Basic Assignment

NOTE: Do not forget to see manual page using "--help" option in command when searching for options/commands for a particular task.

1. Install Docker, either on your native OS or on a VM. Make sure it runs. type "docker -v" to check if it's installed.

try below commands for help

docker --help ---> This command shows all available options and commands to work with images and containers

docker images --help ---> This command shows all the avaialble options and commands to work with docker images

docker ps --help ---> This command shows all the avaialble options and commands to work with docker containers

2. Find a image from dockerhub of your choice(recommeded: nginx), don't use browser, pull the official image from dockerhub

ANS- C:\Users\Sieanna> docker pull nginx

3. List all the available images in your machine/vm, make sure you see recently pulled image in the list.

ANS- C:\Users\Sieanna> docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

myimage 1.0 16bdf41412d4 3 days ago 103MB

<none> <none> 2a8693d8e2e7 3 days ago 103MB

redis latest ddcca4b8a6f0 2 weeks ago 105MB

nginx latest dd34e67e3371 2 weeks ago 133MB

ubuntu latest 1318b700e415 5 weeks ago 72.8MB

hello-world latest d1165f221234 6 months ago 13.3kB

jenkins 2.60.3 cd14cecfdb3a 3 years ago 696MB

4. Find out the "Full" ImageId of the image that you pulled and write it below.

ANS- C:\Users\Sieanna>docker images -q

dd34e67e3371 (nginx image id )

5. Create a container of your image

ANS- C:\Users\Sieanna>docker run --name mynginx -it nginx

6. List all the running containers

ANS-

C:\Users\Sieanna>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

870076ef063e nginx "/docker-entrypoint.…" About a minute ago Up 50 seconds 80/tcp mynginx

7. List all the running and stopped containers

ANS-

C:\Users\Sieanna>docker container ls -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

681a21478b2b hello-world "/hello" 5 minutes ago Exited (0) 5 minutes ago MyImg

870076ef063e nginx "/docker-entrypoint.…" 40 minutes ago Up 40 minutes 80/tcp mynginx

8. Find out the "Full" containerId of the container and write it below.

ANS –

C:\Users\Sieanna>docker ps -aq

681a21478b2b

870076ef063e

9. Find out how many image layers are used to build this image.

ANS-

C:\Users\Sieanna>docker history nginx

IMAGE CREATED CREATED BY SIZE COMMENT

dd34e67e3371 2 weeks ago /bin/sh -c #(nop) CMD ["nginx" "-g" "daemon… 0B

<missing> 2 weeks ago /bin/sh -c #(nop) STOPSIGNAL SIGQUIT 0B

<missing> 2 weeks ago /bin/sh -c #(nop) EXPOSE 80 0B

<missing> 2 weeks ago /bin/sh -c #(nop) ENTRYPOINT ["/docker-entr… 0B

<missing> 2 weeks ago /bin/sh -c #(nop) COPY file:09a214a3e07c919a… 4.61kB

<missing> 2 weeks ago /bin/sh -c #(nop) COPY file:0fd5fca330dcd6a7… 1.04kB

<missing> 2 weeks ago /bin/sh -c #(nop) COPY file:0b866ff3fc1ef5b0… 1.96kB

<missing> 2 weeks ago /bin/sh -c #(nop) COPY file:65504f71f5855ca0… 1.2kB

<missing> 2 weeks ago /bin/sh -c set -x && addgroup --system -… 63.9MB

<missing> 2 weeks ago /bin/sh -c #(nop) ENV PKG\_RELEASE=1~buster 0B

<missing> 2 weeks ago /bin/sh -c #(nop) ENV NJS\_VERSION=0.6.1 0B

<missing> 2 weeks ago /bin/sh -c #(nop) ENV NGINX\_VERSION=1.21.1 0B

<missing> 2 weeks ago /bin/sh -c #(nop) LABEL maintainer=NGINX Do… 0B

<missing> 2 weeks ago /bin/sh -c #(nop) CMD ["bash"] 0B

<missing> 2 weeks ago /bin/sh -c #(nop) ADD file:87b4e60fe3af680c6… 69.3MB

10. Get the Apache Tomcat 7 server image from the docker hub.

ANS- C:\Users\Sieanna>docker pull tomcat:7.0.57-jre8

11. Run the Apache Tomcat 7, I mean create a container of Apache Tomcat.

ANS- C:\Users\Sieanna>docker run --name MyTomCat -p 9090:9090 -p 50000:50000 tomcat:7.0.57-jre8

12. Find out what is the IP Address of the Apache Tomcat Container that it is running on

ANS-

C:\Users\Sieanna> docker inspect <container id>

C:\Users\Sieanna>docker inspect e32489eed508

"IPAddress": "172.17.0.2"

13. Which Port it is using?

ANS- 9090

14. Try to access the Tomcat's home page from your machine/vm.

ANS-

By using docker play ground

Docker run -it –rm -p 8888:8080 tomcat:8.0

15. What is the disk size of Apache Tomcat image?

ANS-

C:\Users\Sieanna>docker container ls -s

OR

C:\Users\Sieanna>docker system df -v

CONTAINER ID IMAGE COMMAND LOCAL VOLUMES SIZE CREATED STATUS NAMES

e32489eed508 tomcat:7.0.57-jre8 "catalina.sh run" 0 38.5kB About an hour ago Up About an hour MyTomCat

16. Find out list of all environment variables that is configured for tomcat image, can you see JAVA\_HOME and CATALINA\_HOME? What did you notice about it?

17. Find out which port is exposed for tomcat?

ANS- after command on Q14 tomcat is running on 8888 port

18. Run multiple conntainers of tomcat on different port and access it's home page.

19. Pull ubuntu os from dockerhub, try to pull 2 images of ubuntu, Except the latest one.

ANS-

C:\Users\Sieanna>docker pull ubuntu:xenial

C:\Users\Sieanna>docker pull ubuntu:rolling

20. Run the container of ubuntu in attached mode.

ANS- C:\Users\Sieanna>docker run -it ubuntu:xenial

21. Run the container of another ubuntu in detached mode.

ANS- C:\Users\Sieanna>docker run -d ubuntu:rolling

22. Check how many ubuntu containers are running and stopped

ANS- C:\Users\Sieanna>docker ps -a

23. Is the tomcat container running? If no, start one.

ANS- C:\Users\Sieanna>docker start MyTomCat

24. Check the logs, generated by tomcat container(don't forget to make request to tomcat's home page to see the log).

ANS- docker logs MyTomCat

25. Check if ubuntu conatiner is running? If no, start one in attached mode to the terminal.

ANS- C:\Users\Sieanna>docker run -it ubuntu:xenial

26. Login as root user in ubuntu container

ANS-

C:\Users\Sieanna>docker exec -it 14ea9b45bcd1 /bin/bash (exec use to show path)

root@14ea9b45bcd1:/# echo "hello from the terminal" (login as root user)

hello from the terminal

27. Create a file with any name in root directory

ANS-

root@14ea9b45bcd1:/# touch test

root@14ea9b45bcd1:/# ls

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys **test** tmp usr var

28. Install software of your choice in ubuntu container using "apt-get install"

ANS-

root@14ea9b45bcd1:/# sudo apt-get install nginx

29. Now exit the ubuntu shell, are you back to your host machine, if not, come back to the host machine.

ANS-

root@14ea9b45bcd1:/# exit

exit

30. Check if the ubuntu container is running.

ANS-

C:\Users\Sieanna>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

e32489eed508 tomcat:7.0.57-jre8 "catalina.sh run" 2 days ago Up About an hour 0.0.0.0:9090->9090/tcp, :::9090->9090/tcp, 8080/tcp, 0.0.0.0:50000->50000/tcp, :::50000->50000/tcp MyTomCat

31. Create a new ubuntu container out of the same image as that previous container in attached mode.

ANS-

C:\Users\Sieanna>docker run -it ubuntu:xenial

root@76d64856cb61:/#

32. Login as a root user

ANS- C:\Users\Sieanna>docker exec -it 76d64856cb61 /bin/bash

root@76d64856cb61:/#

33. Check if you can see the file created in previous container, you will not see the file as well as software that you installed in the previous container. Now kill this Container.

ANS-

**There is no such file called test and previously installed software.**

C:\Users\Sieanna>docker exec -it 76d64856cb61 /bin/bash

root@76d64856cb61:/# ls

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

**Deleting/Killing container**

C:\Users\Sieanna>docker rm -f 76d64856cb61

76d64856cb61

34. Do you have the previous ubuntu container where you created the file and installed the software? If no reapeat step 25 to 29.

ANS-

35. Create an Image out of the existing container.

ANS- We have existing container 14ea9b45bcd1

* C:\Users\Sieanna>docker commit 14ea9b45bcd1 salonivichare/ubuntu:xenial (ubuntu:xential is ubuntus version)
* C:\Users\Sieanna>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

C:\Users\Sieanna>docker images (u can see new created image )

REPOSITORY TAG IMAGE ID CREATED SIZE

**salonivichare/ubuntu xenial bee8045d55f1 17 seconds ago 135MBC:\Users\Sieanna>docker images (u can see img1 is created )**

36. Now Create a Container out of this image and login into it to see if you can see the file and software installed by you in the previous container.

ANS-

C:\Users\Sieanna>docker run -it --name Saloni salonivichare/ubuntu:xenial

root@5481c474bd29:/# ls

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys **test** tmp usr var

37. Do you have running tomcat container? If yes, Stop it and kill all tomcat container.

ANS-

C:\Users\Sieanna>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

e32489eed508 tomcat:7.0.57-jre8 "catalina.sh run" 2 days ago Up 2 hours 0.0.0.0:9090->9090/tcp, :::9090->9090/tcp, 8080/tcp, 0.0.0.0:50000->50000/tcp, :::50000->50000/tcp MyTomCat

**DELETE**

**C:\Users\Sieanna>docker rm -f e32489eed508**

**e32489eed508**

C:\Users\Sieanna>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

38. Create an index.html file with following code in it:-

<h1>This is Tomcat Container</h1>

ANS-

Now, Start a ubuntu container, login into the ubuntu, install nginx, access nginx's home page from the container,

Now, come out of the ubuntu container and replace the existing index.html page of nginx with above index.html page that you created, Now access the nginx server from host machine, it should display the index.html that you have created.

39. type below command:-

docker images --help

Now, try to run command that proves the concept of following three options:-

1. -a = Show all images (default hides intermediate images)

C:\Users\Sieanna>docker images -a

REPOSITORY TAG IMAGE ID CREATED SIZE

<none> <none> 541bf3902c85 2 hours ago 135MB

ubuntu xenial b6f507652425 6 days ago 135MB

ubuntu rolling e5ec58a9f549 6 days ago 80MB

myimage 1.0 16bdf41412d4 6 days ago 103MB

2. -f= Filter output based on conditions provided

C:\Users\Sieanna>docker images -f "dangling=false"

REPOSITORY TAG IMAGE ID CREATED SIZE

ubuntu xenial b6f507652425 6 days ago 135MB

ubuntu rolling e5ec58a9f549 6 days ago 80MB

myimage 1.0 16bdf41412d4 6 days ago 103MB

**AND**

C:\Users\Sieanna>docker images -f "dangling=True"

REPOSITORY TAG IMAGE ID CREATED SIZE

3. -q= Only show image IDs

C:\Users\Sieanna>docker images -q

541bf3902c85

b6f507652425

e5ec58a9f549

16bdf41412d4

write atleast 1 command using each option above and prove their concepts as described in the --help.

40. type below command:-

docker ps --help

Now, try to run command that proves the concept of following six options:-

1. -a = Show all containers (default shows just running)

C:\Users\Sieanna>docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

af3f759ba563 ubuntu:xenial "/bin/bash" 41 minutes ago Exited (127) 9 minutes ago clever\_shirley

6327bbfb05dc ubuntu:xenial "/bin/bash" 2 hours ago Exited ( 0) 2 hours ago vibrant\_wu

2. -f= -f, --filter filter Filter output based on conditions provided

C:\Users\Sieanna>docker ps --filter status=running

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

89d3c6c3942d ubuntu:xenial "/bin/bash" 50 seconds ago Up 22 seconds angry\_leakey

3. -q= Only display container IDs of running container..

-aq = Only display container IDs of all container

C:\Users\Sieanna>docker ps -aq

af3f759ba563

6327bbfb05dc

4. -n/ --last int = Show n last created containers (includes states) (default -1)

C:\Users\Sieanna>docker ps --last 2

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

89d3c6c3942d ubuntu:xenial "/bin/bash" 7 minutes ago Up 6 minutes angry\_leakey

af3f759ba563 ubuntu:xenial "/bin/bash" 2 hours ago Exited (127) About an hour ago clever\_shirley

5. -l= Show the latest created container (includes all states)

C:\Users\Sieanna>docker ps -l

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

af3f759ba563 ubuntu:xenial "/bin/bash" 46 minutes ago Exited (127) 13 minutes ago clever\_shirley

6. -s= C:\Users\Sieanna>docker ps -s Display total file sizes of running container

C:\Users\Sieanna>docker ps -as Display total file sizes of all container

write atleast 1 command using each option above and prove their concepts as described in the --help.

41. Type below command:-

docker --help

you will various sections of commands apart from options like "Managemnet Commands" and "Commands".

Write some texts below describing the use of "Management Commands".

Use each command mentioned below and prove its concepts as described in the --help desription. write what you have understood from the output of the command after its successful execution.

1. cp

2. create: Create a new container only but don’t run.

* **C:\Users\Sieanna>docker create redis**
* **C:\Users\Sieanna>docker ps -a**

**CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES**

**ac59918f51db redis "docker-entrypoint.s…" 19 seconds ago Created gracious\_solomon**

3. export

4. history: Show the history of an image

* **C:\Users\Sieanna>docker history hello-world**

**IMAGE CREATED CREATED BY SIZE COMMENT**

**d1165f221234 6 months ago /bin/sh -c #(nop) CMD ["/hello"] 0B**

**<missing> 6 months ago /bin/sh -c #(nop) COPY file:7bf12aab75c3867a… 13.3kB**

5. info: Display system-wide information

* **C:\Users\Sieanna>docker info**

6. login: Log in to a Docker registry

* **C:\Users\Sieanna>docker login**

**Authenticating with existing credentials...**

**Login Succeeded**

7. logout: Log out from a Docker registry

* **C:\Users\Sieanna>docker logout**

**Removing login credentials for https://index.docker.io/v1/**

8. rename: Rename a container

* **C:\Users\Sieanna>docker rename MyImg MyImage**

9. save: Save one or more images to a tar archive (streamed to STDOUT by default)

* C:\Users\Sieanna>docker save -o angry\_leakey ubuntu:xenial

10. stats: Display a live stream of container(s) resource usage statistics

* docker stats

11. top

42. Kill all running container in one liner command.

ANS- docker rm $(docker ps -a -q)

43. Delete all images in one liner command.

ANS-

docker rmi -f $(docker images -q)

44. Create a simple Dockerfile, build it and run it.

ANS- **To build an image in dockerfile**

* C:\Users\Sieanna\Desktop\DockerFiles>docker build -t myimg1:1.0 .

**To check an image in dockerfile**

* C:\Users\Sieanna\Desktop\DockerFiles>docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

myimg1 1.0 16bdf41412d4 7 days ago 103MB

**To run an image**

* C:\Users\Sieanna\Desktop\DockerFiles>docker run 16bdf41412d4

Hello Worlddddd........... !!! My first docker image

45. Create one or more Dockerfile that demostrate the following commands in Dockerfile (Write a PoC for each in one or more Dockerfile)

**1. USER :** The USER instruction sets the user name (or UID) and optionally the user group (or GID) to use when running the image and for any RUN, CMD and ENTRYPOINT instructions that follow it in the Dockerfile.

**2. RUN :** The RUN instruction will execute any commands in a new layer on top of the current image and commit the results. The resulting committed image will be used for the next step in the Dockerfile.

**3. ENV :** Environment variables (declared with the ENV statement) can also be used in certain instructions as variables to be interpreted by the Dockerfile.

**4. CMD :** There can only be one CMD instruction in a Dockerfile. If you list more than one CMD then only the last CMD will take effect.

**5. ENTRYPOINT :** An ENTRYPOINT allows you to configure a container that will run as an executable.

**6. EXPOSE :** The EXPOSE instruction informs Docker that the container listens on the specified network ports at runtime. You can specify whether the port listens on TCP or UDP, and the default is TCP if the protocol is not specified.

**7. VOLUME :** The VOLUME instruction creates a mount point with the specified name and marks it as holding externally mounted volumes from native host or other containers.

46. Dockerhub:-

Find a application you care about on docker hub.

Launch the container.

Install another application in it.

Save (commit) the image.

Upload that to docker hub in your account and share it with a colleague, ask them to use your image and run the container out of it in their machine/vm.

ANS-

**Step1:** we have used first ubuntu image and run that image to make it container:

C:\Users\Sieanna>docker run --name MyUImag -it ubuntu:rolling

root@53606d81bf05:/#

**Step2:** Then installed redis image in ubuntu container

root@53606d81bf05:/# apt-get update

root@53606d81bf05:/# apt-get install redis

**Step3:** Then we have save (commit) this as image using following command

C:\Users\Sieanna>docker commit 53606d81bf05 salonivichare/ubuntu:rolling

**Step4**: check docker images you will be able to see our image with repository salonivichare/ubuntu:rolling

**Step5:** Check docker login

**Step6:** Push to our docker hub account

**C:\Users\Sieanna>docker push salonivichare/ubuntu:rolling**

The push refers to repository [docker.io/salonivichare/ubuntu]

847494fd6a38: Pushed 376c4e05703c: Mounted from library/ubuntu rolling: digest: sha256:059d7e77d6947ff0b64d173974a00257085bc532b1a27f2912defe062c14eaef size: 741

------------------------------------------------------------------------------------------------------------------------------------

Advanced Assignment:-

Assignment - Dockers and Containers

---Introduction

The goal of this assignment is to get you familiarized with dockers. You will learn about how to dockerize any application by building a simple client and server containers using dockers.

---What is to be done?

You will be building two containers using docker. These containers can be built starting from an official base image. Official Images has a list of different official base images. Start from the image that is suitable for you.

---Server :

Create a volume by name "servervol".

The server container will mount "servervol" in "/serverdata".

This container runs a server application which will create a file of size 1KB with random text data in "/serverdata" and then transfer the file to the client along with the checksum.

The server application itself can be built using any language you are comfortable with. But, the container should include all the packages that are required to run your application. Choose your base image wisely and install only the necessary packages.

The port on which the server runs must be specified as a command line argument when we run docker.

---Client:

Create a volume by name "clientvol".

The client container will mount "clientvol" in "/clientdata".

The client container runs an application that connects to the server, recieves the file that the server sends and saves it in "/clientdata".

Verify that the file is received properly at the clientside by verifying the checksum.

The client application again can be wriiten in any language that you are comfortable with, but the container should include all the necessary packages. Choose your base image wisely and install only necessary packages.

You need to create a user-defined network in docker and run both these containers on the network created. The containers should run these applications by default (i.e, on run command). Mention in README as to what should be done to get into the container shell instead of running the applications by default (i.e, on run command).

NOTE: You should be able to get into the shell of the client container to physically check if the file has been recieved.

---What to submit?

Create a new empty branch in your git repos named "assignment2".

You will be submitting the files that you use to build the server and client containers. Checkout the assignment2 branch and submit the files related to server and client in two folders named "server" and "client" respectively.

As you go about doing the assignment, you will learn that there are multiple ways to create volumes and to have the containers communicate to each other. So submit a README file that expains what method you use.

Create two scripts. one for the server("fileserver.sh") and the other for the client ("fileclient.sh") that has all the commands to build and run server and client containers respectively. These scripts should include the volume creation for client and server respectively. The server script should also include the creation of user-defined network before server runs.

The scripts and the README should be in the root of your branch (outside your server and client directories).

---What is tested?

Run the "fileserver.sh" script on one terminal. It should create the user-defined network, the "servervol", start the server and wait for the client connection.

Run "fileclient.sh" in another terminal. This script should find out the server's IP, create the "clientvol" and run the client container by specifying server's IP and port as command line parameters.

When the client container is run, a connection between the server and client is established and the file is transferred from the server to the client.

Your code and scripts will be tested on creation of volumes, proper running of server and client and file transfer.

You should make sure that the server can transfer any file with random text and the checksum verification succeeds on the client when it receives the file.

---Grade Distribution

Build and run server container - 30 points

Build and run client container - 30 points

Proper communication between the two - 15 points

Scripts to run the containers - 15 points

README file - 10 points

------------------------------------------------------------------------------------------------------------------------------------

Advanced Assignment 2

---Introduction

The goal of this assignment is to get you familiarized with CI-CD Pipeline for Creating docker images after successfull build.

You already built 2 app(client and server) from the "Part 1" Assignment.

NOTE: You should be able to get the docker image for both Client and Server after successfull build using Jenkins Pipeline.

Your Client app and Server app must be in the github.

Whenever developer commits changes to client/server app in github, it should trigger a build on Jenkins.

Jenkins Pipeline should do maven clean and package

Jenkins Pipeline should also do the static-code analysis using (checkstyle/sonarqube)

Jenkins pipeline at the end should create docker image

On build failure, Jenkins pipeline should send an email to any emailId about the failure.

---Grade Distribution

Proper Build trigger - 20 points

Proper Declarative Jenkins file - 20 points

Clean, Package and Code Analysis - 20 points

Creating docker image on successful build - 20 points

Sending email on build failure - 20 points