**Assignment – 3 – Report**

**Name:** Pavan Pandya

**GitHub Link:** <https://github.com/pavanpandya/ECC_Assignment_03>

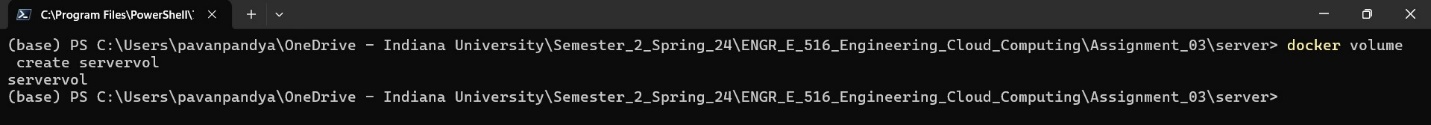
I already have Docker installed on my machine and have written a custom Dockerfile code to have a base environment to run my Python scripts.

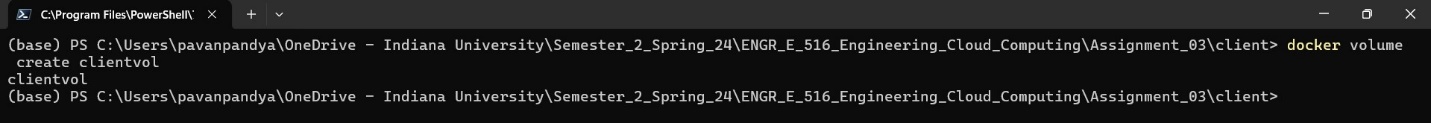
**Docker Volume Creation:**

As per the requirements, I have created two docker images:

1. servervol
2. clientvol

Here is the command used to create those volumes:





**Docker Network Creation:**

As per the requirements, I have created a network ~ “pnpandya”.

Client and Server container build will be running on this network.

**Code:**

1. **server.py**

**A screen shot of a computer program

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

This Python script implements a server for file transfer using sockets. It listens for incoming connections on a specified host and port. Upon establishing a connection with a client, it generates a **random file of size 1KB**, computes its checksum, and sends both the file and the checksum to the client. The server logs various events such as listening for connections, establishing connections with clients, generating files, computing checksums, sending files, and closing connections. The logging messages are formatted with timestamps and severity levels for easy understanding and debugging.

1. **client.py**

**A screen shot of a computer program

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

**A screen shot of a computer screen

Description automatically generated**

This Python script implements a client for file transfer using sockets. It connects to a server at a specified host and port. Upon connection, it receives a file along with its checksum from the server, saves the file to the local directory, and verifies the checksum. The client logs various events such as connecting to the server, receiving files, verifying checksums, and potential errors such as connection refusal or forcible closure. The logging messages are formatted with timestamps and severity levels for better understanding and debugging. Additionally, the script includes a delay of 60 seconds before exiting, allowing time to get the inspect network logs.

**Testing Code on Local Machine:**

In this screenshot, you can observe that server and client are able to transfer the file successfully and that be confirmed by looking at the serverdata and clientdata folders

A screenshot of a computer program

Description automatically generated

In this screenshot, you can observe that I have passed port and host as command line arguments.

A screenshot of a computer program

Description automatically generated

**Creating Dockerfile:**

1. **Server**

**A screenshot of a computer program

Description automatically generated**

1. **Client**

**A screenshot of a computer

Description automatically generated**

**Building Docker Image:**

1. **Server**

A screenshot of a computer

Description automatically generated

1. **Client**

A screenshot of a computer

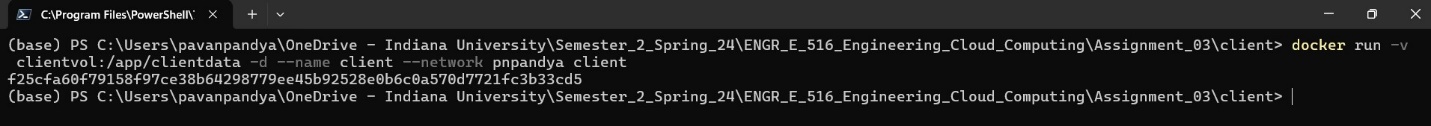
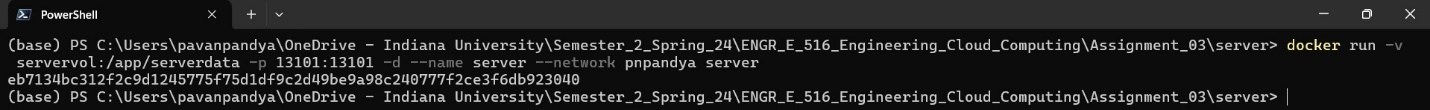
Description automatically generated

Once the images are built successfully, the images are created, and we can see that in the docker desktop dashboard.

A screenshot of a computer

Description automatically generated

**Running Docker Images:**



Now let’s verify the logs by checking the container logs.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

You can observe all the logs in the server and client container logs and confirm that the file is transferred successfully.

**Verifying the data in Volumes:**

1. **servervol**

A computer screen shot of a computer

Description automatically generated

1. **clientvol**

A screenshot of a computer

Description automatically generated

You can observe both the client and server have stored the transferred file after verifying the checksum.

**Confirm Container Network:**

To Confirm if the containers were running on the same network, I used the docker inspect network command:

A screen shot of a computer

Description automatically generated

From the above screenshot, it can be confirmed that the containers are running on the same network i.e. pnpandya

**Automating the Process – Docker Compose:**

**A screenshot of a computer program

Description automatically generated**

**Building the Images using docker-compose.yml:**

A screenshot of a computer

Description automatically generated

**Containers under Assignment\_03**

A screenshot of a computer

Description automatically generated

**Server Container Logs**

A screenshot of a computer

Description automatically generated

**Client Container Logs**

A screenshot of a computer

Description automatically generated

**Server Volume Logs**

A screenshot of a computer

Description automatically generated

**Client Volume Logs**

A screenshot of a computer

Description automatically generated