EXPERIMENT 4

AIM: Working with Docker Network

1. Create Network

Command: - docker network create backend-network

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network create backend-network

de166cf95bbb620125512711a1ddea47c28c0652c8ae4db41fc93591ecd267bd
vishalthakur@vishals-MacBook-Air Exp4 %
vishalthakur@vishals-MacBook-Air Exp4 % docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
de166cf95bbb   backend-network   bridge    local
6b6e6b5c6ad6   bridge            bridge    local
e99ef4e8fe75   host              host      local
94732bfc0c98   none              null      local
```

2. Connect To Network

Command: - docker run -d --name=red_contain --net=backend-network redis

--name refers to the name of the container. We can decide our own name for the container rather than default one.

```
vishalthakur@vishals-MacBook-Air Exp4 % docker run -d --name=red_contain --net=backend-network redis
0debb28315e1ab0632885520d5719f87168241cbf54eb9ff4a9f5535c94d6a37
vishalthakur@vishals-MacBook-Air Exp4 %
```


```
            "Networks": {
                "backend-network": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": [
                        "0debb28315e1"
                    ],
                    "NetworkID": "de166cf95bbb620125512711a1ddea47c28c0652c8ae4db41fc93591ecd267bd",
                    "EndpointID": "26635dbfa32be05fc33c7e998da06515ec8f5c78f4a07d37bb519caa74198f34",
                    "Gateway": "172.18.0.1",
                    "IPAddress": "172.18.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "MacAddress": "02:42:ac:12:00:02",
                    "DriverOpts": null
                }
            }
        }
    }
]
```


3. Network Communication

Command: - docker run --net=backend-network alpine ping -c1 red_contain

In here image of redis container is running in background. So we created a new container of alpine image which will ping container of redis image one time(-c1). And container of alpine image will automatically exit after pinging

```
vishalthakur@vishals-MacBook-Air Exp4 % docker run --net=backend-network alpine ping -c1 red_contain
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
9b18e9b68314: Already exists
Digest: sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
Status: Downloaded newer image for alpine:latest
PING red_contain (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.120 ms

--- red_contain ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.120/0.120/0.120 ms
vishalthakur@vishals-MacBook-Air Exp4 %
```

4. Create New Network

Command: - docker network create frontend-network

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network create frontend-network
92835742a5e397a201f9ad36846f74c4e83eb262c71e18ebc280aded75f09204
vishalthakur@vishals-MacBook-Air Exp4 % docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
de166cf95bbb   backend-network   bridge    local
6b6e6b5c6ad6   bridge            bridge    local
92835742a5e3   frontend-network  bridge    local
e99ef4e8fe75   host              host      local
94732bfc0c98   none              null      local
vishalthakur@vishals-MacBook-Air Exp4 %
```

5. Connect existing container to the Network

Command: - docker network connect frontend-network red_contain

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network connect frontend-network 0debb28315e1
```

We can use container id instead of container name to connect a container to a network.

```
            "Networks": {
                "backend-network": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": [
                        "0debb28315e1"
                    ],
                    "NetworkID": "de166cf95bbb620125512711a1ddea47c28c0652c8ae4db41fc93591ecd267bd",
                    "EndpointID": "26635dbfa32be05fc33c7e998da06515ec8f5c78f4a07d37bb519caa74198f34",
                    "Gateway": "172.18.0.1",
                    "IPAddress": "172.18.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "MacAddress": "02:42:ac:12:00:02",
                    "DriverOpts": null
                },
                "frontend-network": {
                    "IPAMConfig": {},
                    "Links": null,
                    "Aliases": [
                        "0debb28315e1"
                    ],
                    "NetworkID": "92835742a5e397a201f9ad36846f74c4e83eb262c71e18ebc280aded75f09204",
                    "EndpointID": "b6e1dd187b3cca7c2e8caecadf982daf7073f942b2410ca5c9190c973d6015c6",
                    "Gateway": "172.19.0.1",
                    "IPAddress": "172.19.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
```


6. Launch the web server, given it's attached to the same network it will be able to communicate with our Redis instance

Command: - docker run -d -p 3000:3000 --net=frontend-network katacoda/redis-node-docker-example


```
vishalthakur@vishals-MacBook-Air Exp4 % docker run -d -p 3000:3000 --net=frontend-network katacoda/redis-node-
docker-example
Unable to find image 'katacoda/redis-node-docker-example:latest' locally
latest: Pulling from katacoda/redis-node-docker-example
Image docker.io/katacoda/redis-node-docker-example:latest uses outdated schema1 manifest format. Please upgrad
e to a schema2 image for better future compatibility. More information at https://docs.docker.com/registry/spe
c/deprecated-schema-v1/
12b41071e6ce: Pull complete
a3ed95caeb02: Pull complete
49a025abf7e3: Pull complete
1fb1c0be01ab: Pull complete
ae8c1f781cde: Pull complete
db73207ad2ae: Pull complete
446b13034c13: Pull complete
Digest: sha256:1aae9759464f00953c8e078a0e0d0649622fef9dd5655b1491f9ee589ae904b4
Status: Downloaded newer image for katacoda/redis-node-docker-example:latest
WARNING: The requested image's platform (linux/amd64) does not match the detected host platform (linux/arm64/v
8) and no specific platform was requested
ea1b0fc21e53ed6534b4604cedd932767f81ebb23fe0df4a743db6010391fd48
vishalthakur@vishals-MacBook-Air Exp4 % curl docker:3000
curl: (6) Could not resolve host: docker
```


7. Connect Container with Alias

Command: - docker network create frontend-network2
Command: - docker network connect --alias db frontend-network2 red_contain

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network create frontend-network2
92b34105da67659b0be0a97a855ea4fdacb01a416967b3885b0a9e0a8634622e
vishalthakur@vishals-MacBook-Air Exp4 % docker network ls
\NETWORK ID     NAME               DRIVER    SCOPE
de166cf95bbb   backend-network     bridge    local
6b6e6b5c6ad6   bridge             bridge    local
92835742a5e3   frontend-network    bridge    local
92b34105da67   frontend-network2   bridge    local
e99ef4e8fe75   host               host      local
94732bfc0c98   none               null      local
vishalthakur@vishals-MacBook-Air Exp4 %
```

--alias db refers to the nickname given to the conatiner red_contain

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network connect --alias db frontend-network2 red_contain
```

- Container attempt to access a service via the name db, they will be given the IP address or our Redis Instance.

Command: - docker run --net=frontend-network2 alpine ping -c1 db

```
vishalthakur@vishals-MacBook-Air Exp4 % docker run --net=frontend-network2 alpine ping -c1 db
PING db (172.20.0.2): 56 data bytes
64 bytes from 172.20.0.2: seq=0 ttl=64 time=0.173 ms

--- db ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.173/0.173/0.173 ms
vishalthakur@vishals-MacBook-Air Exp4 %
```

8. Disconnect Containers

List all networks

Command: - docker network ls

```
vishalthakur@vishals-MacBook-Air Exp4 % docker network ls
NETWORK ID     NAME               DRIVER    SCOPE
de166cf95bbb   backend-network     bridge    local
6b6e6b5c6ad6   bridge             bridge    local
92835742a5e3   frontend-network    bridge    local
92b34105da67   frontend-network2   bridge    local
e99ef4e8fe75   host               host      local
94732bfc0c98   none               null      local
vishalthakur@vishals-MacBook-Air Exp4 %
```

- Explore the network to see which container are attached and their IP addresses.

Command: - docker network inspect frontend-network

- Disconnect container red_contain from network frontend-network

Command: - docker network disconnect frontend-network red_contain

Submitted By:-
Name: - Vishal
SAPID: - 500084112
Roll no. : - R2142201315

Submitted To: -
Name: - Hitesh Kumar Sharma