Lab Exercise 3- 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
```

1.2. Inspect the Bridge Network

```
docker network inspect bridge
```

This command will show detailed information about the bridge network, including the connected containers and IP address ranges.

```
C:\Users\HP>Docker --version
Docker version 28.3.2, build 578ccf6

C:\Users\HP>docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
6eb9606ea289   bridge    bridge    local
1c5235e57ab3   host      host      local
a617b55682ec   none      null      local

C:\Users\HP>docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "6eb9606ea28936415e7872a872c96cbdcf66a29a6aca5a2ea5b861deda8002df",
        "Created": "2026-01-21T04:54:30.741645302Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv4": true,
        "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": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
    }
]
```

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
```

2.2. Run Containers on the User-Defined Network

Start two containers on the newly created my_bridge network:

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

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

2.3. Test Container Communication

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

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

The containers should be able to communicate since they are on the same network.

```
C:\Users\HP>docker network create my_bridge
53f983bb756c8460427e4524386683121807caa4d67aa48fdc809b3788b376cb

C:\Users\HP>docker run -dit --name container1 --network my_bridge busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
e59838ecfec5: Pull complete
Digest: sha256:2383baad1860bbe9d8a7a843775048fd07d8afe292b94bd876df64a69aae7cb1
Status: Downloaded newer image for busybox:latest
eef95713a80268d2f906a97aa03d255324448a92c3390a91d13081317f1f3cb6

C:\Users\HP>docker run -dit --name container2 --network my_bridge busybox
751f3715f88542aa0cbf153417566fdb9ac7b780c77839178e938c56560b9e78

C:\Users\HP>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=0.088 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.222 ms
64 bytes from 172.18.0.3: seq=2 ttl=64 time=0.115 ms
64 bytes from 172.18.0.3: seq=3 ttl=64 time=0.137 ms
64 bytes from 172.18.0.3: seq=4 ttl=64 time=0.148 ms
64 bytes from 172.18.0.3: seq=5 ttl=64 time=0.330 ms
64 bytes from 172.18.0.3: seq=6 ttl=64 time=0.089 ms
64 bytes from 172.18.0.3: seq=7 ttl=64 time=0.134 ms
```

Step 3: Disconnect and Remove Networks

3.1. Disconnect Containers from Networks

To disconnect container1 from my_bridge:

```
docker network disconnect my_bridge container1
```

```
C:\Users\HP>docker network disconnect my_bridge container1
```

4.2. Remove Networks

To remove the user-defined network:

```
docker network rm my_bridge
```

```
C:\Users\HP>docker stop container2
container2

C:\Users\HP>docker network rm my_bridge
my_bridge
```

Step 4: Clean Up

Stop and remove all containers created during this exercise:

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

```
C:\Users\HP>docker rm -f container1 container2
container1
container2
```