

**CSCI 5410**

**Serverless Data Processing (Summer 2023)**

## MASTER OF APPLIED COMPUTER SCIENCE

Assignment-3 (Part C):

Name: **Jainil Sevalia |** Banner Id : **B00925445 |** Email:[**jn498899@dal.ca**](mailto:jn498899@dal.ca)

**Table of figures:**

[Figure 1 : Overall Architecture of Halifax Taxi 3](#_Toc140574950)

[Figure 2 : HalifaxTaxiQueue created (AWS SQS). 3](#_Toc140574951)

[Figure 3 : Queue created and got Queue URL. 4](#_Toc140574952)

[Figure 4 : Creating Publish Order SNS topic. 5](#_Toc140574953)

[Figure 5: Successfully created PublishOrder SNS topic. 5](#_Toc140574954)

[Figure 6 : SQS subscribed to SNS topic. 6](#_Toc140574955)

[Figure 7 : Create ECR Repository for Lambda Docker Image. 7](#_Toc140574956)

[Figure 8 : Successfully created ECR Repository. 7](#_Toc140574957)

[Figure 9 : PublishOrder lambda function Code. 8](#_Toc140574958)

[Figure 10 : Docker Image of Lambda code build, tagged and Pushed to ECR. 9](#_Toc140574959)

[Figure 11 : ECR repository AWS console. 9](#_Toc140574960)

[Figure 12 : Creating Lambda function using docker image pushed in ECR. 10](#_Toc140574961)

[Figure 13 : Successfully created Lambda Function. 10](#_Toc140574962)

[Figure 14 : Send Mail SNS topic. 11](#_Toc140574963)

[Figure 15 : Successfully created SNS topic. (AWS Console). 12](#_Toc140574964)

[Figure 16 : Subscribe to email in SendMail SNS topic. 13](#_Toc140574965)

[Figure 17 : Subscription list of SNS topic. 13](#_Toc140574966)

[Figure 18 : ECR Repository for second Lambda Docker Image. 14](#_Toc140574967)

[Figure 19 : Code for (poll\_msg\_send\_mail) lambda function. 15](#_Toc140574968)

[Figure 20 : Docker image for second Lambda is build, tagged, and pushed to ECR repository. 16](#_Toc140574969)

[Figure 21 : Successfully Second lambda docker image is pushed. (AWS console). 16](#_Toc140574970)

[Figure 22 : Creating Second Lambda function poll\_msg\_send\_mail. (AWS console). 17](#_Toc140574971)

[Figure 23 : Configure AWS Event bridge for invoking lambda every 2 min. 18](#_Toc140574972)

[Figure 24 : Second lambda(poll\_msg\_send\_mail) AWS console. 18](#_Toc140574973)

[Figure 25: Testing - Triggering first Lambda function to generate order and send to SNS to SQS. 19](#_Toc140574974)

[Figure 26: Message that sent to SNS topic. (Cloud watch AWS console). 19](#_Toc140574975)

[Figure 27 : Got email of with Taxi order Information. 20](#_Toc140574976)

[Figure 28 : Taxi Order Information got in mail. 20](#_Toc140574977)

**Halifax Taxi:**

**Architecture:**

A screenshot of a video game

Description automatically generated

Figure : Overall Architecture of Halifax Taxi

* Created a HalifaxTaxiQueue using AWS SQS service. This Queue will store all messages(Orders Information).

A screenshot of a computer

Description automatically generated

Figure : HalifaxTaxiQueue created (AWS SQS).

* Queue Created successfully.

A screenshot of a computer

Description automatically generated

Figure : Queue created and got Queue URL.

* Create AWS SNS topic. First AWS Lambda(Which randomly generate orders) will send order message to this SNS topic.

A screenshot of a computer

Description automatically generated

Figure : Creating Publish Order SNS topic.

* Successfully created PublishOrder SNS topic.

A screenshot of a computer

Description automatically generated

Figure : Successfully created PublishOrder SNS topic.

* SQS Queue is Subscribed to SNS topic. So, message will publish to SQS.

A screenshot of a computer

Description automatically generated

Figure : SQS subscribed to SNS topic.

* Create ECR Repository for Lambda Docker Image.

A screenshot of a computer

Description automatically generated

Figure : Create ECR Repository for Lambda Docker Image.

* Created ECR Repository for Docker Image.

A screen shot of a computer

Description automatically generated

Figure : Successfully created ECR Repository.

* Lambda function code is mentioned below, which will generate Taxi Order message and send it to SNS Topic(PublishOrder).

A screenshot of a computer program

Description automatically generated

Figure : PublishOrder lambda function Code.

* Docker image build, tagged and pushed to ECR newly created repository using custom deploy.sh script.

A screenshot of a computer program

Description automatically generated

Figure : Docker Image of Lambda code build, tagged and Pushed to ECR.

* Docker image is visible in the ECR repository in AWS Console.

A screenshot of a computer

Description automatically generated

Figure : ECR repository AWS console.

* Created AWS Lambda function using newly pushed docker image on ECR publish\_order Repository.

A screenshot of a computer screen

Description automatically generated

Figure : Creating Lambda function using docker image pushed in ECR.

* Successfully created Lambda function.

A screenshot of a computer

Description automatically generated

Figure : Successfully created Lambda Function.

* Create a new SNS topic for sending mail. Second Lambda will poll message from the SQS and send it to this SNS topic.

A screenshot of a computer

Description automatically generated

Figure : Send Mail SNS topic.

* Successfully SNS topic created for sending mail for car delivery.

A screenshot of a computer

Description automatically generated

Figure : Successfully created SNS topic. (AWS Console).

* Subscribe to email using SNS Subscription functionality.

A screenshot of a computer screen

Description automatically generated

Figure : Subscribe to email in SendMail SNS topic.

A screenshot of a computer

Description automatically generated

Figure : Subscription list of SNS topic.

* Create ECR repository for second lambda function. This function will poll the message from SQS. This function will trigger every 2 min.

A screenshot of a computer

Description automatically generated

Figure : ECR Repository for second Lambda Docker Image.

* Lambda function code is mentioned below, which will poll Taxi Order message from SQS(Every 2 min) and send it to SNS Topic(SendMail).

A screenshot of a computer program

Description automatically generated

Figure : Code for (poll\_msg\_send\_mail) lambda function.

* Docker image for second lambda build, tagged and pushed to ECR newly created repository using custom deploy.sh script.

A screenshot of a computer screen

Description automatically generated

Figure : Docker image for second Lambda is build, tagged, and pushed to ECR repository.

* ECR Repository AWS console, second lambda Docker image is successfully pushed.

A screenshot of a computer

Description automatically generated

Figure : Successfully Second lambda docker image is pushed. (AWS console).

* Create second lambda function which will poll the message from SQS queue and pass it to SNS Topic(SendMail).

A screenshot of a computer

Description automatically generated

Figure : Creating Second Lambda function poll\_msg\_send\_mail. (AWS console).

* Configure Event Bridge which will invoke this lambda function every 2 min to check the SQS Queue for message. Expression cron(\*/2 \* \* \* ? \*) will invoke lambda every 2 min.

A screenshot of a computer

Description automatically generated

Figure : Configure AWS Event bridge for invoking lambda every 2 min.

* Final Architecture for second lambda(poll\_msg\_send\_mail) is mentioned below.

A screenshot of a computer

Description automatically generated

Figure : Second lambda(poll\_msg\_send\_mail) AWS console.

* Everything is set up as mentioned in the assignment guidelines. It’s time for testing this HalifaxTaxi cloud serverless architecture.
* Let’s hit the or trigger manually first lambda function(PublishOrder). This function will randomly generate order for taxi. This order will send to SNS topic(PublishOrder) and lead to SQS(HalifaxTaxiQueue) as that queue is subscribed to SNS topic.

A screenshot of a computer

Description automatically generated

Figure : Testing - Triggering first Lambda function to generate order and send to SNS to SQS.

* At this point SQS(halifaxTaxiQueue) will receive random order message in the queue.
* After every 2min, Second Lambda(poll\_msg\_send\_mail) will trigger. This function will poll message form SQS(HalifaxTaxiQueue) and Send that Message to SNS topic(SendMail). As 2 emails are added as subscriber, that emails will receive taxi order details.
* Here, I’m adding cloud watch log image and received email images as a part of testing.

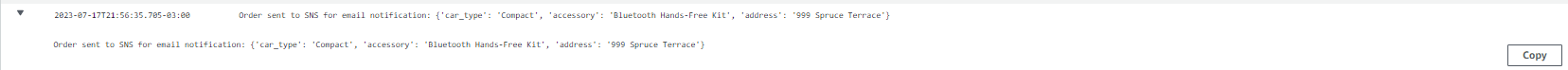


Figure : Message that sent to SNS topic. (Cloud watch AWS console).

* Mail got into my account(Subscribed to SNS topic).

A screenshot of a computer

Description automatically generated

Figure : Got email of with Taxi order Information.

A screenshot of a computer

Description automatically generated

Figure : Taxi Order Information got in mail.