Overview

We demonstrate how to deploy Redis containers in the class. However, we did not really configure and use Redis. This final challenge is to build and test Redis deployment. You will have to work through some of the detailed steps to complete the requirements. If you get stuck on a step, do a little research on the documentation site or review previous assignments and exercises.

Let's start and finish strong!

First, create a new GitHub Classroom repository by clicking on this link: https://classroom.github.com/a/SbyfhlfR

Launch a Jenkins stack in the **us-east-1** region using the template: **jenkins-cf.json**

Create a new Jenkins pipeline job called `final` and configure it to use a Jenkinsfile located in your Classroom repository. Note, you do not need to setup a pipeline build trigger or GitHub webhook for this.

Update a CloudFormation template

- Download the **docker-single-server.json**. Modify this template to install redis-tools on the docker1. (Hint: run 'apt install redis-tools -y' in the UserData section)
- You will use this updated template later.

Your Jenkinsfile will have 4 stages:

Let's look at the requirements for each of these stages.

Stage 1: Create a Test Stack

- Create a CloudFormation stack called **final-test** using the updated docker-single-server.json template (IP (Jenkins slave's public IP), region (us-east-1))
- Wait for the stack creation process to complete
- Describe the stack using 'aws cloudformation describe-stacks'. You need to specify the stack name, and its region.
- The output of the docker-single-server.json template is the IP of a Docker host (docker1). Get that IP and run the uptime command on docker1 over SSH. You will need to setup SSH Agent on Jenkins. Here is an example of adding a private key to Jenkins credential store (https://support.cloudbees.com/hc/en-us/articles/222121807-Using-an-SSH-agent-to-authenticate-SSH-connections)

Stage 2: Deploy Redis Standalone

- Run a container on docker1 using the image redis:latest. The container is running in a detached mode and mapping its port 6379 to host port 6379.

Stage 3: Test Redis Standalone

- Run the following two commands over SSH on docker1 to confirm that Redis is working:
- 'redis-cli set hello world'
- 'redis-cli get hello'

Stage 4: Delete Test Stack

- Delete the final-test stack and wait for the stack deletion to complete.

Console Output

Once you have the pipeline working properly, copy the console output from the last successful build into a file called `console.txt` and check it into the GitHub repository.

Check your work

Here is what the contents of your git repository should look like before final submission:

- Jenkinsfile
- docker-single-server.json
- console.txt

Terminate all resources

The last step is to terminate your CloudFormation stacks on AWS.

Submitting your assignment

I will review your work on GitHub after the due date.

Thank you! Wish you all the best in your future!