School of Computer Science

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

Containers & Docker Security

Lab File (2022-2026)

5th Semester

Submitted To:

Dr. Hitesh Kumar Sharma

Submitted By:

Khushi Jain

500105912

R2142220336

B Tech CSE

DevOps(B1)

EXPERIMENT 4

AIM: Working with Docker Networking

Step 1: Understanding Docker Default Networks

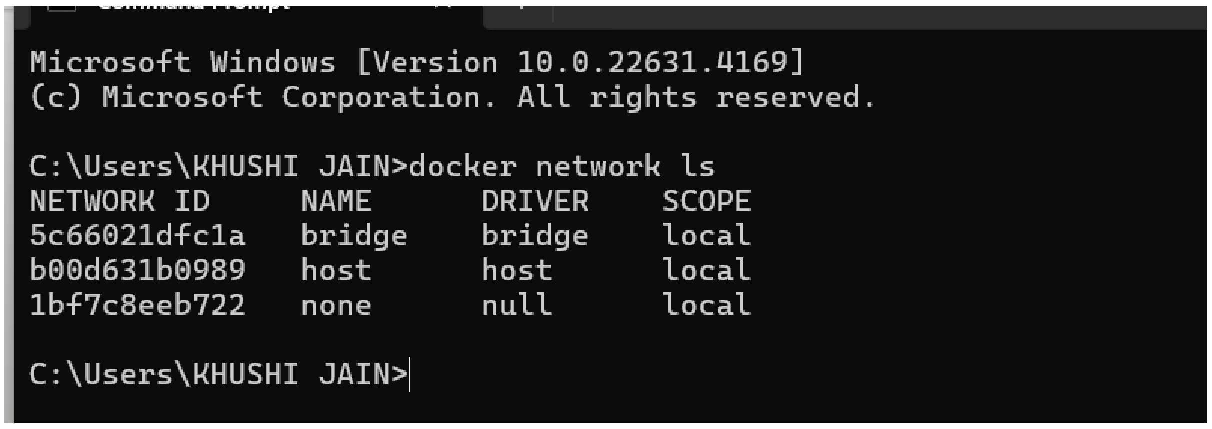
Docker provides three default networks:

- bridge: The default network when a container starts.
- host: Bypasses Docker's network isolation and attaches the container directly to the host network.
- none: No networking is available for the container.

1.1. Inspect Default Networks

Check Docker's default networks using:

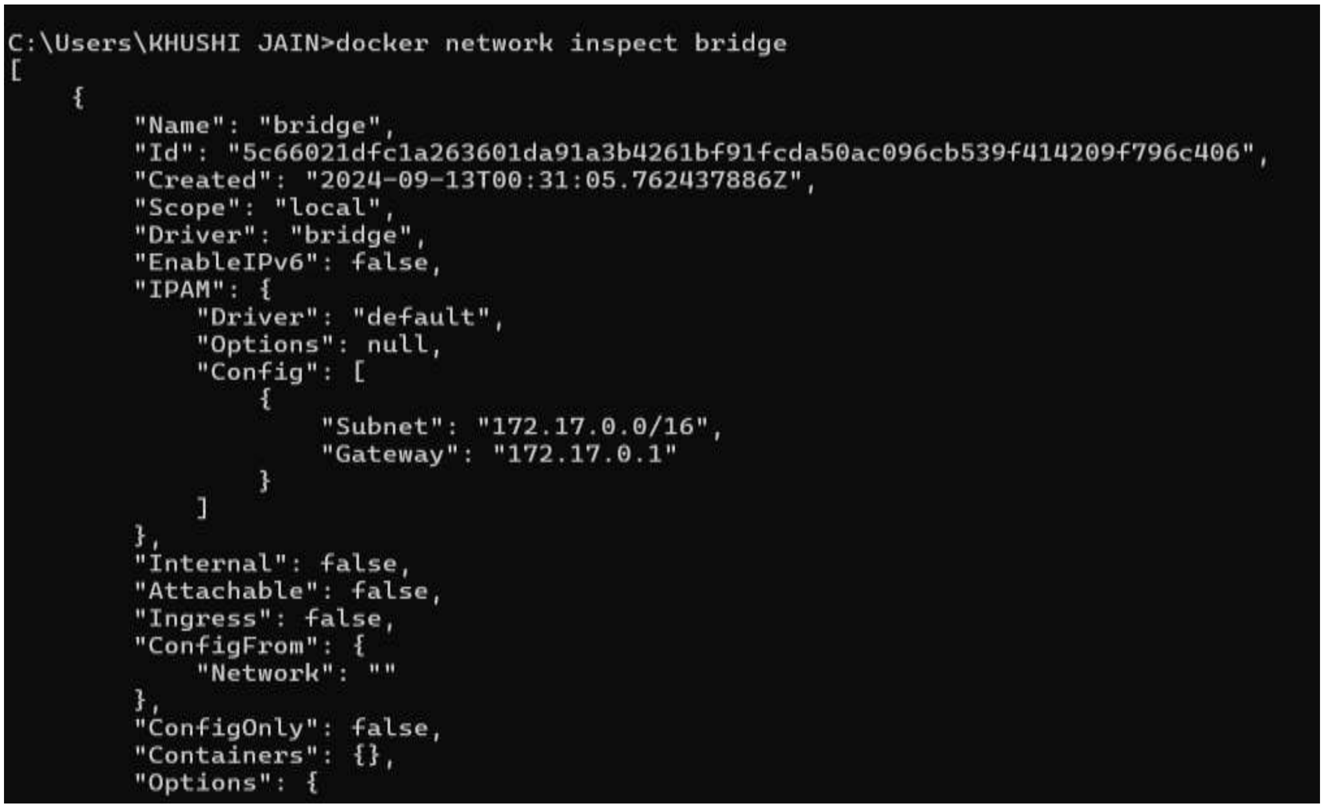
```
docker network ls
```



```
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\KHUSHI JAIN>docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
5c66021dfc1a   bridge    bridge    local
b00d631b0989   host      host      local
1bf7c8eeb722   none      null      local

C:\Users\KHUSHI JAIN>
```

1.2. Inspect the Bridge Network

```
docker network inspect bridge
```



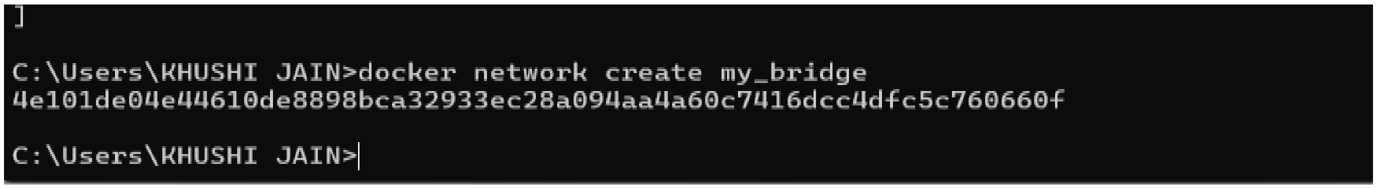
```
C:\Users\KHUSHI JAIN>docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "5c66021dfc1a263601da91a3b4261bf91fcda50ac096cb539f414209f796c406",
        "Created": "2024-09-13T00:31:05.762437886Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
```

Step 2: Create and Use a Bridge Network

2.1. Create a User-Defined Bridge Network

A user-defined bridge network allows containers to communicate by name instead of IP.

```
docker network create my_bridge
```

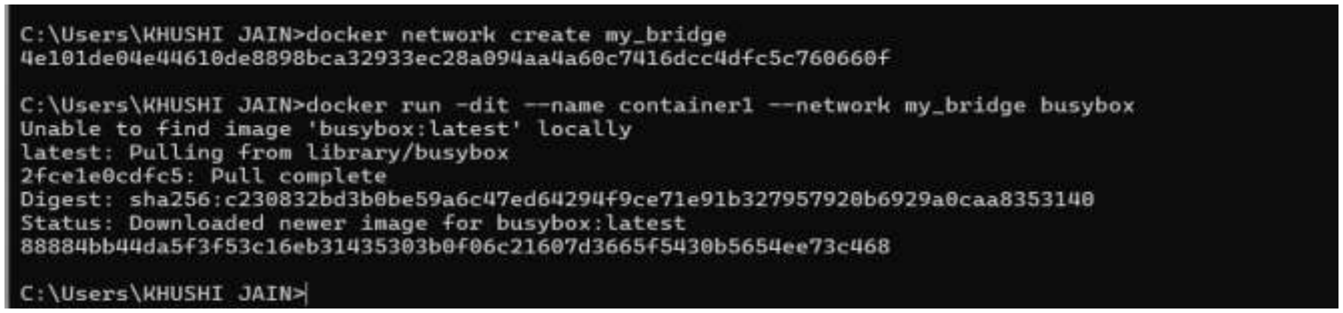


```
]

C:\Users\KHUSHI JAIN>docker network create my_bridge
4e101de04e44610de8898bca32933ec28a094aa4a60c7416dcc4dfc5c760660f

C:\Users\KHUSHI JAIN>
```

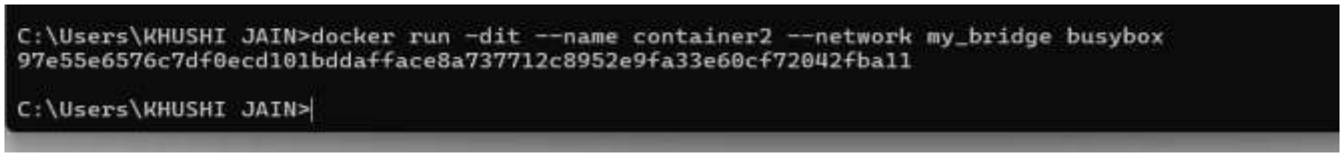
2.2. Run Containers on the User-Defined Network

Start two containers on the newly created my_bridge network:

```
docker run -dit --name container2 --network my_bridge busybox

docker run -dit --name container1 --network my_bridge busybox
```



```
C:\Users\KHUSHI JAIN>docker network create my_bridge
4e101de04e44610de8898bca32933ec28a094aa4a60c7416dcc4dfc5c760660f

C:\Users\KHUSHI JAIN>docker run -dit --name container1 --network my_bridge busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
2fce1e0cdfc5: Pull complete
Digest: sha256:c230832bd3b0be59a6c47ed64294f9ce71e91b327957920b6929a0caa8353140
Status: Downloaded newer image for busybox:latest
88884bb44da5f3f53c16eb31435303b0f06c21607d3665f5430b5654ee73c468

C:\Users\KHUSHI JAIN>
```

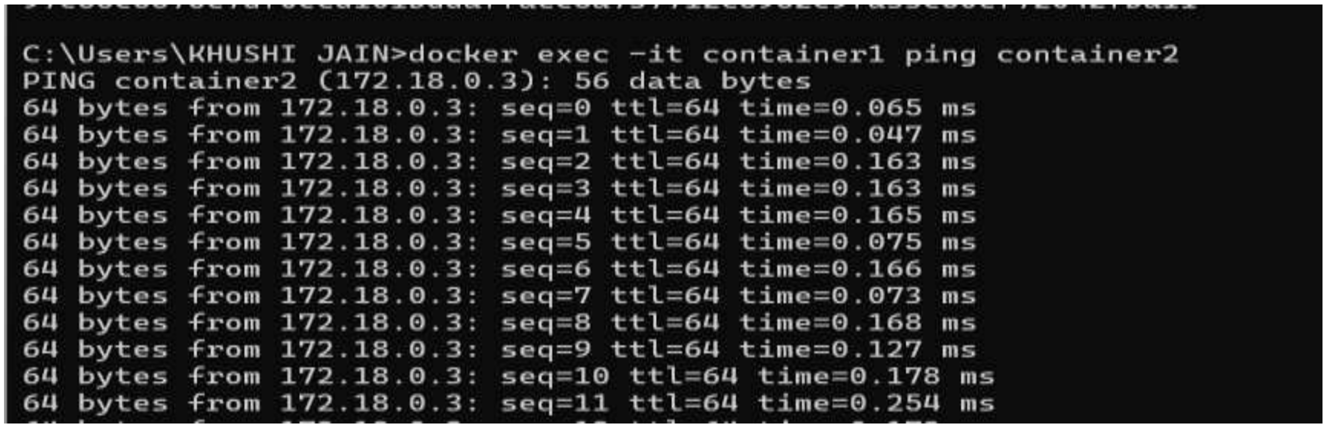


```
C:\Users\KHUSHI JAIN>docker run -dit --name container2 --network my_bridge busybox
97e55e6576c7df0ecd101bddafface8a737712c8952e9fa33e60cf72042fba11

C:\Users\KHUSHI JAIN>
```

2.3. Test Container Communication

Execute a ping command from container1 to container2 using container names:

```
docker exec -it container1 ping container2
```



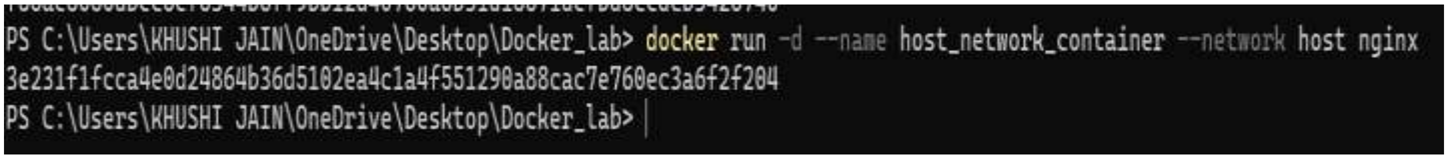
```
C:\Users\KHUSHI JAIN>docker exec -it container1 ping container2
PING container2 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: seq=0 ttl=64 time=0.065 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.047 ms
64 bytes from 172.18.0.3: seq=2 ttl=64 time=0.163 ms
64 bytes from 172.18.0.3: seq=3 ttl=64 time=0.163 ms
64 bytes from 172.18.0.3: seq=4 ttl=64 time=0.165 ms
64 bytes from 172.18.0.3: seq=5 ttl=64 time=0.075 ms
64 bytes from 172.18.0.3: seq=6 ttl=64 time=0.166 ms
64 bytes from 172.18.0.3: seq=7 ttl=64 time=0.073 ms
64 bytes from 172.18.0.3: seq=8 ttl=64 time=0.168 ms
64 bytes from 172.18.0.3: seq=9 ttl=64 time=0.127 ms
64 bytes from 172.18.0.3: seq=10 ttl=64 time=0.178 ms
64 bytes from 172.18.0.3: seq=11 ttl=64 time=0.254 ms
```

Step 3: Create and Use a Host Network

3.1. Run a Container Using the Host Network

The host network allows the container to use the host machine's networking stack:

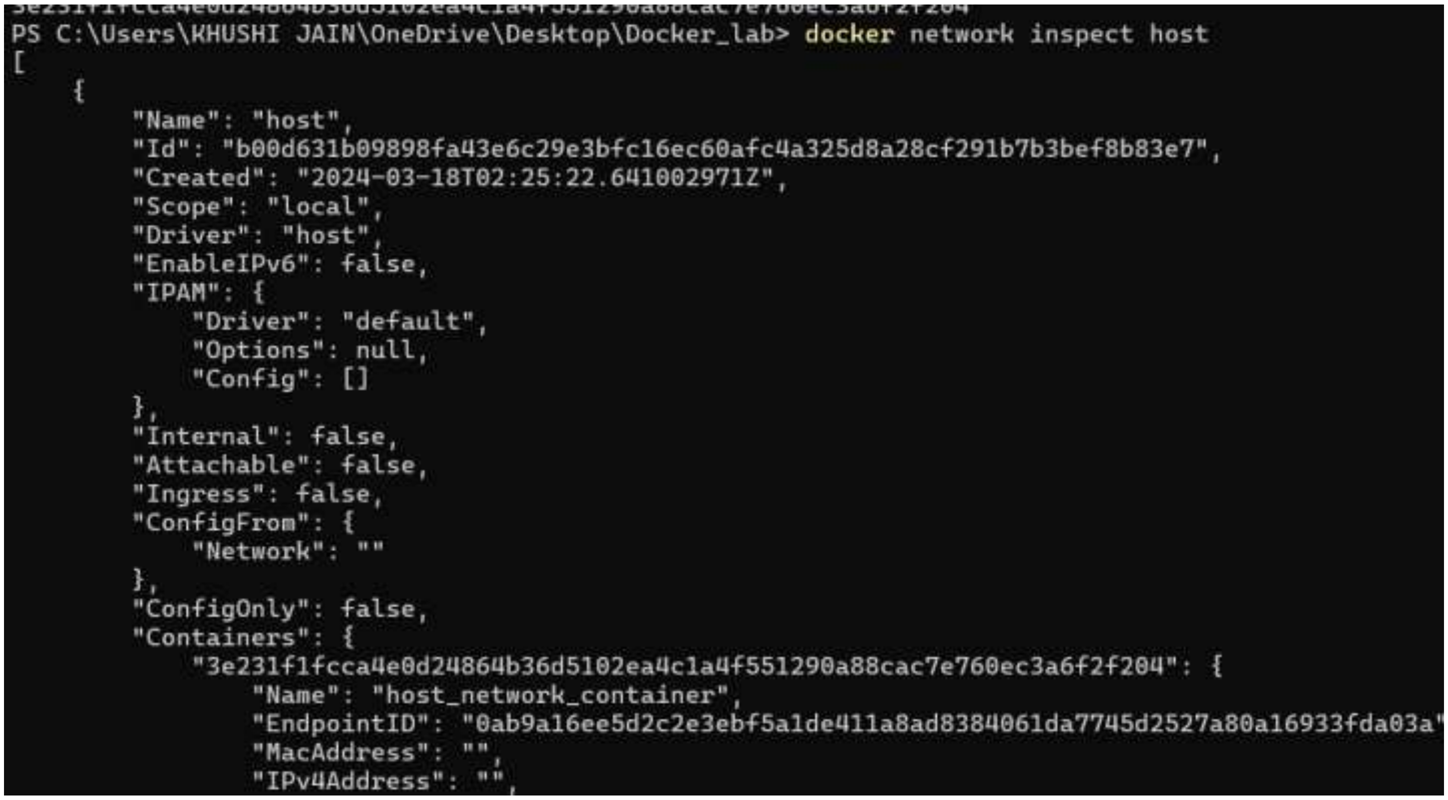
```
docker run -d --name host_network_container --network host nginx
```



```
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab> docker run -d --name host_network_container --network host nginx
3e231f1fcca4e0d24864b36d5102ea4c1a4f551290a88cac7e760ec3a6f2f204
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab> |
```

3.2. Check Network

```
docker network inspect host
```



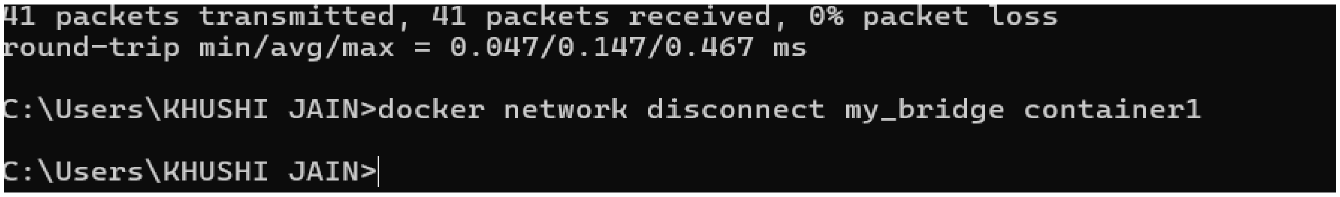
```
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab> docker network inspect host
[
    {
        "Name": "host",
        "Id": "b00d631b09898fa43e6c29e3bfc16ec60afc4a325d8a28cf291b7b3bef8b83e7",
        "Created": "2024-03-18T02:25:22.641002971Z",
        "Scope": "local",
        "Driver": "host",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": []
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "3e231f1fcca4e0d24864b36d5102ea4c1a4f551290a88cac7e760ec3a6f2f204": {
                "Name": "host_network_container",
                "EndpointID": "0ab9a16ee5d2c2e3ebf5a1de411a8ad8384061da7745d2527a80a16933fda03a"
                "MacAddress": "",
                "IPv4Address": "",
```

Step 4: Disconnect and Remove Networks

4.1. Disconnect Containers from Networks

To disconnect container1 from my_bridge:

```
docker network disconnect my_bridge container1
```

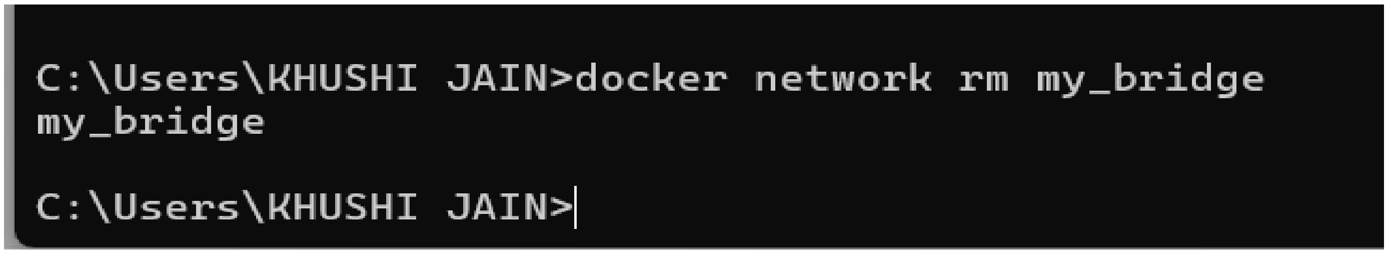


```
41 packets transmitted, 41 packets received, 0% packet loss
round-trip min/avg/max = 0.047/0.147/0.467 ms

C:\Users\KHUSHI JAIN>docker network disconnect my_bridge container1

C:\Users\KHUSHI JAIN>
```

4.2. Remove Networks

To remove the user-defined network:

```
docker network rm my_bridge
```

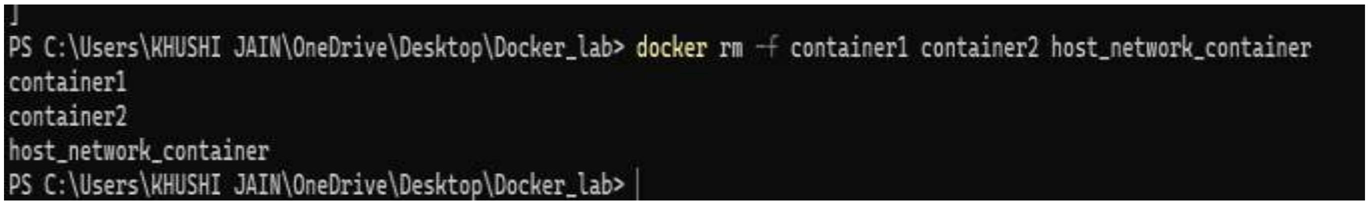


```
C:\Users\KHUSHI JAIN>docker network rm my_bridge
my_bridge

C:\Users\KHUSHI JAIN>
```

Step 5: Clean Up

Stop and remove all containers created during this exercise:

```
docker rm -f container1 container2 host_network_container
```



```
]
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab> docker rm -f container1 container2 host_network_container
container1
container2
host_network_container
PS C:\Users\KHUSHI JAIN\OneDrive\Desktop\Docker_lab>
```