Cloud Computing and Cloud Networking

CS 208

Assignment 2

# Problem Statement

In this assignment, you have to use a client-server socket program using TCP sockets

from Assignment 1 and containerize your applications. There will be two containers, one

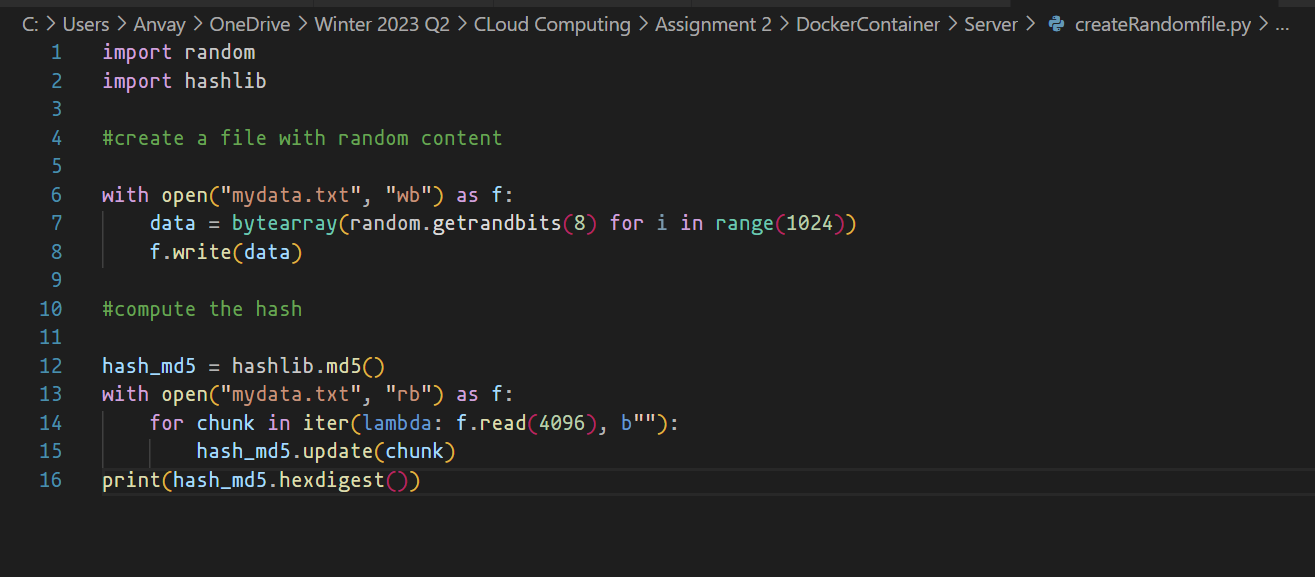
for the server and another for the client application. You can create two containers in the

same machine.

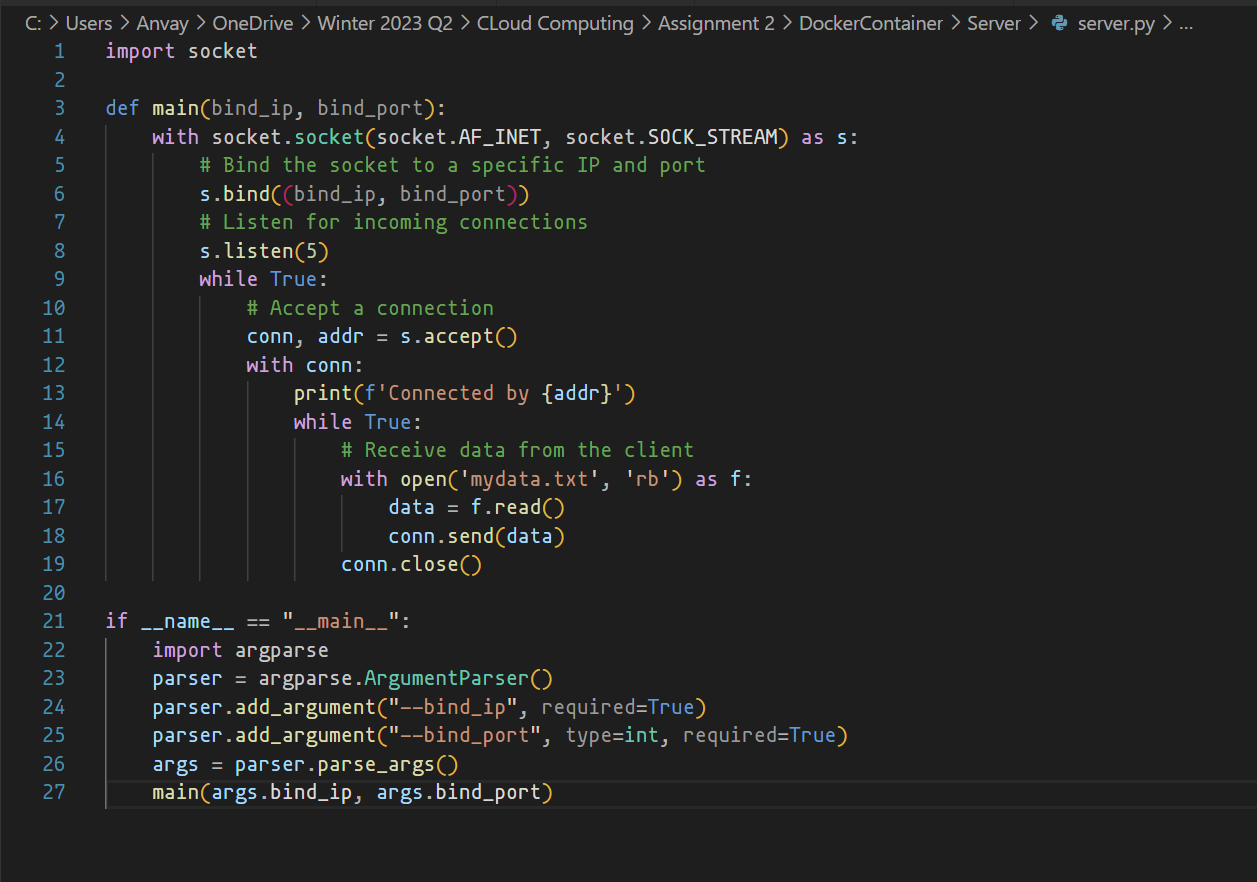
# Server Container

## Resources:

1. The program used to create a random file

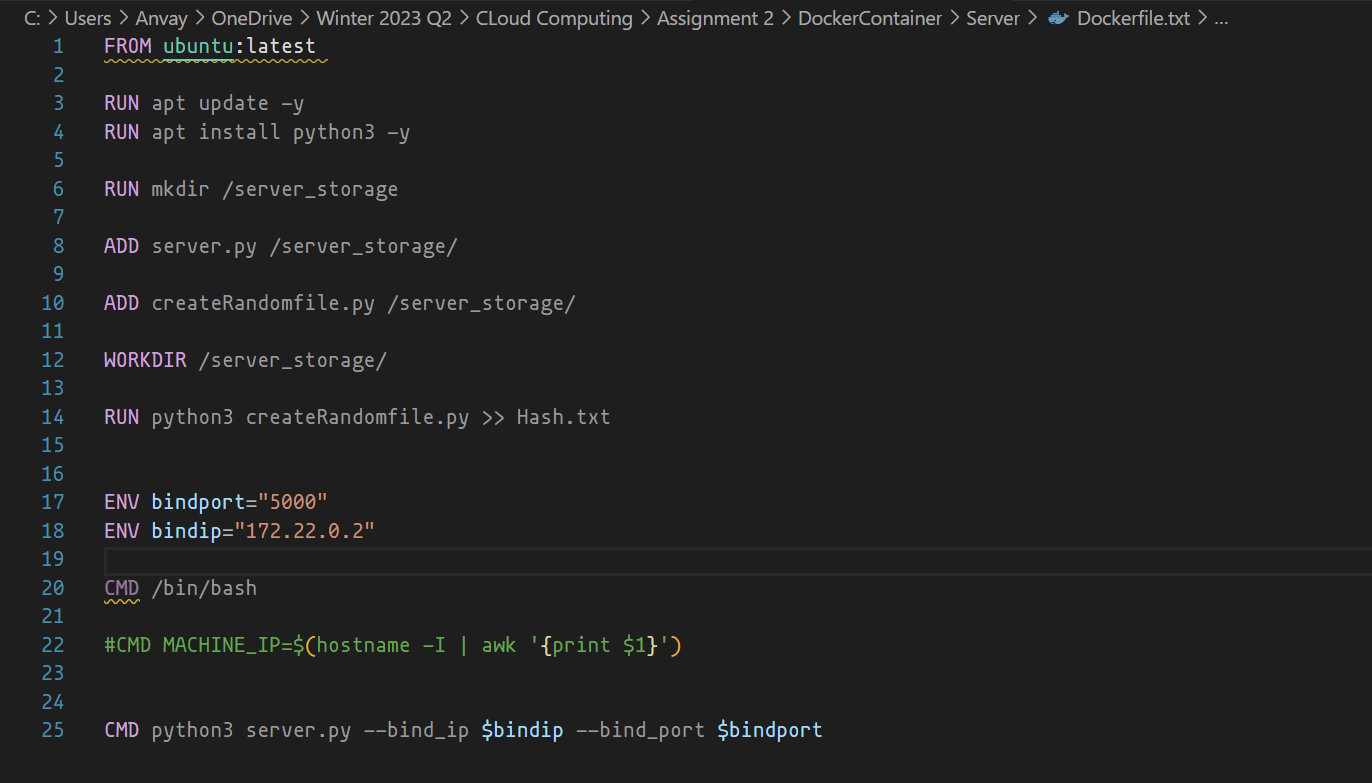


1. The server socket program for the server container runs

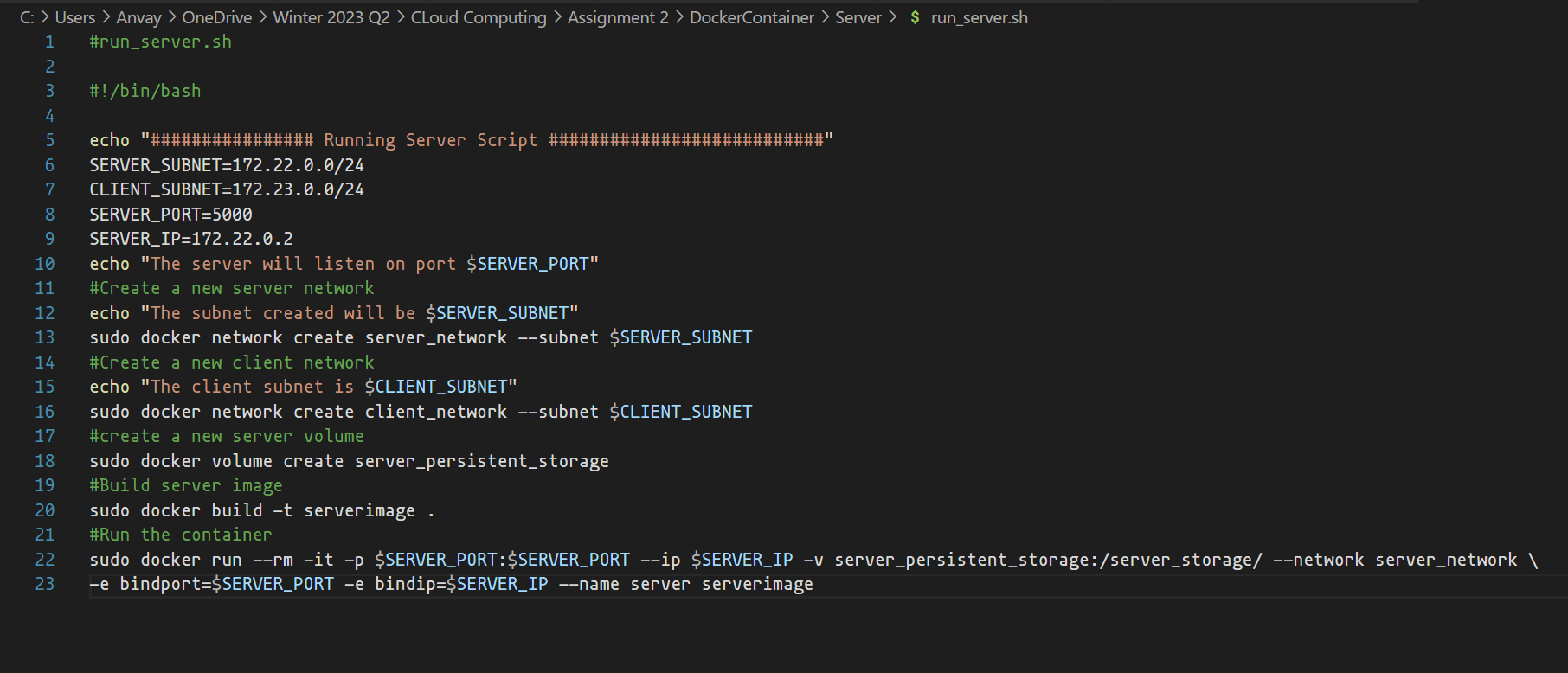


1. The Docker file that is used to build and run the server container.

Here the container uses a ubuntu image and following with required package download. The directory **server\_storage** is created which will be attached to the volume. The socket program and the create random file program is added to the directory and the program is run during container runtime. The **bindport** and **bindip** that are hardcoded into the file can be overiden during **docker run**.

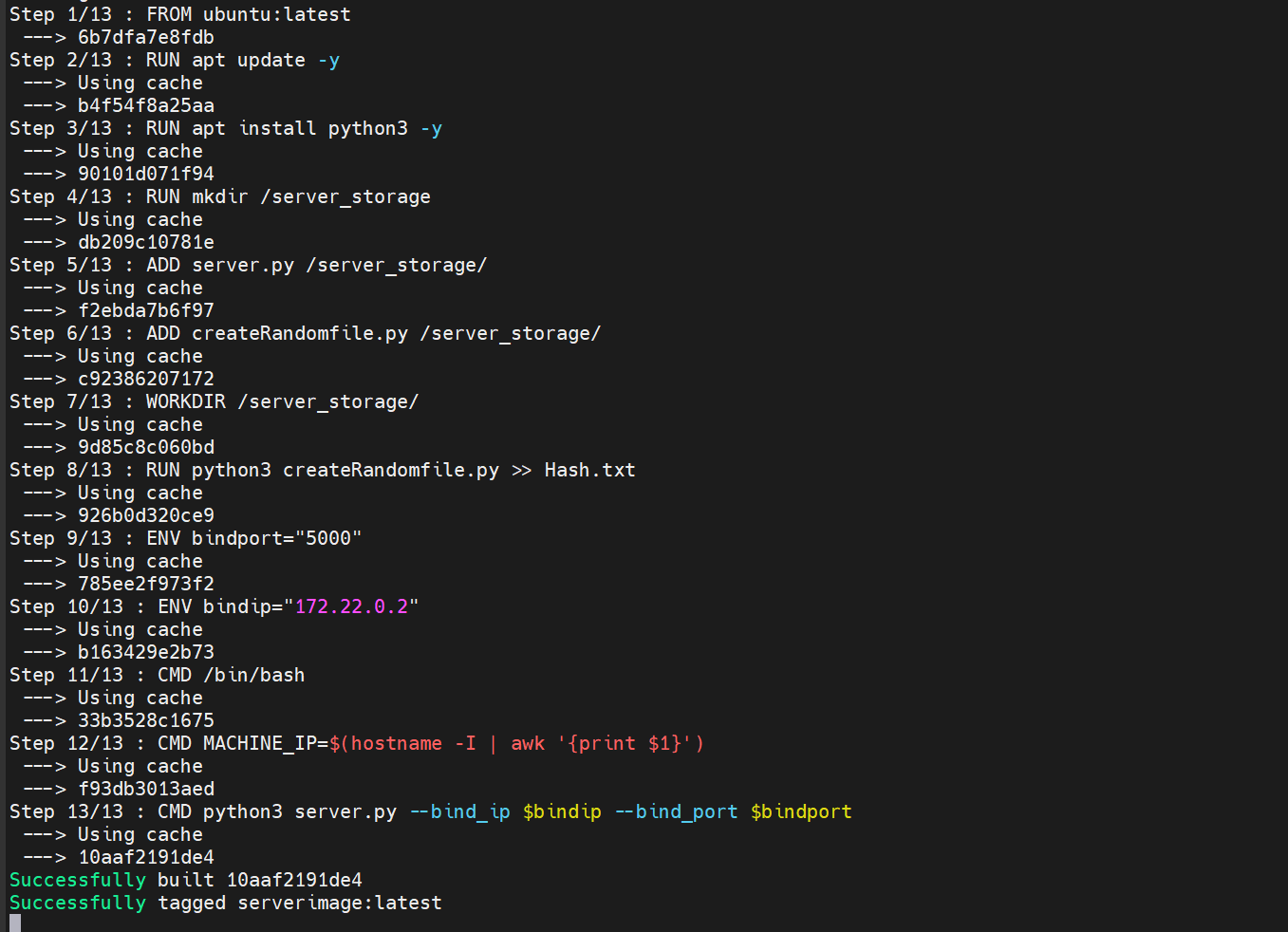


1. **run\_server.sh** is the script that creates a bunch of resources required for the server container. It starts of with creating few variables that can be changed as required. Followed by that is the network and volume create. The docker run command with -**e** is given to **override the default variables** in the **dockerfile**

****

## SERVER CONTAINER RUNTIME

After running the server container using the script. The following output is received. We see that the server is listening for incoming connection.



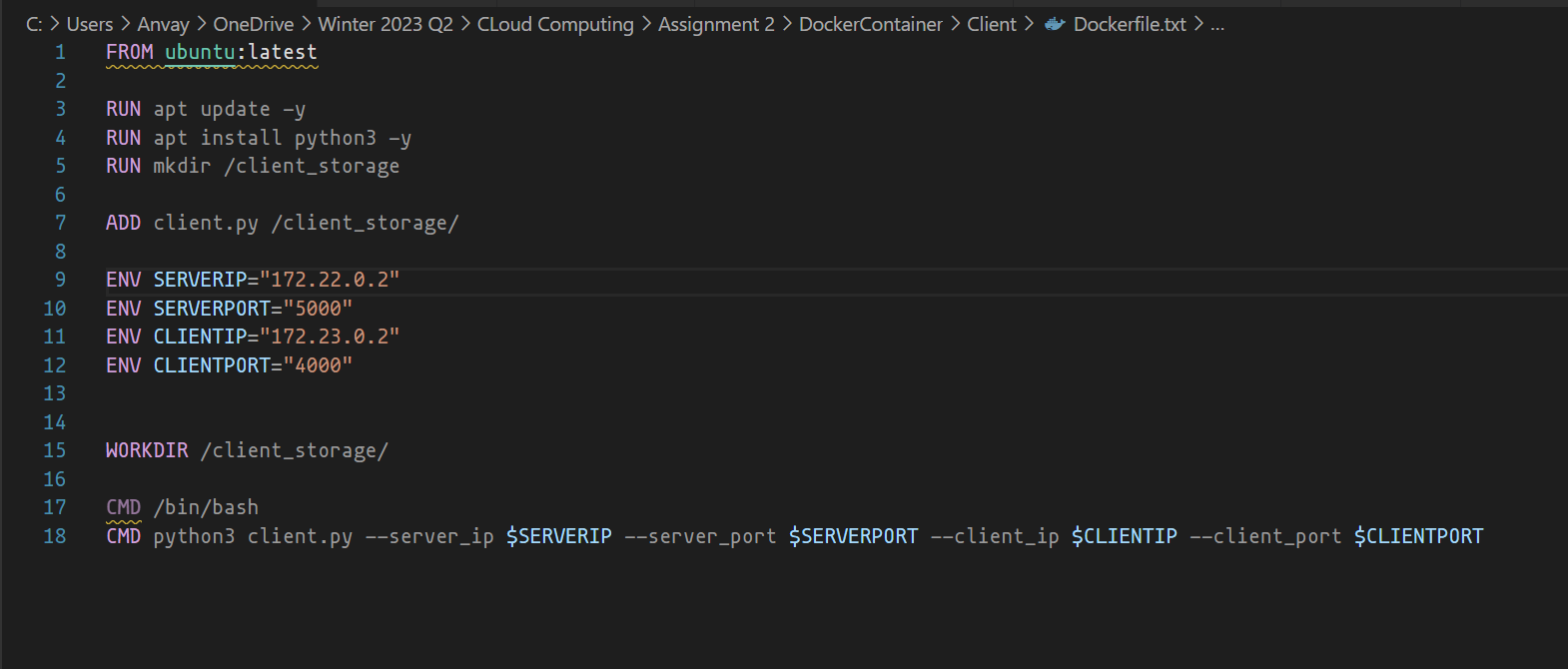
# Client Container

## Resources:

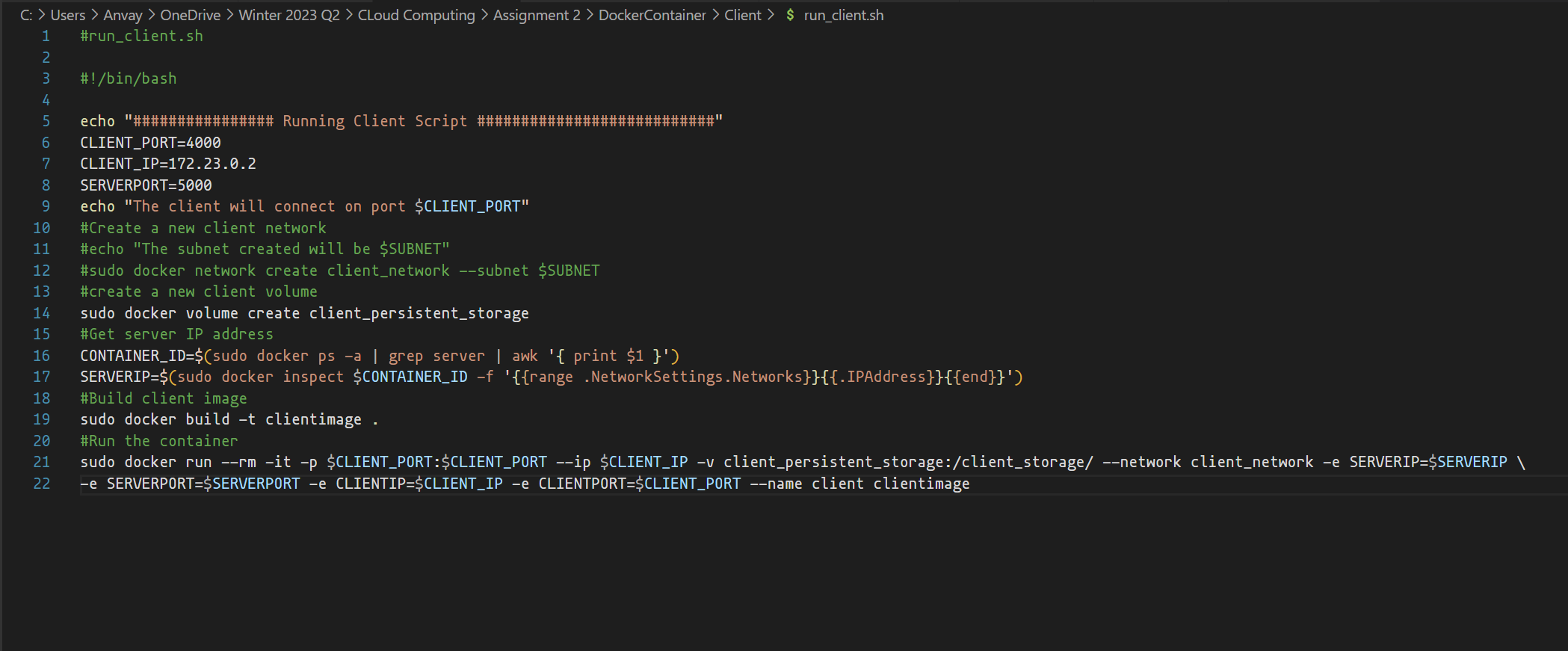
1. The client socket program is defined as follows.



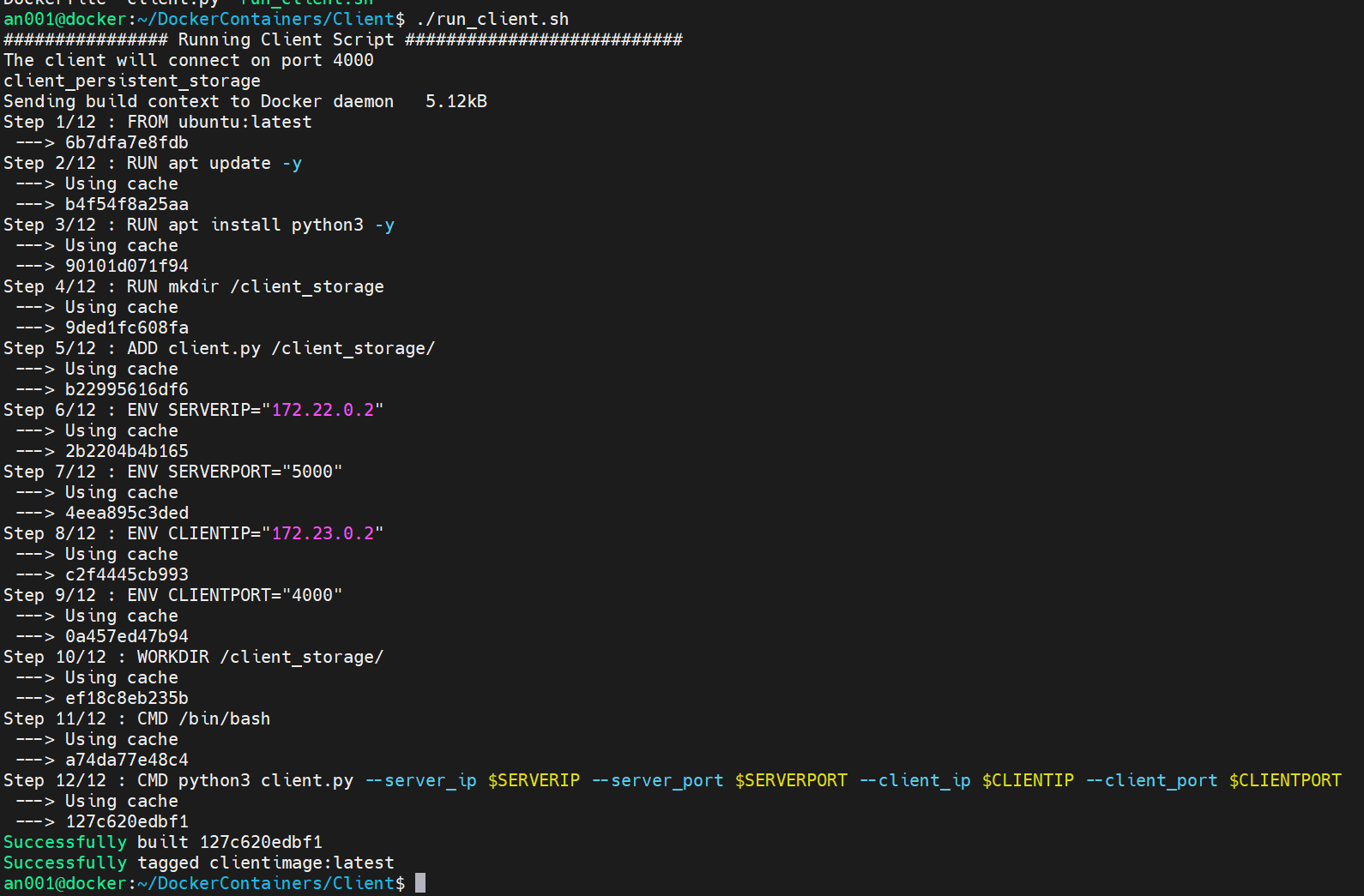
1. The **Dockerfile** for the client server is mentioned below. It is made from the latest image of ubuntu. Followed by creation of **client\_storage** directory to map the client volume. The **ENV variables are the server IP, port and client IP and port**. This can be overridden at docker run time.



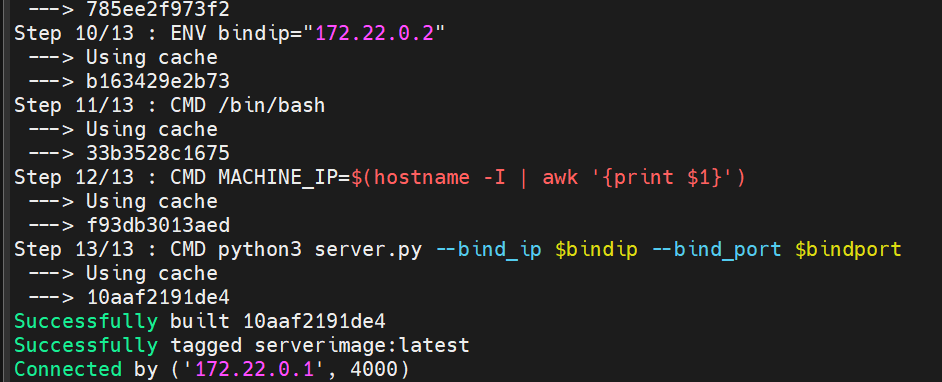
1. The **run\_client.sh** is the script that is used to run the container with bunch of other resource creation required for the container.



1. The script exits after receiving the file.



1. The server container exits after sending the data file.

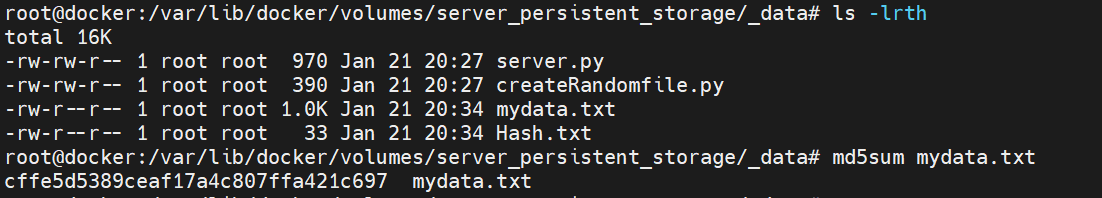


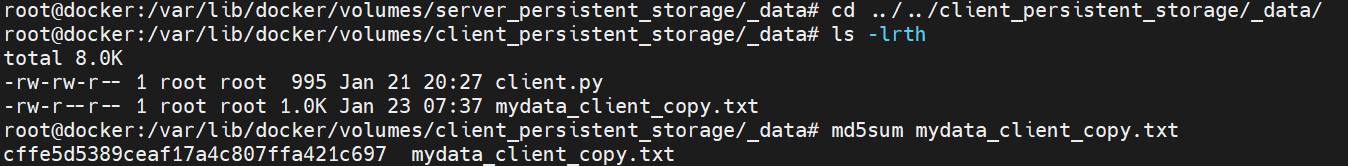
# IPTABLE CONFIGURATION

The iptables should be configured for the communication between the containers. The process/ commands to configure this is in the README file. Please refer to it.

# Verifying the Checksums

By checking the path of the volumes created by docker and navigating to the path. We can find the data. We can check the checksum of the data files to verify accurate data transfer between the containers.





# Conclusion

The Docker containers successfully transfer files between the server and client container. The checksum of the files on both containers are same and verifies data integrity and accuracy while transfer.

# References

Please refer to the github link for full assignment. The readme file will guide you to run the applications on your environment.   
  
[**https://github.com/anvayabn/DockerContainers.git**](https://github.com/anvayabn/DockerContainers.git)