Cloud Developer

Serverless

**Lab Steps**

Task 1: Sign in to AWS Management Console

1. Click on the  button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
   * Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
   * Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.
3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

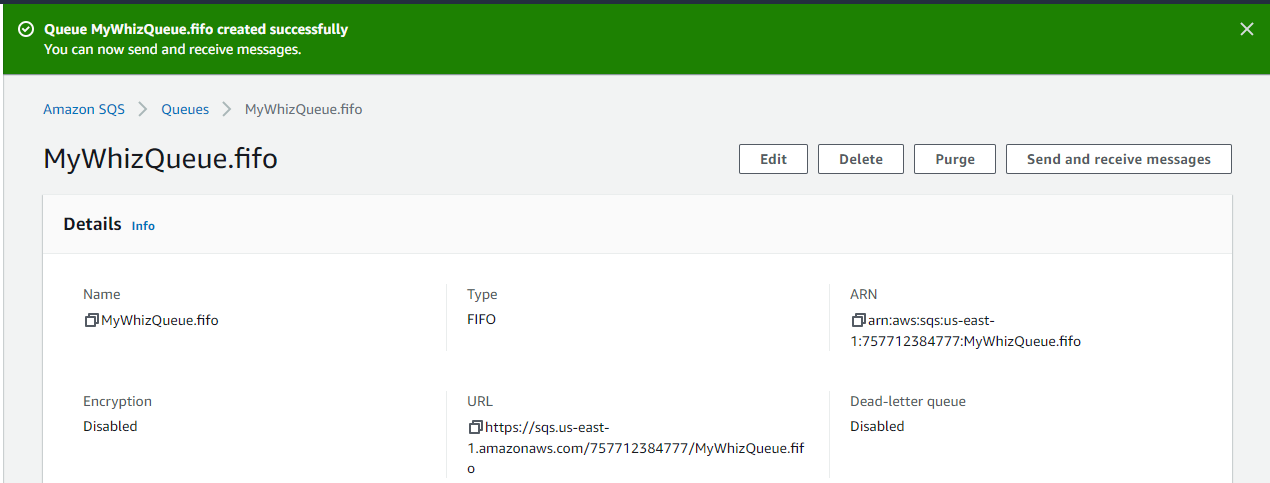
**Note :**If you face any issues, please go through [**FAQs and Troubleshooting for Labs**](https://business.whizlabs.com/labs/support-document/faqs-and-troubleshooting).

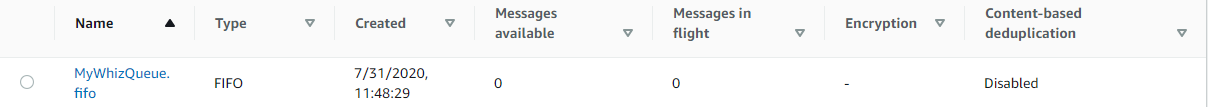
Task 2 : Create FIFO and Standard Queue using the Console

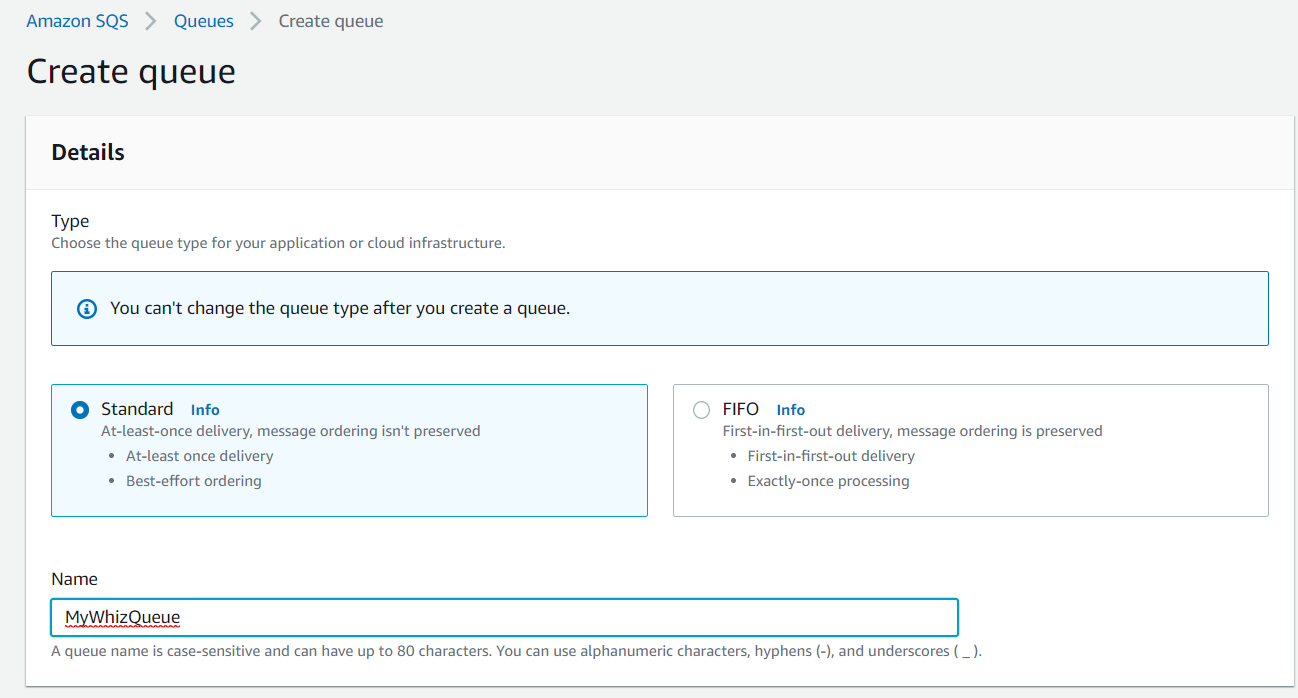
1. Make sure you are in the **N. Virginia**region.
2. Navigate to the Services menu on the top, search for **SQS**and select it.
3. You’ll be redirected to the SQS console page. Click on the 
4. **Details :**
   * Type : Select **FIFO**
   * Name : Enter ***MyWhizQueue.fifo***

* **Note**The name of a FIFO queue must end with the  **.fifo** suffix.

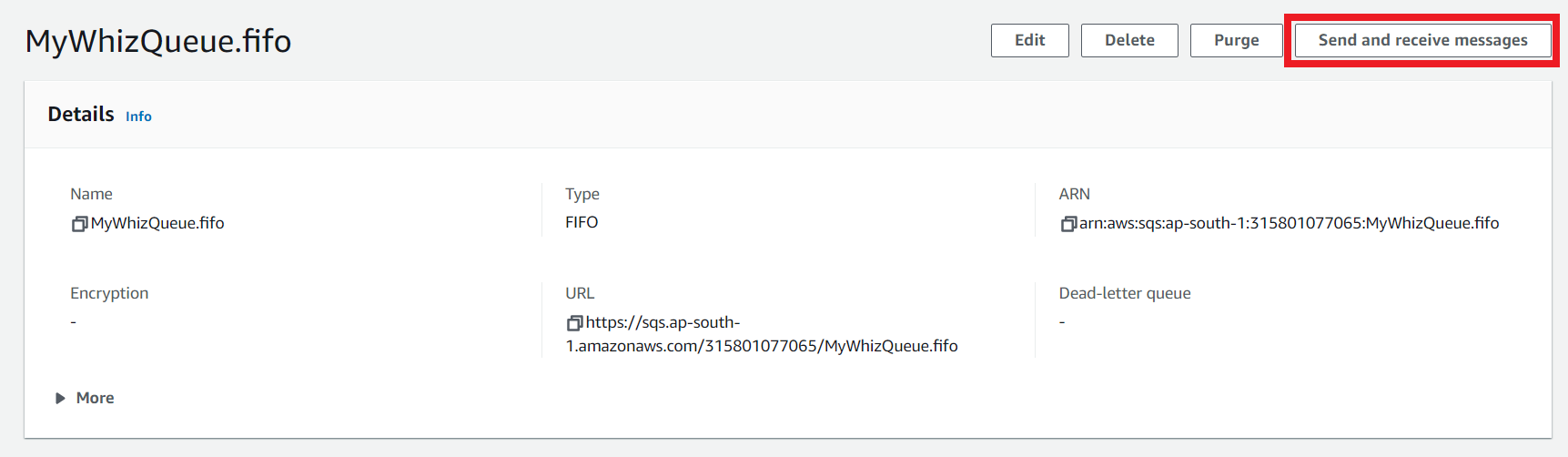
1. Leave everything as **default** and click on 



1. Once you click on **Create Queue**you will get a success message*.* Go back to the queues dashboard, a FIFO queue gets created as given in the image below.  
   
2. We’ll be creating a **Standard Queue**, with all the default options. The only difference is we don't provide the suffix **.fifo** while creating the queue.
3. Click on 
4. **Details :**
   1. Type : Select **Standard**
5. Name : Enter ***MyWhizQueue***



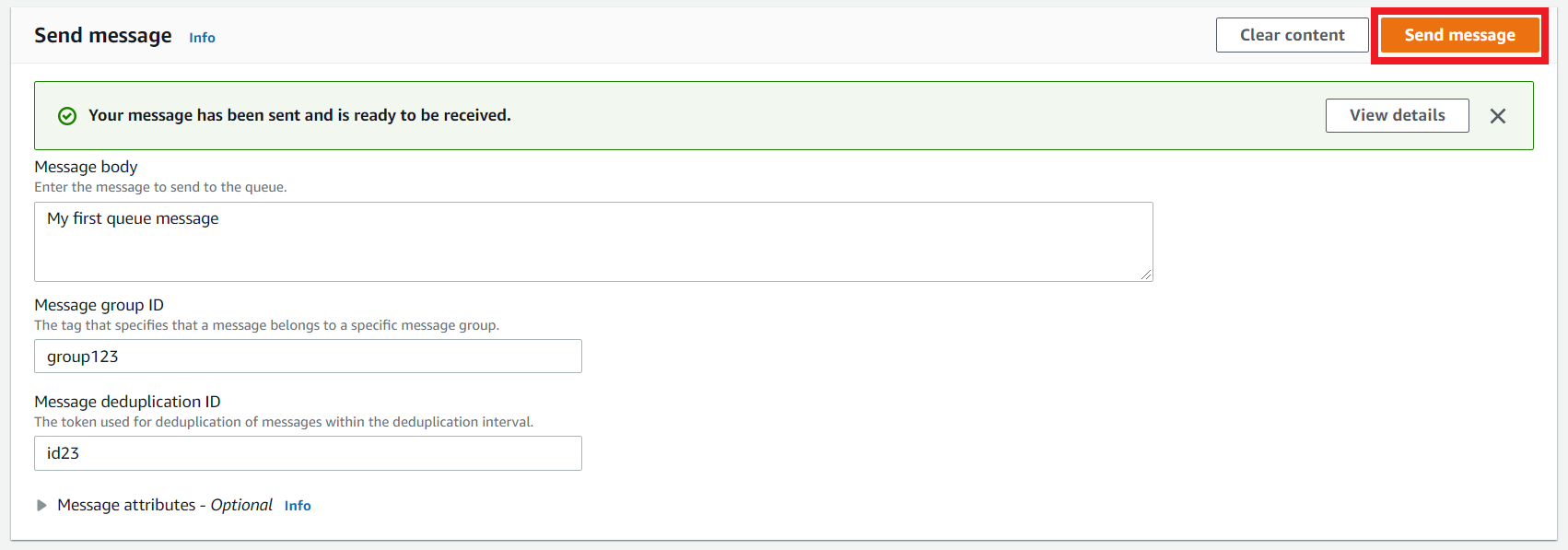
1. The **Type** column helps you distinguish the Standard queue from the FIFO queue at a glance. For a FIFO queue, the **Content-Based deduplication** column displays whether **each of your messages has a unique body** enabled or not.
2. Click on the **MyWhizQueue.fifo** to get detailed information about all the important parameters including **ARN, Name and URL**of the queue.
3. We’ll be now sending a message from our FIFO queue.
4. Click on **Send and receive messages**.



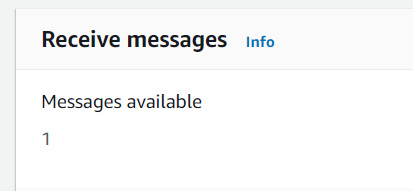
1. Send message

* Message body : Enter ***My first queue message***
* Message group ID : Enter ***group123***
* Message deduplication ID : Enter ***id23***

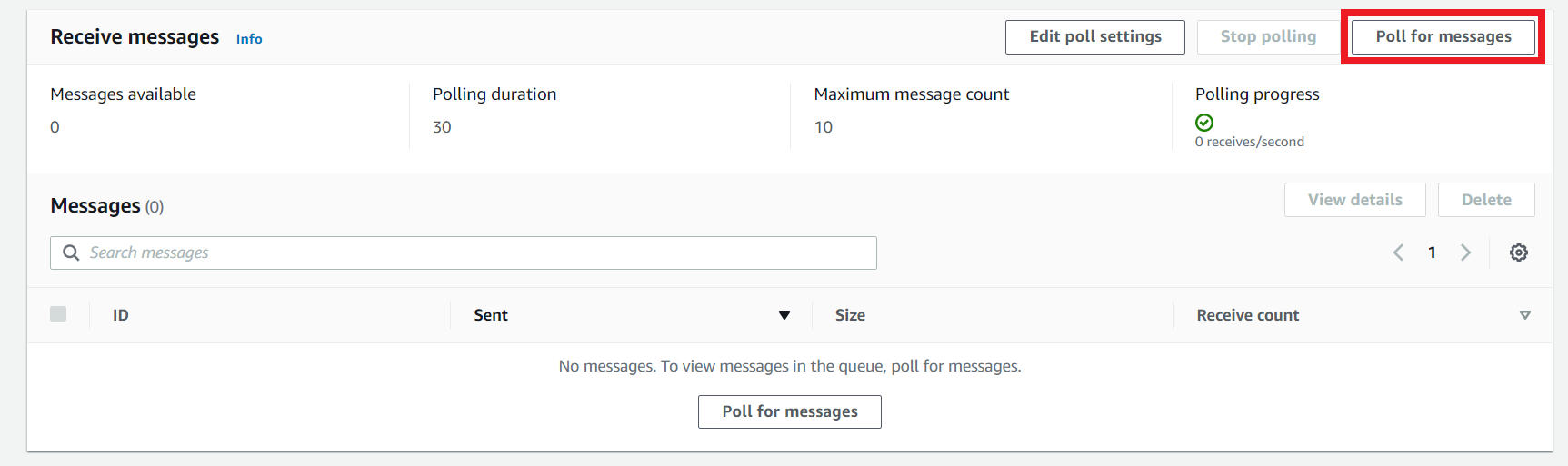
1. Now click on **Send message.**
2. Once you have sent a message you get an acknowledgment.

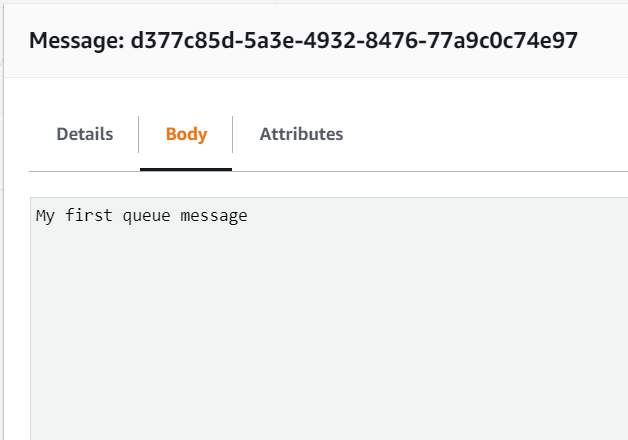


1. After sending the message, we can see that **Message available** changes to 1.

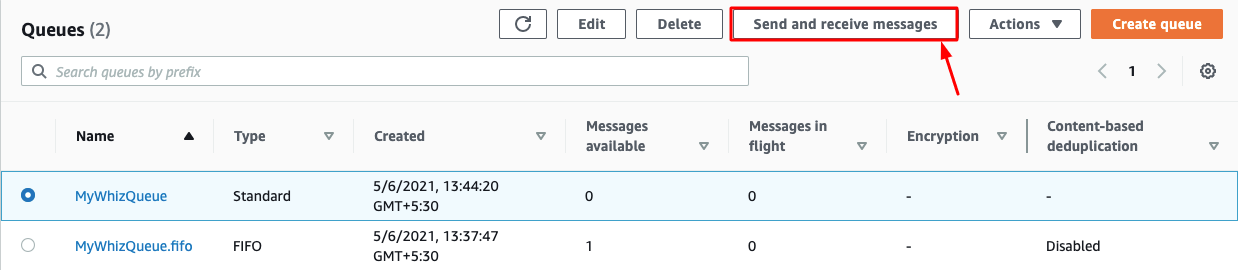


1. Similarly, send few more messages with different **Message body**, **Message group ID** and **Message deduplication**.
2. After sending the messages, **Message available** changes to the number of messages sent.
3. Now, to get all the messages we have sent, click on **Poll for messages** in the **Receive messages section**.
4. We can now see the message column has been filled with the messages.
5. Then click on the message to see its body.





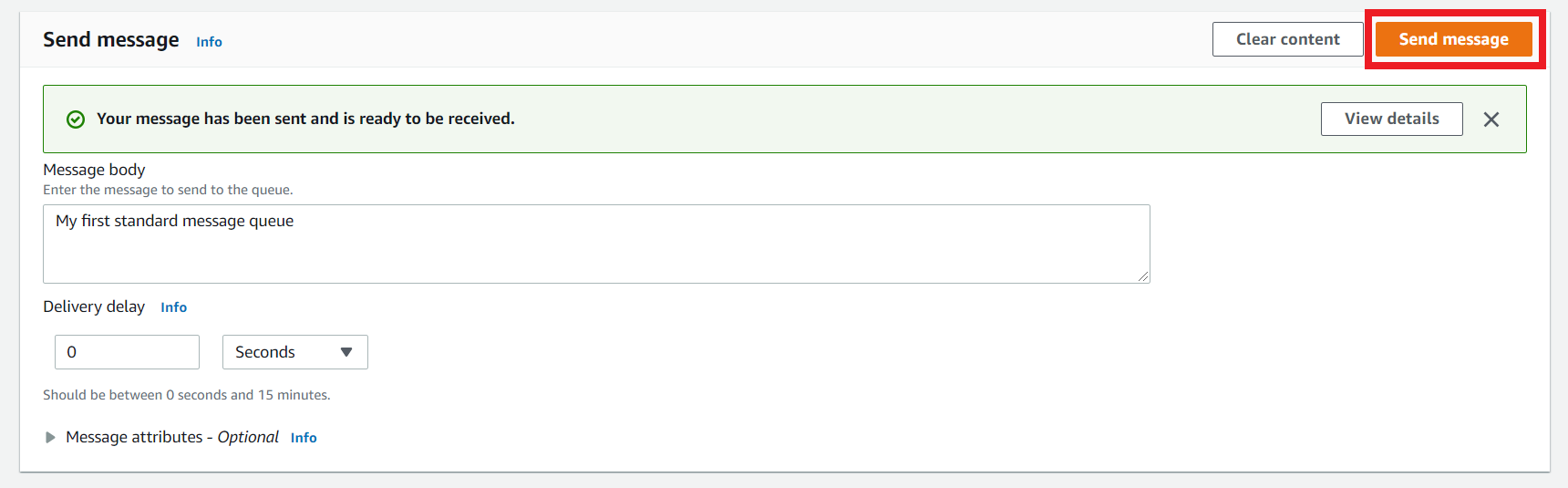
1. **Hence, we have successfully sent and received messages in the console.**
2. Similarly, we’ll try with the **Standard**queue for sending the message.
3. Click on **Send and receive messages**.



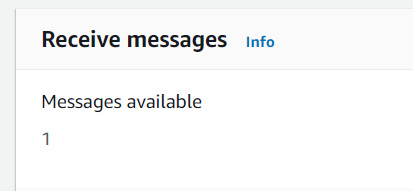
1. Send message:

* Message body : Enter ***My first standard message queue***
* Now click on **Send message**

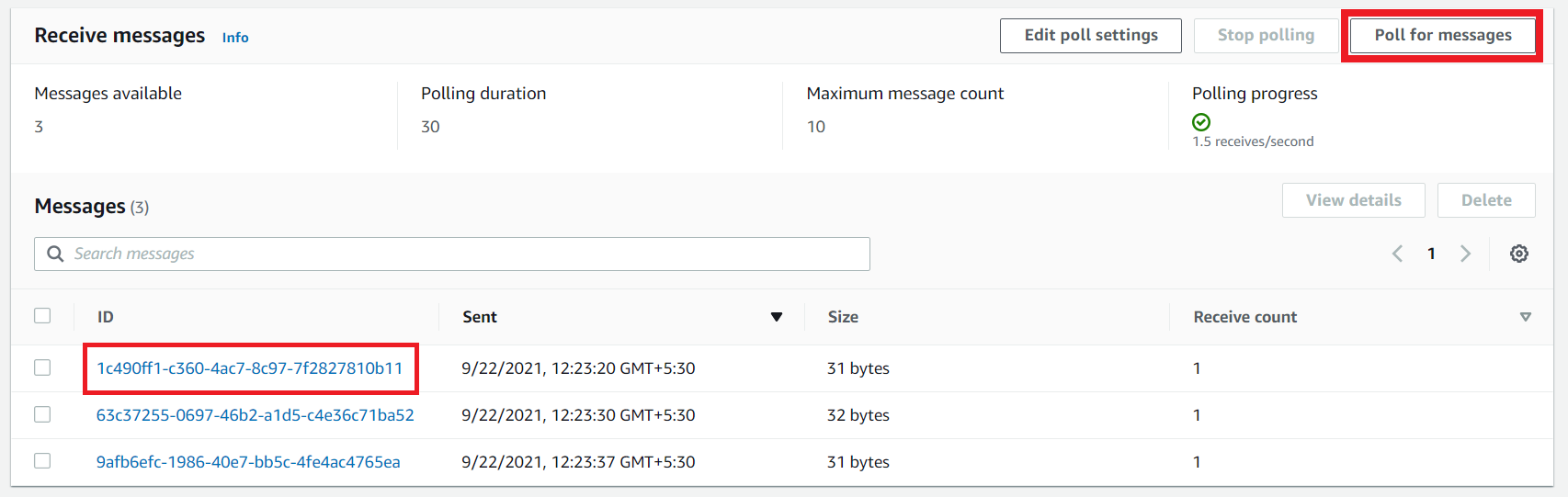
1. Once the message is delivered, you’ll receive an acknowledgment for successful delivery.

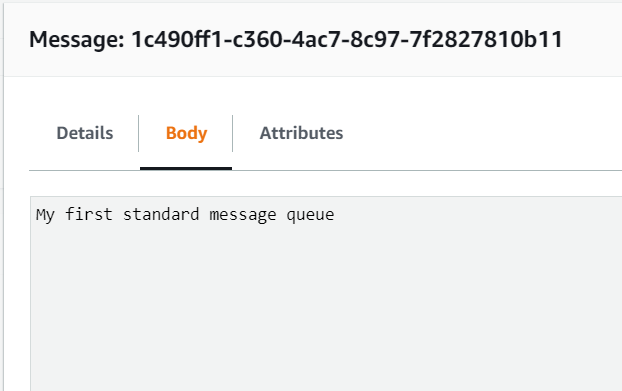


1. After sending the message, we can see that **Message available** changes to 1.



1. Similarly, send few more messages with different **Message body**.
2. After sending the messages, **Message available** changes to the number of messages sent.
3. Now, to get all the messages we have sent click on **Poll for messages** in the **Receive messages** section.
4. Then click on the messages to see its body.



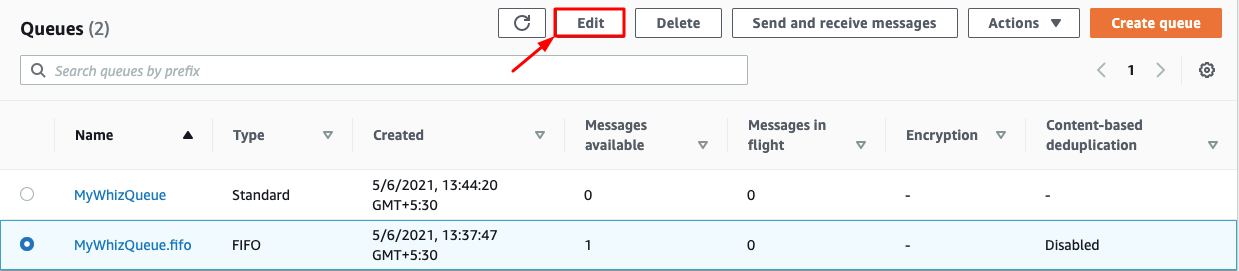


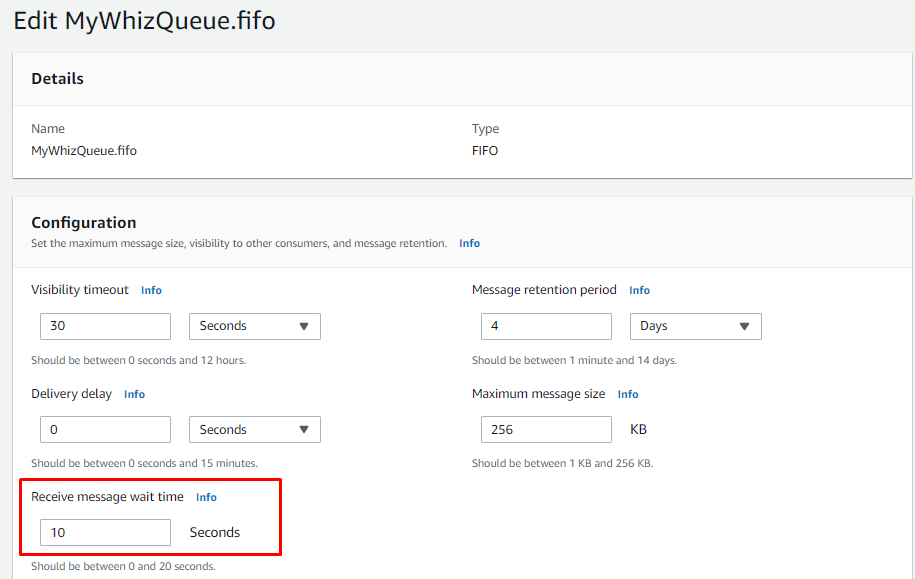
1. **Hence, we have successfully sent and received messages in the console.**
2. Let's try to understand the other parameters like **Long Polling** and how it works in SQS. Before moving to the lab, try to focus on what exactly the Long polling is and when it should be used.

Task 3 : What is Long Polling & Configuring Long Polling

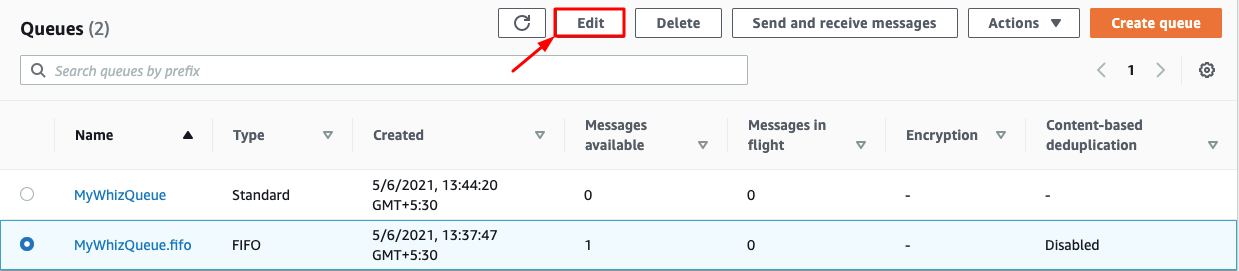
1. **Long Polling:**Let's try to understand how long polling works and why we should use it.

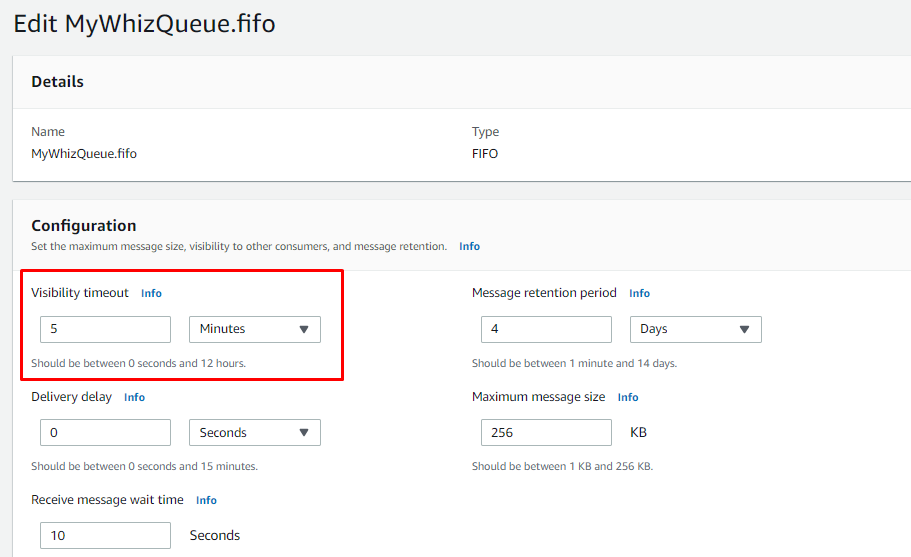
* For example, if our application requires SQS messages, it will call the **RecieveMessage** function in the background. **ReceiveMessage** will check the presence of any messages in the queue and return immediately, with or without messages.
* Calling a **ReceiveMessage** function in our application is fine as per the requirement, but what if the SQS client repeatedly checks for the message in the queue for any new messages. This is a problem, as continuous calls for the **ReceiveMessage** function will require lots of CPU cycles and tie up a thread. In this situation, we'll be using long polling. The only modification that we have to make is to update our **WaitTimeSeconds**argument to 1-20 seconds.
* Now if the queue is empty, the call will wait up to **WaitTimeSeconds**for a message to come in the queue before returning. If messages come in before the timeout the call will return the message right away.

1. Remember if the wait time for the **ReceiveMessage** API action is greater than 0, long polling is in effect. Long polling is cost-effective because you're not constantly polling an empty queue.
2. By default Amazon SQS uses short polling, querying only a subset of its servers (based on a weighted random distribution) to determine whether any messages are available for a response.
3. Let’s try to make this work in our existing queue by configuring Long Polling. Select any queue (standard or FIFO) from the list and click on **Edit** to make changes in our queue. We’ll be selecting the FIFO queue as an example.  
   
4. Once you have selected the **Edit Queue,**update the **Receive Message Wait Time**parameter. It can be any value between 0 to 20 seconds. In our example, we have changed it to **10 seconds**. This will make our long polling come into effect. If we keep the value as 0 or default, it is considered short polling. Once you have made the change, click on **Save changes**.

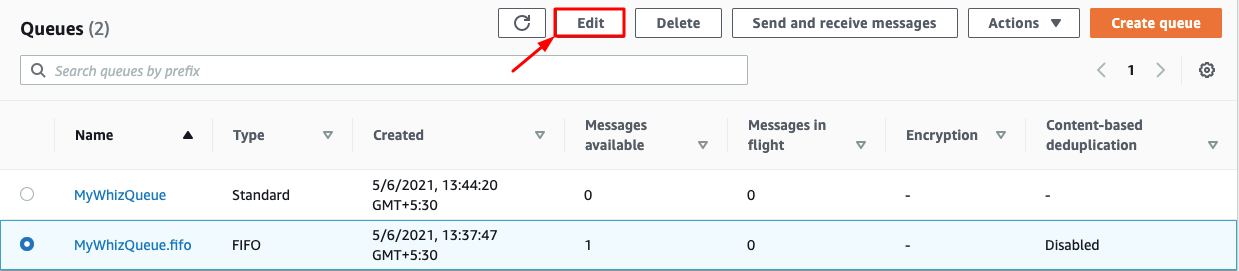


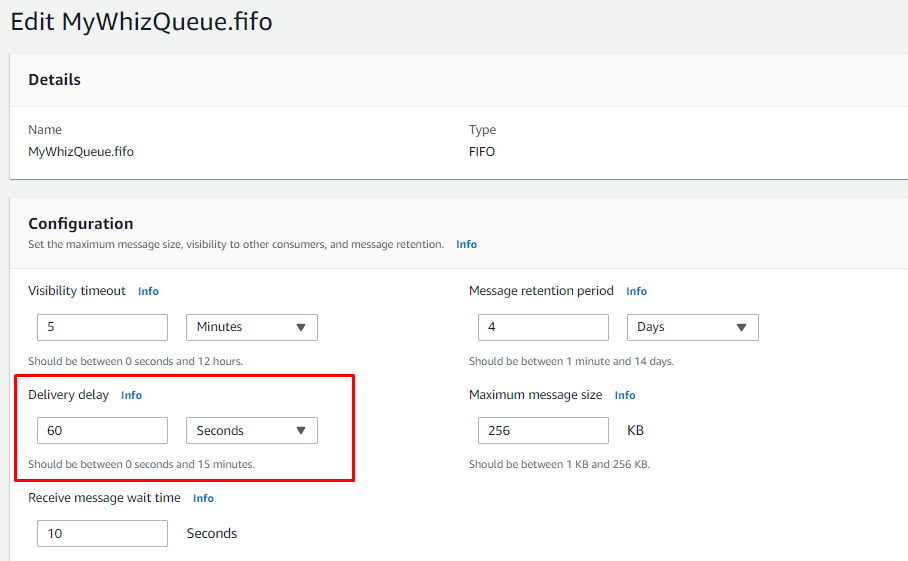
Task 4 : What is Visibility Timeout & Configuring Visibility TimeOut

1. Visibility timeout is a time duration in which AWS SQS avoids the other components from receiving and processing the messages.
2. **Case Study:**
   * Let's try to understand the definition by an example. So if you keep your visibility timeout to 1 minute for a big data job, what's going to happen is the message is going to come back into the queue because it's not going to complete in 1 minute.
   * Let's say it takes five minutes to actually process a large amount of data. The message is going to become visible in the queue and then another EC2 instance will pick it up. Because of this, you could be delivering your messages multiple times because your visibility timeout is too low.
3. Default Visibility timeout is **30 seconds**.
4. Increase it if your task takes > **30 seconds**.
5. Maximum it can go up to is **12 hours**.
6. Let’s try to make this work in our existing queue by configuring the visibility timeout. Select any queue (standard or FIFO) from the list and click on **Edit** to make changes in our queue. We’ll be selecting the FIFO queue as an example.  
   
7. Once you have selected the **Edit,**update the **Default Visibility Timeout**parameter. It can be any value between 0 seconds to 12 hours. In our example, we have changed it to **5 mins**.
8. Once you have made the change click on **Save** at the bottom to bring changes in effect.



Task 5 : What is Delay Queue & Configuring Delay Queue

1. Delay queue allows us to delay/postpone the delivery of a new message for a given period of time. We can easily turn any queue into a delay queue by configuring **SetQueueAttributes**to set the queue’s **DelaySeconds**attribute.
   * If we create a delay queue, any message that has been sent to this queue will remain invisible to the consumer for the configured delay time.
   * To create a delay queue set **DelaySeconds**attribute to a value between 0 to 900 seconds.
2. **Case Study:**
   * Let's try to understand the definition by an example. Consider a case where an application is trying to insert millions of data into a database. The application will send a message about the availability of this new data that has been recently inserted to other subsystems, which in turn process this message and subsequently make updates to the same row. Now there is a dependency that until the first batch completes its job and commits after the updates the next batch should not get triggered. If the message gets triggered before the updates are complete, it would fail the next batch. In this case, delayed delivery helps.
3. Let’s try to make this work in our existing queue by configuring the **Delivery Delay**. Select any queue (standard or FIFO) from the list and click on **Edit** to make changes in our queue. We’ll be selecting the FIFO queue as an example.  
   
4. Once you have selected the **Edit,**update the **Delivery Delay**parameter. It can be any value between 0 seconds to 15 Mins. In our example, we have changed it to **60 secs**.
5. Once you have changed click on **Save** at the bottom to bring changes into effect.

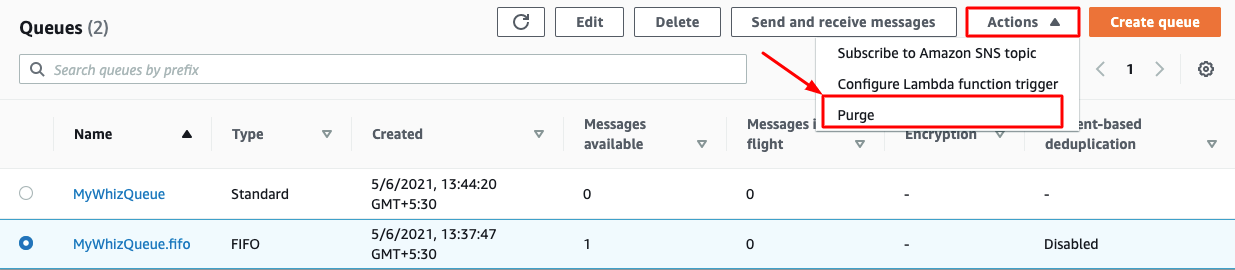


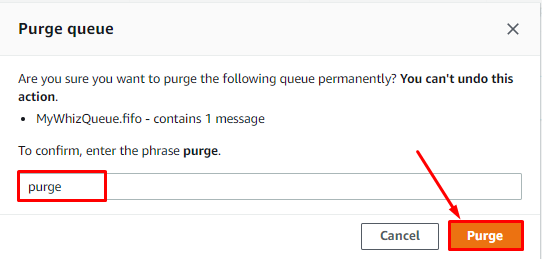
Task 6 : Purge Queue and Verify the same

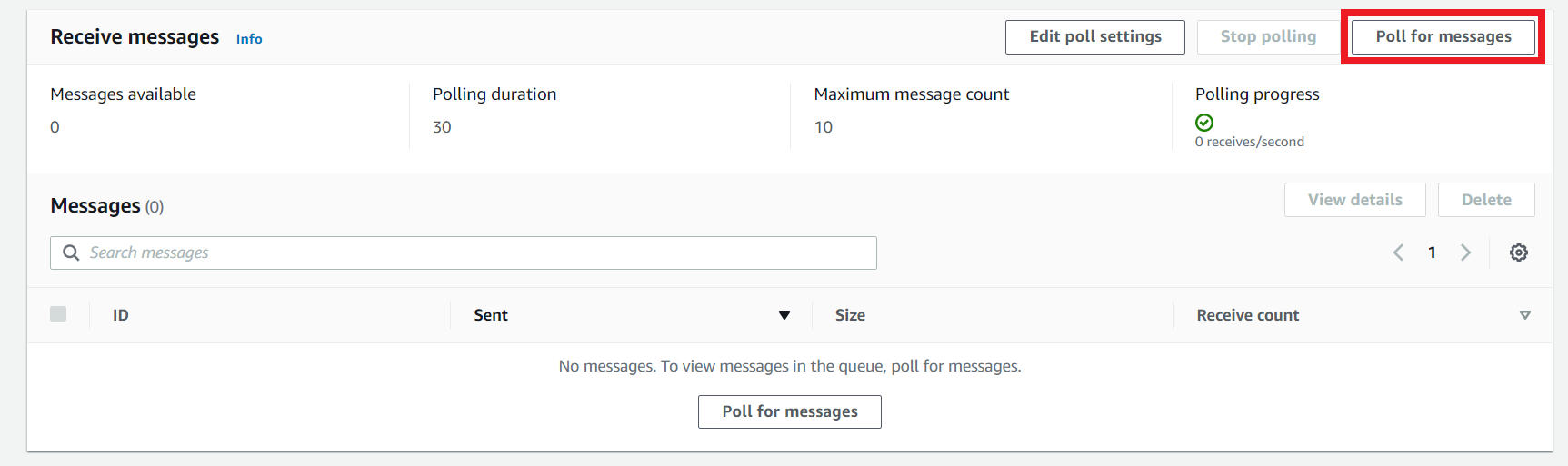
1. In this topic, we’ll try to purge the queue. Before we jump into purging, let's understand first what happens if we purge a queue.

* Purge Queue option allows us to delete the messages in the queue.
* The message deletion process can take up to 60 sec, depending upon the size of the queue.
* **Note**: Once you call the **Purge Queues**action, messages cannot be retrieved from the queue.

1. Let's try to make this change to our existing queue. Click on Queue Option and under **Actions** select **Purge**.

****

1. Once you select the **purge** option it will ask for a confirmation. Enter the phrase **purge** and click on purge.  
     
   
2. To verify if purge has occurred on the fifo queue, click on send and receive messages and then click on Poll for messages.
3. It will show no messages which verify that the purge is successful.



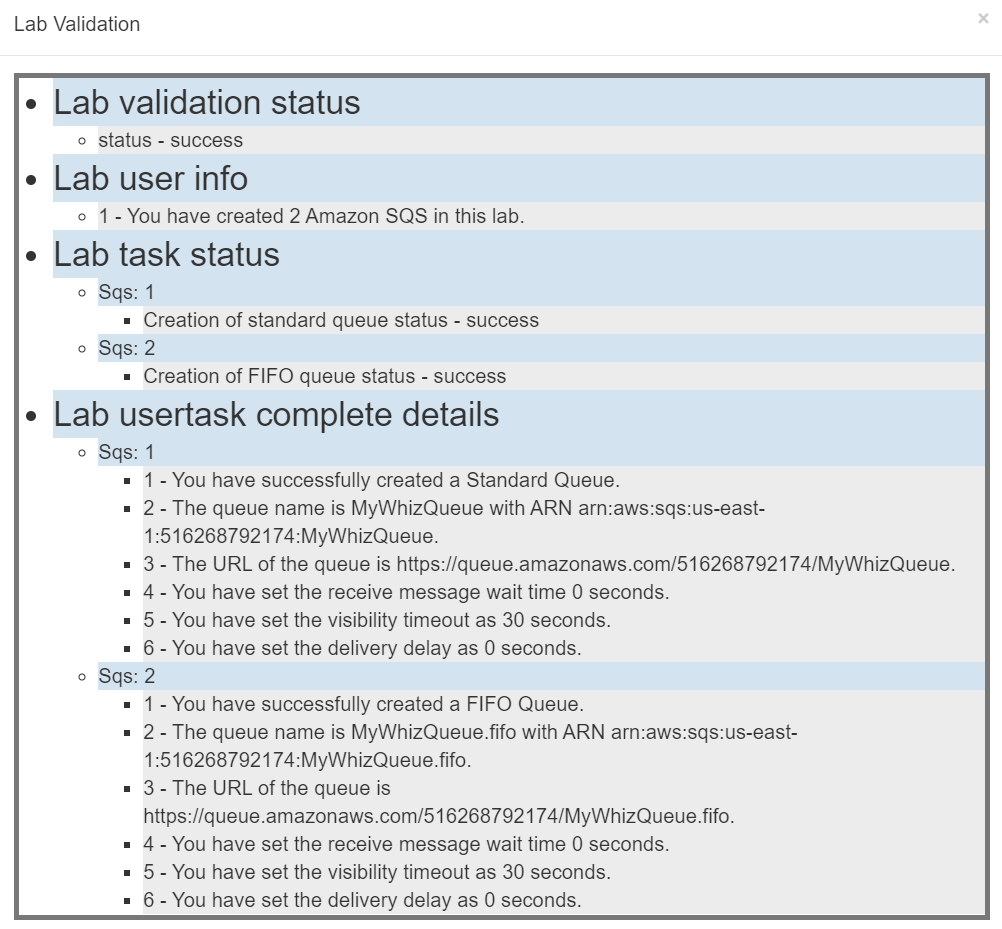
1. Similarly, repeat the purge on the standard queue and verify the same.

Task 7 : SQS points to remember

1. The basic difference between **Delay Queue** and **Visibility Timeout** is **Delay Queue** hides the message when it is first added in the queue whereas visibility timeout hides a message only after the message is retrieved from the queue.
2. **Inflight messages** are messages in SQS that have been received by a consumer but not yet deleted.
3. Max 120,000 messages can stay in the queue.
4. Message size is **256KB**.
5. It has a Default Retention period.
6. SQS is pull-based not push-based.

Task 8 : Validation Test

1. Once the lab steps are completed, please click on the  button on the left side panel.
2. This will validate the resources in the AWS account and shows you whether you have completed this lab successfully or not.
3. Sample output :



**Completion and Conclusion**

1. You have successfully logged into the AWS Management Console.
2. You have successfully created FIFO and standard queue using the console.
3. You have successfully understood Long Polling and configure it.
4. You have successfully understood Visibility Timeout and configure it.
5. You have successfully understood Delay Queue and configure it.
6. You have successfully purged the queue and verify the same.
7. You have successfully understood SQS.

**End Lab**

1. Sign out of the AWS Account.
2. You have successfully completed the lab.