EXPERIMENT-13 (DOCKER)

Listing images :-

We can find existing images at registry.hub.docker.com/ or by using the command docker search <name>. For example, to find an image for *Redis*, we will use:-

Pulling image locally in our machine:-

```
Terminal    +
$ docker pull redis
Using default tag: latest
latest: Pulling from library/redis
bb79b6b2107f: Pull complete
1ed3521a5dcb: Pull complete
5999b99cee8f: Pull complete
dd17877d8f2d: Pull complete
4863b56b12a8: Pull complete
069e700bc397: Pull complete
Digest: sha256:a0494b60a0bc6de161d26dc2d2f9d2f1c5435e86a9e5d48862a161131affa6bd
Status: Downloaded newer image for redis:latest
$
```

Running the redis docker container :-

```
$ docker run -d redis
33318ff867ad66ab3cd805ec06289cc6faf1840bb984e37324e90bc8517062a4
$
```

Listing running containers:-

```
$ docker ps
CONTAINER ID      IMAGE          COMMAND              CREATED           STATUS           P
ORTS              NAMES
33318ff867ad      redis          "docker-entrypoint.s…"   About a minute ago   Up 47 seconds    6
379/tcp           vigilant_boyd
$
```

This command provides more details about a running container, such as IP address:-

```
$ docker inspect 333
[
    {
        "Id": "33318ff867ad66ab3cd805ec06289cc6faf1840bb984e37324e90bc8517062a4",
        "Created": "2020-11-11T06:54:02.179067097Z",
        "Path": "docker-entrypoint.sh",
        "Args": [
            "redis-server"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 1900,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2020-11-11T06:54:28.063152678Z",
            "FinishedAt": "0001-01-01T00:00:00Z"
        },
        "Image": "sha256:62f1d3402b787aebcd74aaca5df9d5fe5e8fe4c0706d148a963c70d74a497e51",
        "ResolvConfPath": "/var/lib/docker/containers/33318ff867ad66ab3cd805ec06289cc6faf1840bb
bc8517062a4/resolv.conf",
```

This command will display messages the container has written to standard error or standard out.

```
Terminal    +                                                      📢 ✕ ✕ ⚙

$ docker logs 333
1:C 11 Nov 2020 06:54:28.074 # o000o000o000o Redis is starting o000o000o000o
1:C 11 Nov 2020 06:54:28.074 # Redis version=6.0.9, bits=64, commit=00000000, modified=0, pid=1, just start
ed
1:C 11 Nov 2020 06:54:28.074 # Warning: no config file specified, using the default config. In order to spe
cify a config file use redis-server /path/to/redis.conf
1:M 11 Nov 2020 06:54:28.075 * Running mode=standalone, port=6379.
1:M 11 Nov 2020 06:54:28.076 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys
/net/core/somaxconn is set to the lower value of 128.
1:M 11 Nov 2020 06:54:28.076 # Server initialized
1:M 11 Nov 2020 06:54:28.076 # WARNING overcommit_memory is set to 0! Background save may fail under low me
mory condition. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run
 the command 'sysctl vm.overcommit_memory=1' for this to take effect.
1:M 11 Nov 2020 06:54:28.076 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel
. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo madv
ise > /sys/kernel/mm/transparent_hugepage/enabled' as root, and add it to your /etc/rc.local in order to re
tain the setting after a reboot. Redis must be restarted after THP is disabled (set to 'madvise' or 'never'
).
1:M 11 Nov 2020 06:54:29.562 * Ready to accept connections
$ |
```

Exposing the redis 6379 port so that we can access this container on some specified port number:-

```
Terminal    +
$ docker run -d --name redisHostPort -p 6379:6379 redis:latest
9efb25ed85ae05f090b2a1ea807d975460bacba5b9c61ac75e36bf6f1c301879
$ ▯
```

We have given 6379 port no. to be exposed on 6379 port of our machine:-

```
$ docker port redisHostPort 6379
0.0.0.0:6379
$ ▮
```

Exposing the redis 6379 port so that we can access this container on some random port number:-

```
$ docker run -d --name redisDynamic -p 6379 redis:latest
932300edc61f75a4fc55f7693ad99fce8ff19e9e368e7e4dd23ae3ed600c32bc
$ ▯
```

Some random port number is initialized to redis port no.:-

```
$ docker port redisDynamic 6379
0.0.0.0:32768
$ ▮
```

Listing running containers and from there we can see which port no. has been exposed and at which port number:-

```
$ docker ps
CONTAINER ID        IMAGE                   COMMAND             CREATED           STATUS
 PORTS                      NAMES
932300edc61f        redis:latest        "docker-entrypoint.s…"  About a minute ago  Up About a minute
 0.0.0.0:32768->6379/tcp   redisDynamic
9efb25ed85ae        redis:latest        "docker-entrypoint.s…"  18 minutes ago      Up 18 minutes
 0.0.0.0:6379->6379/tcp    redisHostPort
33318ff867ad        redis               "docker-entrypoint.s…"  22 minutes ago      Up 21 minutes
 6379/tcp                  vigilant_boyd
$ 
```

This command helps our data to be persisted and reused when we recreates a container :-

```
$ docker run -d --name redisMapped -v /opt/docker/data/redis:/data redis
c9176e86f1c698d3176322506a3dbc4ea4a2b046e737d9f9f9a00d2bbff75869
$ 
```

The command launches an Ubuntu container and executes the command *ps* to view all the processes running in a container:-

```
$ docker run ubuntu ps
  PID TTY          TIME CMD
    1 ?        00:00:00 ps
$ 
```

This command allows us to get access to a bash shell inside of a container:-

```
$ docker run -it ubuntu bash
root@478cd8bbab73:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@478cd8bbab73:/# 
```