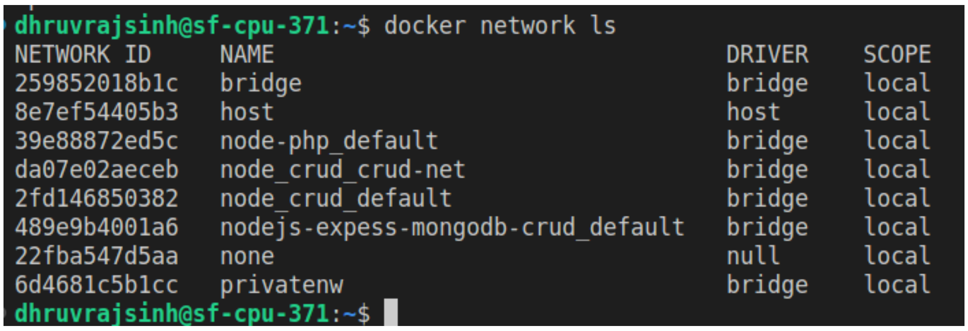
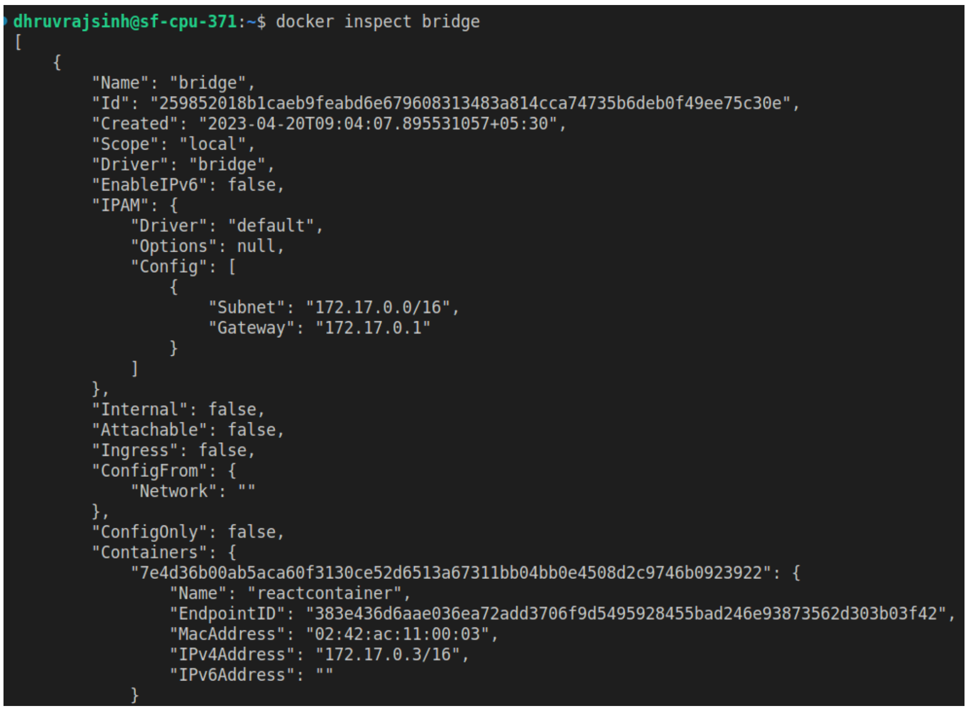
<u>Exercise_6: Network</u>

- To list all available network in docker



```
dhruvrajsinh@sf-cpu-371:~$ docker network ls
NETWORK ID     NAME                                  DRIVER    SCOPE
259852018b1c   bridge                                bridge    local
8e7ef54405b3   host                                  host      local
39e88872ed5c   node-php_default                      bridge    local
da07e02aeceb   node_crud_crud-net                    bridge    local
2fd146850382   node_crud_default                     bridge    local
489e9b4001a6   nodejs-express-mongodb-crud_default   bridge    local
22fba547d5aa   none                                  null      local
6d4681c5b1cc   privatenw                             bridge    local
dhruvrajsinh@sf-cpu-371:~$
```

- To inspect the bridge network, we can run:



```
dhruvrajsinh@sf-cpu-371:~$ docker inspect bridge
[
    {
        "Name": "bridge",
        "Id": "259852018b1caeb9feabd6e679608313483a814cca74735b6deb0f49ee75c30e",
        "Created": "2023-04-20T09:04:07.895531057+05:30",
        "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": {
            "7e4d36b00ab5aca60f3130ce52d6513a67311bb04bb0e4508d2c9746b0923922": {
                "Name": "reactcontainer",
                "EndpointID": "383e436d6aae036ea72add3706f9d5495928455bad246e93873562d303b03f42",
                "MacAddress": "02:42:ac:11:00:03",
                "IPv4Address": "172.17.0.3/16",
                "IPv6Address": ""
            }
```

- To add the another container to the same network as the dummy container, we need to include the --network flag when running the another container.

- Assuming the network we created earlier is named **"privatenw**", we can add the another container to the network with the following command:

- To start the two postgres databases and add them to the same network, you can use the following commands:
    - docker run --rm -d --name widgetdb --network privatenw -e POSTGRES_PASSWORD=mysecretpassword postgres

- This will start two postgres containers with the names widgetdb and **privatenw,** respectively, and add them to the privatenw network. The -e flag sets the POSTGRES_PASSWORD environment variable, which is required for postgres to start up.

- Once the databases are running and on the same network, you can connect to one from the other using its container name as the host name. For example, to connect to the widgetdb database from the privatenw database.

- Sometimes its useful to access an application running in a Docker container directly, as if it were running on your host machine.