**Problem Statement:** To install docker, build two images: one for server and the other for client. Mount a volume for the server, and another for the client. Create a user defined network and connect the containers to them with ports specified on the command line. Once the Docker run command is given, the server container must immediately generate a text file of size 1KB, store it in the volume and then the client upon connecting to the server, must receive the text file along with the checksum to verify the integrity of the file. Once the file is received by the client, its integrity should be verified.

---

**Server:**

**First, create a volume by name "servervol"**
-> docker volume create servervol

**Next, mount the "servervol" in "/serverdata"**
-> This will be done in the Docker run command itself. It will be mentioned at the final part of the server.

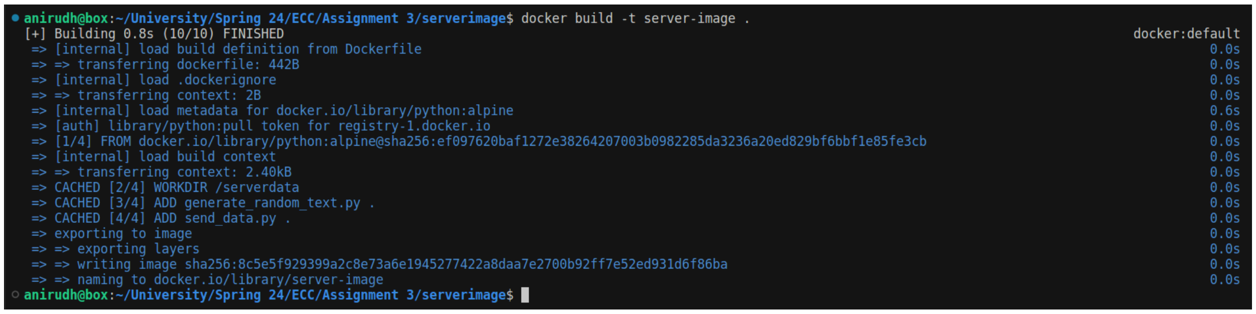
**Container should run an application on startup which will create a file of size 1KB with random text data and store it in "/serverdata"**
-> To generate the text file, I used the random library file in python to generate random text of size 1024 and store it in the directory "/serverdata". The file generate_random_text.py has the code to this.

**The container should include all the packages that are required to run your application. Choose an appropriate base image and install only the necessary packages**
-> The base image I chose was python:alpine. The reasoning behind this is python's docker images come with all the basic pip packages installed to run python based applications. And as for alpine, it's for the sake of a lightweight file system for running other commands in interactable mode such as ping, ssh and so on. I did not have to specify any additional python libraries to be installed as the python:alpine base image already had all the necessary dependencies installed.

**The port on which the server runs must be specified as a command line argument when we run docker**
-> The port I exposed in the Dockerfile is 8888.

**Commands used for the Server Container:**

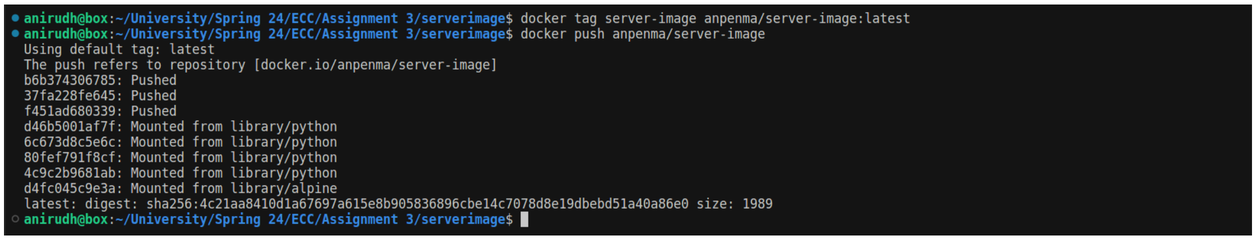
-> docker build -t server-image .



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker build -t server-image .
  [+] Building 0.8s (10/10) FINISHED                                                                                       docker:default
   => [internal] load build definition from Dockerfile                                                                              0.0s
   => => transferring dockerfile: 442B                                                                                              0.0s
   => [internal] load .dockerignore                                                                                                 0.0s
   => => transferring context: 2B                                                                                                   0.0s
   => [internal] load metadata for docker.io/library/python:alpine                                                                  0.6s
   => [auth] library/python:pull token for registry-1.docker.io                                                                     0.0s
   => [1/4] FROM docker.io/library/python:alpine@sha256:ef097620baf1272e38264207003b0982285da3236a20ed829bf6bbf1e85fe3cb           0.0s
   => [internal] load build context                                                                                                 0.0s
   => => transferring context: 2.40kB                                                                                               0.0s
   => CACHED [2/4] WORKDIR /serverdata                                                                                              0.0s
   => CACHED [3/4] ADD generate_random_text.py .                                                                                    0.0s
   => CACHED [4/4] ADD send_data.py .                                                                                               0.0s
   => exporting to image                                                                                                            0.0s
   => => exporting layers                                                                                                           0.0s
   => => writing image sha256:8c5e5f929399a2c8e73a6e1945277422a8daa7e2700b92ff7e52ed931d6f86ba                                      0.0s
   => => naming to docker.io/library/server-image                                                                                   0.0s
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ █
```

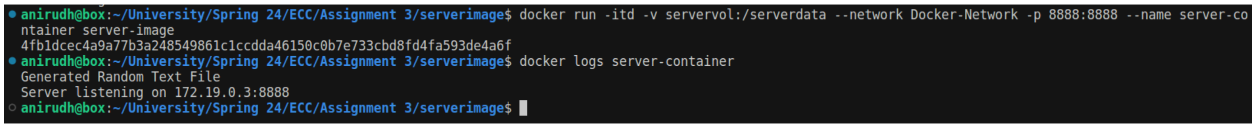
-> docker tag server-image anpenma/server-image:latest

-> docker push anpenma/server-image



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker tag server-image anpenma/server-image:latest
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker push anpenma/server-image
  Using default tag: latest
  The push refers to repository [docker.io/anpenma/server-image]
  b6b374306785: Pushed
  37fa228fe645: Pushed
  f451ad680339: Pushed
  d46b5001af7f: Mounted from library/python
  6c673d8c5e6c: Mounted from library/python
  80fef791f8cf: Mounted from library/python
  4c9c2b9681ab: Mounted from library/python
  d4fc045c9e3a: Mounted from library/alpine
  latest: digest: sha256:4c21aa8410d1a67697a615e8b905836896cbe14c7078d8e19dbebd51a40a86e0 size: 1989
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ █
```

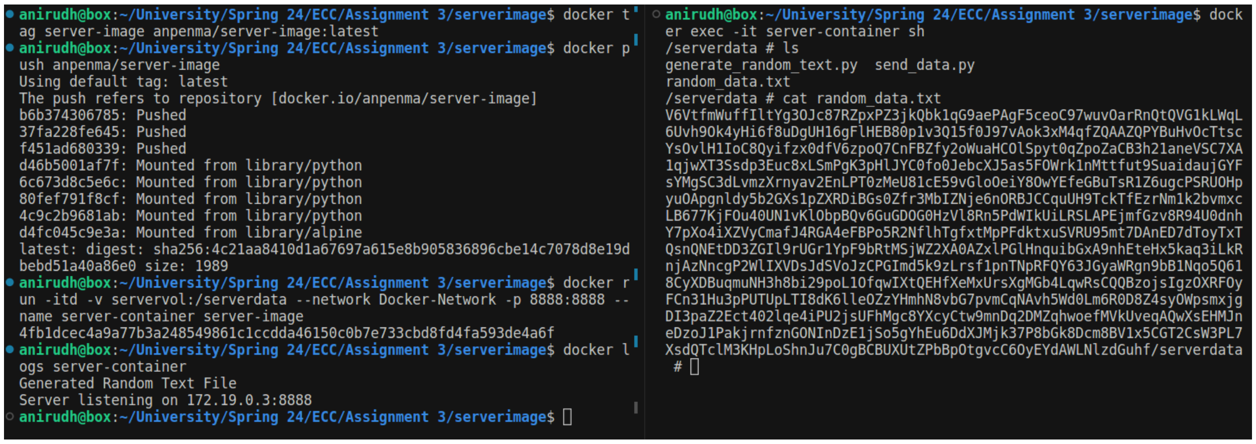
-> docker run -itd -v servervol:/serverdata --network Docker-Network -p 8888:8888 --name server-container server-image



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker run -itd -v servervol:/serverdata --network Docker-Network -p 8888:8888 --name server-co
  ntainer server-image
  4fb1dcec4a9a77b3a248549861c1ccdda46150c0b7e733cbd8fd4fa593de4a6f
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker logs server-container
  Generated Random Text File
  Server listening on 172.19.0.3:8888
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ █
```

-> docker exec -it server-container sh

-> ls && cat random_data.txt



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker t
  ag server-image anpenma/server-image:latest
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker p
  ush anpenma/server-image
  Using default tag: latest
  The push refers to repository [docker.io/anpenma/server-image]
  b6b374306785: Pushed
  37fa228fe645: Pushed
  f451ad680339: Pushed
  d46b5001af7f: Mounted from library/python
  6c673d8c5e6c: Mounted from library/python
  80fef791f8cf: Mounted from library/python
  4c9c2b9681ab: Mounted from library/python
  d4fc045c9e3a: Mounted from library/alpine
  latest: digest: sha256:4c21aa8410d1a67697a615e8b905836896cbe14c7078d8e19d
  bebd51a40a86e0 size: 1989
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker r
  un -itd -v servervol:/serverdata --network Docker-Network -p 8888:8888 --
  name server-container server-image
  4fb1dcec4a9a77b3a248549861c1ccdda46150c0b7e733cbd8fd4fa593de4a6f
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ docker l
  ogs server-container
  Generated Random Text File
  Server listening on 172.19.0.3:8888
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ 
```

```
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/serverimage$ dock
  er exec -it server-container sh
  /serverdata # ls
  generate_random_text.py  send_data.py
  random_data.txt
  /serverdata # cat random_data.txt
  V6VtfmWuffIltYg3OJc87RZpxPZ3jkQbk1qG9aePAgF5ceoC97wuvOarRnQtQVG1kLWqL
  6Uvh9Ok4yHi6f8uDgUH16gFlHEB80p1v3Q15f0J97vAok3xM4qfZQAAZQPYBuHvOcTtsc
  YsOvlH1IoC8Qyifzx0dfV6zpoQ7CnFBZfy2oWuaHCOlSpyt0qZpoZaCB3h21aneVSC7XA
  1qjwXT3Ssdp3Euc8xLSmPgK3pHlJYC0fo0JebcXJ5as5FOWrk1nMttfut9SuaidaujGYF
  sYMgSC3dLvmzXrnyav2EnLPT0zMeU81cE59vGloOeiY8OwYEfeGBuTsR1Z6ugcPSRUOHp
  yuOApgnldy5b2GXs1pZXRDiBGs0Zfr3MbIZNje6nORBJCCquUH9TckTfEzrNm1k2bvmxc
  LB677KjFOu40UN1vKlObpBQv6GuGDOG0HzVl8Rn5PdWIkUiLRSLAPEjmfGzv8R94U0dnh
  Y7pXo4iXZVyCmafJ4RGA4eFBPo5R2NflhTgfxtMpPFdktxuSVRU95mt7DAnED7dToyTxT
  QsnQNEtDD3ZGIl9rUGr1YpF9bRtMSjWZ2XA0AZxlPGlHnquibGxA9nhEteHx5kaq3iLkR
  njAzNncgP2WlIXVDsJdSVoJzCPGImd5k9zLrsf1pnTNpRFQY63JGyaWRgn9bB1Nqo5Q61
  8CyXDBuqmuNH3h8bi29poL1OfqwIXtQEHfXeMxUrsXgMGb4LqwRsCQQBzojsIgzOXRFOy
  FCn31Hu3pPUTUpLTI8dK6lleOZzYHmhN8vbG7pvmCqNAvh5Wd0Lm6R0D8Z4syOWpsmxjg
  DI3paZ2Ect402lqe4iPU2jsUFhMgc8YXcyCtw9mnDq2DMZqhwoefMVkUveqAQwXsEHMJn
  eDzoJ1PakjrnfznGONInDzE1jSo5gYhEu6DdXJMjk37P8bGk8Dcm8BV1x5CGT2CsW3PL7
  XsdQTclM3KHpLoShnJu7C0gBCBUXUtZPbBpOtgvcC6OyEYdAWLNlzdGuhf/serverdata
   # 
```

In the above image, we can see on the left side that the “random_data.txt” has been generated by “generate_random_text.py” file and immediately after that the “send_data.py” is run with the server’s ip address showing the print statement listening on ip:port. And on the right side we can see the content of random_data.txt

**Client:**

**First, create a volume by name “clientvol”**
-> docker volume create clientvol

**Next, mount the “clientvol” in “/clientdata”**
-> This will be done in the Docker run command itself. It will be mentioned at the final part of the server.

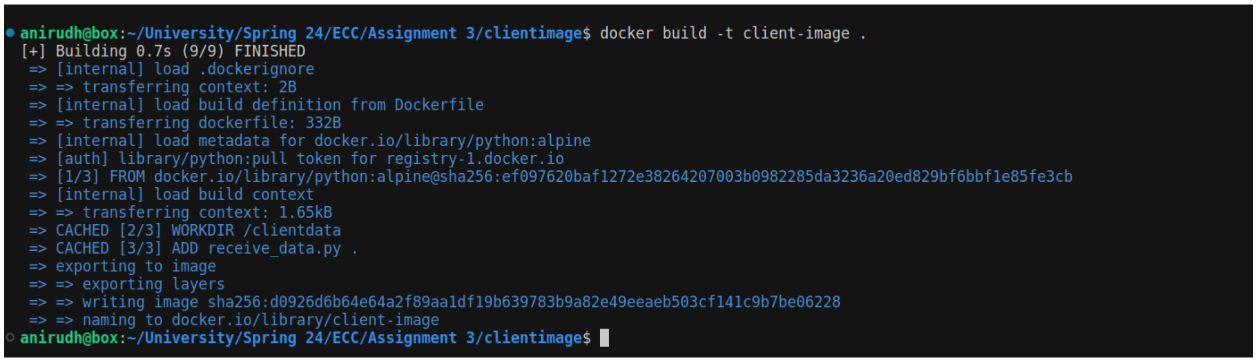
**Container should run an application on startup which will connect to the server, receive the file that the server sends and save it in “/clientdata”**
-> The script receive_data

**Verify that the file is received properly at the clientside by verifying the checksum**
-> To do this hashlib has a function called hexdigest that can check if it’s the same as the checksum

**The container should include all the packages that are required to run your application. Choose an appropriate base image and install only the necessary packages**
-> The base image I chose for the client is the same as the server. And it comes with all necessary packages already installed.

**Commands used for the Client Container:**

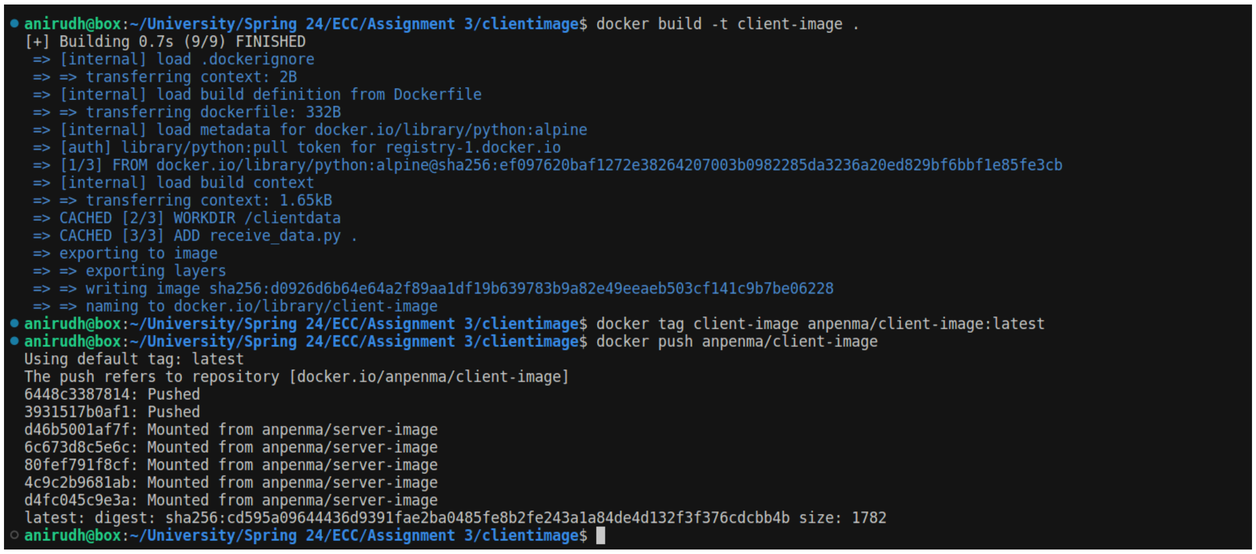
-> docker build -t client-image .



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker build -t client-image .
  [+] Building 0.7s (9/9) FINISHED
   => [internal] load .dockerignore
   => => transferring context: 2B
   => [internal] load build definition from Dockerfile
   => => transferring dockerfile: 332B
   => [internal] load metadata for docker.io/library/python:alpine
   => [auth] library/python:pull token for registry-1.docker.io
   => [1/3] FROM docker.io/library/python:alpine@sha256:ef097620baf1272e38264207003b0982285da3236a20ed829bf6bbf1e85fe3cb
   => [internal] load build context
   => => transferring context: 1.65kB
   => CACHED [2/3] WORKDIR /clientdata
   => CACHED [3/3] ADD receive_data.py .
   => exporting to image
   => => exporting layers
   => => writing image sha256:d0926d6b64e64a2f89aa1df19b639783b9a82e49eeaeb503cf141c9b7be06228
   => => naming to docker.io/library/client-image
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ █
```

-> docker tag client-image anpenma/client-image:latest

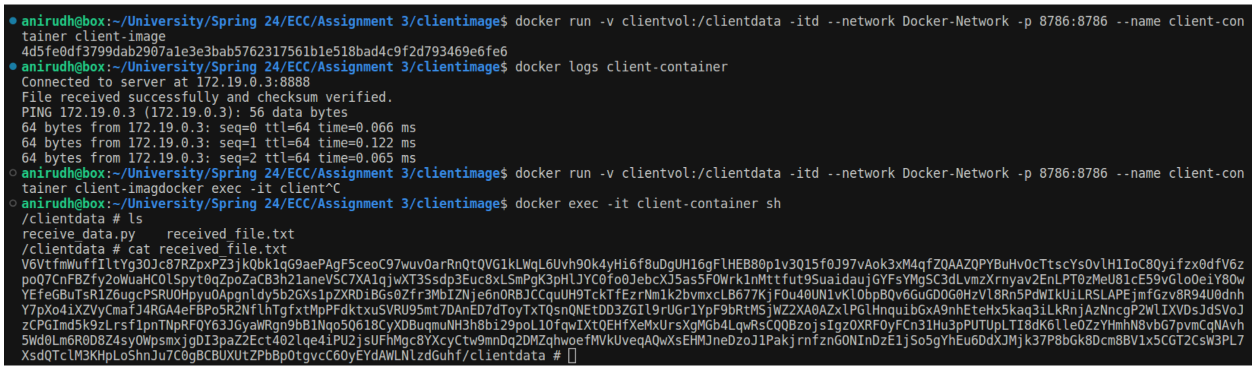
-> docker push anpenma/client-image



```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker build -t client-image .
  [+] Building 0.7s (9/9) FINISHED
   => [internal] load .dockerignore
   => => transferring context: 2B
   => [internal] load build definition from Dockerfile
   => => transferring dockerfile: 332B
   => [internal] load metadata for docker.io/library/python:alpine
   => [auth] library/python:pull token for registry-1.docker.io
   => [1/3] FROM docker.io/library/python:alpine@sha256:ef097620baf1272e38264207003b0982285da3236a20ed829bf6bbf1e85fe3cb
   => [internal] load build context
   => => transferring context: 1.65kB
   => CACHED [2/3] WORKDIR /clientdata
   => CACHED [3/3] ADD receive_data.py .
   => exporting to image
   => => exporting layers
   => => writing image sha256:d0926d6b64e64a2f89aa1df19b639783b9a82e49eeaeb503cf141c9b7be06228
   => => naming to docker.io/library/client-image
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker tag client-image anpenma/client-image:latest
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker push anpenma/client-image
  Using default tag: latest
  The push refers to repository [docker.io/anpenma/client-image]
  6448c3387814: Pushed
  3931517b0af1: Pushed
  d46b5001af7f: Mounted from anpenma/server-image
  6c673d8c5e6c: Mounted from anpenma/server-image
  80fef791f8cf: Mounted from anpenma/server-image
  4c9c2b9681ab: Mounted from anpenma/server-image
  d4fc045c9e3a: Mounted from anpenma/server-image
  latest: digest: sha256:cd595a09644436d9391fae2ba0485fe8b2fe243a1a84de4d132f3f376cdcbb4b size: 1782
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ █
```

-> docker run -v clientvol:/clientdata -itd --network Docker-Network -p 8786:8786 --name client-container client-image

-> docker exec -it client-container sh

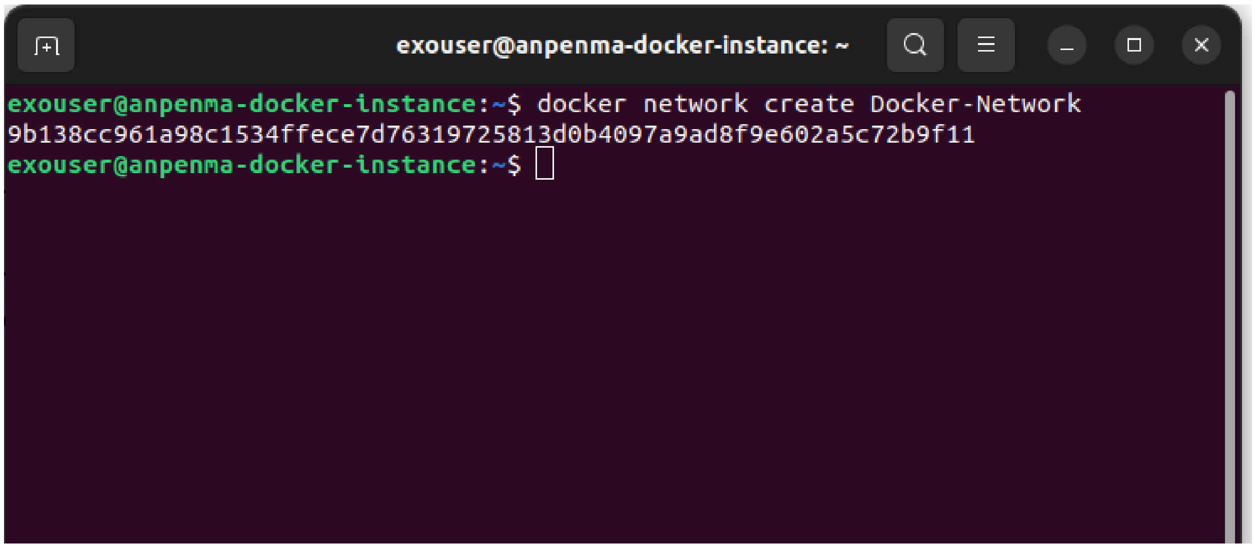


```
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker run -v clientvol:/clientdata -itd --network Docker-Network -p 8786:8786 --name client-con
  tainer client-image
  4d5fe0df3799dab2907a1e3e3bab5762317561b1e518bad4c9f2d793469e6fe6
● anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker logs client-container
  Connected to server at 172.19.0.3:8888
  File received successfully and checksum verified.
  PING 172.19.0.3 (172.19.0.3): 56 data bytes
  64 bytes from 172.19.0.3: seq=0 ttl=64 time=0.066 ms
  64 bytes from 172.19.0.3: seq=1 ttl=64 time=0.122 ms
  64 bytes from 172.19.0.3: seq=2 ttl=64 time=0.065 ms
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker run -v clientvol:/clientdata -itd --network Docker-Network -p 8786:8786 --name client-con
  tainer client-imagdocker exec -it client^C
○ anirudh@box:~/University/Spring 24/ECC/Assignment 3/clientimage$ docker exec -it client-container sh
  /clientdata # ls
  receive_data.py    received_file.txt
  /clientdata # cat received_file.txt
  V6VtfmWuffIltYg3OJc87RZpxPZ3jkQbk1qG9aePAgF5ceoC97wuvOarRnQtQVG1kLWqL6Uvh9Ok4yHi6f8uDgUH16gFlHEB80p1v3Q15f0J97vAok3xM4qfZQAAZQPYBuHvOcTtscYsOvlH1IoC8Qyifzx0dfV6z
  poQ7CnFBZfy2oWuaHCOlSpyt0qZpoZaCB3h21aneVSC7XA1qjwXT3Ssdp3Euc8xLSmPgK3pHlJYC0fo0JebcXJ5as5FOWrk1nMttfut9SuaidaujGYFsYMgSC3dLvmzXrnyav2EnLPT0zMeU81cE59vGloOeiY8Ow
  YEfeGBuTsR1Z6ugcPSRUOHpyuOApgnldy5b2GXs1pZXRDiBGs0Zfr3MbIZNje6nORBJCCquUH9TckTfEzrNm1k2bvmxcLB677KjFOu40UN1vKlObpBQv6GuGDOG0HzVl8Rn5PdWIkUiLRSLAPEjmfGzv8R94U0dnh
  Y7pXo4iXZVyCmafJ4RGA4eFBPo5R2NflhTgfxtMpPFdktxuSVRU95mt7DAnED7dToyTxTQsnQNEtDD3ZGIl9rUGr1YpF9bRtMSjWZ2XA0AZxlPGlHnquibGxA9nhEteHx5kaq3iLkRnjAzNncgP2WlIXVDsJdSVoJ
  zCPGImd5k9zLrsf1pnTNpRFQY63JGyaWRgn9bB1Nqo5Q618CyXDBuqmuNH3h8bi29poL1OfqwIXtQEHfXeMxUrsXgMGb4LqwRsCQQBzojsIgzOXRFOyFCn31Hu3pPUTUpLTI8dK6lleOZzYHmhN8vbG7pvmCqNAvh
  5Wd0Lm6R0D8Z4syOWpsmxjgDI3paZ2Ect402lqe4iPU2jsUFhMgc8YXcyCtw9mnDq2DMZqhwoefMVkUveqAQwXsEHMJneDzoJ1PakjrnfznGONInDzE1jSo5gYhEu6DdXJMjk37P8bGk8Dcm8BV1x5CGT2CsW3PL7
  XsdQTclM3KHpLoShnJu7C0gBCBUXUtZPbBpOtgvcC6OyEYdAWLNlzdGuhf/clientdata # 
```

We can see in the logs that the file is received and checksum has been verified and by manually also checking the content by entering the client container, we see that the received file is the same as the originally generated random text in the server.

**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).**

-> docker network create Docker-Network



```
exouser@anpenma-docker-instance:~$ docker network create Docker-Network
9b138cc961a98c1534ffece7d76319725813d0b4097a9ad8f9e602a5c72b9f11
exouser@anpenma-docker-instance:~$ 
```