CSC 470 – Special Topics in Computer Science: Cloud Computing

Small Project #4 – Simple Queue Service (SQS)

Objective

Develop a small Java application that utilizes the SQS service to raise your awareness and comprehension of these services and their documented API.

Due Date

This assignment is due by 9:00am, Friday, April 17th, 2015.

Specifics

Create a small Java application, which demonstrates the use of the SQS APIs. Your application should be coded against the AWS SDK for Java API and all required implementation activities should be executed programmatically by your application (the client).

Your solution can be a command line Java application, or a GUI-based Java application; the choice is yours. In either case, the interface that the user (me) uses should allow for me to confirm the execution of the action (PUT object, GET object) step-by-step via the source code, the output, and via the AWS console. That is, your application should not run without pausing allowing me to verify the most recent action. You will need to build in programmatic pauses that are controlled by the user (e.g. "Press return to continue.") so that I may verify each action.

This project will be graded by the amount of demonstrable, functioning features that your implementation incorporates as follows. Bolded items below are actions and should be executed as such. Each item for a grade must be implemented and clearly executing successfully for that grade to be awarded. That is, you cannot receive a "B" grade if you don't implement and successfully implement all three points (Put Object – Copy, Get Service, and Put Object). You cannot receive an "A" unless all of the "B", "C", and "D" items are satisfied.

The table below contains bolded actions. These actions should correspond directly to RESTful[1] operations on SQS. However, you should use the AWS SDK for Java to form proper requests of the service (allowing for the API to do the work for you to make the REST call to the SQS service). Thus, you'll want to understand the REST operation and refer to the API documentation to determine how make a similar call to the service via the API. Note that the sample code in the SDK for Java download will be a helpful resource.

[1] See http://en.wikipedia.org/wiki/Representational_state_transfer for more information on REST.

Grade / # Points	Implementation Required *(higher grades must include required implementation from lower grades)*

A (4)	• **SetQueueAttributes** – Change the following attributes on the 'csc470test' queue as follows:
	  ○ Set the queue's visibility timeout to 60 minutes
	  ○ Set the maximum message size to 1024 KiB
	  ○ Set the message retention period to 2 days
	  ○ Set the receive message wait time attribute to 15 seconds.
	• **PurgeQueue** – Purge all remaining messages in the 'csc470test' queue.

B (3)	• **SendMessage** – Send five messages to the 'csc470test' queue. The messages should be five famous quotes of your choice. Include the quote's originator in the message. With each submission, print the message's ID (returned from the request). • **ReceiveMessage** – Issue three receive message requests on the 'csc470test' queue and print the message returned. If no message is returned by a request, print an informative string to the user indicating that the queue returned no message. • **DeleteMessage** – For each message returned above (ReceiveMessage), immediately delete it from the queue. Print a confirmation message on the deletion of the message from the queue.

C (2)	• **GetQueueAttributes -** Fetch and print (individually with labels) the following attributes of the 'csc470test' queue:
	  ○ The creation time of the queue
	  ○ The ARN of the queue
	  ○ The message retention period (properly labeled with a time label)
	  ○ The time the queue was last modified
	  ○ The current approximate number of messages
	• **DeleteQueue** – Delete the 'csc470test2' queue.

D (1)	• **CreateQueue** - Create two message queues in the SQS service named 'csc470test' and 'csc470test2'. • **ListQueues** - List the queues associated with your account.

F (0)	Fails to execute, fails to compile, no working functionality

Finally, change the IAM group policy for my user account (which you created in Project 1) so that I can use the AWS console to confirm all of the above actions. Be sure to test my user account by creating a new user account with the same IAM group policy so that you can see what I see when using the console.

These actions will be verified by source code and user display/review from the application.

Resources

- AWS SDK for Java - http://goo.gl/de5Hby
- AWS SDK for Java API Reference - http://goo.gl/4WZO1
- SQS Developer Guide - http://goo.gl/Bn8qaD
- SQS API Reference - http://goo.gl/ZvUj3F

Delivery

- Your delivery of this assignment should be in the form of a Java jar file that contains both your source code and .class files. Include in your deliverable a valid working Maven pom file that addresses the project dependencies, handles possible compilation and execution.
- Upload only your jar file as a deliverable to the corresponding assignment in Canvas.