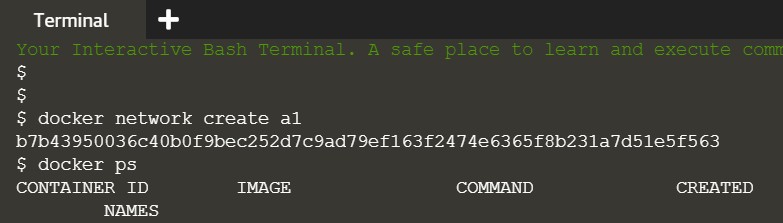
**EXPERIMENT - 3**

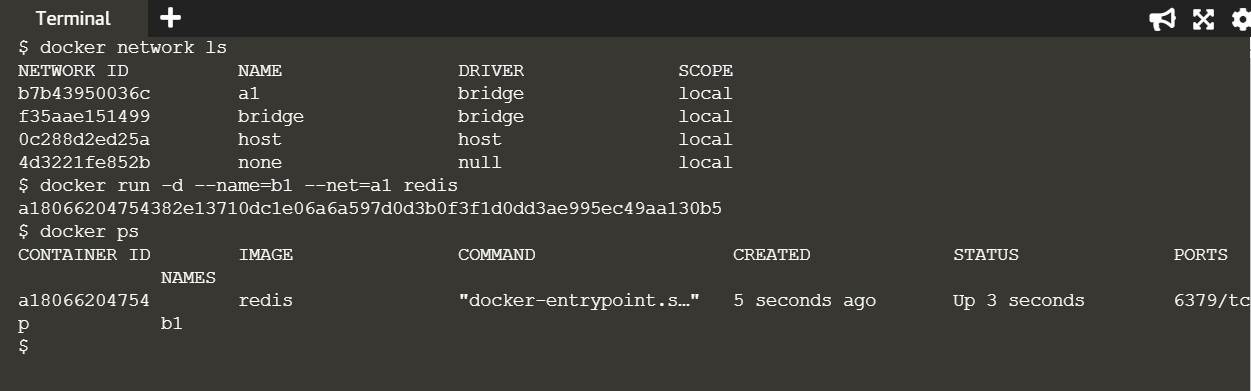
Check if any container is running

Run command – $ docker ps



Creating a container

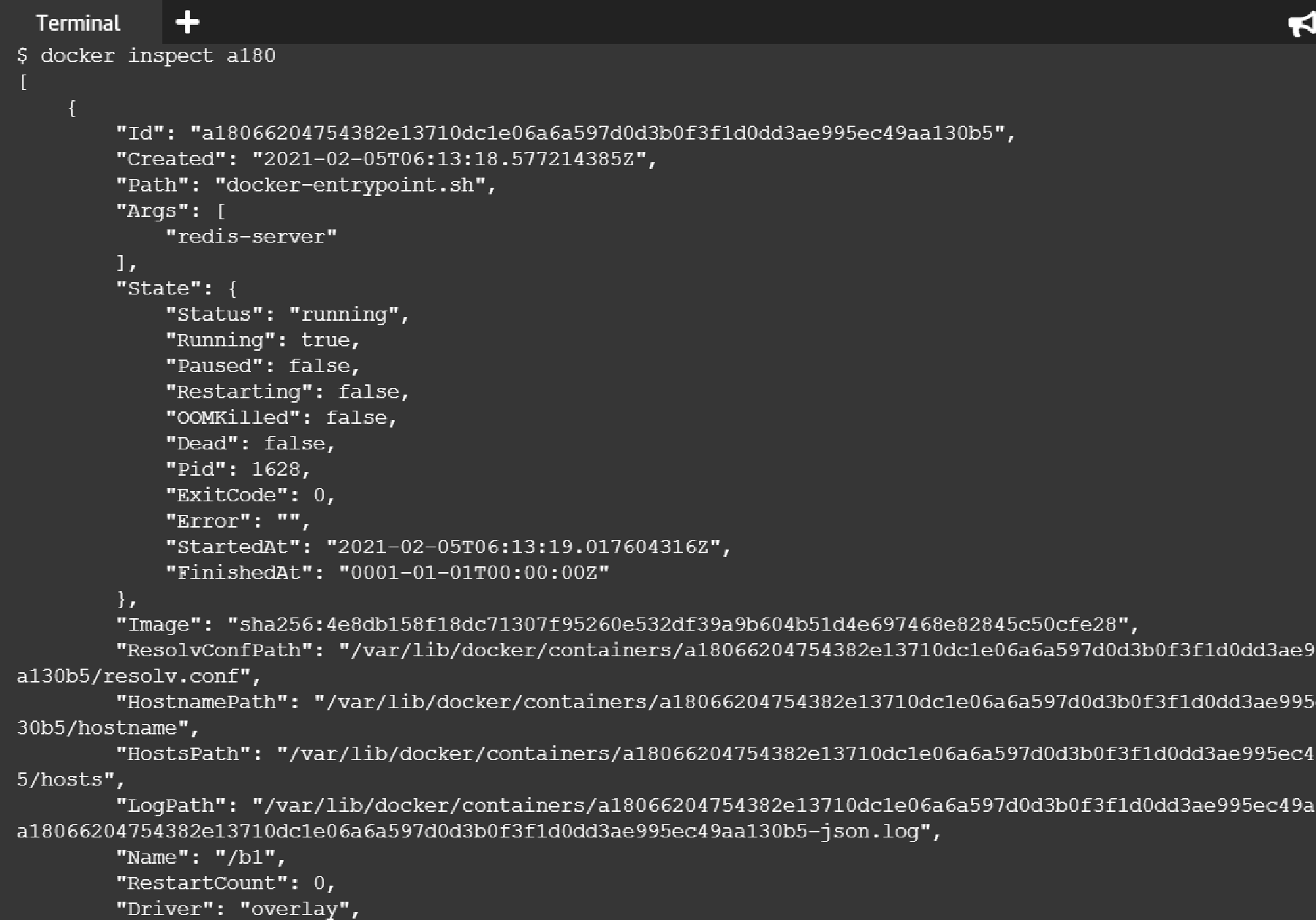
Run command – $ docker run –d –name=b1 –net=a1 redis



To check IP address

Run command - $ docker inspect a180

(Note : a180 is unique initials of container ID you want ip)





"NetworkID": "b7b4?95I5?6c40b0f9bec252d7c9ad79ef

"EndpointID": "0bda2 d 9a° 9b4?ae54b d2552f9ee9??

"Gat

"ay"

,

y 2

,-

"IPAddress": " 72. 9.u.?", "IPPrefir. fifi"4 VG; "IPvGGateway": "", "GLobalIPvGAddress": "", "GLobalIPv6Prefixlen": v,

"MacAddress": "v2:42:ac: ?::':v2",

"DriverOpts": null



"NetworkID": "b7b4?95003GC40b0f9bec252d7c9ad79ef G?f2474e

"EndpointID": "e94Gac tJdf4a522G?525d CG549?ceaG?aGdGt t?

"Gat

"ay"

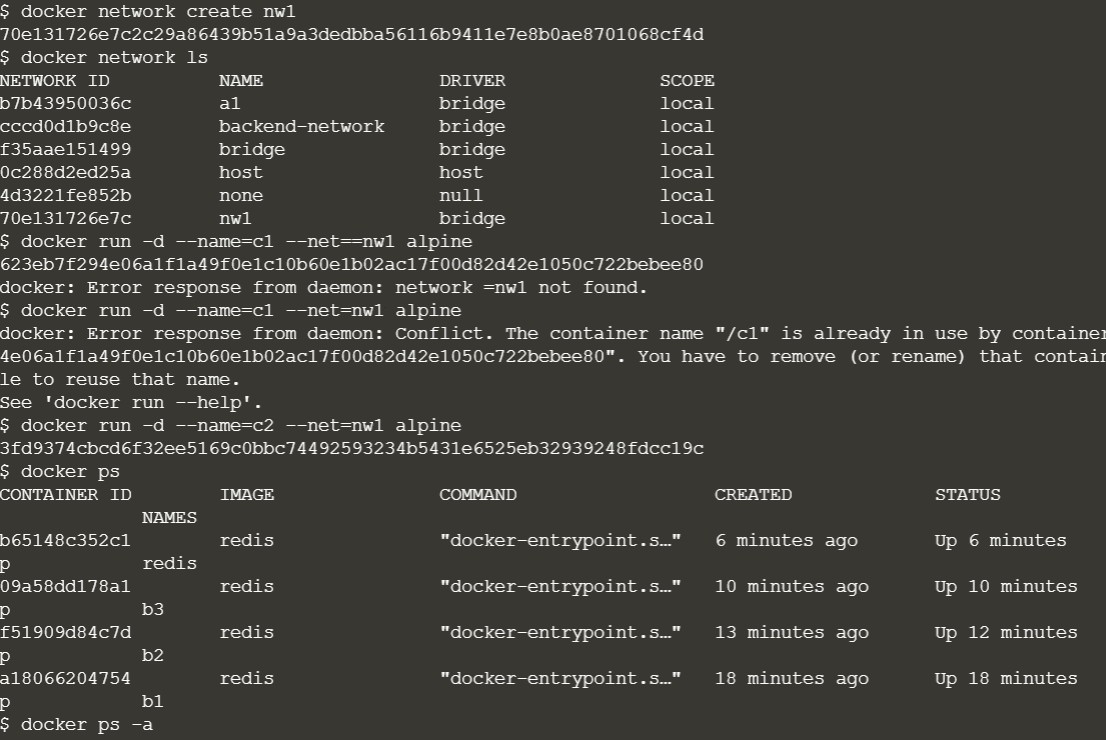
yy

,-,

"IPA.H.Press": " 72. ?•.1.?",

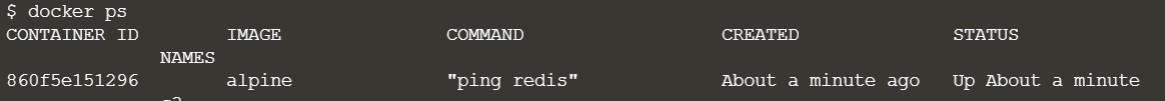
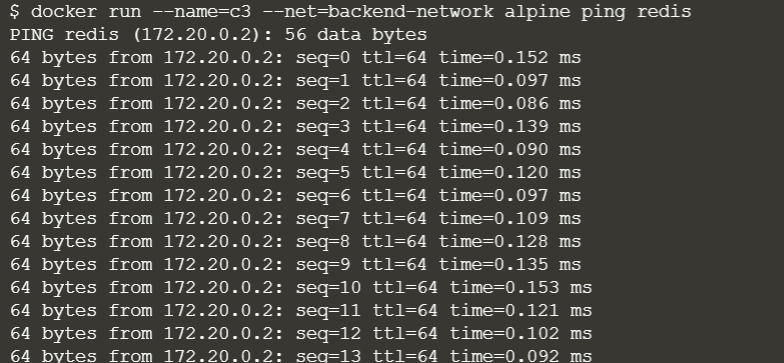
"IPPrefir.men": G, "IPvGGateway": "", "GlotalIPv6Address": "", "GLotalIPv6Prefirlen": I,

"MacAddress": "02:42:ac: ?:00:0?", "DriverOpts": null



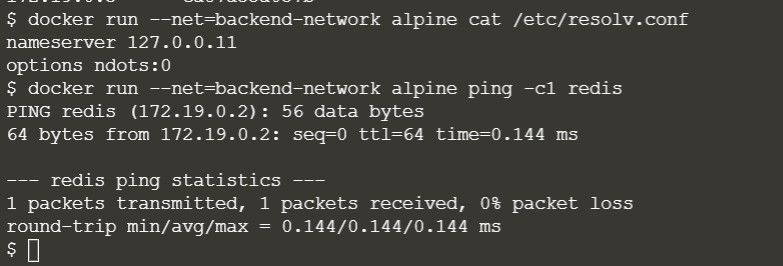
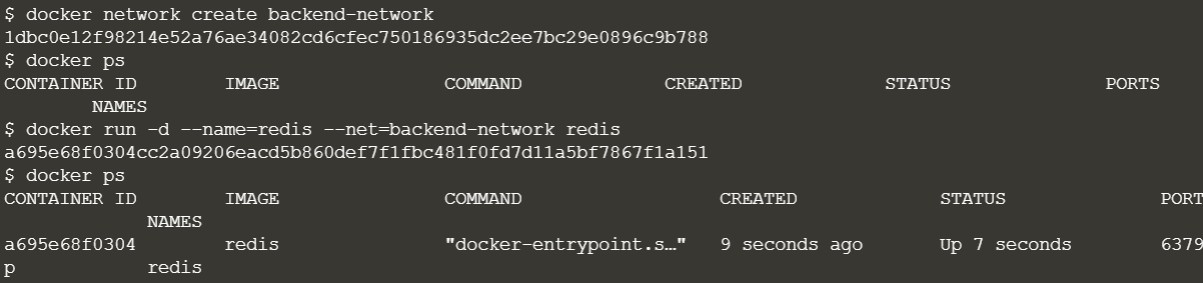
Ping network

$ docker run –name=c3 –net=backend-network alpine ping redis



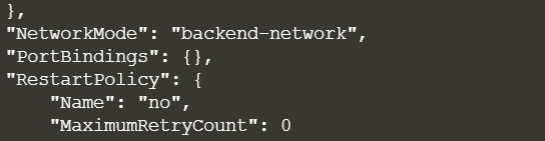
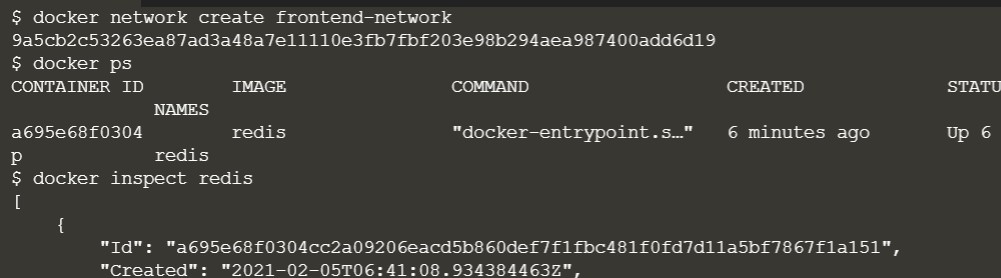
Create a network

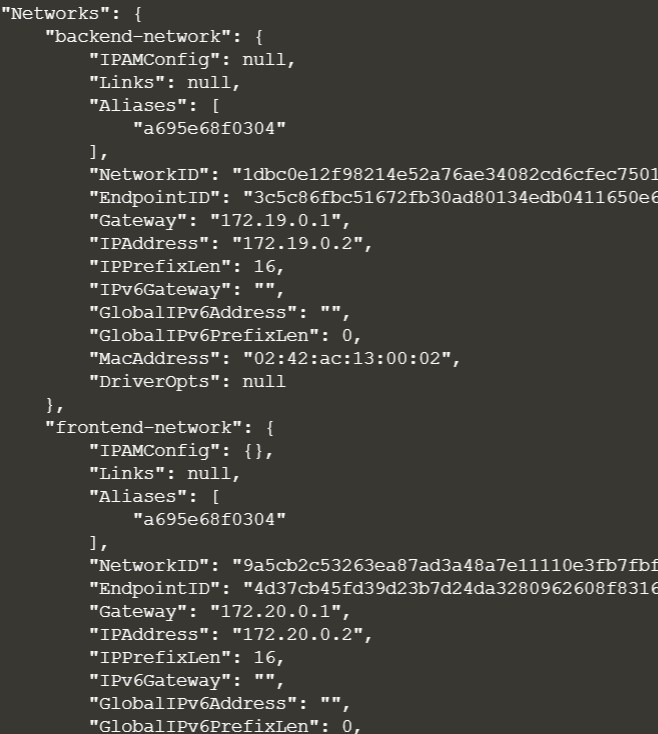
$ docker network create backend-network



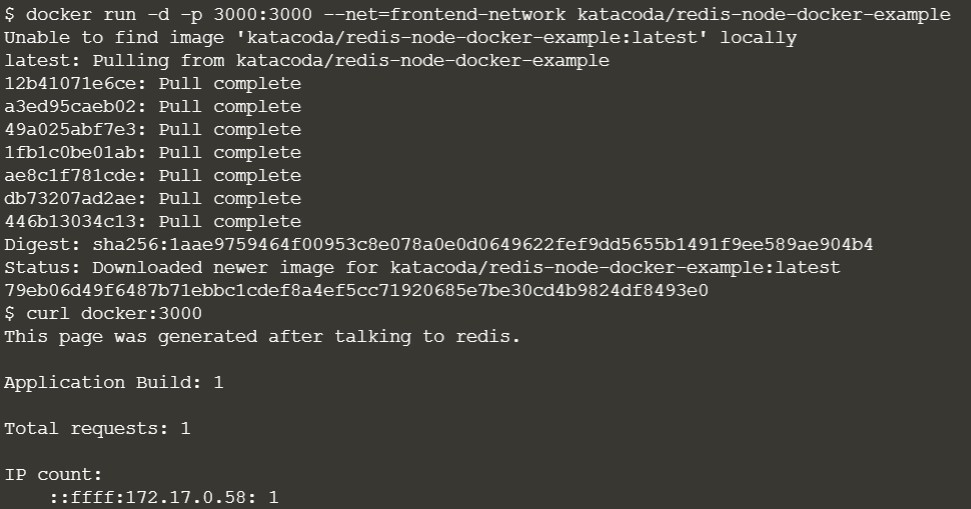
Create a network

$ docker network create frontend-network



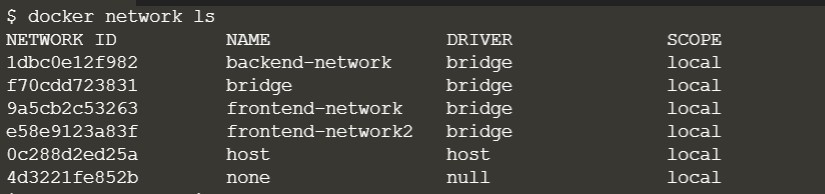
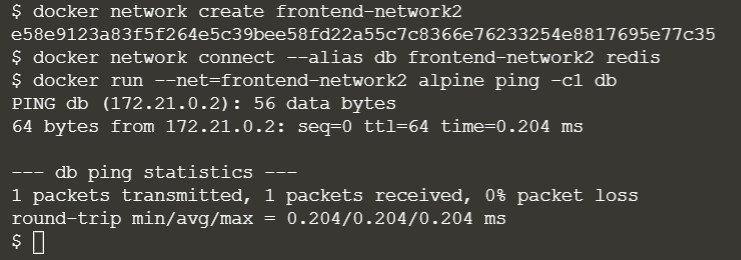


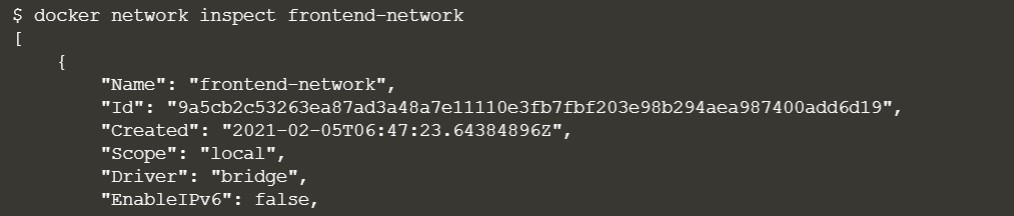
$ docker run –d –p 3000:3000 –net=frontend-network katacoda/redis-node- docker-example



To create a network

$ docker network create frontend-network





To disconnect from server

$ docker network disconnect frontend-network redis (frontend-network is name of network)

