CSC 470 – Special Topics in Computer Science: Cloud Computing

Small Project #3 – AWS EC2 and Simple Email Service (SES)

Objective

Develop a small Java application that utilizes various EC2 and SES services and features to raise your awareness and comprehension of these services and their documented API.

Due Date

This assignment is due by 9:00am, Friday, March 6th, 2015.

Specifics

Create a small Java application, which demonstrates the use of the EC2 and SES 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 EC2 and SES. 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 EC2 and SES 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

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

SDK for Java download will be a helpful resource.

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

A (4)	• ***SES***
	  ○ **DeleteVerifiedEmailAddress -** In your program, prompt the user to enter an email address and then submit that address to be deleted/removed from the verified sender list for your account.
	• ***EC2***
	  ○ **TerminateInstances** – Allow the user to select an instance to terminate (I'll select the one I launched below) and submit the termination request.

B (3)	• ***SES***
	  ○ **SendEmail** - In your program, prompt the user to enter any CC and BCC fields (support multiple of each) and send the same message as below (**including** the TO field) to all of the addresses.
	  ○ **VerifyEmailAddress** - In your program, allow the user to enter a new email address and submit this address to be verified for your account. (Tip: mailinator.com is a fine resource for throwaway email addresses.)
	• ***EC2***
	  ○ **DescribeInstances** – obtain and print the instance description as specified by the user. That is, allow the user to select or enter the instance for which the description is fetched (and then printed). Print all information that is returned. You may limit this query to a specific region, if necessary. However, I am looking to see the instance that I launched (see below).

C (2)	• ***SES*** ○ **SendEmail** - In your program, prompt the user to enter a destination email, and then programmatically send an email with the subject "Hello from SES" and the body of the message "Greetings <entered email address>, from Amazon SES!" ○ In your program, query the SES to obtain and display the following data of your account. Be sure to display all returned data. • the current sending quota (**GetSendQuota**), and • the current sending statistics (**GetSendStatistics**) • ***EC2*** ○ **RunInstances** – start a micro instance of your selection (t1 or t2) on EC2. To achieve this, you likely will need to set the region, availability zone, security group, and more as discussed in class. If possible (and if it works with the other EC2 commands above), allow the user to select the AWS region and availability zone. The command should result in the instance being launched, and the output should (at the least) indicate the public IP address of the instance.
D (1)	• ***SES*** ○ Use the AWS console to submit a request (and complete the approval process) to have your @tcnj.edu email address verified as an identity (allowing emails to be sent). We will add other addresses later in this assignment (see above). This step should be completed in advance of submitting your project. ○ **ListVerifiedEmailAddresses** - In your program query the SES to obtain and display a list of verified email identities.
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. That is, I should be able to (using my AWS credentials through your account) verify your SES verified addresses, see the sending statistics, observe the sent message count,

and see that a verified email address will be deleted. Also, for EC2, ensure that I can verify the launching, existence, and termination of the instance I launch in the commands above.

Your EC2 instance should at the least support ssh (port 22) open, so that I can attempt to connect to it. I will not log in, and I am not asking for the sharing of private keys for the instance. However, I will attempt to log in to ensure that the machine is running.

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
- SES Developer Guide - http://goo.gl/5krBR
- EC2 API Reference - http://goo.gl/H4toi
- SES API Reference - http://goo.gl/VmPNxS

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.