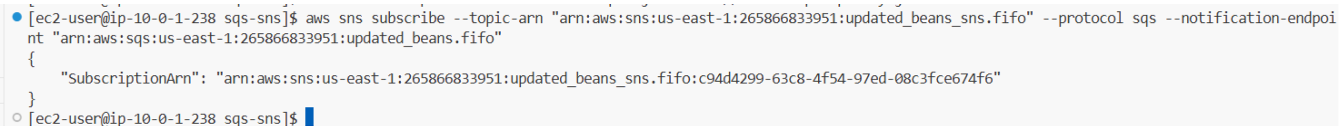
```
● [ec2-user@ip-10-0-1-238 sqs-sns]$ aws sns set-topic-attributes --cli-input-json file://beans-topic-policy.json
○ [ec2-user@ip-10-0-1-238 sqs-sns]$ 
```

Task 5: Linking Amazon SQS and Amazon SNS

20. Create a subscription from the SQS queue to the SNS topic.

- Update the following command:
    - Replace the *<FMI_1>* placeholder with the *update_beans_sns.fifo TopicArn*.
    - Replace the *<FMI_2>* placeholder with the *updated_beans.fifo QueueArn*.

```
aws sns subscribe --topic-arn "<FMI_1>" --protocol sqs --notification-endpoint "<FMI_2>"
```



```
● [ec2-user@ip-10-0-1-238 sqs-sns]$ aws sns subscribe --topic-arn "arn:aws:sns:us-east-1:265866833951:updated_beans_sns.fifo" --protocol sqs --notification-endpoi
nt "arn:aws:sqs:us-east-1:265866833951:updated_beans.fifo"
{
    "SubscriptionArn": "arn:aws:sns:us-east-1:265866833951:updated_beans_sns.fifo:c94d4299-63c8-4f54-97ed-08c3fce674f6"
}
○ [ec2-user@ip-10-0-1-238 sqs-sns]$ 
```

Task 6: Testing message publishing

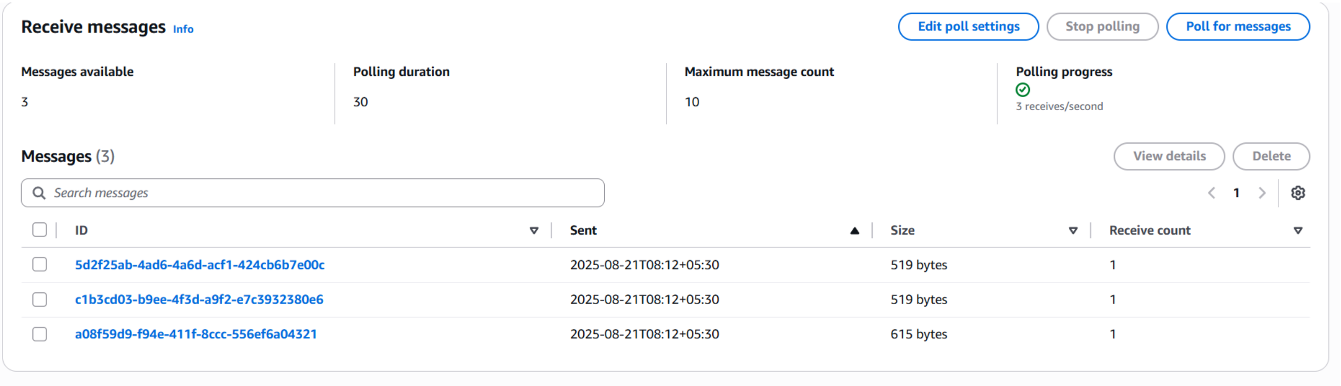
21. In the *send_beans_update.py* file, replace the <*FMI_1*> placeholder with your AWS account ID to update the *sns_topic* value.

22. In the terminal, use the following commands to run the Python publisher:

```
cd ~/environment/resources/sqs-sns
python3 send_beans_update.py beans_update_1.txt
```

23. Navigate to the Amazon SQS console.

- In the AWS Management Console, search for and select Simple Queue Service
- The *DeadLetterQueue.fifo* and *updated_beans.fifo* queues are listed.
- Choose the link for the *updated_beans.fifo* queue and choose **Send and receive messages**
- Poll for new messages:In the bottom pane, choose **Poll for messages**.



Receive messages Info

Messages available: 3

Polling duration: 30

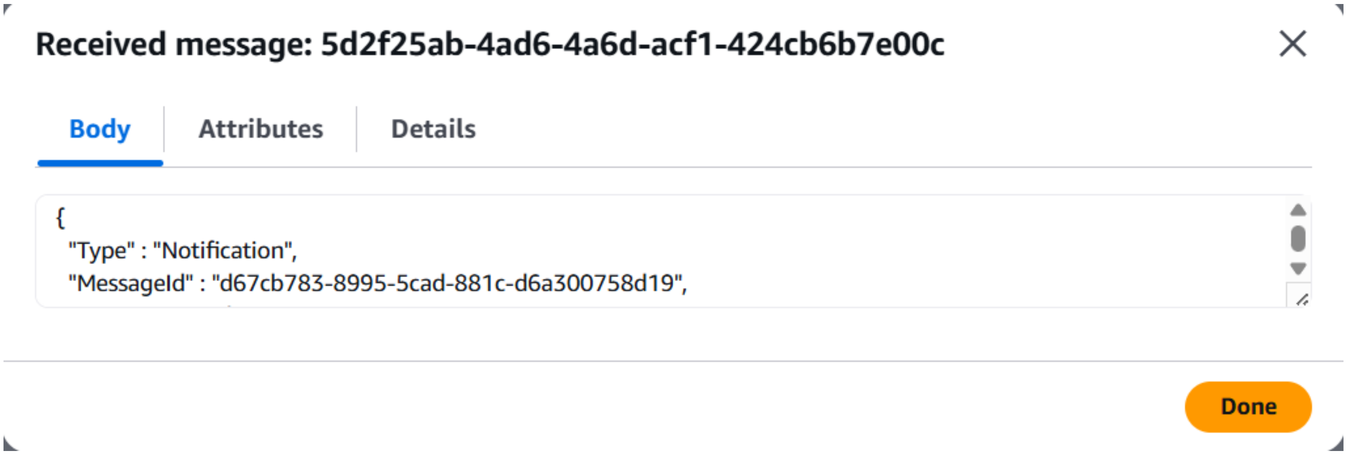
Maximum message count: 10

Polling progress: 3 receives/second

Messages (3)

ID	Sent	Size	Receive count
5d2f25ab-4ad6-4a6d-acf1-424cb6b7e00c	2025-08-21T08:12+05:30	519 bytes	1
c1b3cd03-b9ee-4f3d-a9f2-e7c3932380e6	2025-08-21T08:12+05:30	519 bytes	1
a08f59d9-f94e-411f-8ccc-556ef6a04321	2025-08-21T08:12+05:30	615 bytes	1

- Locate the message with the largest **Size**.
- To examine the message details, choose the **ID** hyperlink.
- Review the message body.

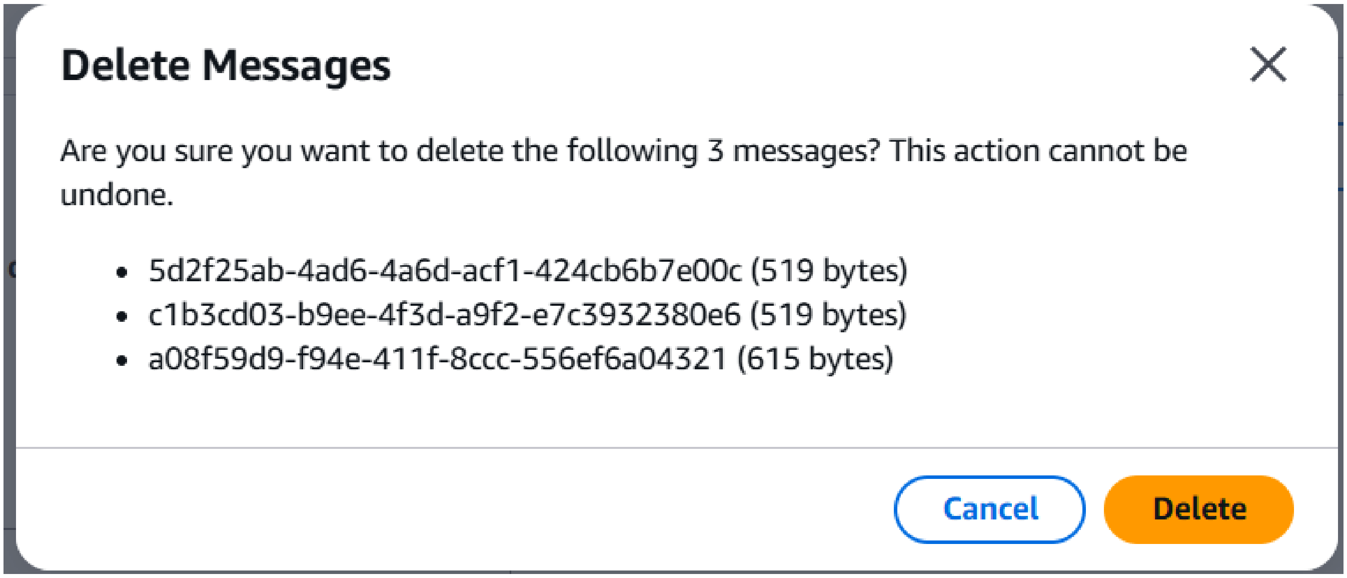


Received message: 5d2f25ab-4ad6-4a6d-acf1-424cb6b7e00c

```
{
 "Type" : "Notification",
 "MessageId" : "d67cb783-8995-5cad-881c-d6a300758d19",
```

24. Delete messages from the queue.

- Select all messages in the queue.
- Choose **Delete**, and choose **Delete** again when you are prompted to confirm this action.



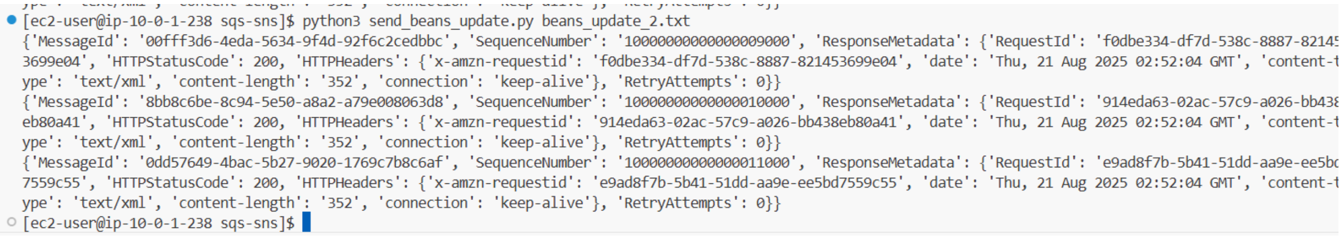
Delete Messages

Are you sure you want to delete the following 3 messages? This action cannot be undone.

- 5d2f25ab-4ad6-4a6d-acf1-424cb6b7e00c (519 bytes)
- c1b3cd03-b9ee-4f3d-a9f2-e7c3932380e6 (519 bytes)
- a08f59d9-f94e-411f-8ccc-556ef6a04321 (615 bytes)

25. In the VS Code IDE bash terminal window, run the Python publisher again. This time, send a new set of records:

```
python3 send_beans_update.py beans_update_2.txt
```



```
ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
● [ec2-user@ip-10-0-1-238 sqs-sns]$ python3 send_beans_update.py beans_update_2.txt
  {'MessageId': '00fff3d6-4eda-5634-9f4d-92f6c2cedbbc', 'SequenceNumber': '10000000000000009000', 'ResponseMetadata': {'RequestId': 'f0dbe334-df7d-538c-8887-82145
  3699e04', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'f0dbe334-df7d-538c-8887-821453699e04', 'date': 'Thu, 21 Aug 2025 02:52:04 GMT', 'content-t
  ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
  {'MessageId': '8bb8c6be-8c94-5e50-a8a2-a79e008063d8', 'SequenceNumber': '10000000000000010000', 'ResponseMetadata': {'RequestId': '914eda63-02ac-57c9-a026-bb438
  eb80a41', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '914eda63-02ac-57c9-a026-bb438eb80a41', 'date': 'Thu, 21 Aug 2025 02:52:04 GMT', 'content-t
  ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
  {'MessageId': '0dd57649-4bac-5b27-9020-1769c7b8c6af', 'SequenceNumber': '10000000000000011000', 'ResponseMetadata': {'RequestId': 'e9ad8f7b-5b41-51dd-aa9e-ee5bc
  7559c55', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'e9ad8f7b-5b41-51dd-aa9e-ee5bd7559c55', 'date': 'Thu, 21 Aug 2025 02:52:04 GMT', 'content-t
  ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
○ [ec2-user@ip-10-0-1-238 sqs-sns]$
```

Task 7: Configuring the application to poll the queue

26. View the coffee bean inventory.

- Return to the browser tab that is open to the Amazon SQS console.
- Choose **Services**, and choose **Elastic Beanstalk**.
- Choose the hyperlink for **MyEnv**.
- To open the currently deployed coffee suppliers application, choose the URL directly under **MyEnv**.
- The coffee suppliers application opens in a new browser tab.
- Append /beans to the URL in your browser tab.
- The URL is similar to the following:

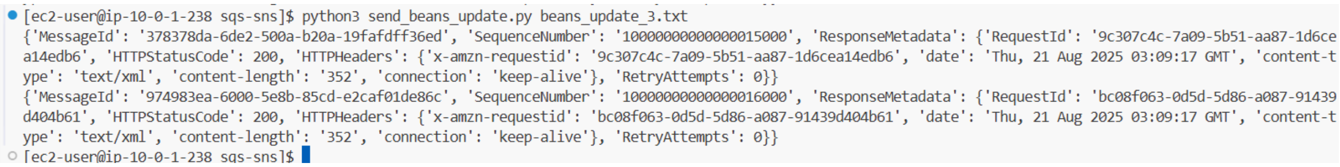
`https://myenv.eba-xxxxxxxx.us-east-1.elasticbeanstalk.com/beans`

27. Review the messages in the dead-letter queue.

- Return to the browser tab that is open to the Elastic Beanstalk console.
- Choose **Services**, and then choose **Simple Queue Service**.
- Choose the hyperlink for the **DeadLetterQueue.fifo** queue.
- Choose **Send and receive messages**.
- In the bottom pane, choose **Poll for messages**.
- Three messages are listed in the dead-letter queue.

28. Use the following command to run the Python publisher again. This time you will send a new set of records:

```
python3 send_beans_update.py beans_update_3.txt
```



```
● [ec2-user@ip-10-0-1-238 sqs-sns]$ python3 send_beans_update.py beans_update_3.txt
  {'MessageId': '378378da-6de2-500a-b20a-19fafdff36ed', 'SequenceNumber': '10000000000000015000', 'ResponseMetadata': {'RequestId': '9c307c4c-7a09-5b51-aa87-1d6ce
  a14edb6', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '9c307c4c-7a09-5b51-aa87-1d6cea14edb6', 'date': 'Thu, 21 Aug 2025 03:09:17 GMT', 'content-t
  ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
  {'MessageId': '974983ea-6000-5e8b-85cd-e2caf01de86c', 'SequenceNumber': '10000000000000016000', 'ResponseMetadata': {'RequestId': 'bc08f063-0d5d-5d86-a087-91439
  d404b61', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'bc08f063-0d5d-5d86-a087-91439d404b61', 'date': 'Thu, 21 Aug 2025 03:09:17 GMT', 'content-t
  ype': 'text/xml', 'content-length': '352', 'connection': 'keep-alive'}, 'RetryAttempts': 0}}
○ [ec2-user@ip-10-0-1-238 sqs-sns]$
```

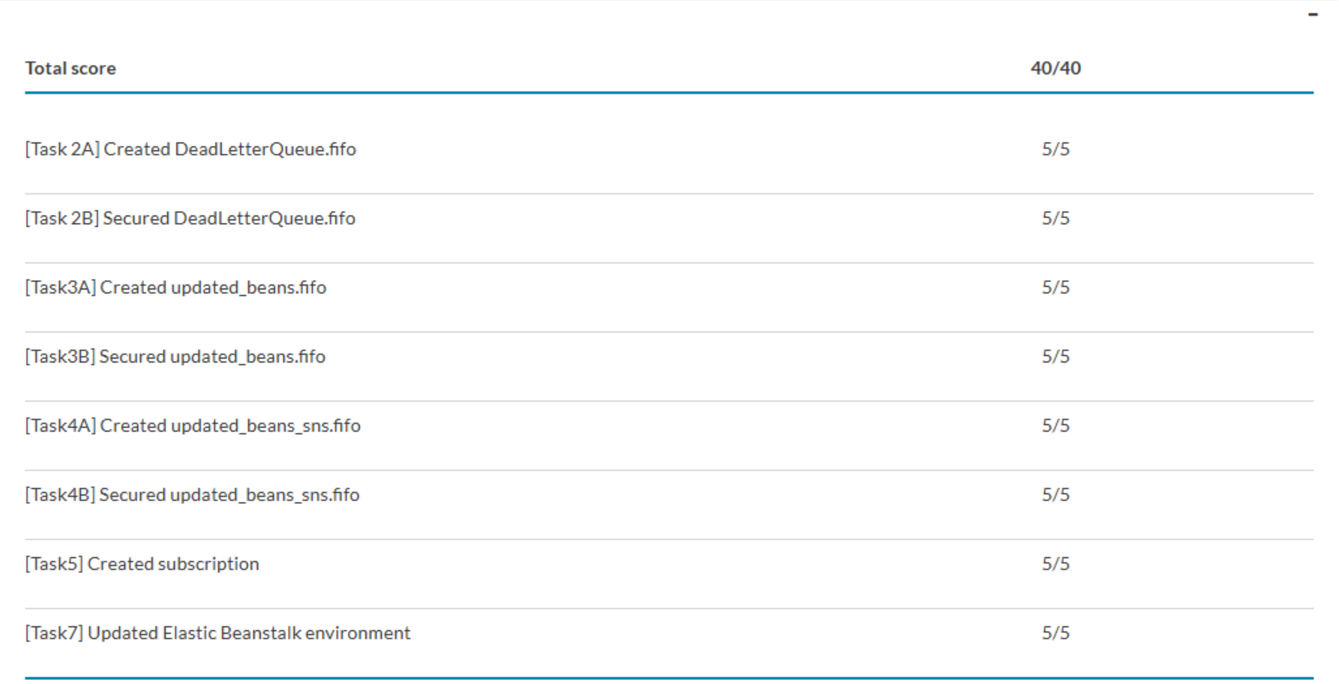
Conclusion:-

In this lab, I successfully implemented a messaging system using **Amazon SNS** and **Amazon SQS** to enable automated coffee inventory updates. The following steps, configurations, and commands were performed:

- Connected to **VS Code IDE** using provided credentials.
- Downloaded and extracted the lab resources:
  - `wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/10-lab-sqs/code.zip -P /home/ec2-user/environment`
  - `unzip code.zip`
- Updated AWS CLI and verified installation:
  - `chmod +x ./resources/setup.sh && ./resources/setup.sh`
  - `aws --version`
  - `pip3 show boto3`
  - **Created Dead Letter Queue (DLQ):**
  - `cd ~/environment/resources/sqs-sns`
  - `aws sqs create-queue --queue-name DeadLetterQueue.fifo --attributes file://create-dlq.json`
  - `aws sqs set-queue-attributes --queue-url "<DLQ_URL>" --attributes file://dlq-policy.json`
- **Created SQS queue (updated_beans.fifo)** and applied policy:
  - `aws sqs create-queue --queue-name updated_beans.fifo --attributes file://create-beans-queue.json`
  - `aws sqs set-queue-attributes --queue-url "<QUEUE_URL>" --attributes file://beans-queue-policy.json`
  - `aws sqs get-queue-attributes --queue-url "https://sqs.us-east-1.amazonaws.com/<AccountID>/updated_beans.fifo" --attribute-names QueueArn`
- **Created SNS topic** and applied topic policy:
  - `aws sns create-topic --name updated_beans_sns.fifo --attributes DisplayName="updated beans sns",ContentBasedDeduplication="true",FifoTopic="true"`
  - `aws sns set-topic-attributes --cli-input-json file://beans-topic-policy.json`
- **Linked SQS with SNS (subscription):**
  - `aws sns subscribe --topic-arn "<TopicArn>" --protocol sqs --notification-endpoint "<QueueArn>"`
- **Tested message publishing with Python publisher:**
  - `cd ~/environment/resources/sqs-sns`
  - `python3 send_beans_update.py beans_update_1.txt`
  - `python3 send_beans_update.py beans_update_2.txt`
  - `python3 send_beans_update.py beans_update_3.txt`
- Verified messages in **Amazon SQS Console** and reviewed processing with **Elastic Beanstalk application**.

- Checked **Dead Letter Queue** for failed messages.

Result :-



Total score	40/40
[Task 2A] Created DeadLetterQueue.fifo	5/5
[Task 2B] Secured DeadLetterQueue.fifo	5/5
[Task3A] Created updated_beans.fifo	5/5
[Task3B] Secured updated_beans.fifo	5/5
[Task4A] Created updated_beans_sns.fifo	5/5
[Task4B] Secured updated_beans_sns.fifo	5/5
[Task5] Created subscription	5/5
[Task7] Updated Elastic Beanstalk environment	5/5