Lab 1:

1. Network Type Custom Bridge 2. Network Name: hr-app-net 3. Dedicated Subnet: 192.168.20.0/24 4. Gateway: 192.168.20.1 5. Application Stack: Server: An NGINX container (acting as the web frontend). Client/Tester: An Alpine container (for internal diagnostics and connectivity testing).

First we will make the network with the subnet /24

The used command is

Docker network create “network name ” --driver “host-null-bridge” -–subnet “192.168.20.0/24”

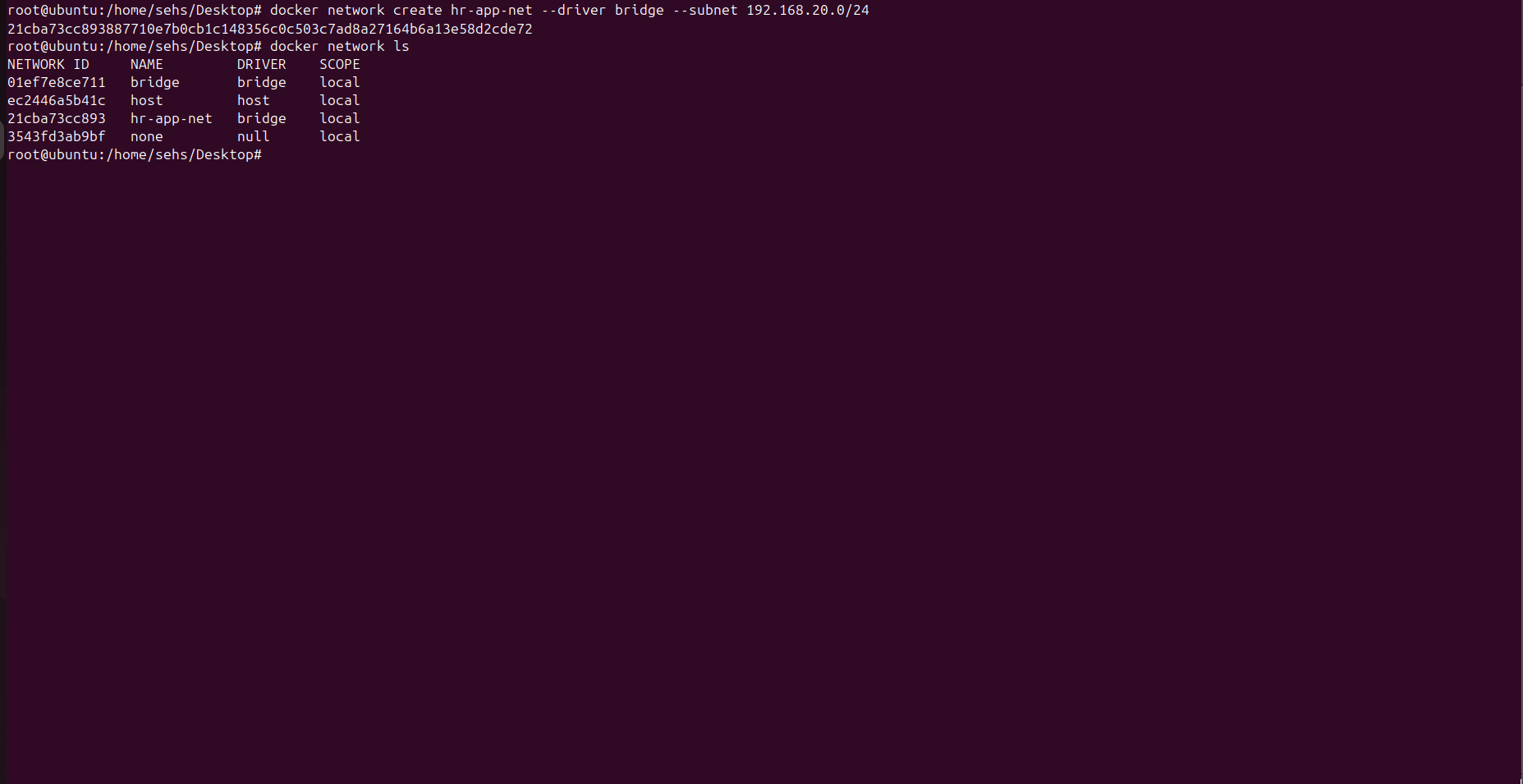
Here we used the bridge driver as requested NOTE if u used null all the containers in this network will be disconnected from the internet

If u used host it will use the NIC of the machine directly

And u can not add –driver bridge because this is the default

For the full command it will be

Docker network create hr-app-net –driver bridge –subnet 192.168.20.0/24



Then we used docker network ls

To show all networks

For the nginx image if there is no on we will use

Docker pull “image name ” for latest version

Same for the Alpine

Then we will run the container using the pulled image and assign it to the network using

Docker run -d –name “container name ” -it --network “network name ” -–memory “100m” --cpus “”0.5”” “image name”

NOTE

Its better to specify a limited resources for the container because if you did not it will be set as unlimited resources and can consume all the resources BUT if you specify it you can upgrade or downgrade it later

The command : docker run -d –name nginx-server -it -- network hr-app-net –memory 50m –cpus “0.5” nginx

-d to make it run in the background

-it for interactive terminal -i to keet the STDIN active even if there is no input -t for terminal

A computer screen with text

AI-generated content may be incorrect.

Using docker ps to show all running containers

Using docker exec -it “docker name” “shell”

Docker exec alpine-tester sh

To connect to the terminal and test ping

A screenshot of a computer program

AI-generated content may be incorrect.