Overview

In this final challenge, you will create a code pipeline to build a website in Docker images and deploy the website to a production environment. This challenge is based on past assignments, exercises, and hands-on activities. 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 AWS documentation site or review previous assignments and exercises.

Create a GitHub repo for the Final

Click the link below to accept a repo from GitHub Classroom.

https://classroom.github.com/a/HSxmoQkP

You need to add/update the following files into the repo.

1. Update html/index.php

Replace [YOUR NAME HERE] with your name.

2. Add a Dockerfile

Let's build a simple apache+php webserver container using a Dockerfile. Create the Dockerfile in the root directory in the repo. Here are the configuration requirements for the Docker image:

- The new image must be based off the `php:7.2-apache` image.
- Set the maintainer of the image to your name and email address.
- The image should expose port 80.
- Copy the file from the html directory to the /var/www/html directory in the container.

3. Update buildspec.yml

We won't manually build a docker image in this project. Instead, we will build it in the build stage of a code pipeline. Replace [TO-DO] in the buildspec.yml with a command to create a new Docker image called phpweb with a tag of 1.0 using the Dockerfile.

4. Add a Docker Compose File

Create a basic Docker compose file called docker-compose.yaml in the root directory in the repo to launch the finalweb1 container with the following configuration settings:

- The name of the service should be finalweb1.
- The container should map port 80 within the container to port 80 on the host server.
- Use the image phpweb with a tag of 1.0

5. Update scripts/start_server.sh

In the deployment stage, the code pipeline will follow instructions in appspec.yml to install and configure applications. Please update start_server.sh in the scripts folder. Replace [TO-DO] with the docker-compose command to create all of the container services defined in the docker-compose.yaml in a detached mode.

6. Update final-docker-pipeline-template.json

Replace the AMI ID with the AMI used in docker-single-server.json. You also need to finish the projectURL in the output section. This output needs to refer GitHubUsername and GitHubRepositoryName in the parameter section.

Create the Final Website

Create a CloudFormation stack with final-docker-pipeline-template.json. Note that parameter "GitHubUsername" is UST-SEIS665. Show me your work to get the full grade!

Check your work

If you follow the instructions, you repo should have

- Two folders: "html" and "scripts".
- One index.php in html directory.
- Three scripts in the scripts directory
- final-docker-pipeline-template.json, Dockerfile, docker-compose.yaml, buildspec.yml, appspec.yml, and README.md

Remember to Terminate Resources. THANK YOU!