**Table

Description automatically generated with low confidenceExperiment 3 Date: - 05-02-2021**

**Aim: - To gain knowledge and practice for the docker networks.**

**Solution: -**

1. To start with we create the network with our predefined name.

 docker network create backend-network

Text

Description automatically generated

1. When we launch new containers, we can use the *--net* attribute to assign which network they should be connected to.

 docker run -d --name=redis --net=backend-network redis

Text

Description automatically generated

1. The first thing you'll notice is that Docker no longer assigns environment variables or updates the hosts file of containers. Explore using the following two commands and you'll notice it no longer mentions other containers.

docker run --net=backend-network alpine env

Graphical user interface, text

Description automatically generated

docker run --net=backend-network alpine cat /etc/hosts

Graphical user interface, text

Description automatically generated

1. Instead, the way containers can communicate via an Embedded DNS Server in Docker. This DNS server is assigned to all containers via the IP 127.0.0.11 and set in the resolv.conf file.

docker run --net=backend-network alpine cat /etc/resolv.conf

Graphical user interface, text

Description automatically generated

1. When containers attempt to access other containers via a well-known name, such as Redis, the DNS server will return the IP address of the correct Container. In this case, the fully qualified name of Redis will be redis.backend-network.

docker run --net=backend-network alpine ping -c1 redis

Graphical user interface, text

Description automatically generated

1. The first task is to create a new network in the same way.

docker network create frontend-network

Graphical user interface, text

Description automatically generated

1. When using the connect command it is possible to attach existing containers to the network.

docker network connect frontend-network redis

Graphical user interface, text

Description automatically generated

1. When we launch the web server, given it's attached to the same network it will be able to communicate with our Redis instance.

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

Graphical user interface, text

Description automatically generated

1. The following command will connect our Redis instance to the frontend-network with the alias of db.

docker network create frontend-network2

docker network connect --alias db frontend-network2 redis

Text

Description automatically generated

1. When containers attempt to access a service via the name db, they will be given the IP address of our Redis container.

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

Text

Description automatically generated

1. The following command will list all the networks on our host.

docker network ls

Text

Description automatically generated

1. We can then explore the network to see which containers are attached and their IP addresses.

docker network inspect frontend-network

Text

Description automatically generated

1. The following command disconnects the redis container from the frontend-network.

docker network disconnect frontend-network redis

A screenshot of a computer

Description automatically generated