CS312 :: Homework 4

Follow the instructions below to create a simple pair of Docker containers. This will give you lots of experience with Docker itself, as well as provide chances to practice some other fundamental skills, such as shell scripting, understanding of network ports, `vi`, basic knowledge of HTTP, HTML, Javascript, and NodeJS, and version control.

To get started in with this homework, follow these instructions first, then answer the questions below.

Setup and Study Tasks

1. Create a user account on hub.docker.com. Your Docker account username will be used below where <USERNAME> appears.
2. Create 2 public repositories called
    a. cs312hw4_webserver
    b. cs312hw4_webclient
3. Create 2 new images using Dockerfiles and store the Images on Docker Hub in the repositories you just created. Here are the details on the two new images:
    a. Image 1: a webserver
        i. Built from the alpine:latest image.
        ii. Tag the final gradable version with: "<USERNAME>/cs312hw4_webserver:1.0".
        iii. This container operates as a NodeJS web server as demonstrated in lecture. Hardcode the port it will use as 8080, so that the other container can access it easily.
        iv. When requested to produce the default web page, it must return the Dockerfile used to produce this Image. You could, for example, use the Dockerfile COPY command to import the Dockerfile, and then figure out a way to send it using the response.end() JS command, or maybe just hard-code the Dockerfile into a string in your NodeJS server code. The second way is inferior but allowed. *Note that because your container returns the Dockerfile here, you do not otherwise submit the Dockerfile as part of this assignment.*
    b. Image 2: a webclient
        i. Built from the alpine:latest image.
        ii. Tag the final gradable version with: "<USERNAME>/ cs312hw4_webclient:1.0".
        iii. When it starts up, the client simply performs this loop until it is killed:
            1. Prints a line that only has a timestamp on it (whatever form you want this to take is fine, but it needs to be recognizable as the current time)
            2. Requests the default web page from the web server using `curl` and prints it.
            3. Wait 3 seconds, then repeat back to step 1.
4. You will probably need to set up some sort of network, link, or service to make these two containers be able to see each other. To test that this is working, see Question 2, below.

Questions

Submit your answers as a .PDF on Canvas. Note that you are NOT submitting your Dockerfiles, code, Images, or Containers, but instead are providing instructions that the TAs can follow that allow them to retrieve and run your images via standard docker commands (see below).

1. What is your Docker Hub username? *(1 point)*
2. Write up a simple, short set of instructions that the TAs can follow that include docker commands that show how your webclient can get data from your webserver. This will probably involve starting a Docker network, link, or service of some kind, then a few "docker run" commands to get the data from your Docker Hub account and display the looping results on the screen.

   By following your instructions, the TA should see on their screen a loop that alternately prints out a timestamp and then your Dockerfile, as indicated in Setup and Study Tasks.

   Note that your instructions must include the usage of these docker flags in the docker commands:

   ~Webserver: **`-d`** *and* **`--rm`**
   ~Webclient: **`--rm`** *and* **`-it`**

   (39 points for appropriate contents)