Lab Assignment 5: Docker and Containerization - Documentation

Due Date Submit before 23:59 on Friday, November 3, 2023

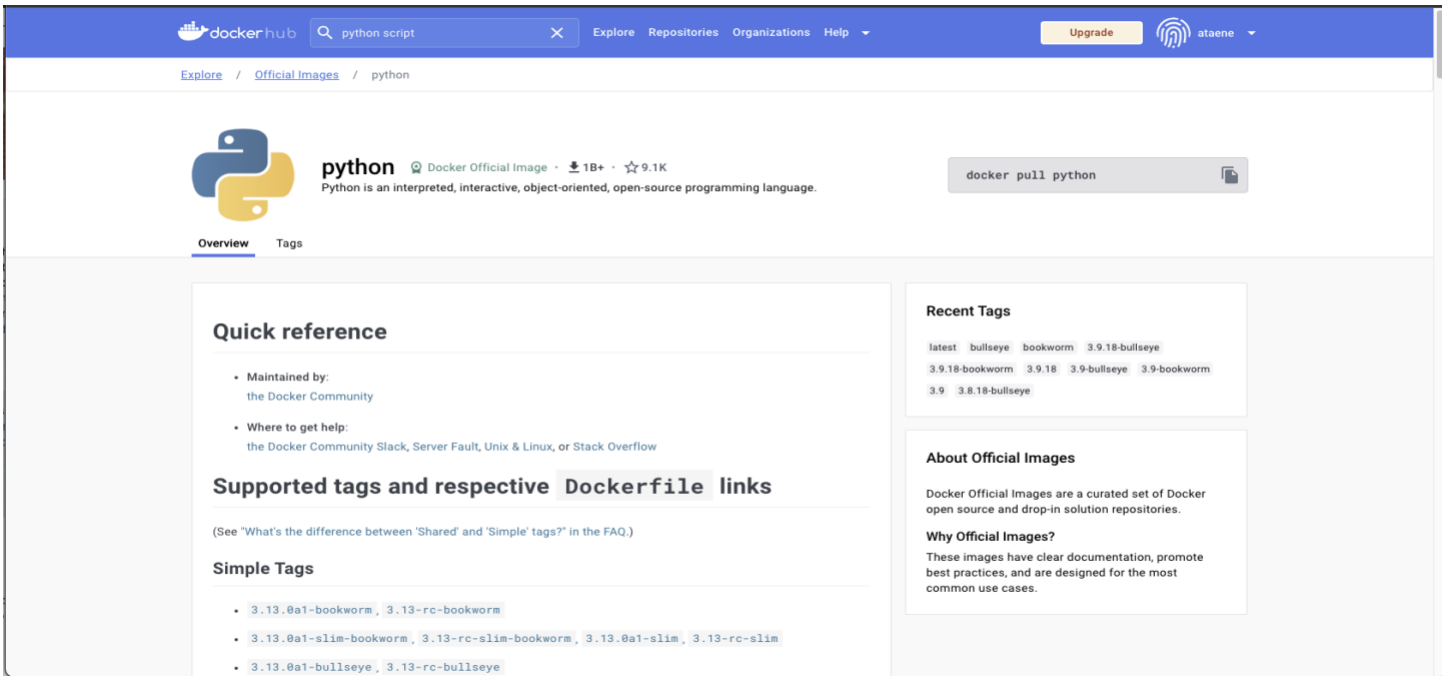
Student Name: Emmanuel Alafonye

Step 1: Install Docker Desktop: I installed Docker Desktop on my Mac computer by following the instructions in the Docker documentation downloaded from https://docs.docker.com/engine/install/

Step 2: Create a Docker Hub Account: I created a Docker Hub account at https://hub.docker.com and signed in.

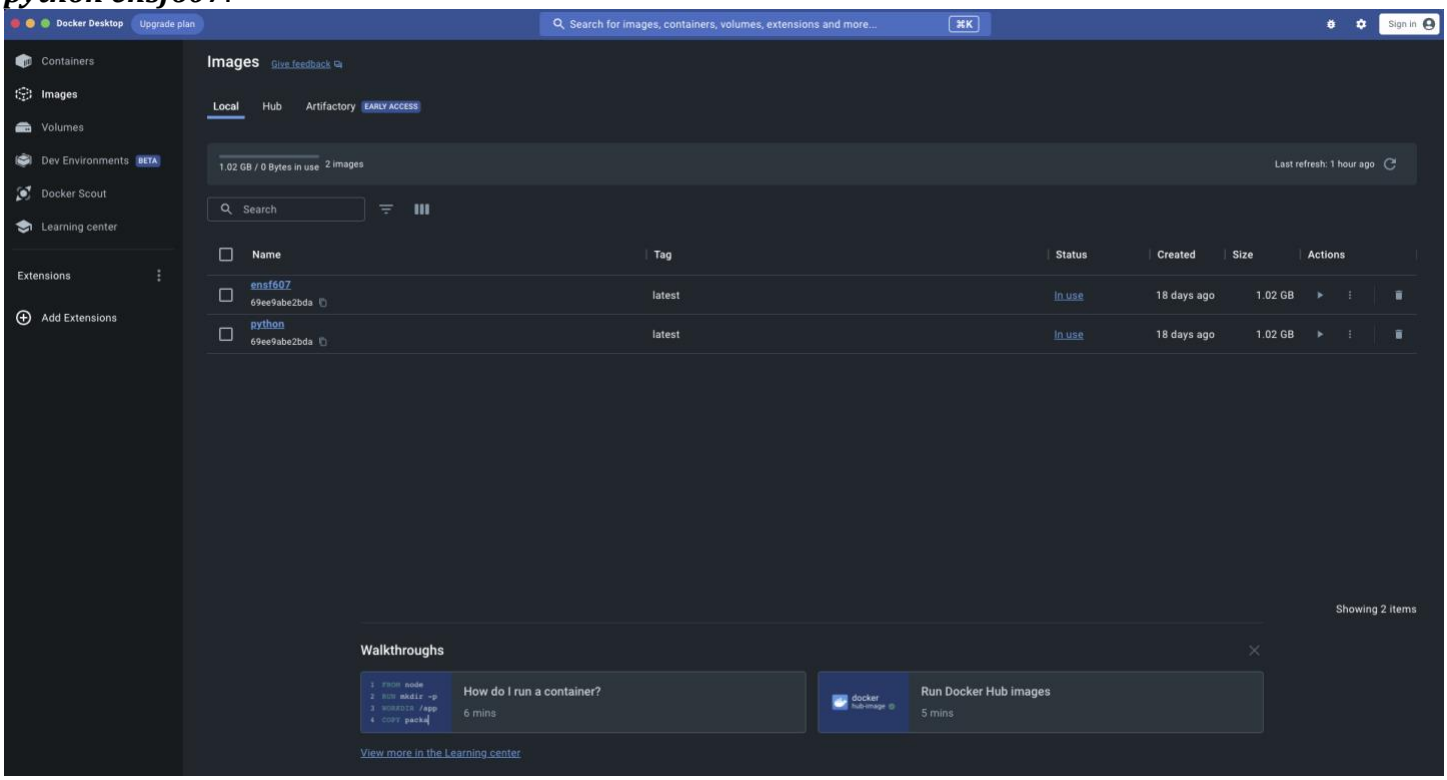
Step 3: Find a Python Image: I searched for a Python image on Docker Hub, and I found the official Python image.

Step 4: Pull Python Image: I used the command, ***docker pull python*** to download the Python image from Docker Hub.



Step 5: List Docker Images: To confirm the image download, I ran ***docker image ls***, and I observed the Python image in the list.

Step 6: Rename Python Image: I renamed the Python image to "ensf607" using the command ***docker tag python ensf607***.



Step 7: Run a Container: I ran a Docker container using the renamed image: ***docker run -itd ensf607***. The `-itd` options were used, and I understood that they control interactive and detached execution of the container.

Description of -itd: “i” means interactive, which allows the user to interact with the docker containers input standard – stdin. This means that the user can interact and send input commands without which there is no interaction with the docker container.

The “t” is a pseudo terminal of the container that provides a more interactive and responsive experience for the command line interactions with docker containers.

The “d” means detached, is used to run the docker container in the background processes and a response is immediately given back to the user that provides the prompt. Hence it is very useful for continuous running of the docker container without blocking the terminal.

Step 8: List Active Containers: To see the active containers, I executed ***docker ps***. Below is the random id: 69ee9abe2bda3367c4de61b7e0ee4af2fc72b28e24b74ffede77fc1017c198d8

Step 10: Rename Container: I renamed the Python container using the command ***docker rename ensf607 ensf607_alafonye***. I confirmed the change with docker ps.

Step 11: Enter the Container

```
(base) ataene@Emmanuels-MacBook-Pro ~ % docker exec -it ensf607_alafonye sh
# ls -ltr
total 52
drwxr-xr-x   2 root root 4096 Sep 29 20:04 boot
drwxr-xr-x   1 root root 4096 Oct 30 00:00 var
drwxr-xr-x   1 root root 4096 Oct 30 00:00 usr
drwxr-xr-x   2 root root 4096 Oct 30 00:00 srv
lrwxrwxrwx   1 root root    8 Oct 30 00:00 sbin -> usr/sbin
drwxr-xr-x   2 root root 4096 Oct 30 00:00 opt
drwxr-xr-x   2 root root 4096 Oct 30 00:00 mnt
drwxr-xr-x   2 root root 4096 Oct 30 00:00 media
lrwxrwxrwx   1 root root    7 Oct 30 00:00 lib -> usr/lib
lrwxrwxrwx   1 root root    7 Oct 30 00:00 bin -> usr/bin
drwxr-xr-x   1 root root 4096 Nov  1 02:04 run
drwx------   1 root root 4096 Nov  1 09:23 root
drwxrwxrwt   1 root root 4096 Nov  1 09:33 tmp
drwxr-xr-x   1 root root 4096 Nov  2 14:28 etc
dr-xr-xr-x  12 root root    0 Nov  2 14:28 sys
dr-xr-xr-x 235 root root    0 Nov  2 14:28 proc
drwxr-xr-x   5 root root  360 Nov  2 14:28 dev
drwxr-xr-x   1 root root 4096 Nov  2 14:36 home
# ./home
sh: 2: ./home: Permission denied
# cd ./home
# ls
python_scripts
# exit
(base) ataene@Emmanuels-MacBook-Pro ~ %
```

I entered the container using the command ***docker exec -it ensf607_alafonye sh***. I understood that "sh" stands for the shell environment, allowing me to run Linux commands inside the container.

Command Line within Container

Inside the container, I executed various Linux commands, such as ***ls -ltr***, ***cd ./home***, ***mkdir ./python_scripts***, and more. I also used ***exit*** to exit the container and return to my computer's OS.

Uploading Python Script: I uploaded the Python script, ***testprint.py***, to the container using the command: ***docker cp Desktop/ENSF607/Assignments/Assignment5/testprint.py ensf607_alafonye:/home/python_scripts/testprint.py.***

To verify the script's presence, I re-entered the container with ***docker exec -it ensf607_alafonye sh,*** navigated to the ***/home/python_scripts*** directory, and used ***ls -ltr***.

Running the Python Script: I executed the Python script with the command ***python ./testprint.py***, and it produced the expected output: "This is a container test."

Conclusion: I have successfully completed this lab assignment, which involved creating a Docker container, interacting with it, and running a Python script within the container. This exercise has given me a hands-on understanding of containerization and Docker.

```
(base) ataene@Emmanuels-MacBook-Pro Assignment5 % docker cp Desktop/ENSF607/Assignments/Assignment5/testprint.py
 ensf607_alafonye:/home/python_scripts/testprint.py

lstat /Users/ataene/Desktop/ENSF607/Assignments/Assignment5/Desktop: no such file or directory
(base) ataene@Emmanuels-MacBook-Pro Assignment5 % docker cp Desktop/ENSF607/Assignments/Assignment5/testprint.py
 ensf607_alafonye:/home/python_scripts/testprint.py

lstat /Users/ataene/Desktop/ENSF607/Assignments/Assignment5/Desktop: no such file or directory
(base) ataene@Emmanuels-MacBook-Pro Assignment5 % cd /Users/ataene/Desktop/ENSF607/Assignments/Assignment5
docker cp testprint.py ensf607_alafonye:/home/python_scripts/testprint.py

                                        Successfully copied 2.05kB to ensf607_alafonye:/home/python_scripts
/testprint.py
(base) ataene@Emmanuels-MacBook-Pro Assignment5 % docker exec -it ensf607_alafonye sh
# ./home
sh: 1: ./home: Permission denied
# ./home/python_scripts
sh: 2: ./home/python_scripts: Permission denied
# cd ./home/python_scripts
# python ./testprint/py
python: can't open file '/home/python_scripts/./testprint/py': [Errno 2] No such file or directory
# python ./testprint.py
This is a container test
#
```