CSCI 5410 – Assignment 5

Assignment 5 – Part B

Name: Benny Daniel Tharigopala
Banner ID: B00899629
GitLab URL: https://git.cs.dal.ca/benny/csci5410_B00899629_Benny_Tharigopala

AWS Lambda-SQS-SNS

Introduction

Problem Statement:
Part-B of Assignment 5 requires students to mimic a Car Rental Agency's notification system. It considers a typical scenario where customers request a specific vehicle, on a specific date, through the agency's online portal. In this scenario, once a customer submits a request to the agency, one of the personnel (Bob) who is responsible for observing the request queue, will prepare the paperwork relevant to the request. Subsequently, Bob will send a notification to Alice (another personnel responsible for delivering vehicles to customers).

Implementation Approach:

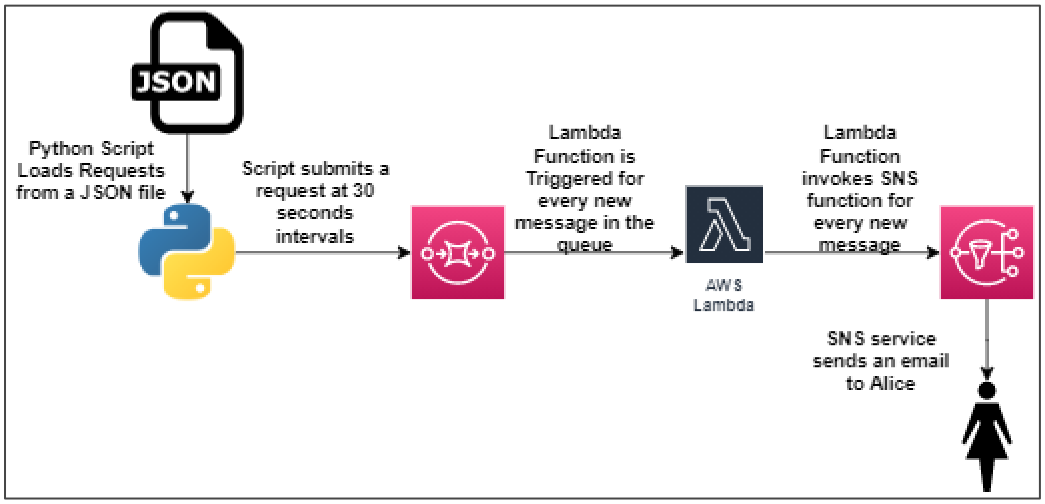
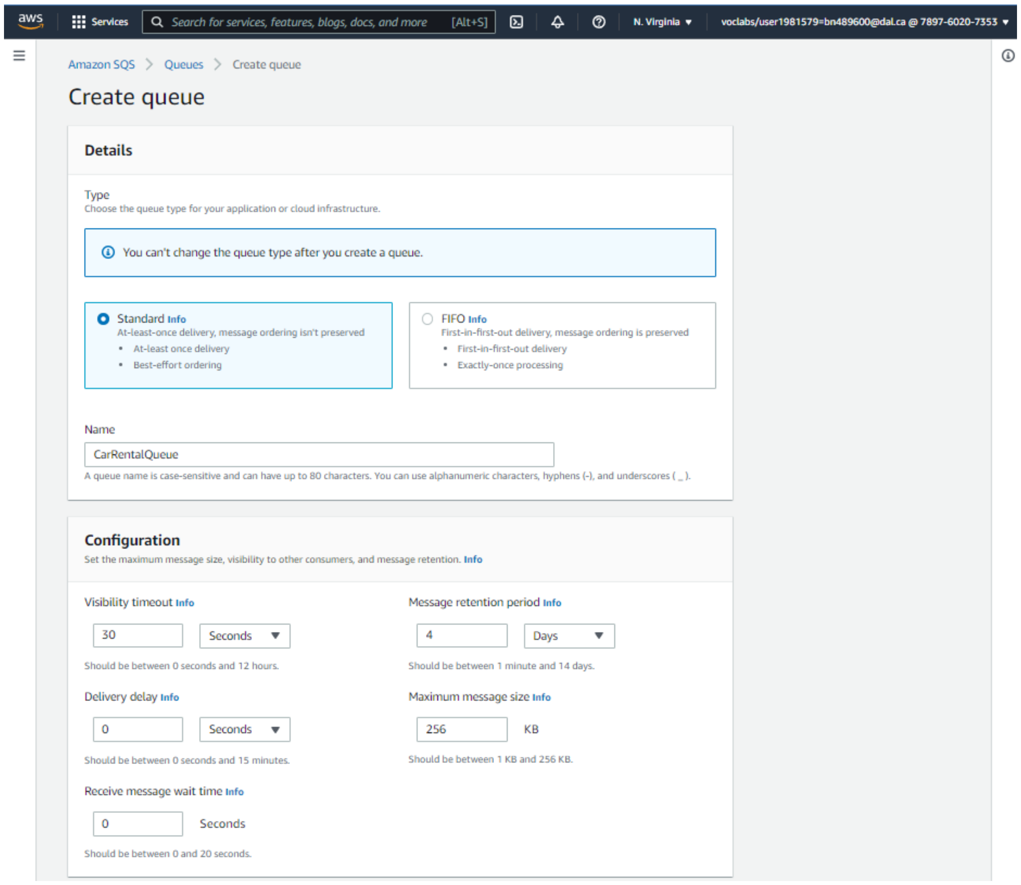

JSON
Python Script Loads Requests from a JSON file
Script submits a request at 30 seconds intervals
Lambda Function is Triggered for every new message in the queue
AWS Lambda
Lambda Function invokes SNS function for every new message
SNS service sends an email to Alice

Figure 1: Process Flow Diagram

A Python script ("main.py") loads request data from a JSON file. Subsequently, the script sends messages at 30 seconds intervals to Amazon's Simple Queue Service (SQS). Upon receiving a new message, a Lambda function, which is configured to invoke the Simple Notification Service (SNS) in AWS [1], triggers the infrastructure which then submits an email with the vehicle's name and date on which the vehicle is required, to Alice (my Dalhousie Email Id).

Creating a Standard Queue with AWS SQS


aws
Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1981579=bn489600@dal.ca @ 7897-6020-7353
Amazon SQS > Queues > Create queue

Create queue

Details

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

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

[✓ 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

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

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

Visibility timeout Info: 30 Seconds
Should be between 0 seconds and 12 hours.

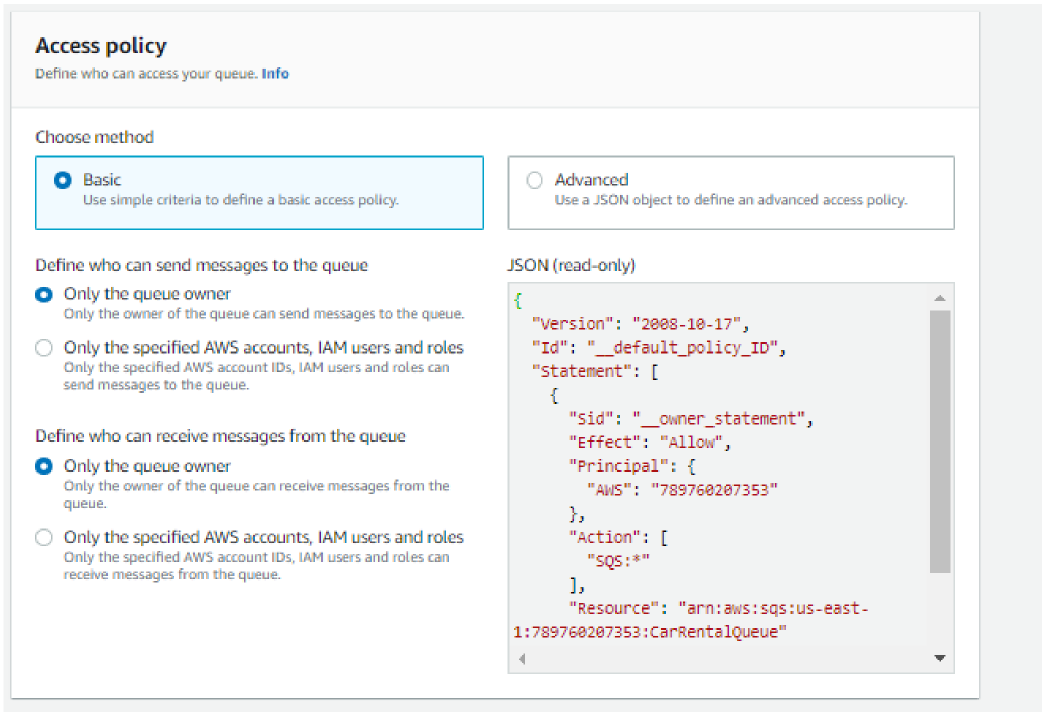
Message retention period Info: 4 Days
Should be between 1 minute and 14 days.

Delivery delay Info: 0 Seconds
Should be between 0 seconds and 15 minutes.

Maximum message size Info: 256 KB
Should be between 1 KB and 256 KB.

Receive message wait time Info: 0 Seconds
Should be between 0 and 20 seconds.

Figure 2: Standard Queue in AWS SQS



Access policy
Define who can access your queue. Info

Choose method

[✓ Basic]
Use simple criteria to define a basic access policy.

Advanced
Use a JSON object to define an advanced access policy.

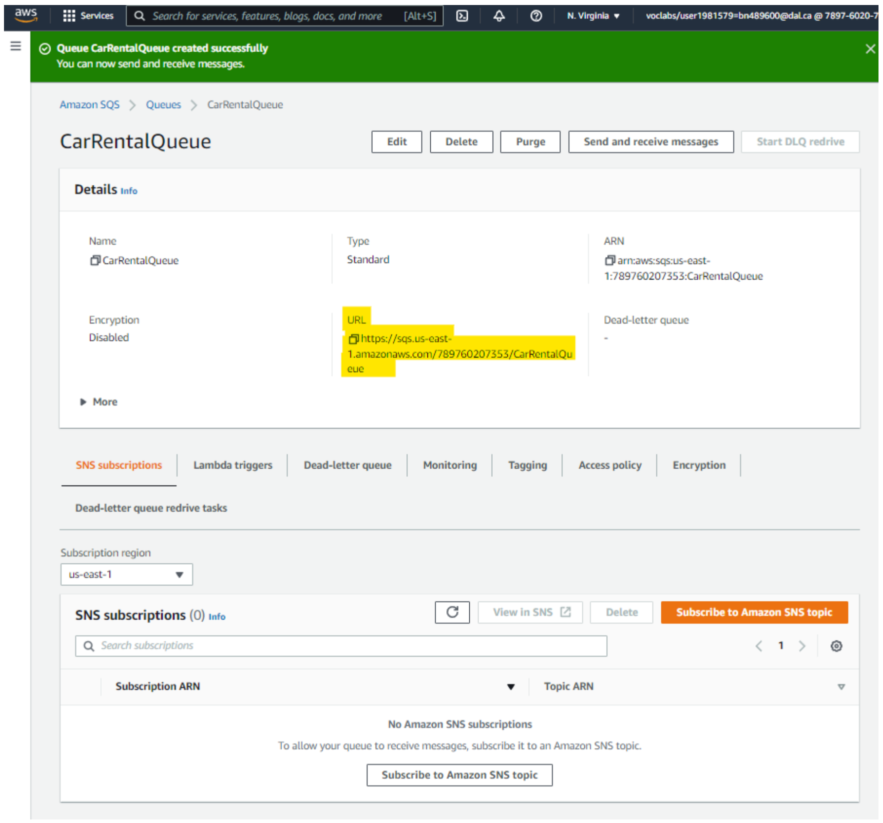
Define who can send messages to the queue
- [✓ Only the queue owner]
  Only the owner of the queue can send messages to the queue.
- Only the specified AWS accounts, IAM users and roles
  Only the specified AWS account IDs, IAM users and roles can send messages to the queue.

Define who can receive messages from the queue
- [✓ Only the queue owner]
  Only the owner of the queue can receive messages from the queue.
- Only the specified AWS accounts, IAM users and roles
  Only the specified AWS account IDs, IAM users and roles can receive messages from the queue.

JSON (read-only)

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__owner_statement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "789760207353"
      },
      "Action": [
        "SQS:*"
      ],
      "Resource": "arn:aws:sqs:us-east-
1:789760207353:CarRentalQueue"
```

Figure 3: Choosing the basic Access Policy



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

Amazon SQS > Queues > CarRentalQueue

CarRentalQueue

Details

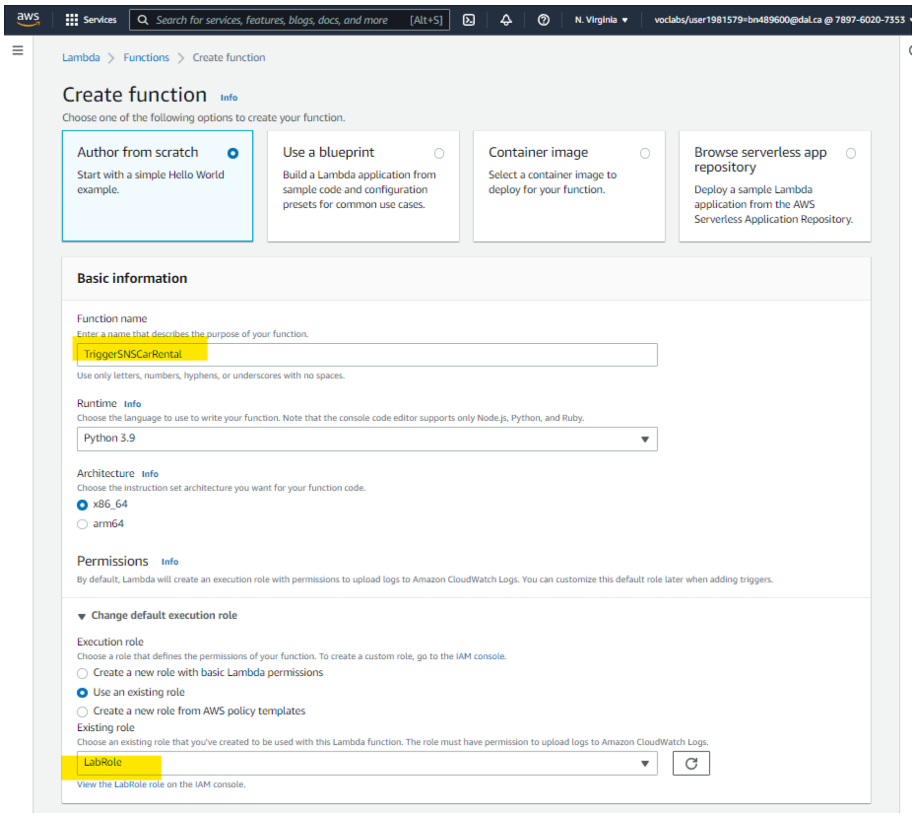
Name	Type	ARN
CarRentalQueue	Standard	arn:aws:sqs:us-east-1:789760207353:CarRentalQueue
Encryption	URL	Dead-letter queue
Disabled	https://sqs.us-east-1.amazonaws.com/789760207353/CarRentalQueue	-

SNS subscriptions (0)

No Amazon SNS subscriptions
To allow your queue to receive messages, subscribe it to an Amazon SNS topic.

Figure 4: Copy the URL to access the SQS service from the Python Script

Creating and Configuring a Lambda Function



Lambda > Functions > Create function

Create function

Choose one of the following options to create your function.

[✓ Author from scratch] Start with a simple Hello World example.
Use a blueprint: Build a Lambda application from sample code and configuration presets for common use cases.
Container image: Select a container image to deploy for your function.
Browse serverless app repository: Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
TriggerSNSCarRental
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.9

Architecture
Choose the instruction set architecture you want for your function code.
[✓ x86_64] arm64

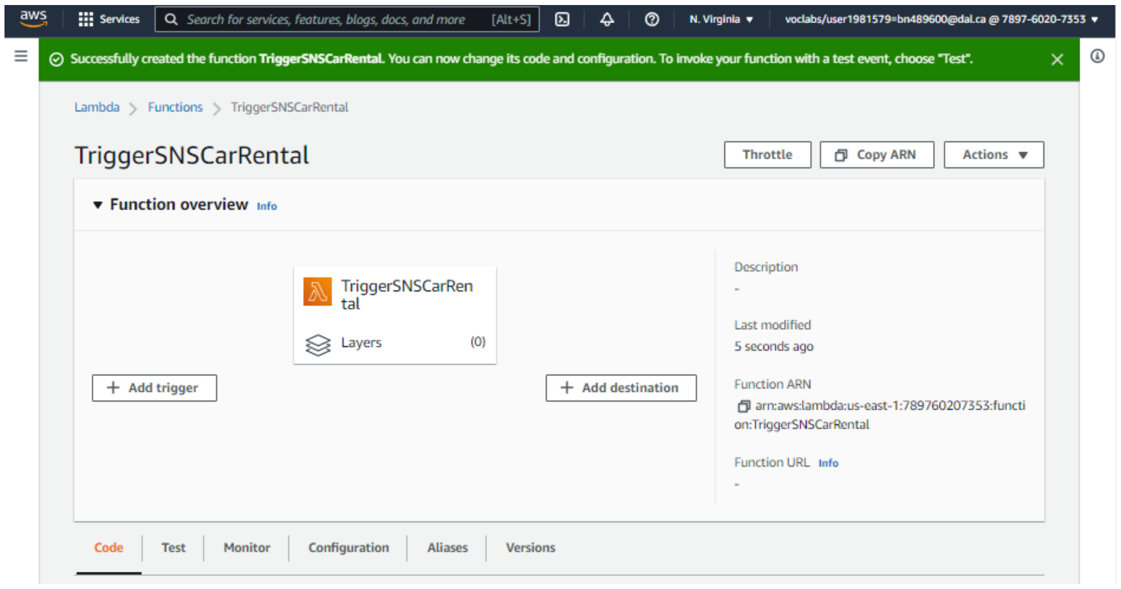
Permissions
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
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.
LabRole
View the LabRole role on the IAM console.

Figure 5: Configuring a Lambda Function


Successfully created the function TriggerSNSCarRental. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > TriggerSNSCarRental

TriggerSNSCarRental

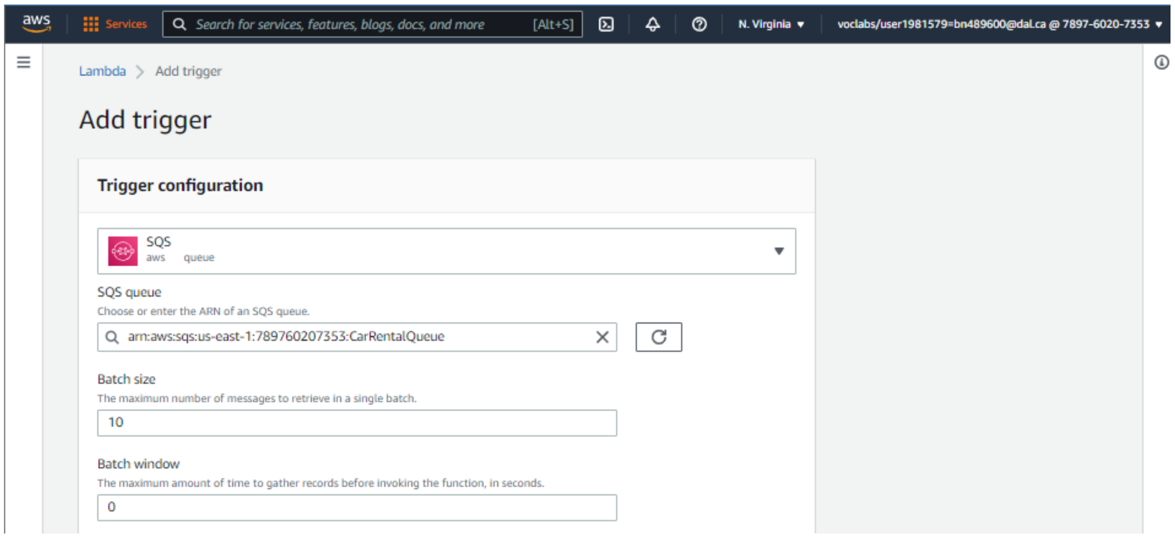
Description: -

Last modified: 5 seconds ago

Function ARN: arn:aws:lambda:us-east-1:789760207353:function:TriggerSNSCarRental

Function URL: -

Figure 6: Lambda Function is created successfully


Lambda > Add trigger

Add trigger

Trigger configuration

SQS

SQS queue: arn:aws:sqs:us-east-1:789760207353:CarRentalQueue

Batch size: 10

Batch window: 0

Figure 7: Add a Trigger to the Lambda Function

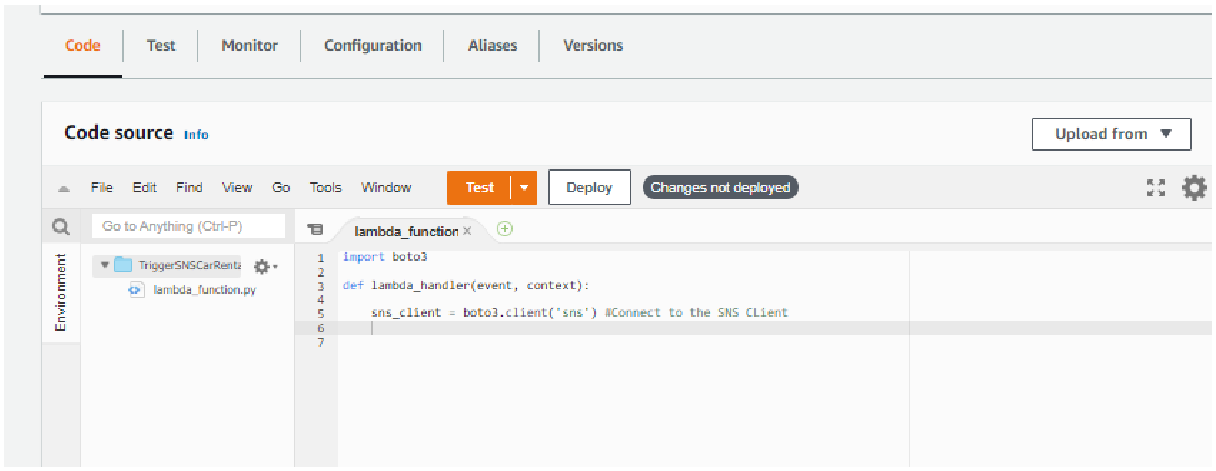
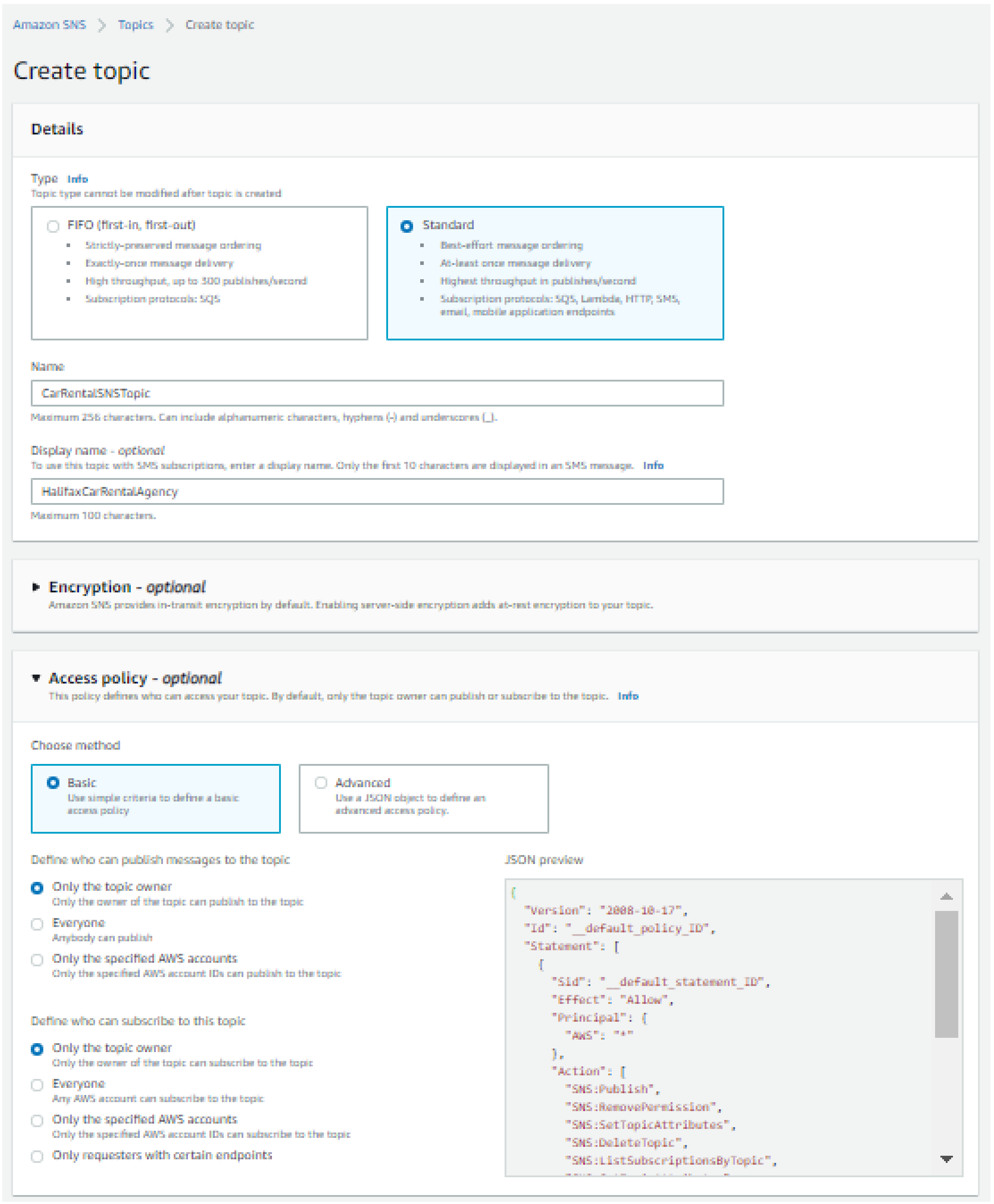

```
import boto3

def lambda_handler(event, context):

    sns_client = boto3.client('sns') #Connect to the SNS CLient
```

Figure 8: Establish a connection with AWS SNS through the SNS client

Creating and Configuring a SNS Topic



Amazon SNS > Topics > Create topic

Create topic

Details

Type Info
Topic type cannot be modified after topic is created

- ○ FIFO (first-in, first-out)
  - Strictly-preserved message ordering
  - Exactly-once message delivery
  - High throughput, up to 300 publishes/second
  - Subscription protocols: SQS
- ● Standard
  - Best-effort message ordering
  - At-least once message delivery
  - Highest throughput in publishes/second
  - Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name
CarRentalSNSTopic
Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_).

Display name - *optional*
To use this topic with SMS subscriptions, enter a display name. Only the first 10 characters are displayed in an SMS message. Info
HalifaxCarRentalAgency
Maximum 100 characters.

▸ Encryption - *optional*
Amazon SNS provides in-transit encryption by default. Enabling server-side encryption adds at-rest encryption to your topic.

▾ Access policy - *optional*
This policy defines who can access your topic. By default, only the topic owner can publish or subscribe to the topic. Info

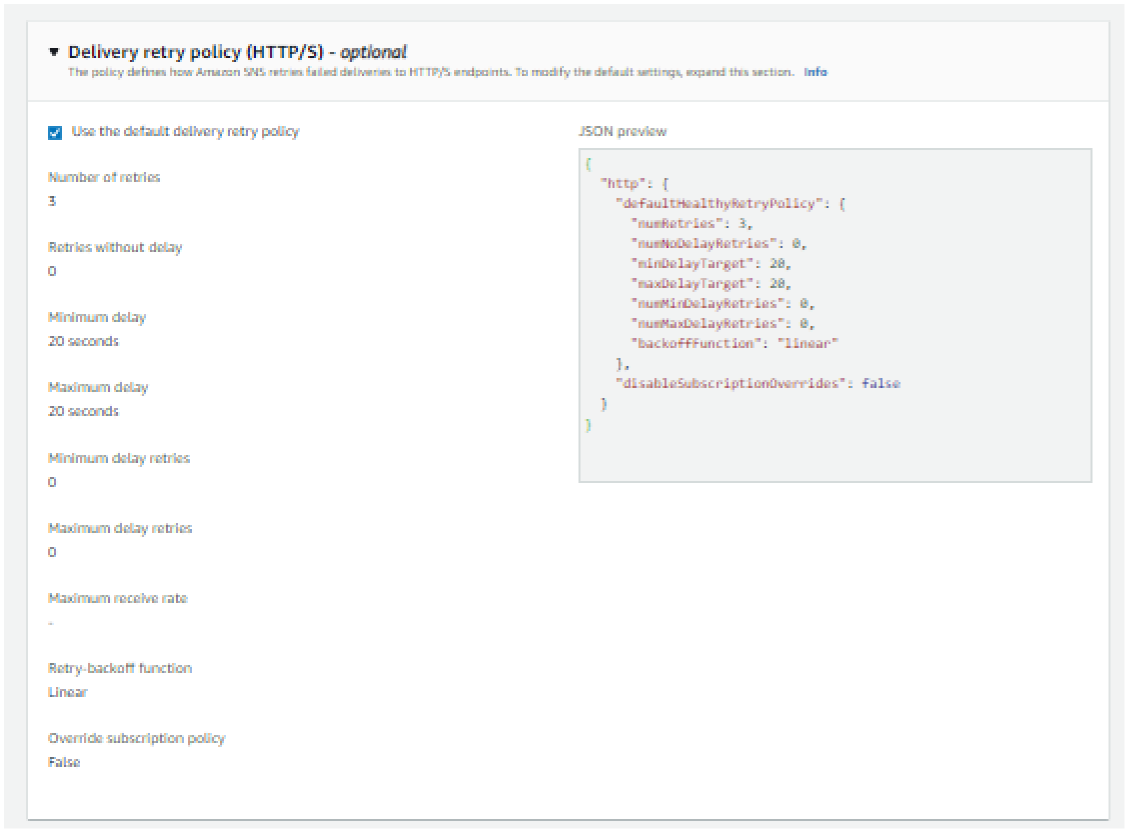
Choose method
- ● Basic – Use simple criteria to define a basic access policy
- ○ Advanced – Use a JSON object to define an advanced access policy.

Define who can publish messages to the topic
- ● Only the topic owner – Only the owner of the topic can publish to the topic
- ○ Everyone – Anybody can publish
- ○ Only the specified AWS accounts – Only the specified AWS account IDs can publish to the topic

Define who can subscribe to this topic
- ● Only the topic owner – Only the owner of the topic can subscribe to the topic
- ○ Everyone – Any AWS account can subscribe to the topic
- ○ Only the specified AWS accounts – Only the specified AWS account IDs can subscribe to the topic
- ○ Only requesters with certain endpoints

JSON preview
```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "SNS:Publish",
        "SNS:RemovePermission",
        "SNS:SetTopicAttributes",
        "SNS:DeleteTopic",
        "SNS:ListSubscriptionsByTopic",
```

Figure 9: Configure SNS Topic & Access Policy



Delivery retry policy (HTTP/S) - *optional*

The policy defines how Amazon SNS retries failed deliveries to HTTP/S endpoints. To modify the default settings, expand this section. Info

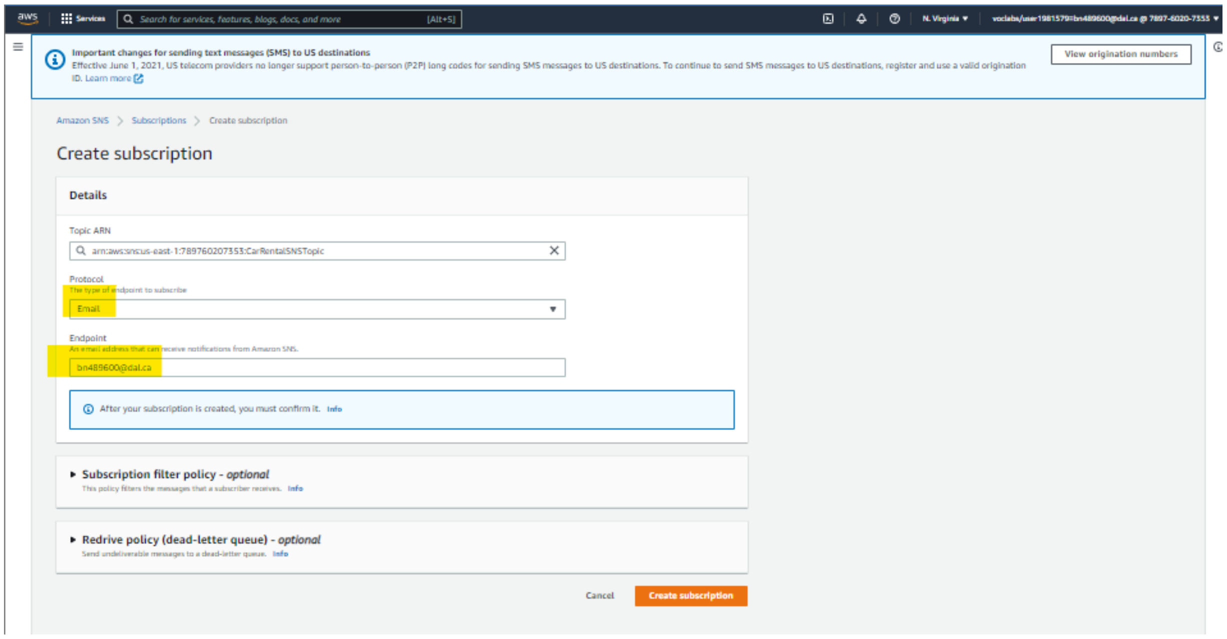
- [✓] Use the default delivery retry policy

Number of retries	3
Retries without delay	0
Minimum delay	20 seconds
Maximum delay	20 seconds
Minimum delay retries	0
Maximum delay retries	0
Maximum receive rate	-
Retry-backoff function	Linear
Override subscription policy	False

JSON preview

```
{
  "http": {
    "defaultHealthyRetryPolicy": {
      "numRetries": 3,
      "numNoDelayRetries": 0,
      "minDelayTarget": 20,
      "maxDelayTarget": 20,
      "numMinDelayRetries": 0,
      "numMaxDelayRetries": 0,
      "backoffFunction": "linear"
    },
    "disableSubscriptionOverrides": false
  }
}
```

Figure 10: Configure Retry Policy



Search for services, features, blogs, docs, and more [Alt+S] N. Virginia voclabs/user1981579=bn489600@dal.ca @ 7897-6020-7353

Important changes for sending text messages (SMS) to US destinations
Effective June 1, 2021, US telecom providers no longer support person-to-person (P2P) long codes for sending SMS messages to US destinations. To continue to send SMS messages to US destinations, register and use a valid origination ID. Learn more

Amazon SNS > Subscriptions > Create subscription

Create subscription

Details

Topic ARN: arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic

Protocol
The type of endpoint to subscribe
Email

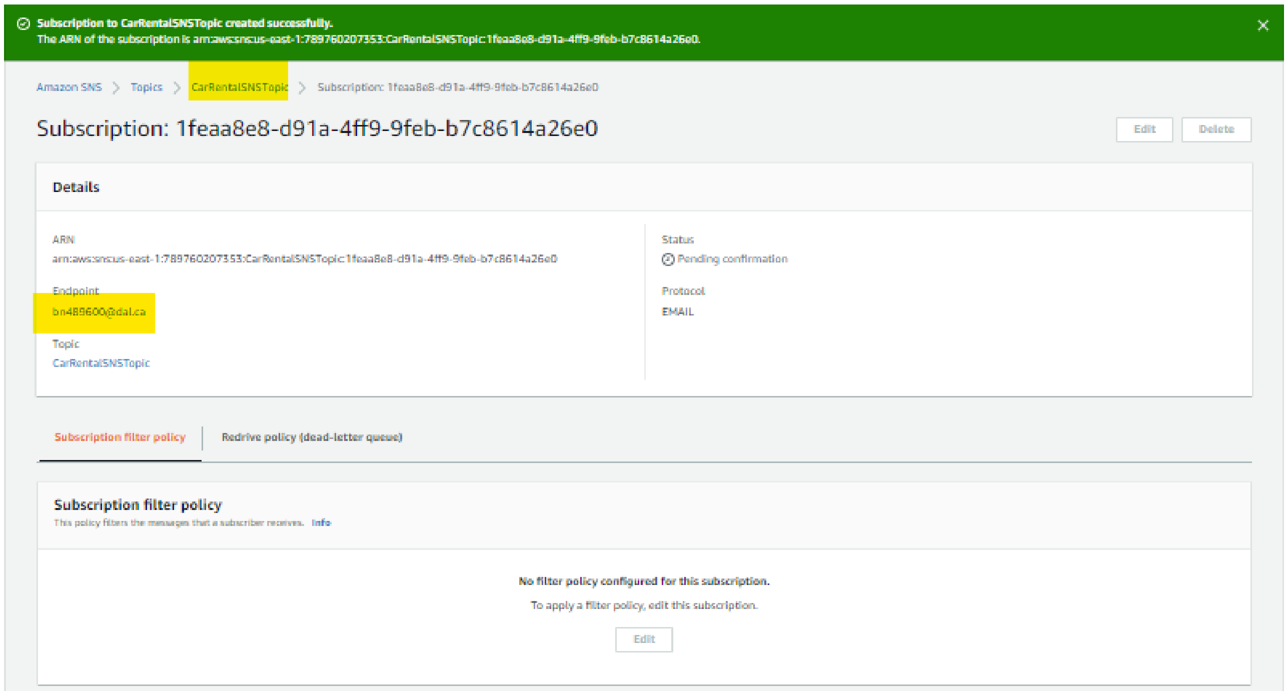
Endpoint
An email address that can receive notifications from Amazon SNS.
bn489600@dal.ca

After your subscription is created, you must confirm it. Info

Subscription filter policy - *optional*
This policy filters the messages that a subscriber receives. Info

Redrive policy (dead-letter queue) - *optional*
Send undeliverable messages to a dead-letter queue. Info

Figure 11: Creating a subscription for the notification



Subscription to CarRentalSNSTopic created successfully.
The ARN of the subscription is arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic:1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0.

Amazon SNS > Topics > CarRentalSNSTopic > Subscription: 1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0

Subscription: 1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0

Details

ARN
arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic:1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0

Endpoint
bn489600@dal.ca

Topic
CarRentalSNSTopic

Status
Pending confirmation

Protocol
EMAIL

Subscription filter policy

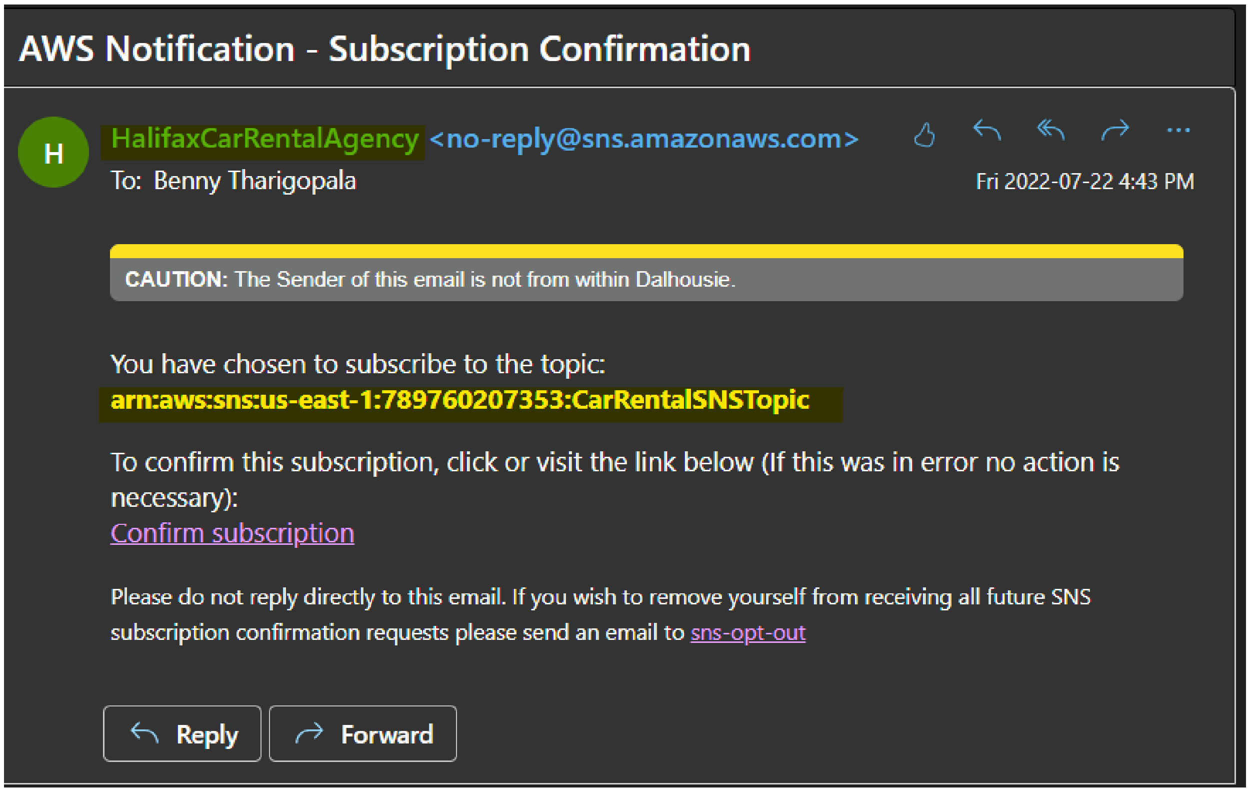
This policy filters the messages that a subscriber receives. Info

No filter policy configured for this subscription.

To apply a filter policy, edit this subscription.

Edit
Delete
Subscription filter policy
Redrive policy (dead-letter queue)
Edit

Figure 12: Subscription created Successfully for the Topic



AWS Notification - Subscription Confirmation

HalifaxCarRentalAgency <no-reply@sns.amazonaws.com>
To: Benny Tharigopala
Fri 2022-07-22 4:43 PM

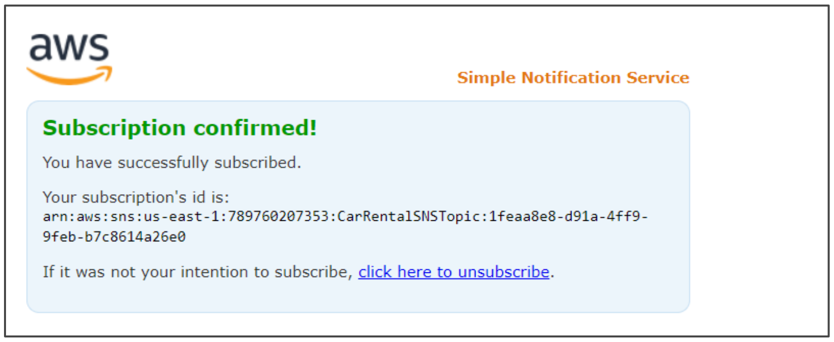
CAUTION: The Sender of this email is not from within Dalhousie.

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out

Figure 13: Subscription Confirmation Email



Simple Notification Service

Subscription confirmed!

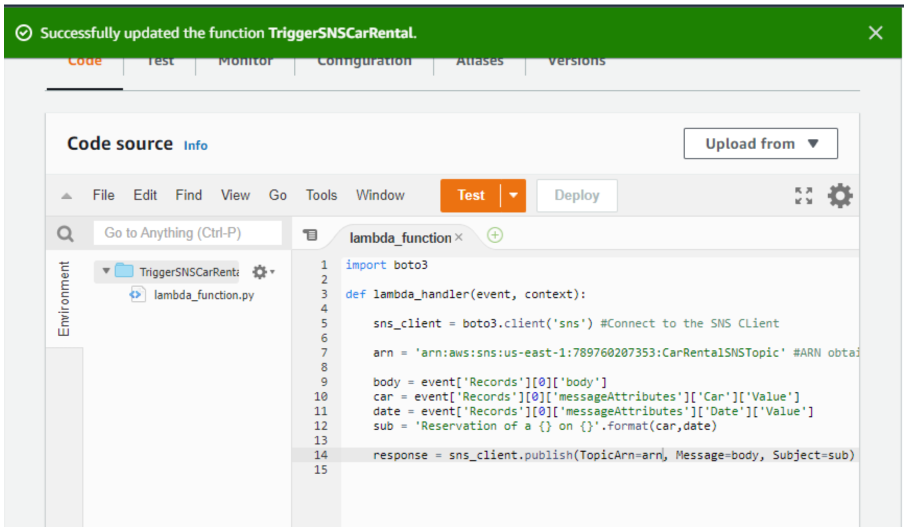
You have successfully subscribed.

Your subscription's id is:
arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic:1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0

If it was not your intention to subscribe, click here to unsubscribe.

Figure 14: Subscription Successfully Confirmed

Updating Lambda Function with the Amazon Resource Name (ARN) of the SNS Topic & SQS Message Attributes and Body

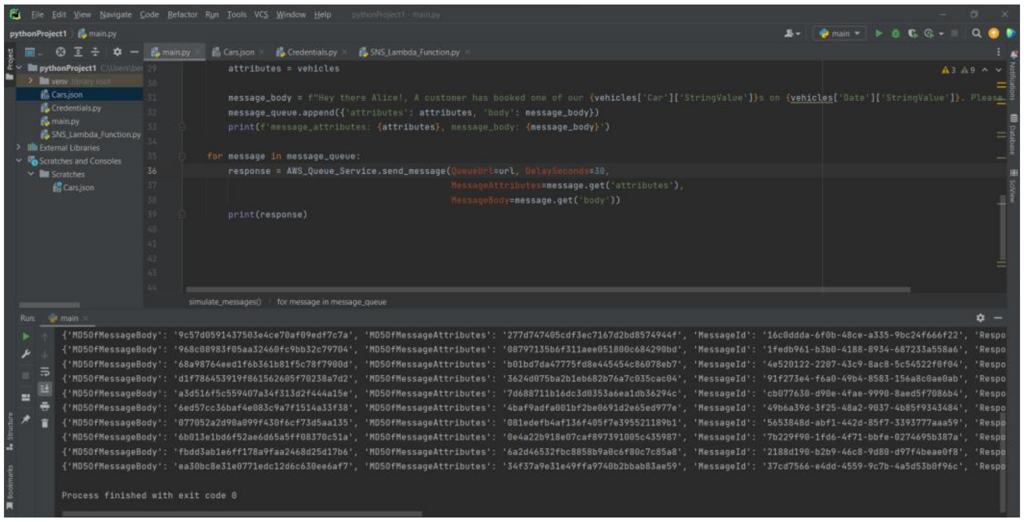


Successfully updated the function TriggerSNSCarRental.

```python
import boto3

def lambda_handler(event, context):

    sns_client = boto3.client('sns') #Connect to the SNS CLient

    arn = 'arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic' #ARN obtai

    body = event['Records'][0]['body']
    car = event['Records'][0]['messageAttributes']['Car']['Value']
    date = event['Records'][0]['messageAttributes']['Date']['Value']
    sub = 'Reservation of a {} on {}'.format(car,date)

    response = sns_client.publish(TopicArn=arn, Message=body, Subject=sub)
```

Figure 15: Update Lambda Function with Message Body and SNS ARN

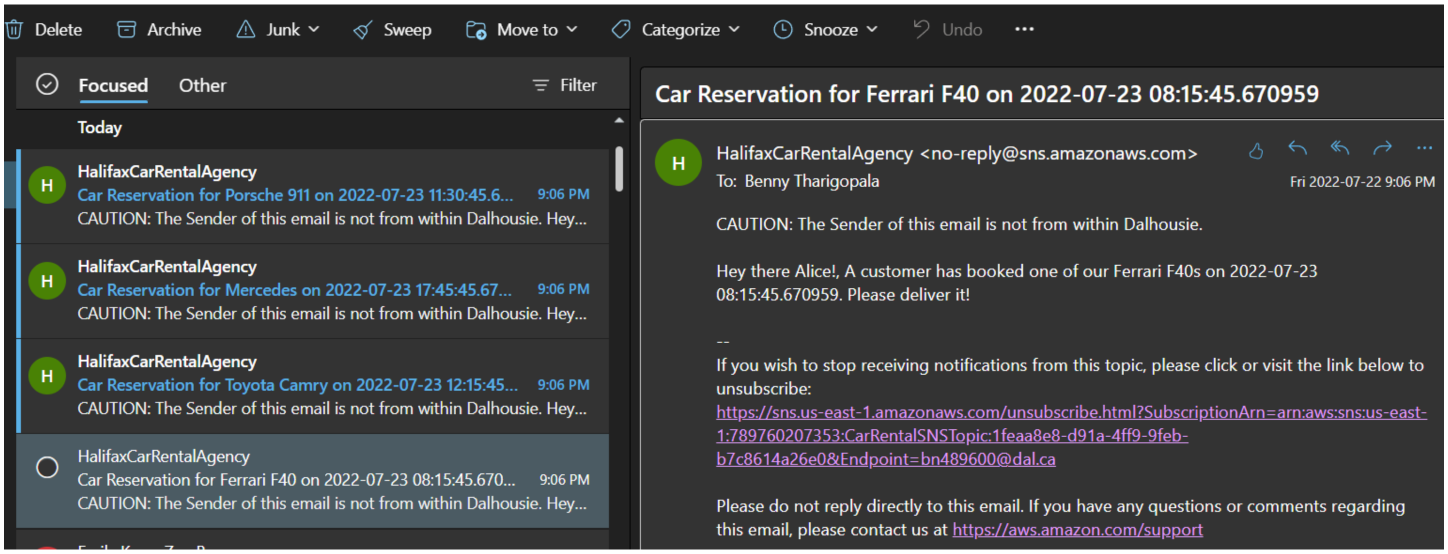
Executing the Script to submit Requests to the SQS Queue



```python
        attributes = vehicles

        message_body = f"Hey there Alice!, A customer has booked one of our {vehicles['Car']['StringValue']}s on {vehicles['Date']['StringValue']}. Please
        message_queue.append({'attributes': attributes, 'body': message_body})
        print(f'message_attributes: {attributes}, message_body: {message_body}')

    for message in message_queue:
        response = AWS_Queue_Service.send_message(QueueUrl=url, DelaySeconds=30,
                                                  MessageAttributes=message.get('attributes'),
                                                  MessageBody=message.get('body'))
        print(response)
```

```
{'MD5OfMessageBody': '9c57d0591437503e4ce70af09edf7c7a', 'MD5OfMessageAttributes': '277d747405cdf3ec7167d2bd8574944f', 'MessageId': '16c0ddda-6f0b-48ce-a335-9bc24f666f22', 'Respo
{'MD5OfMessageBody': '968c08983f05aa32460fc9bb32c79704', 'MD5OfMessageAttributes': '08797135b6f311aee051800c684290bd', 'MessageId': '1fedb961-b3b0-4188-8934-687233a558a6', 'Respo
{'MD5OfMessageBody': '68a98764eed1f6b361b81f5c78f7900d', 'MD5OfMessageAttributes': 'b01bd7da47775fd8e445454c86078eb7', 'MessageId': '4e520122-2207-43c9-8ac8-5c54522f0f04', 'Respo
{'MD5OfMessageBody': 'd1f786453919f861562605f70238a7d2', 'MD5OfMessageAttributes': '3624d075ba2b1eb682b76a7c035cac04', 'MessageId': '91f273e4-f6a0-49b4-8583-156a8c0ae0ab', 'Respo
{'MD5OfMessageBody': 'a3d516f5c559407a34f313d2f444a15e', 'MD5OfMessageAttributes': '7d688711b16dc3d0353a6ea1db36294c', 'MessageId': 'cb077630-d90e-4fae-9990-8aed5f7086b4', 'Respo
{'MD5OfMessageBody': '6ed57cc36baf4e083c9a7f1514a33f38', 'MD5OfMessageAttributes': '4baf9adfa001bf2be0691d2e65ed977e', 'MessageId': '49b6a39d-3f25-48a2-9037-4b85f9343484', 'Respo
{'MD5OfMessageBody': '077052a2d90a099f430f6cf73d5aa135', 'MD5OfMessageAttributes': '081edefb4af136f405f7e395521189b1', 'MessageId': '5653848d-abf1-442d-85f7-3393777aaa59', 'Respo
{'MD5OfMessageBody': '6b013e1bd6f52ae6d65a5ff08370c51a', 'MD5OfMessageAttributes': '0e4a22b918e07caf897391005c435987', 'MessageId': '7b229f90-1fd6-4f71-bbfe-0274695b387a', 'Respo
{'MD5OfMessageBody': 'fbdd3ab1e6ff178a9faa2468d25d17b6', 'MD5OfMessageAttributes': '6a2d46532fbc8858b9a0c6f80c7c85a8', 'MessageId': '2188d190-b2b9-46c8-9d80-d97f4beae0f8', 'Respo
{'MD5OfMessageBody': 'ea30bc8e31e0771edc12d6c630ee6af7', 'MD5OfMessageAttributes': '34f37a9e31e49ffa9740b2bbab83ae59', 'MessageId': '37cd7566-e4dd-4559-9c7b-4a5d53b0f96c', 'Respo

Process finished with exit code 0
```

Figure 16: Script Execution To Simulate Messages from Customers



HalifaxCarRentalAgency
Car Reservation for Porsche 911 on 2022-07-23 11:30:45.6... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Mercedes on 2022-07-23 17:45:45.67... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Toyota Camry on 2022-07-23 12:15:45... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Ferrari F40 on 2022-07-23 08:15:45.670... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

Car Reservation for Ferrari F40 on 2022-07-23 08:15:45.670959

HalifaxCarRentalAgency <no-reply@sns.amazonaws.com>
To: Benny Tharigopala
Fri 2022-07-22 9:06 PM

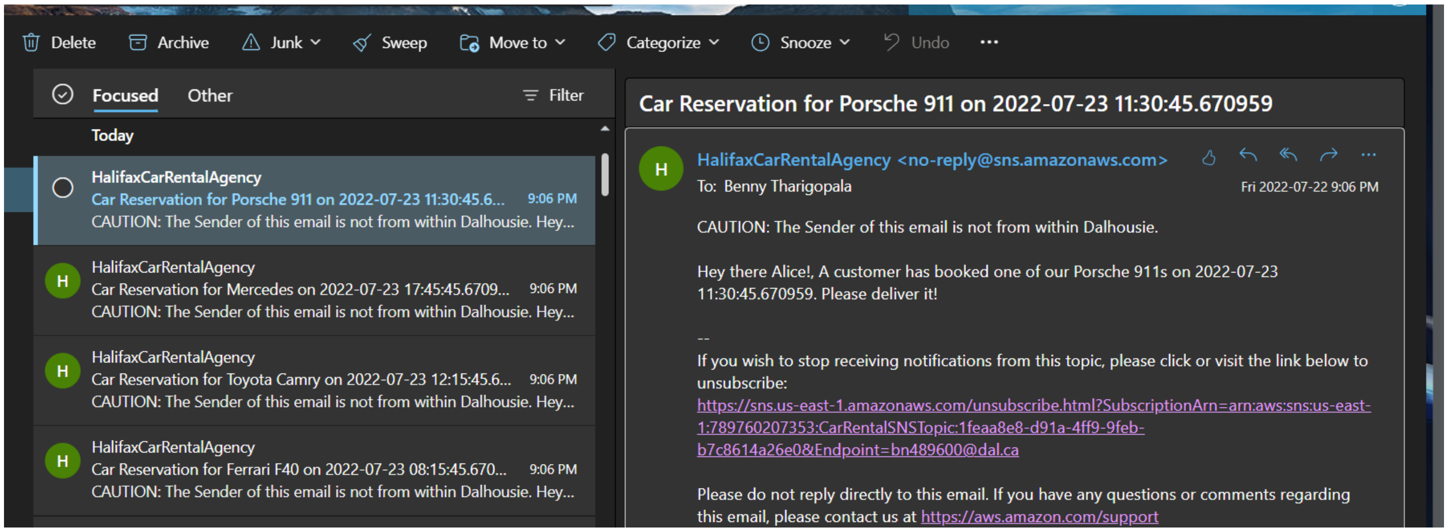
CAUTION: The Sender of this email is not from within Dalhousie.

Hey there Alice!, A customer has booked one of our Ferrari F40s on 2022-07-23 08:15:45.670959. Please deliver it!

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic:1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0&Endpoint=bn489600@dal.ca

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Figure 17: Snip of an Email from the SNS service



HalifaxCarRentalAgency
Car Reservation for Porsche 911 on 2022-07-23 11:30:45.6... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Mercedes on 2022-07-23 17:45:45.6709... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Toyota Camry on 2022-07-23 12:15:45.6... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

HalifaxCarRentalAgency
Car Reservation for Ferrari F40 on 2022-07-23 08:15:45.670... 9:06 PM
CAUTION: The Sender of this email is not from within Dalhousie. Hey...

Car Reservation for Porsche 911 on 2022-07-23 11:30:45.670959

HalifaxCarRentalAgency <no-reply@sns.amazonaws.com>
To: Benny Tharigopala
Fri 2022-07-22 9:06 PM

CAUTION: The Sender of this email is not from within Dalhousie.

Hey there Alice!, A customer has booked one of our Porsche 911s on 2022-07-23 11:30:45.670959. Please deliver it!

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic:1feaa8e8-d91a-4ff9-9feb-b7c8614a26e0&Endpoint=bn489600@dal.ca

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

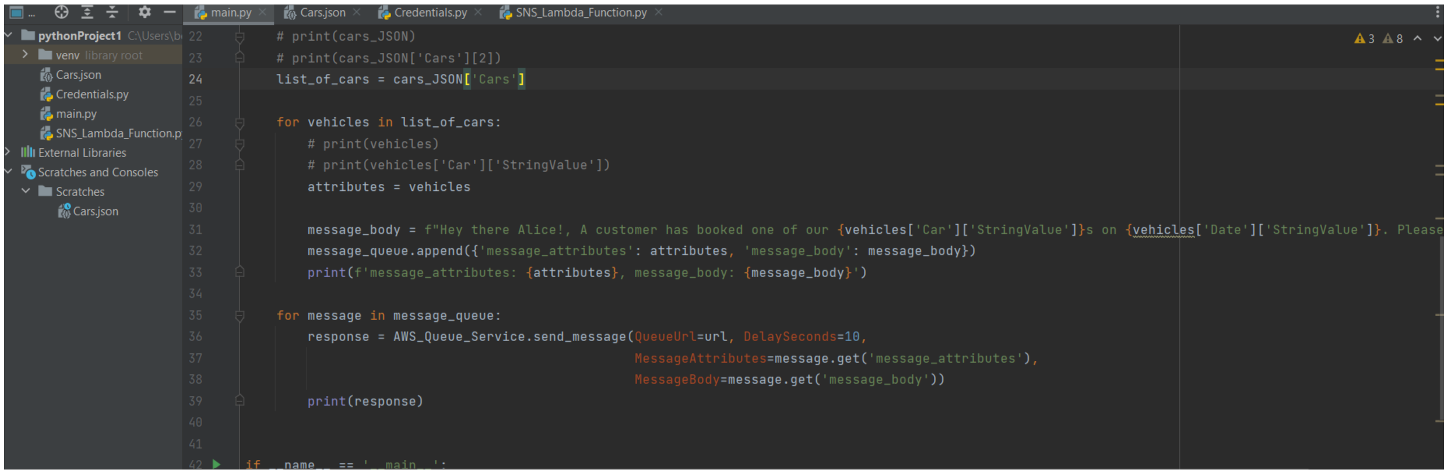
Figure 18: Another Snip of an Email from the SNS service

Code Blocks

JSON File - Sample

```
{
"Cars": [
    {
        "Car": {
"DataType": "String",
"StringValue": "Mercedes"
},
"Cost": {
    "DataType": "String",
    "StringValue": "850"
},
"Date": {
    "DataType": "String",
    "StringValue": "2022-07-23 17:45:45.670959"
}},
        {"Car": {
    "DataType": "String",
    "StringValue": "Porsche 911"
},
"Cost": {
    "DataType": "String",
    "StringValue": "1000"
},
"Date": {
    "DataType": "String",
    "StringValue": "2022-07-23 11:30:45.670959"
}},
        {"Car": {
    "DataType": "String",
    "StringValue": "Honda Civic"
},
"Cost": {
    "DataType": "String",
    "StringValue": "875"
},
"Date": {
    "DataType": "String",
    "StringValue": "2022-07-23 16:00:45.670959"
}},


    ]
    }
```

Script to Submit Messages to the SQS Queue [2-3,5]



```python
# print(cars_JSON)
# print(cars_JSON['Cars'][2])
list_of_cars = cars_JSON['Cars']

for vehicles in list_of_cars:
    # print(vehicles)
    # print(vehicles['Car']['StringValue'])
    attributes = vehicles

    message_body = f"Hey there Alice!, A customer has booked one of our {vehicles['Car']['StringValue']}s on {vehicles['Date']['StringValue']}. Please
    message_queue.append({'message_attributes': attributes, 'message_body': message_body})
    print(f'message_attributes: {attributes}, message_body: {message_body}')

for message in message_queue:
    response = AWS_Queue_Service.send_message(QueueUrl=url, DelaySeconds=10,
                                              MessageAttributes=message.get('message_attributes'),
                                              MessageBody=message.get('message_body'))
    print(response)


if __name__ == '__main__':
```

Figure 19: Message Simulation

Lambda Function [4]

```
import boto3


def lambda_handler(event, context):
    sns_client = boto3.client('sns')  # Connect to the SNS service with the client
    arn = 'arn:aws:sns:us-east-1:789760207353:CarRentalSNSTopic'

    print(event)

    message = event['Records'][0]['body']
    car = event['Records'][0]['messageAttributes']['Car']['stringValue']
    date = event['Records'][0]['messageAttributes']['Date']['stringValue']
    subject = 'Car Reservation for {} on {}'.format(car, date)

    response = sns_client.publish(TopicArn=arn, Message=message, Subject=subject)
```

Citations

[1] Dumbre, “Working with SQS in Python using Boto3,” *Hands-On-Cloud*, Sep. 02, 2021. https://hands-on.cloud/working-with-sqs-in-python-using-boto3/ (accessed Jul. 21, 2022).

[2] “SQS — Boto3 Docs 1.24.35 documentation,” *boto3.amazonaws.com*, Apr. 25, 2015. https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/sqs.html#SQS.Client.send_message (accessed Jul. 23, 2022).

[3] “Sample Amazon SQS function code - AWS Lambda,” *docs.aws.amazon.com*, Feb. 11, 2015. https://docs.aws.amazon.com/lambda/latest/dg/with-sqs-create-package.html (accessed Jul. 21, 2022).

[4] “Sample function code - AWS Lambda,” *docs.aws.amazon.com*, Apr. 25, 2016. https://docs.aws.amazon.com/lambda/latest/dg/with-sns-create-package.html#with-sns-example-deployment-pkg-python (accessed Jul. 23, 2022).

[5] theglitchblog, “Trigger AWS Lambda with AWS SQS using Python,” *the glitch blog*, Jul. 10, 2021. https://theglitchblog.com/2021/07/11/trigger-aws-lambda-with-aws-sqs-using-python/ (accessed Jul. 21, 2022).