SNS AND SQS IN Amazon Web Services

1. ABSTRACT

The AWS Cloud provides a broad set of infrastructure services, such as computing power, storage options, networking and databases that are delivered as a utility: on-demand, available in seconds, with pay-as-you-go pricing. From data warehousing to deployment tools, directories to content delivery, over 90 AWS services are available. New services can be provisioned quickly, without upfront capital expense. This allows enterprises, start-ups, small and medium-sized businesses, and customers in the public sector to access the building blocks they need to respond quickly to changing business requirements. This provides you with an overview of the benefits of the AWS Cloud and introduces you to the services that make up the platform.

We are creating application which will allow registration of participant for a an event .once the participant have been registered they are subscribed member of SNS event topic. The application displays all the existing topic and also the different attribute of the message for each participant. These participant are in queue and message sent by Event manager(SNS topic) is been notified to all the existing subscribers through one of the action such as email.

2. **INTRODUCTION**

In 2006, Amazon Web Services (AWS) began offering IT infrastructure services to businesses in the form of web services—now commonly known as cloud computing. One of the key benefits of cloud computing is the opportunity to replace up-front capital infrastructure expenses with low variable costs that scale with your business.

With the cloud, businesses no longer need to plan for and procure servers and other IT infrastructure weeks or months in advance. Instead, they can instantly spin up hundreds or thousands of servers in minutes and deliver results faster. Today, AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that powers hundreds of thousands of businesses in 190 countries around the world.

The AWS Cloud infrastructure is built around Regions and Availability Zones (AZs). A Region is a physical location in the world where we have multiple AZs. AZs consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities.

These AZs offer you the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center. The AWS Cloud operates 42 AZs within 16 geographic Regions around the world, with five more Availability Zones and two more Regions coming online in 2017.

Cloud security at AWS is the highest priority. As an AWS customer, you will benefit from a data center and network architecture built to meet the requirements of the most security sensitive organizations. Security in the cloud is much like security in your on-premises data centers—only without the costs of maintaining facilities and hardware.

In the cloud, you don’t have to manage physical servers or storage devices. Instead, you use software-based security tools to monitor and protect the flow of information into and of out of your cloud resources

1.1SERVICES USED BY THE PROJECT

Amazon SQS Amazon Simple Queue Service (Amazon SQS)

SQS is a fast, reliable, scalable, fully managed message queuing service.120 Amazon SQS makes it simple and cost-effective to decouple the components of a cloud application.You can use Amazon SQS to transmit any volume of data, without losing messages or requiring other services to be always available. Amazon SQS includes standard queues with high throughput and at-least-once processing, and FIFO queues that provide FIFO (first-in, first-out) delivery and exactly-once processing.

Amazon SNS Amazon Simple Notification Service (Amazon SNS)

SNS is a fast, flexible, fully managed push notification service that lets you send individual messages or to fan-out messages to large numbers of recipients. Amazon SNS makes it simple and cost effective to send push notifications to mobile device users, email recipients or even send messages to other distributed services.

With Amazon SNS, you can send notifications to Apple, Google, Fire OS, and Windows devices, as well as to Android devices in China with Baidu Cloud Push. You can use Amazon SNS to send SMS messages to mobile device users worldwide. Beyond these endpoints, Amazon SNS can also deliver messages to Amazon Simple Queue Service (SQS), AWS Lambda functions, or to any HTTP endpoint.

Architecture flow:-

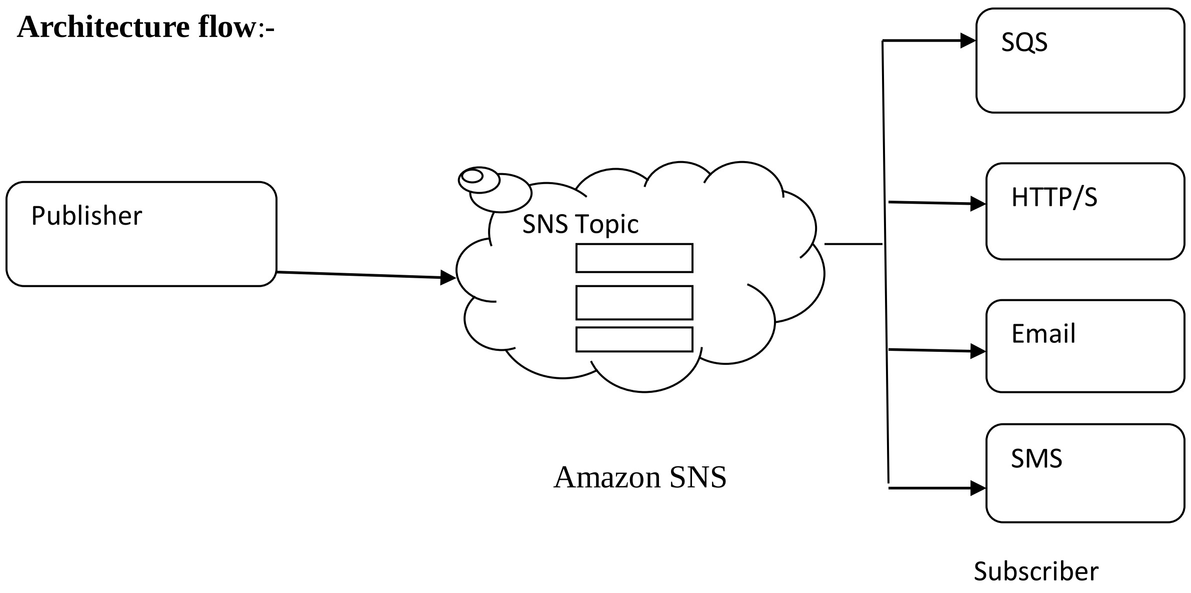
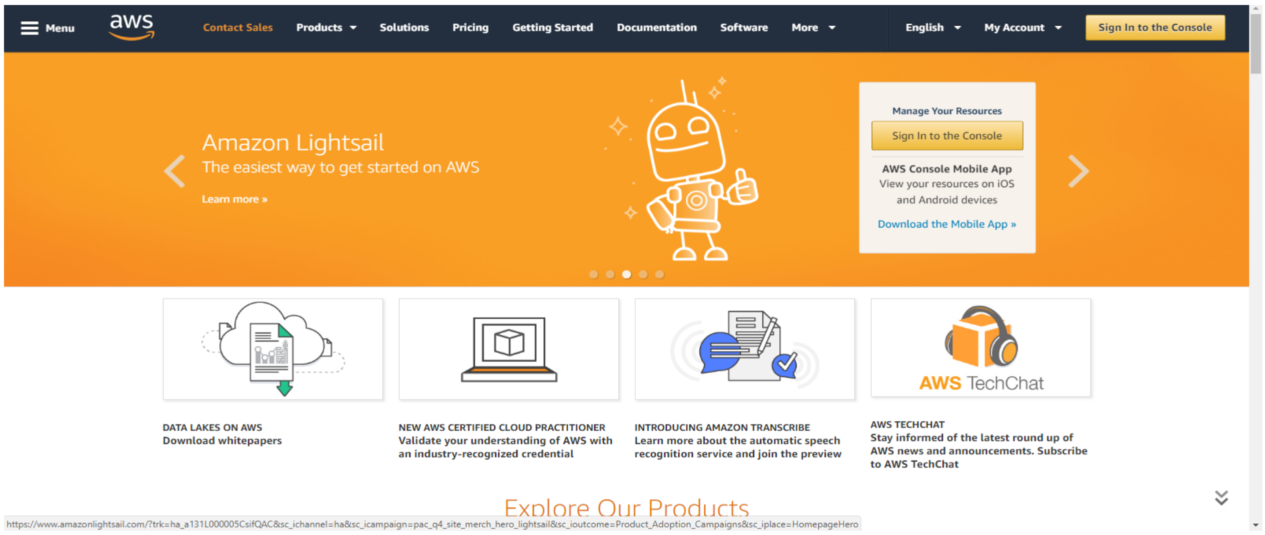


FIG 1

3. GETTING STARTED WITH AMAZON WEB SERVICES

Step 1 :: We login intoAmazon web services account through Amazon portal, link is as follows.https://portal.amazon.com/ . Using mail id and password.



Amazon Lightsail
The easiest way to get started on AWS

Manage Your Resources

AWS Console Mobile App
View your resources on iOS and Android devices

DATA LAKES ON AWS
Download whitepapers

NEW AWS CERTIFIED CLOUD PRACTITIONER
Validate your understanding of AWS with an industry-recognized credential

INTRODUCING AMAZON TRANSCRIBE
Learn more about the automatic speech recognition service and join the preview

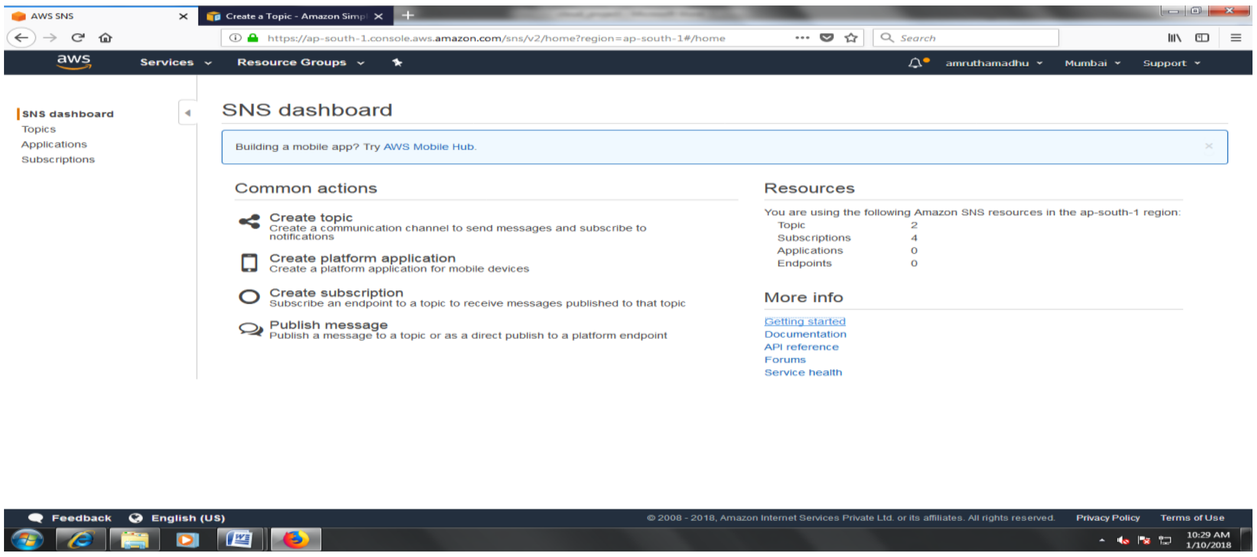
AWS TECHCHAT
Stay informed of the latest round up of AWS news and announcements. Subscribe to AWS TechChat

Explore Our Products

https://www.amazonlightsail.com/?trk=ha_a131L000005CsifQAC&sc_ichannel=ha&sc_icampaign=pac_q4_site_merch_hero_lightsail&sc_ioutcome=Product_Adoption_Campaigns&sc_iplace=HomepageHero

FIG 2

Step 2: selecting the service simple notification service and Fig 2 shows the dashboard where we can create topic.



https://ap-south-1.console.aws.amazon.com/sns/v2/home?region=ap-south-1#/home

SNS dashboard

Building a mobile app? Try AWS Mobile Hub.

Common actions

Create topic
Create a communication channel to send messages and subscribe to notifications

Create platform application
Create a platform application for mobile devices

Create subscription
Subscribe an endpoint to a topic to receive messages published to that topic

Publish message
Publish a message to a topic or as a direct publish to a platform endpoint

Resources

You are using the following Amazon SNS resources in the ap-south-1 region:

Topic	2
Subscriptions	4
Applications	0
Endpoints	0

More info

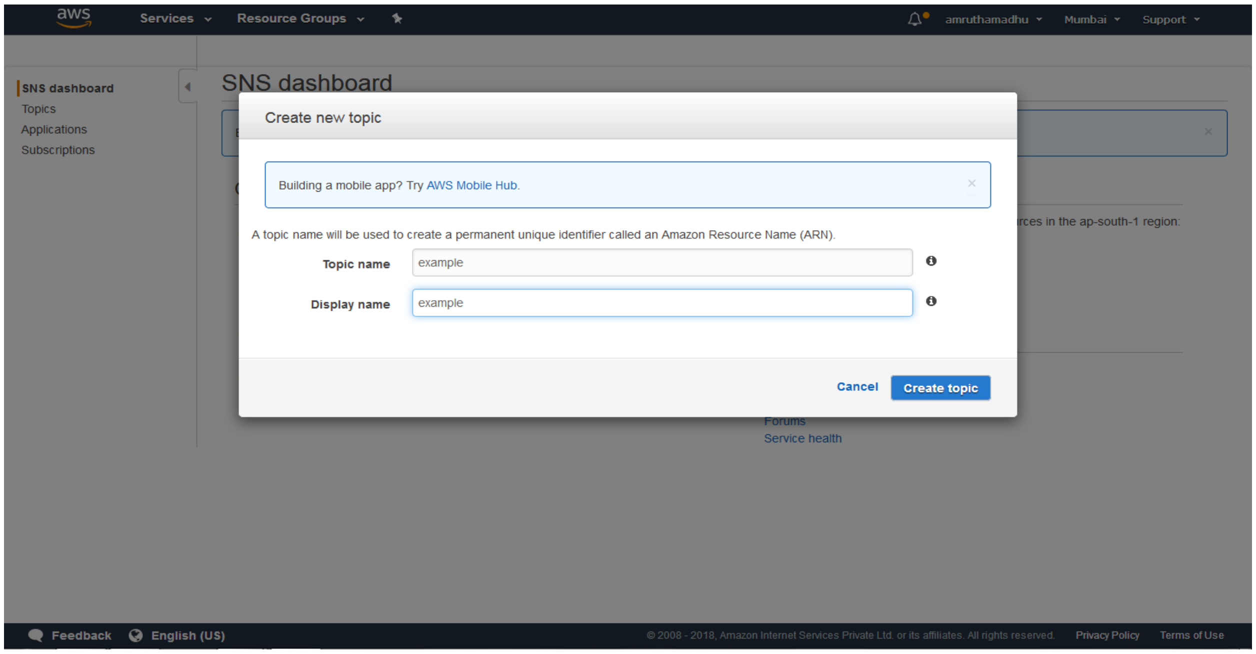
Getting started
Documentation
API reference
Forums
Service health

© 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

10:29 AM
1/10/2018

FIG 3

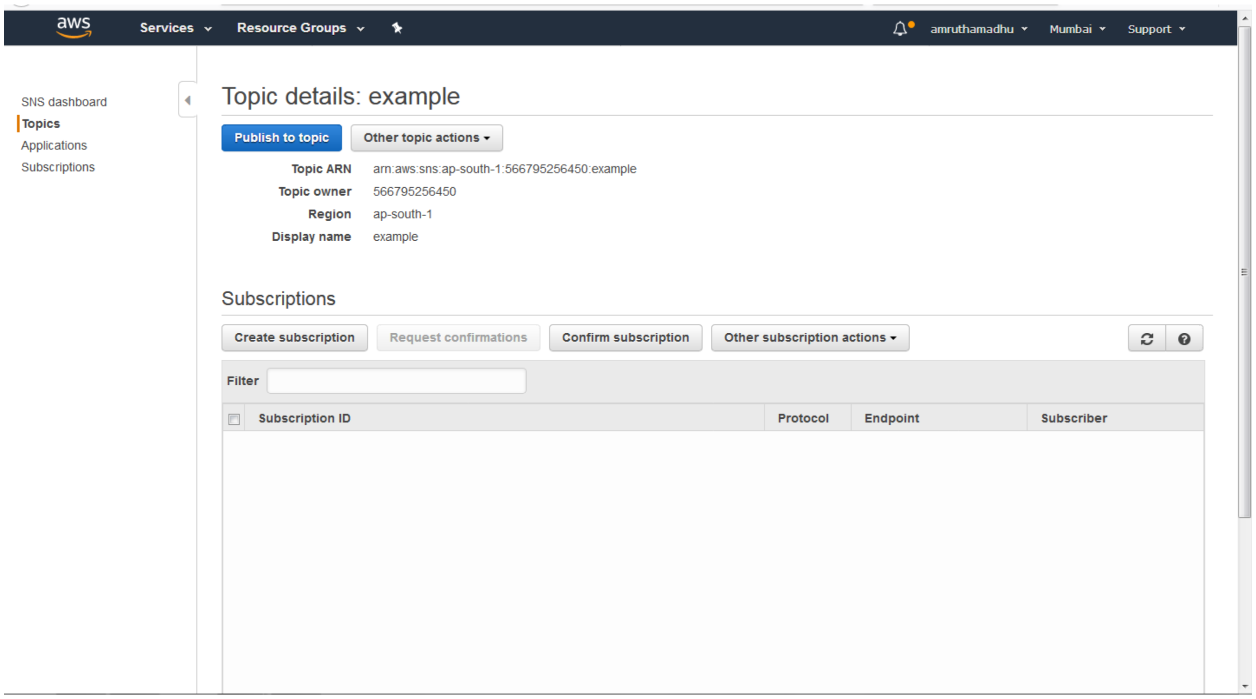
Step 3: In the Amazon SNS console, click Create topic. The Create topic dialog box appears. In the Topic name box, type a topic name. Click Create topic. The new topic appears in the Topics page.Select the new topic and then clicks the topic ARN.



A topic name will be used to create a permanent unique identifier called an Amazon Resource Name (ARN).

Topic name	example
Display name	example

FIG 4

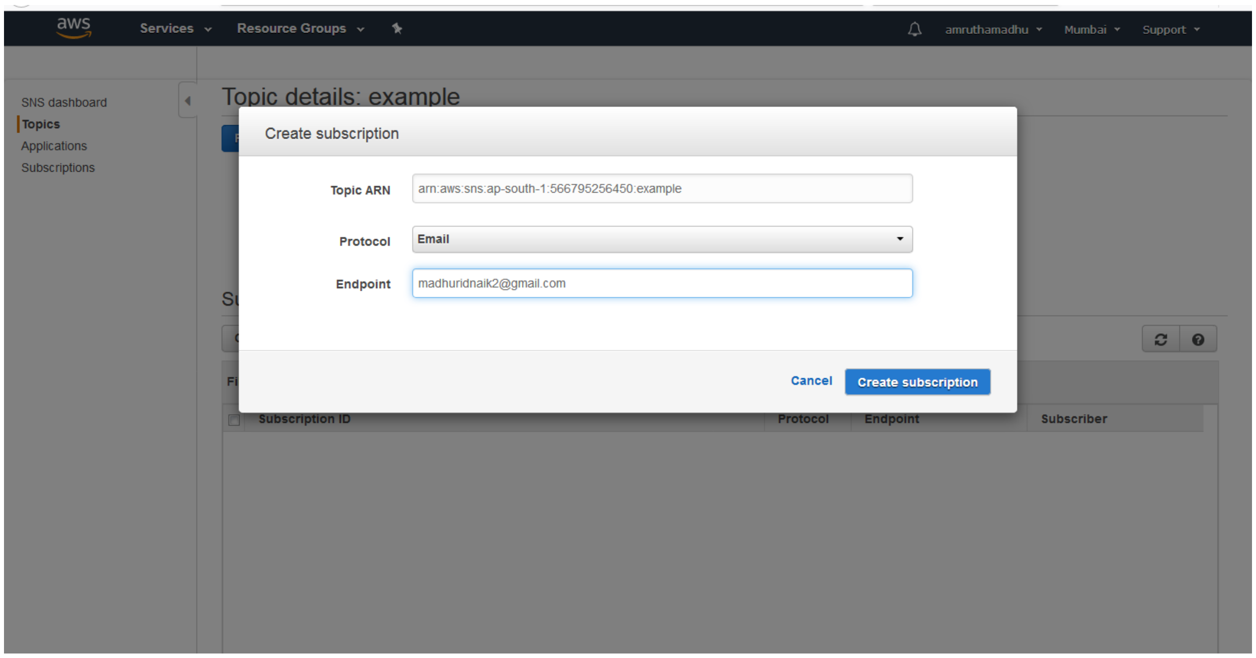


Topic ARN	arn:aws:sns:ap-south-1:566795256450:example
Topic owner	566795256450
Region	ap-south-1
Display name	example

Subscription ID	Protocol	Endpoint	Subscriber

FIG 5

Step 4 : Open the Amazon SNS console at https://console.aws.amazon.com/sns/v2/home. Click Create subscription. The Create Subscription dialog box appears. In the Topic ARN field, paste the topic ARN you created in the previous task.



Create subscription

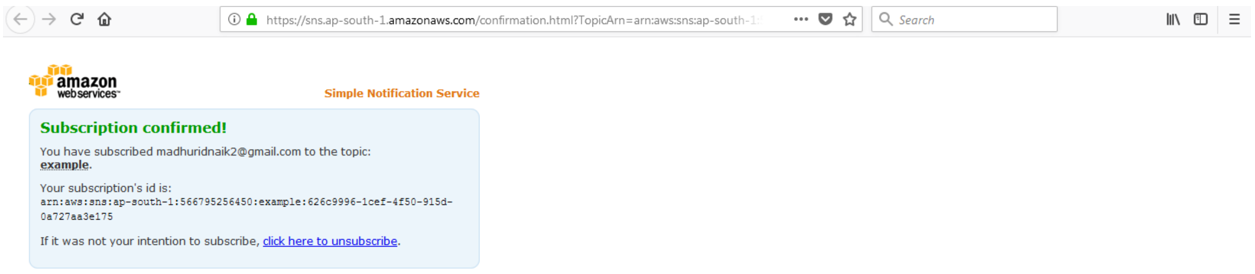
Topic ARN: arn:aws:sns:ap-south-1:566795256450:example

Protocol: Email

Endpoint: madhuridnaik2@gmail.com

FIG 6

In the Protocol drop-down box, select Email. In the Endpoint box, type an email address you can use to receive the notification. Go to your email application and open the message from AWS Notifications, and then click the link to confirm your subscription.


https://sns.ap-south-1.amazonaws.com/confirmation.html?TopicArn=arn:aws:sns:ap-south-1:
Search
amazon web services
Simple Notification Service

Subscription confirmed!

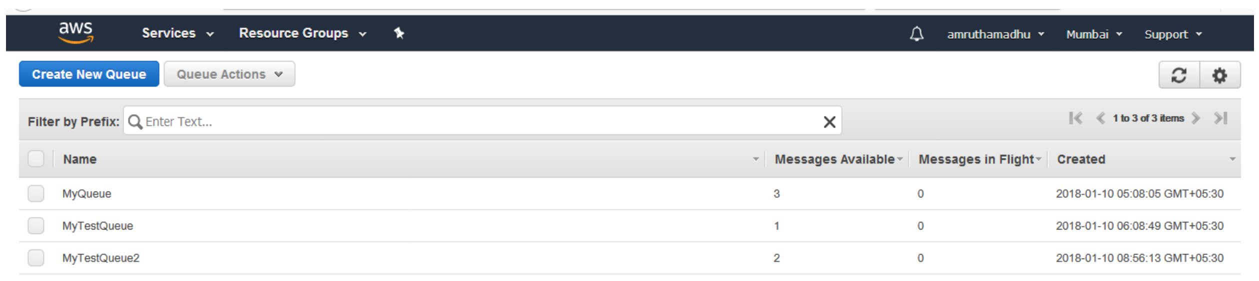
You have subscribed madhuridnaik2@gmail.com to the topic: **example**.

Your subscription's id is:
arn:aws:sns:ap-south-1:566795256450:example:626c9996-1cef-4f50-915d-0a727aa3e175

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

FIG 7

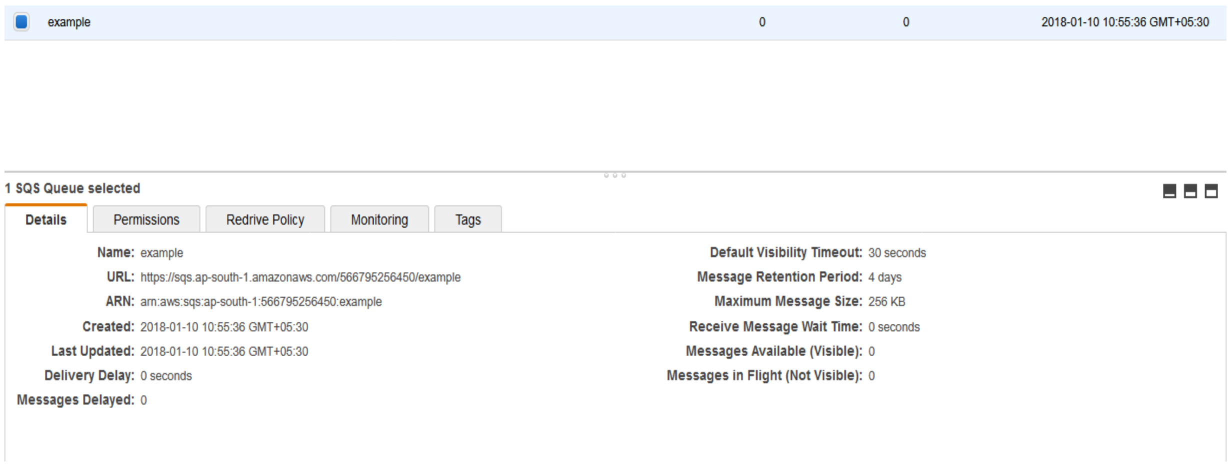
Step 5 : for simple queuing service we need to create a new queue in the console



Name	Messages Available	Messages in Flight	Created
MyQueue	3	0	2018-01-10 05:08:05 GMT+05:30
MyTestQueue	1	0	2018-01-10 06:08:49 GMT+05:30
MyTestQueue2	2	0	2018-01-10 08:56:13 GMT+05:30

FIG 8

Choose Create New Queue.On the Create New Queue page, ensure that you're in the correct region and then type the Queue Name.



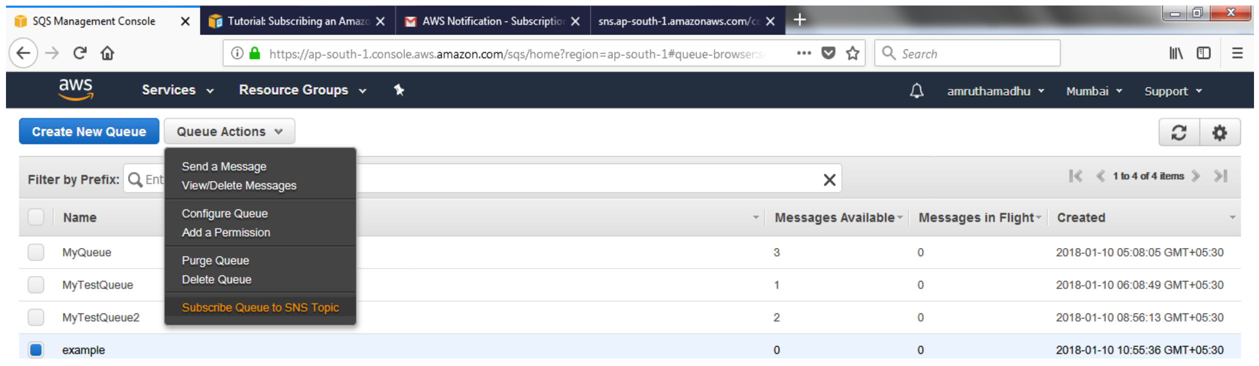
example	0	0	2018-01-10 10:55:36 GMT+05:30

Name: example
URL: https://sqs.ap-south-1.amazonaws.com/566795256450/example
ARN: arn:aws:sqs:ap-south-1:566795256450:example
Created: 2018-01-10 10:55:36 GMT+05:30
Last Updated: 2018-01-10 10:55:36 GMT+05:30
Delivery Delay: 0 seconds
Messages Delayed: 0

Default Visibility Timeout: 30 seconds
Message Retention Period: 4 days
Maximum Message Size: 256 KB
Receive Message Wait Time: 0 seconds
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0

FIG 9

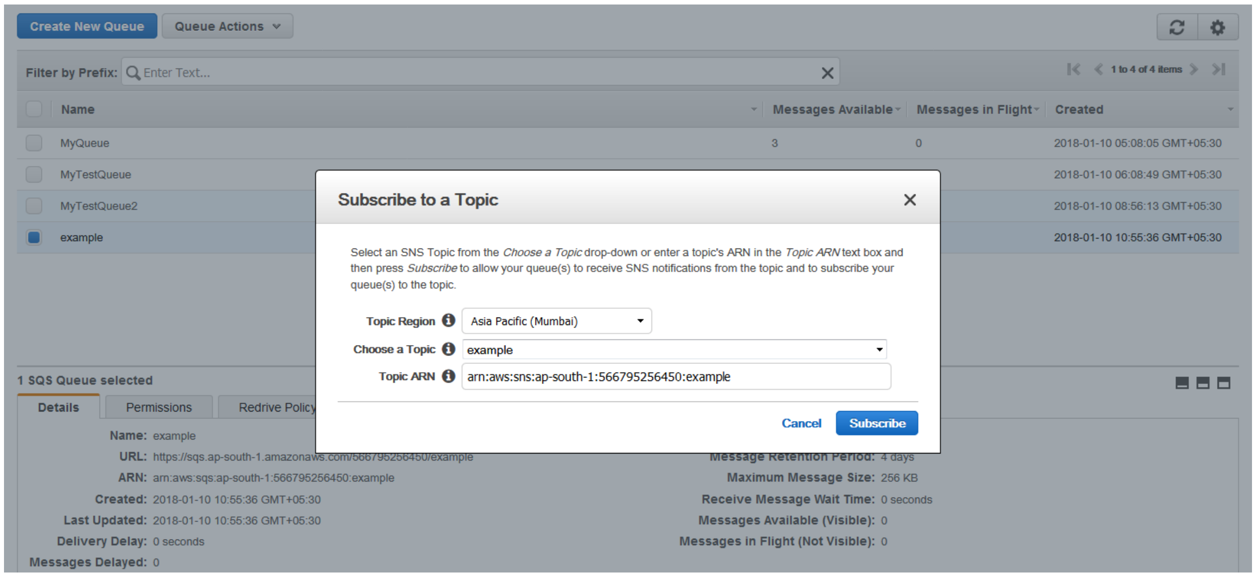
Step 6: From Queue Actions, select Subscribe Queue to SNS Topic (or Subscribe Queues to SNS Topic).



Name	Messages Available	Messages in Flight	Created
MyQueue	3	0	2018-01-10 05:08:05 GMT+05:30
MyTestQueue	1	0	2018-01-10 06:08:49 GMT+05:30
MyTestQueue2	2	0	2018-01-10 08:56:13 GMT+05:30
example	0	0	2018-01-10 10:55:36 GMT+05:30

FIG 10

Choose the existing sns topic or create a new sns topic and enter the



Name	Messages Available	Messages in Flight	Created
MyQueue	3	0	2018-01-10 05:08:05 GMT+05:30
MyTestQueue			2018-01-10 06:08:49 GMT+05:30
MyTestQueue2			2018-01-10 08:56:13 GMT+05:30
example			2018-01-10 10:55:36 GMT+05:30

Subscribe to a Topic

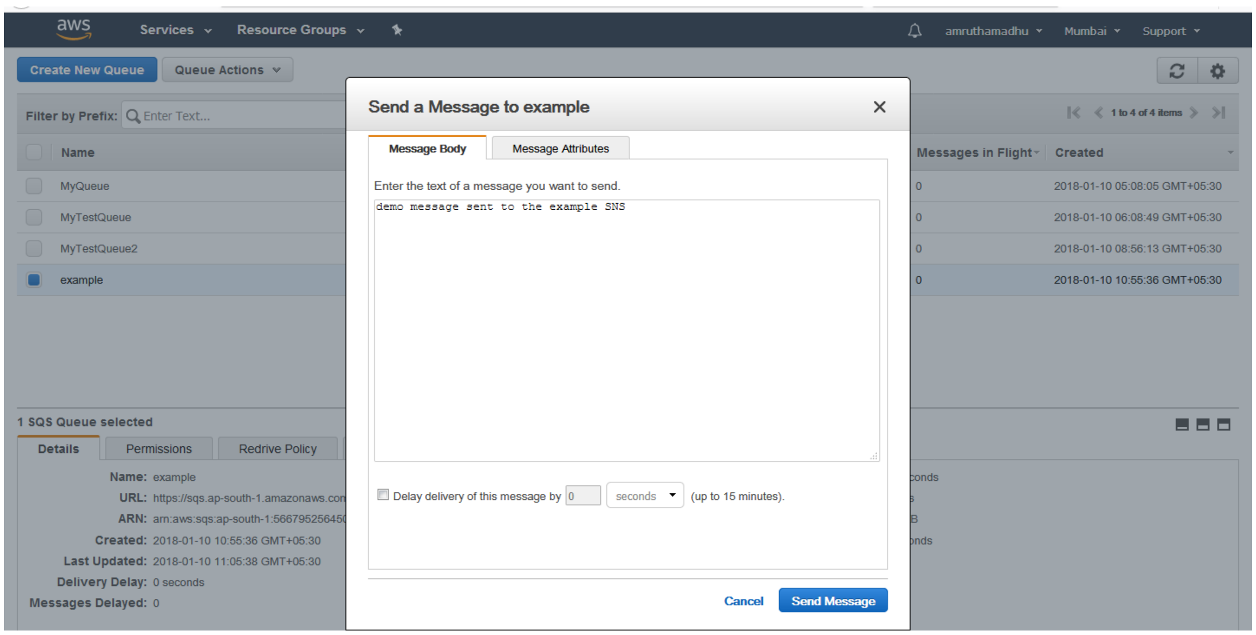
Select an SNS Topic from the *Choose a Topic* drop-down or enter a topic's ARN in the *Topic ARN* text box and then press *Subscribe* to allow your queue(s) to receive SNS notifications from the topic and to subscribe your queue(s) to the topic.

Topic Region: Asia Pacific (Mumbai)
Choose a Topic: example
Topic ARN: arn:aws:sns:ap-south-1:566795256450:example

Name: example
URL: https://sqs.ap-south-1.amazonaws.com/566795256450/example
ARN: arn:aws:sqs:ap-south-1:566795256450:example
Created: 2018-01-10 10:55:36 GMT+05:30
Last Updated: 2018-01-10 10:55:36 GMT+05:30
Delivery Delay: 0 seconds
Messages Delayed: 0
Message Retention Period: 4 days
Maximum Message Size: 256 KB
Receive Message Wait Time: 0 seconds
Messages Available (Visible): 0
Messages in Flight (Not Visible): 0

FIG 11

Write the message that need to be published to all the subscriber



Name	Messages in Flight	Created
MyQueue	0	2018-01-10 05:08:05 GMT+05:30
MyTestQueue	0	2018-01-10 06:08:49 GMT+05:30
MyTestQueue2	0	2018-01-10 08:56:13 GMT+05:30
example	0	2018-01-10 10:55:36 GMT+05:30

Send a Message to example

Enter the text of a message you want to send.

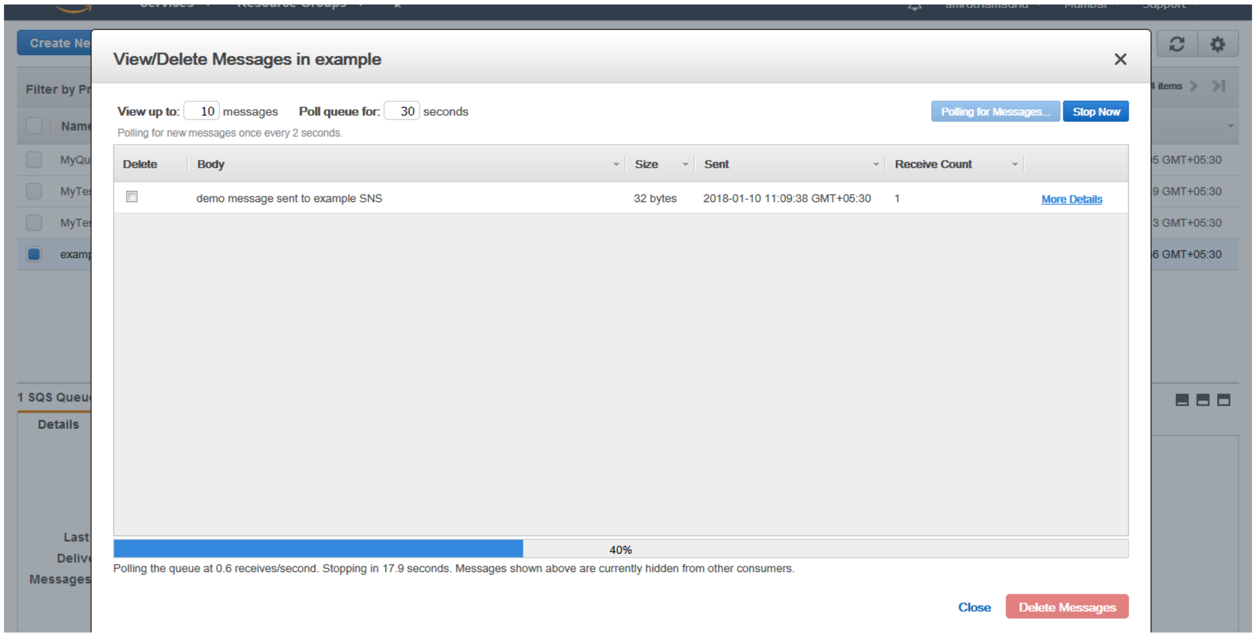
```
demo message sent to the example SNS
```

Delay delivery of this message by 0 seconds (up to 15 minutes).

Name: example
URL: https://sqs.ap-south-1.amazonaws.com
ARN: arn:aws:sqs:ap-south-1:566795256450
Created: 2018-01-10 10:55:36 GMT+05:30
Last Updated: 2018-01-10 11:05:38 GMT+05:30
Delivery Delay: 0 seconds
Messages Delayed: 0

FIG12

View of the message and also polling the message



View/Delete Messages in example

View up to: 10 messages Poll queue for: 30 seconds

Polling for new messages once every 2 seconds.

Delete	Body	Size	Sent	Receive Count	
☐	demo message sent to example SNS	32 bytes	2018-01-10 11:09:38 GMT+05:30	1	More Details

40%

Polling the queue at 0.6 receives/second. Stopping in 17.9 seconds. Messages shown above are currently hidden from other consumers.

FIG 13

4. EXECUTION AND RESULT:-

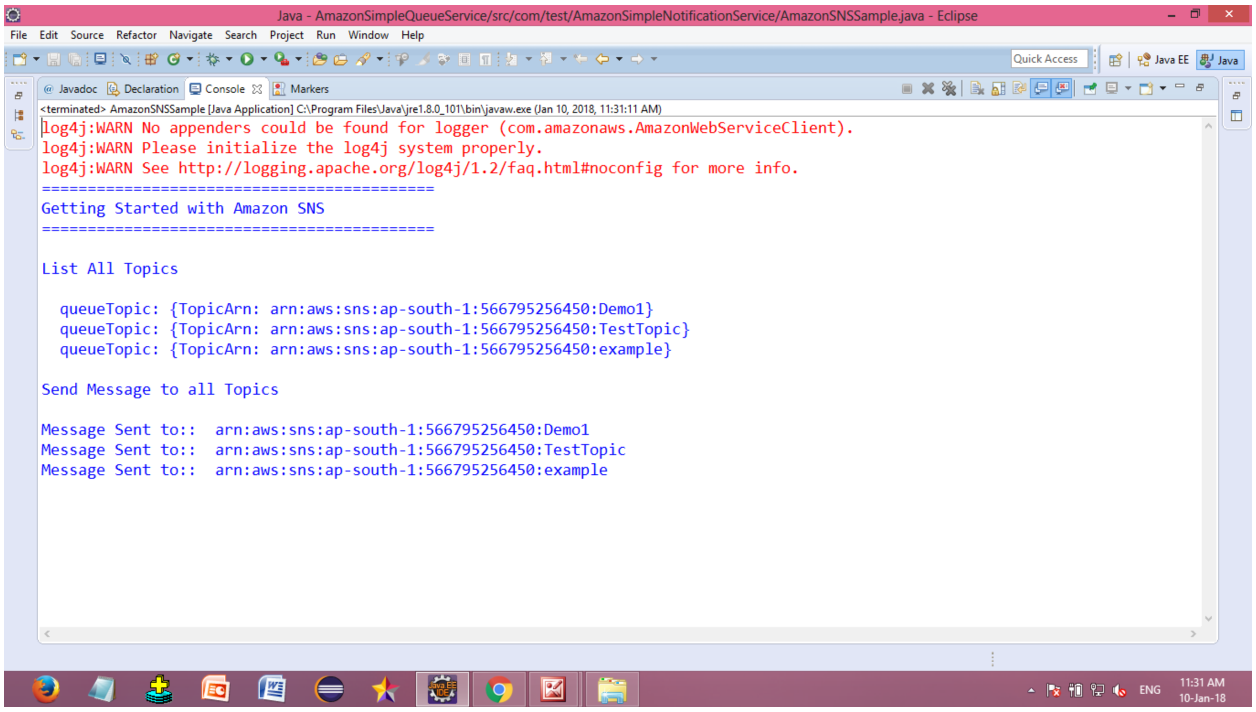
In order to connect the platform to the cloud we need to have the credential data so that user can access the cloud API .
First we need to create a folder named .aws under C:\users\username.Inside .aws folder include credentials, which contains the details such as

- aws_access_key
- aws_secret_key

4.1 SNS code:

```
package com.test.AmazonSimpleNotificationService;

import java.io.IOException;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.sns.AmazonSNS;
import com.amazonaws.services.sns.AmazonSNSClientBuilder;
import com.amazonaws.services.sns.model.Topic;


public class AmazonSNSSample {

    public static void main(String[] args) throws IOException {

        ProfileCredentialsProvider credentialsProvider = new ProfileCredentialsProvider();
        try {
            credentialsProvider.getCredentials();
        } catch (Exception e) {
            throw new AmazonClientException(
                    "Cannot load the credentials from the credential profiles file. "+e);
        }

         AmazonSNS sns = AmazonSNSClientBuilder.standard()
                       .withCredentials(credentialsProvider)
                       .withRegion(Regions.AP_SOUTH_1)
                       .build();

        System.out.println("===========================================");
        System.out.println("Getting Started with Amazon SNS");
        System.out.println("===========================================\n");

        try {
            // List All Topics and Send Message to all Topics
            System.out.println("List All Topics\n");

            for (Topic queueTopic : sns.listTopics().getTopics()) {
                System.out.println("  queueTopic: " + queueTopic);
```

```
          }
          System.out.println("\nSend Message to all Topics\n");

          for (Topic queueTopic : sns.listTopics().getTopics()) {
              sns.publish(queueTopic.getTopicArn(),
                      "This is Message from Java API to :: " + queueTopic.getTopicArn(),
                      "Test Message Title");
                System.out.println("Message Sent to::  "+ queueTopic.getTopicArn());

              }

        } catch (Exception ex) {
            System.out.println("Error Message: " + ex.getMessage());
        }
    }
}
```



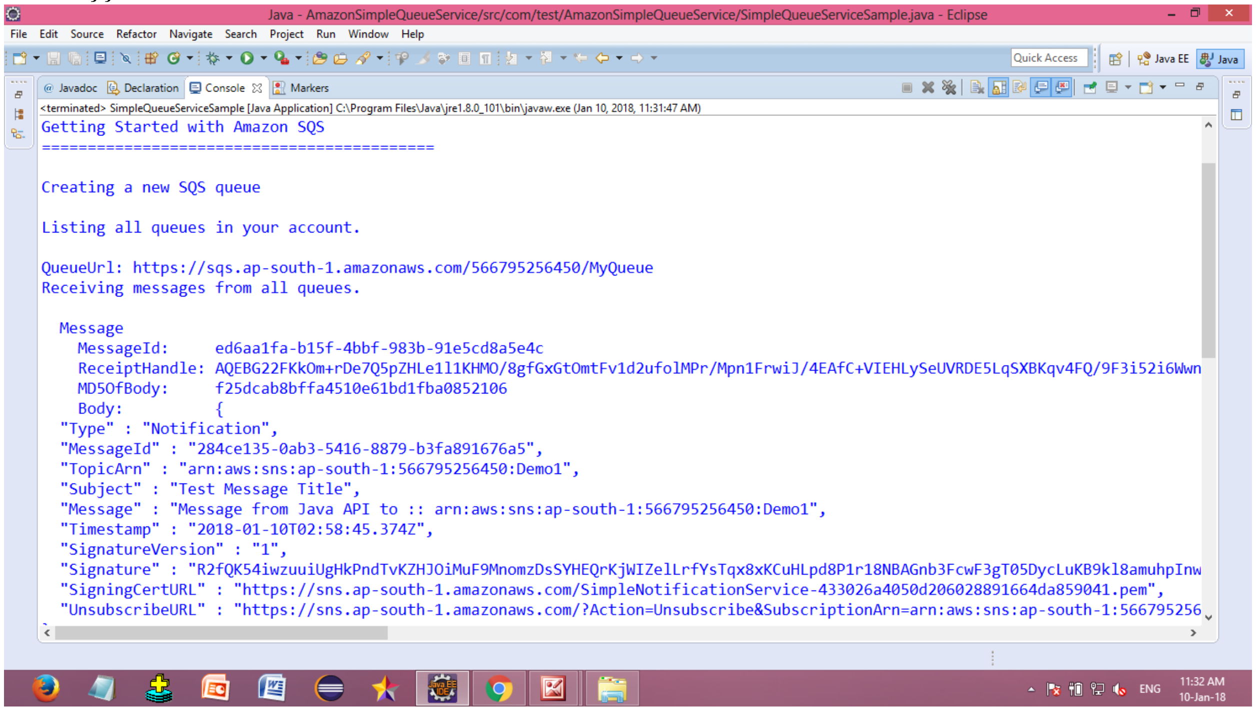
```
Java - AmazonSimpleQueueService/src/com/test/AmazonSimpleNotificationService/AmazonSNSSample.java - Eclipse
<terminated> AmazonSNSSample [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (Jan 10, 2018, 11:31:11 AM)
log4j:WARN No appenders could be found for logger (com.amazonaws.AmazonWebServiceClient).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
===========================================
Getting Started with Amazon SNS
===========================================

List All Topics

  queueTopic: {TopicArn: arn:aws:sns:ap-south-1:566795256450:Demo1}
  queueTopic: {TopicArn: arn:aws:sns:ap-south-1:566795256450:TestTopic}
  queueTopic: {TopicArn: arn:aws:sns:ap-south-1:566795256450:example}

Send Message to all Topics

Message Sent to::  arn:aws:sns:ap-south-1:566795256450:Demo1
Message Sent to::  arn:aws:sns:ap-south-1:566795256450:TestTopic
Message Sent to::  arn:aws:sns:ap-south-1:566795256450:example
```

FIG 14

It will list out all the topics that were created and sends the notification to all the subscribers

4.2 SQS code

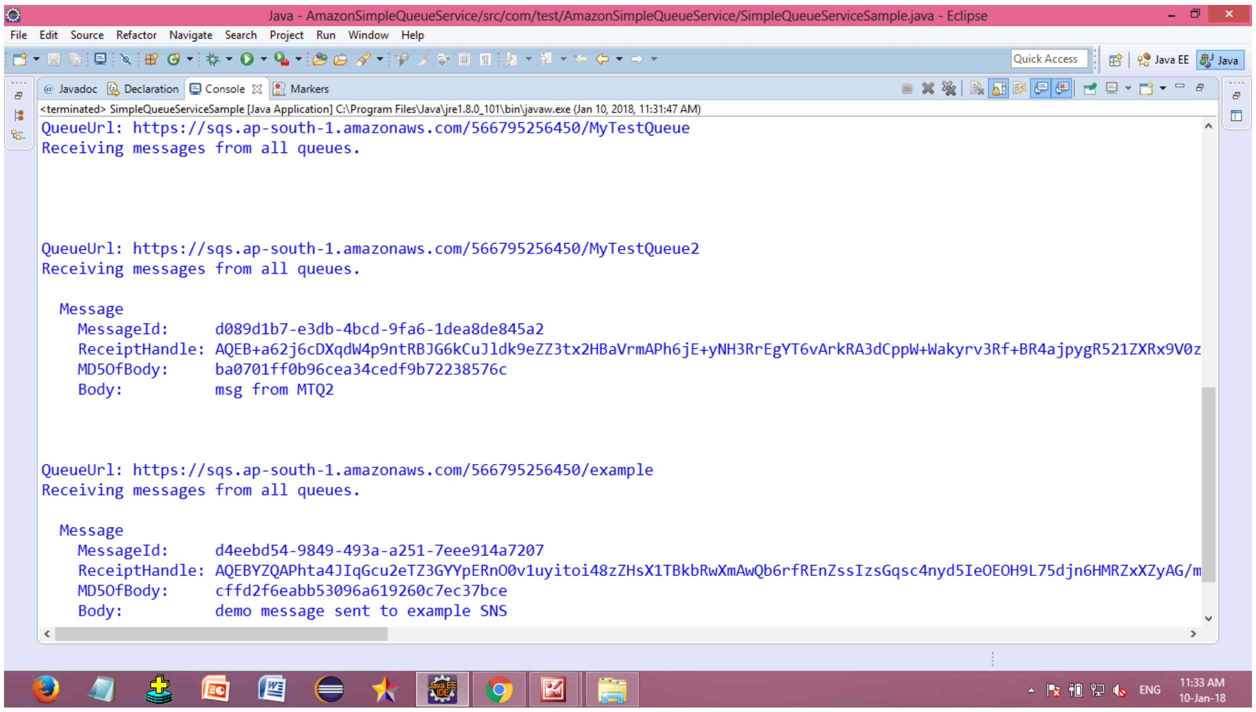
```
package com.test.AmazonSimpleQueueService;

import java.util.List;
import java.util.Map.Entry;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.DeleteMessageRequest;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SendMessageRequest;

public class SimpleQueueServiceSample {

    public static void main(String[] args) throws Exception {

        ProfileCredentialsProvider credentialsProvider = new ProfileCredentialsProvider();
        try {
            credentialsProvider.getCredentials();
        } catch (Exception e) {
            throw new AmazonClientException(
                    "Cannot load the credentials from the credential profiles file. "+e);
        }

        AmazonSQS sqs = AmazonSQSClientBuilder.standard()
                              .withCredentials(credentialsProvider)
                              .withRegion(Regions.AP_SOUTH_1)
                              .build();
        System.out.println("===========================================");
        System.out.println("Getting Started with Amazon SQS");
        System.out.println("===========================================\n");

        try {
            // Create a queue
            System.out.println("Creating a new SQS queue \n");
            CreateQueueRequest createQueueRequest = new CreateQueueRequest("MyTestQueue2");
            String myQueueUrl = sqs.createQueue(createQueueRequest).getQueueUrl();

            // List queues
            System.out.println("Listing all queues in your account.\n");
            for (String queueUrl : sqs.listQueues().getQueueUrls()) {
                System.out.println("QueueUrl: " + queueUrl);
            //System.out.println();

            // Send a message
            //System.out.println("Sending a message to all Queues.\n");
           // sqs.sendMessage(new SendMessageRequest(queueUrl, "This is the message from SQS API."));

            // Receive messages
            System.out.println("Receiving messages from all queues.\n");
            ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest(queueUrl);
            List<Message> messages = sqs.receiveMessage(receiveMessageRequest).getMessages();
            for (Message message : messages) {
                System.out.println("  Message");
                System.out.println("    MessageId:     " + message.getMessageId());
                System.out.println("    ReceiptHandle: " + message.getReceiptHandle());
                System.out.println("    MD5OfBody:     " + message.getMD5OfBody());
                System.out.println("    Body:          " + message.getBody());
                for (Entry<String, String> entry : message.getAttributes().entrySet()) {
                    System.out.println("  Attribute");
                    System.out.println("    Name:  " + entry.getKey());
                    System.out.println("    Value: " + entry.getValue());
                }
            }
            System.out.println("\n\n");
            //System.out.println("Deleting all message.\n");
            //String messageReceiptHandle = messages.get(0).getReceiptHandle();
            //sqs.deleteMessage(new DeleteMessageRequest(queueUrl, messageReceiptHandle));
        }
         //  System.out.println("Deleting the MyTestQueue queue.\n");
         //  sqs.deleteQueue(new DeleteQueueRequest(myQueueUrl));

        } catch (AmazonServiceException ase) {

            System.out.println("Error Message:    " + ase.getMessage());

        } catch (AmazonClientException ace) {
            System.out.println("Error Message: " + ace.getMessage());
            }}
```



Java - AmazonSimpleQueueService/src/com/test/AmazonSimpleQueueService/SimpleQueueServiceSample.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java EE
Java
Javadoc
Declaration
Console
Markers
ENG
11:32 AM
10-Jan-18

```
<terminated> SimpleQueueServiceSample [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (Jan 10, 2018, 11:31:47 AM)
Getting Started with Amazon SQS
===========================================

Creating a new SQS queue

Listing all queues in your account.

QueueUrl: https://sqs.ap-south-1.amazonaws.com/566795256450/MyQueue
Receiving messages from all queues.

  Message
    MessageId:     ed6aa1fa-b15f-4bbf-983b-91e5cd8a5e4c
    ReceiptHandle: AQEBG22FKkOm+rDe7Q5pZHLe1l1KHMO/8gfGxGtOmtFv1d2ufolMPr/Mpn1FrwiJ/4EAfC+VIEHLySeUVRDE5LqSXBKqv4FQ/9F3i52i6Wwn
    MD5OfBody:     f25dcab8bffa4510e61bd1fba0852106
    Body:          {
  "Type" : "Notification",
  "MessageId" : "284ce135-0ab3-5416-8879-b3fa891676a5",
  "TopicArn" : "arn:aws:sns:ap-south-1:566795256450:Demo1",
  "Subject" : "Test Message Title",
  "Message" : "Message from Java API to :: arn:aws:sns:ap-south-1:566795256450:Demo1",
  "Timestamp" : "2018-01-10T02:58:45.374Z",
  "SignatureVersion" : "1",
  "Signature" : "R2fQK54iwzuuiUgHkPndTvKZHJOiMuF9MnomzDsSYHEQrKjWIZelLrfYsTqx8xKCuHLpd8P1r18NBAGnb3FcwF3gT05DycLuKB9kl8amuhpInw
  "SigningCertURL" : "https://sns.ap-south-1.amazonaws.com/SimpleNotificationService-433026a4050d206028891664da859041.pem",
  "UnsubscribeURL" : "https://sns.ap-south-1.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:ap-south-1:566795256
```

FIG 15


Java - AmazonSimpleQueueService/src/com/test/AmazonSimpleQueueService/SimpleQueueServiceSample.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java EE
Java
Javadoc
Declaration
Console
Markers
11:33 AM
10-Jan-18

```
<terminated> SimpleQueueServiceSample [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (Jan 10, 2018, 11:31:47 AM)
QueueUrl: https://sqs.ap-south-1.amazonaws.com/566795256450/MyTestQueue
Receiving messages from all queues.



QueueUrl: https://sqs.ap-south-1.amazonaws.com/566795256450/MyTestQueue2
Receiving messages from all queues.

  Message
    MessageId:     d089d1b7-e3db-4bcd-9fa6-1dea8de845a2
    ReceiptHandle: AQEB+a62j6cDXqdW4p9ntRBJG6kCuJldk9eZZ3tx2HBaVrmAPh6jE+yNH3RrEgYT6vArkRA3dCppW+Wakyrv3Rf+BR4ajpygR521ZXRx9V0z
    MD5OfBody:     ba0701ff0b96cea34cedf9b72238576c
    Body:          msg from MTQ2



QueueUrl: https://sqs.ap-south-1.amazonaws.com/566795256450/example
Receiving messages from all queues.

  Message
    MessageId:     d4eebd54-9849-493a-a251-7eee914a7207
    ReceiptHandle: AQEBYZQAPhta4JIqGcu2eTZ3GYYpERnO0v1uyitoi48zZHsX1TBkbRwXmAwQb6rfREnZssIzsGqsc4nyd5IeOEOH9L75djn6HMRZxXZyAG/m
    MD5OfBody:     cffd2f6eabb53096a619260c7ec37bce
    Body:          demo message sent to example SNS
```

FIG 16

Here it will allow to create new queue or we can use the existing queue. Lists all the queue and send the subscription request to SNS and displays all the attributes of the sender and also displays the message on console as well as send the email to the subscriber.