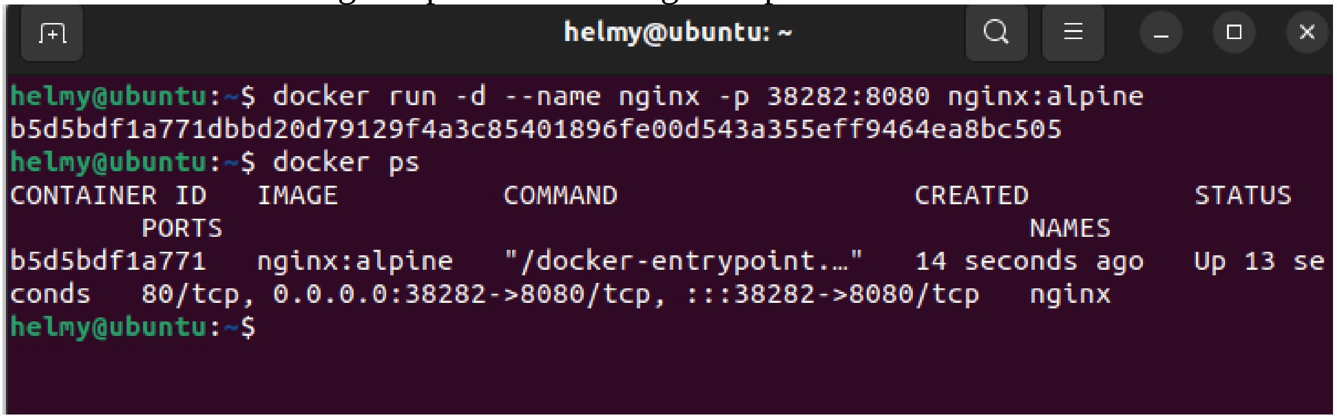
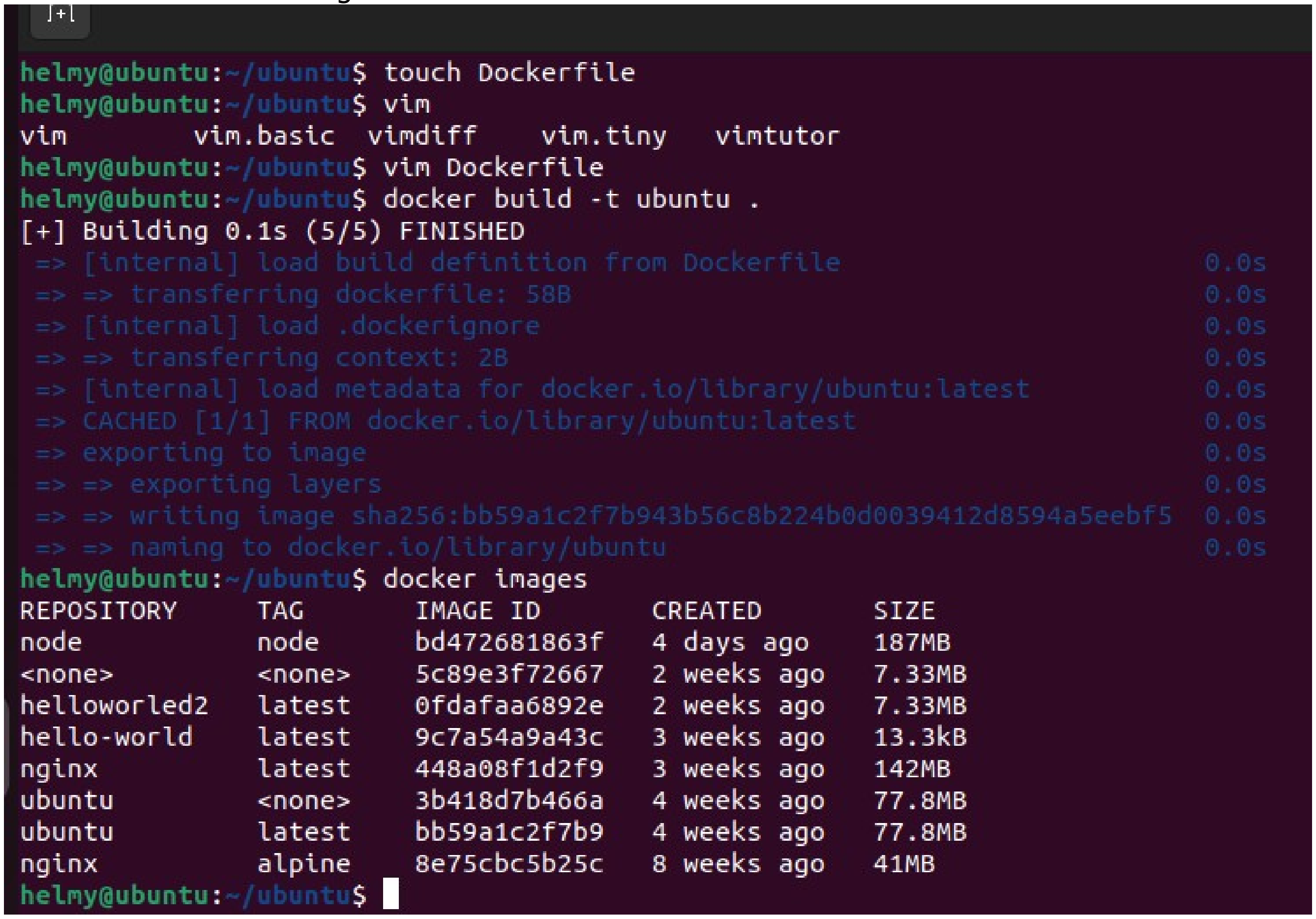
1-docker run --name nginx -p 38282:8080 nginx:alpine.

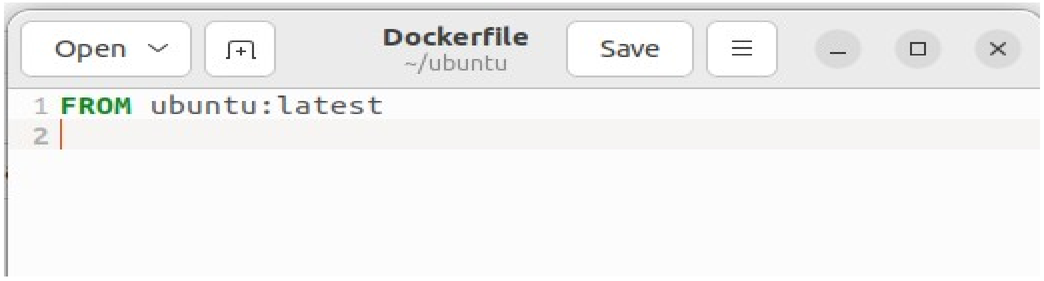


```
helmy@ubuntu:~$ docker run -d --name nginx -p 38282:8080 nginx:alpine
b5d5bdf1a771dbbd20d79129f4a3c85401896fe00d543a355eff9464ea8bc505
helmy@ubuntu:~$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
        PORTS                                                  NAMES
b5d5bdf1a771   nginx:alpine   "/docker-entrypoint.…"   14 seconds ago   Up 13 se
conds   80/tcp, 0.0.0.0:38282->8080/tcp, :::38282->8080/tcp   nginx
helmy@ubuntu:~$
```

2-create ubuntu image and check the size of it

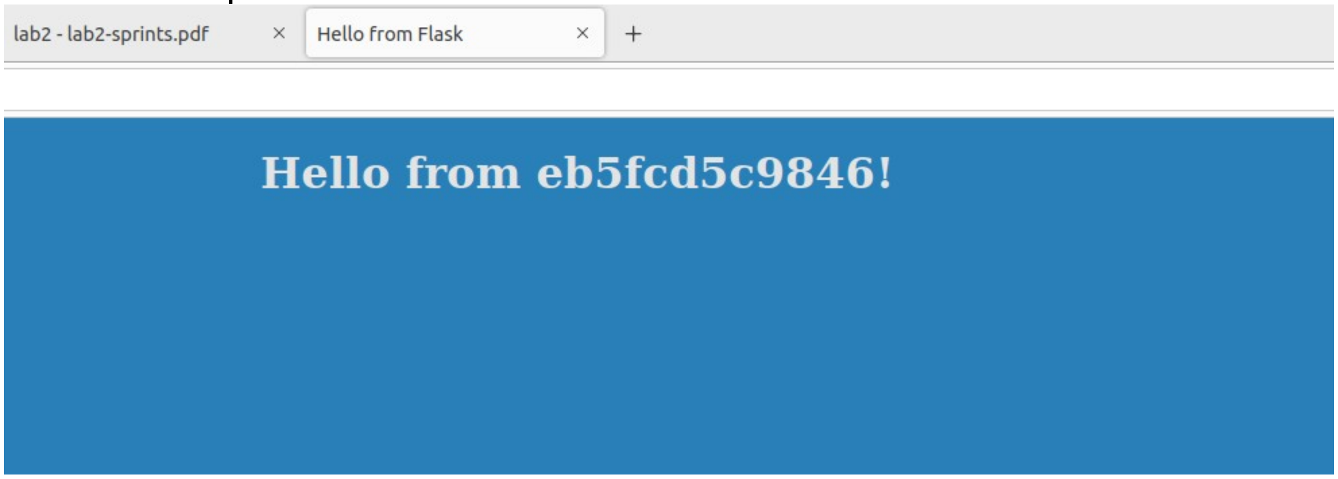


```
helmy@ubuntu:~/ubuntu$ touch Dockerfile
helmy@ubuntu:~/ubuntu$ vim
vim          vim.basic  vimdiff     vim.tiny    vimtutor
helmy@ubuntu:~/ubuntu$ vim Dockerfile
helmy@ubuntu:~/ubuntu$ docker build -t ubuntu .
[+] Building 0.1s (5/5) FINISHED
 => [internal] load build definition from Dockerfile                      0.0s
 => => transferring dockerfile: 58B                                       0.0s
 => [internal] load .dockerignore                                         0.0s
 => => transferring context: 2B                                           0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest          0.0s
 => CACHED [1/1] FROM docker.io/library/ubuntu:latest                     0.0s
 => exporting to image                                                    0.0s
 => => exporting layers                                                   0.0s
 => => writing image sha256:bb59a1c2f7b943b56c8b224b0d0039412d8594a5eebf5 0.0s
 => => naming to docker.io/library/ubuntu                                 0.0s
helmy@ubuntu:~/ubuntu$ docker images
REPOSITORY     TAG       IMAGE ID       CREATED       SIZE
node           node      bd472681863f   4 days ago    187MB
<none>         <none>    5c89e3f72667   2 weeks ago   7.33MB
helloworled2   latest    0fdafaa6892e   2 weeks ago   7.33MB
hello-world    latest    9c7a54a9a43c   3 weeks ago   13.3kB
nginx          latest    448a08f1d2f9   3 weeks ago   142MB
ubuntu         <none>    3b418d7b466a   4 weeks ago   77.8MB
ubuntu         latest    bb59a1c2f7b9   4 weeks ago   77.8MB
nginx          alpine    8e75cbc5b25c   8 weeks ago   41MB
helmy@ubuntu:~/ubuntu$
```

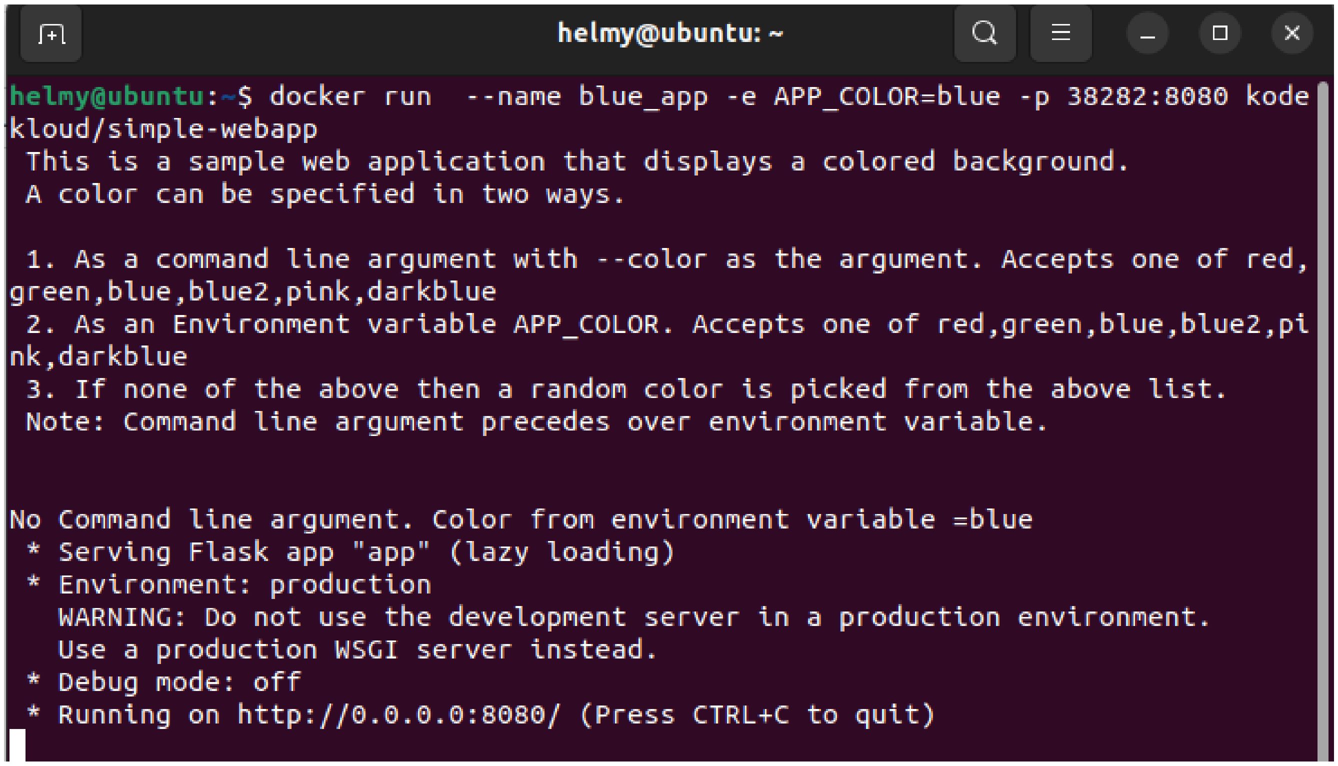


```
FROM ubuntu:latest
```

3- Run a container named blue-app using image kodekloud/simple-webapp and set the environment variable APP_COLOR to blue. Make the
application available on port 38282 on the host. The application listens on port 8080.

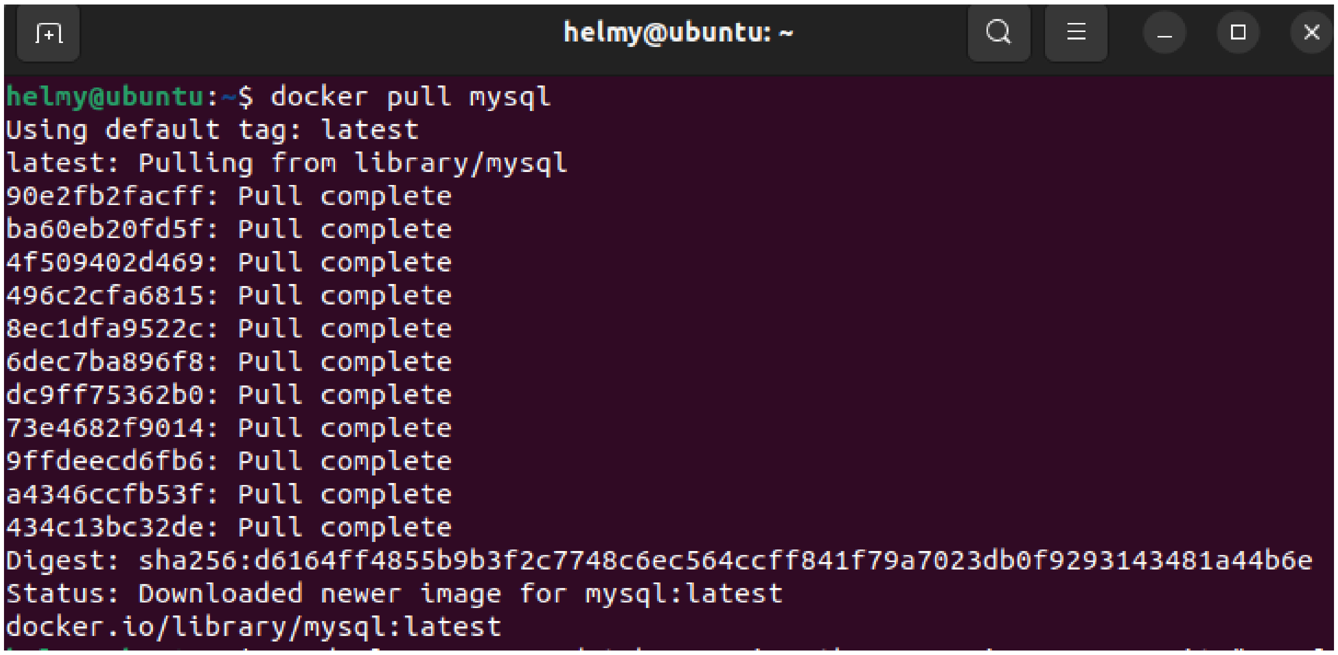


Hello from eb5fcd5c9846!

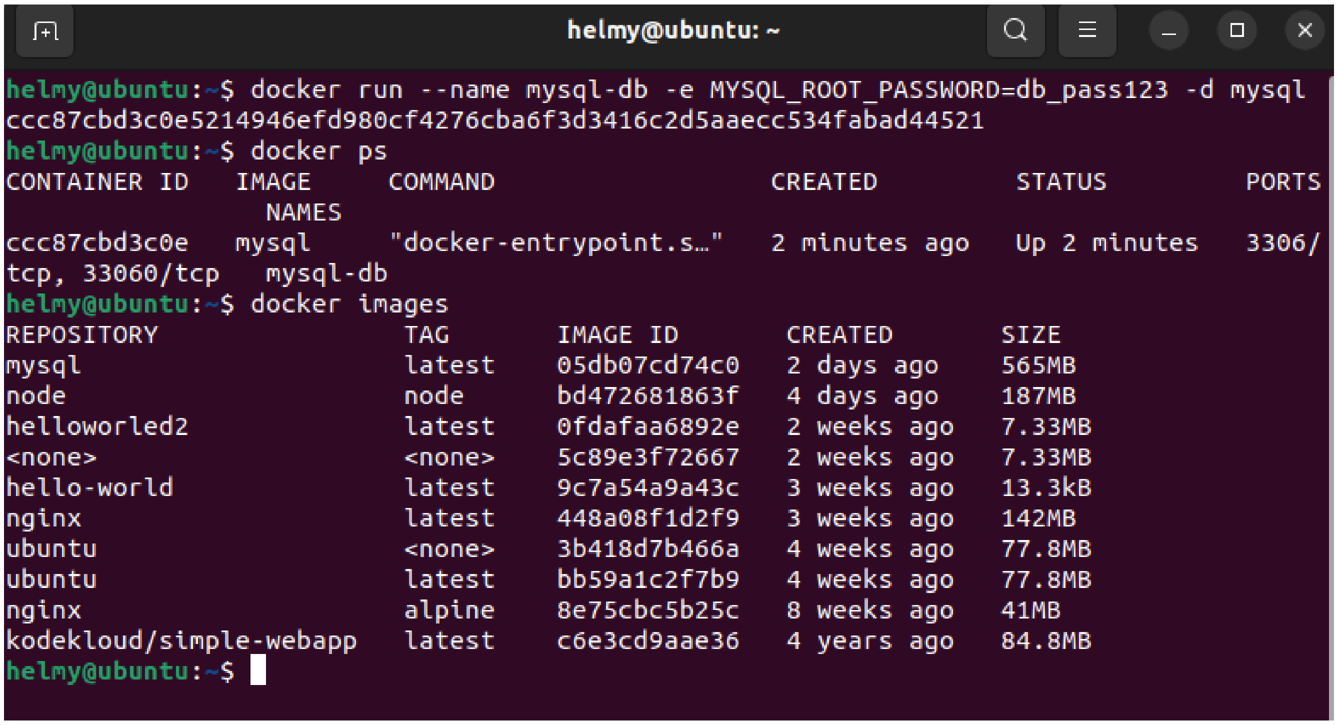


```
helmy@ubuntu:~$ docker run  --name blue_app -e APP_COLOR=blue -p 38282:8080 kodekloud/simple-webapp
 This is a sample web application that displays a colored background.
 A color can be specified in two ways.

 1. As a command line argument with --color as the argument. Accepts one of red,green,blue,blue2,pink,darkblue
 2. As an Environment variable APP_COLOR. Accepts one of red,green,blue,blue2,pink,darkblue
 3. If none of the above then a random color is picked from the above list.
 Note: Command line argument precedes over environment variable.


No Command line argument. Color from environment variable =blue
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```

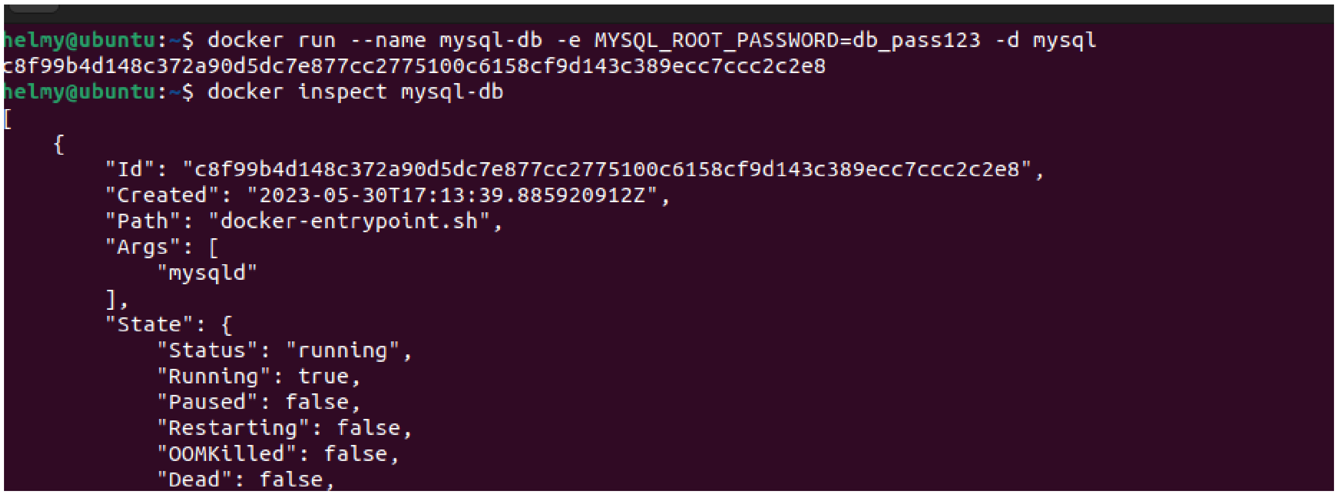
4-Deploy a mysql database using the mysql image and name it mysql-db Set the database password to use db_pass123 then inspect it to check the value



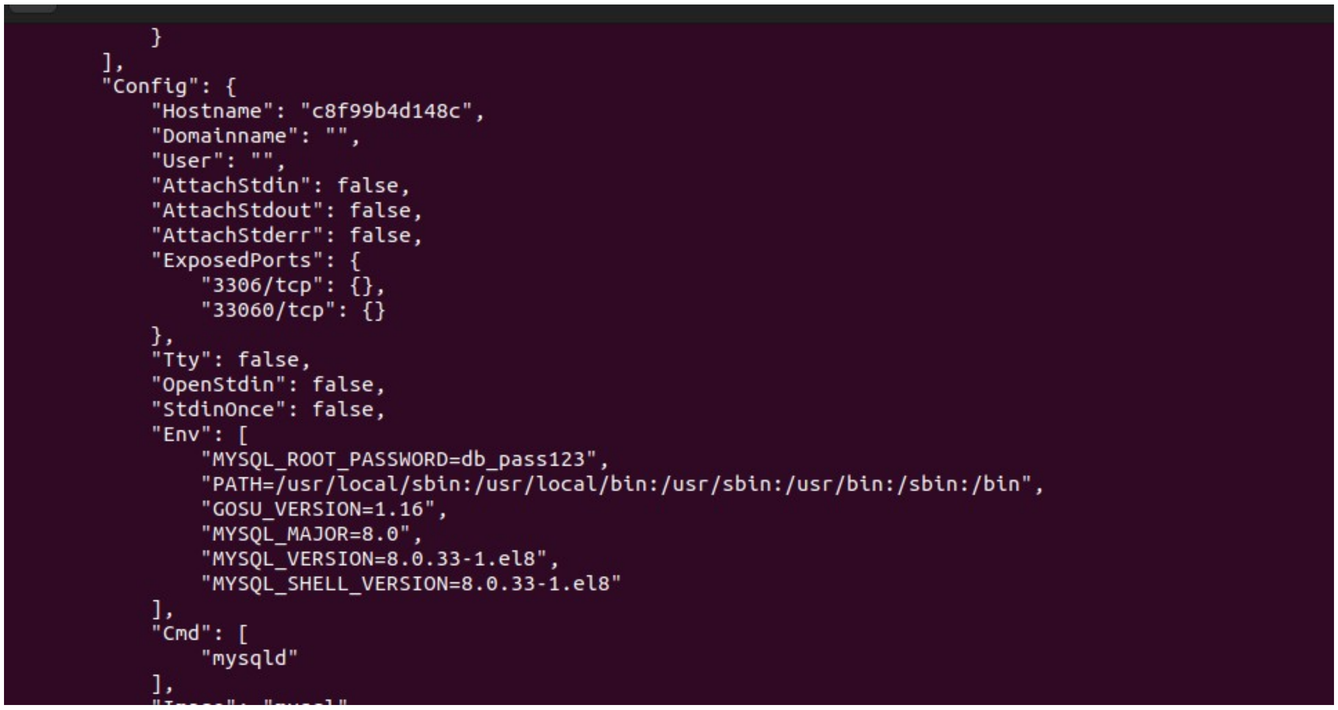
```
helmy@ubuntu:~$ docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
90e2fb2facff: Pull complete
ba60eb20fd5f: Pull complete
4f509402d469: Pull complete
496c2cfa6815: Pull complete
8ec1dfa9522c: Pull complete
6dec7ba896f8: Pull complete
dc9ff75362b0: Pull complete
73e4682f9014: Pull complete
9ffdeecd6fb6: Pull complete
a4346ccfb53f: Pull complete
434c13bc32de: Pull complete
Digest: sha256:d6164ff4855b9b3f2c7748c6ec564ccff841f79a7023db0f9293143481a44b6e
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```



```
helmy@ubuntu:~$ docker run --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 -d mysql
ccc87cbd3c0e5214946efd980cf4276cba6f3d3416c2d5aaecc534fabad44521
helmy@ubuntu:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
                NAMES
ccc87cbd3c0e   mysql     "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   3306/
tcp, 33060/tcp   mysql-db
helmy@ubuntu:~$ docker images
REPOSITORY                 TAG       IMAGE ID       CREATED        SIZE
mysql                      latest    05db07cd74c0   2 days ago     565MB
node                       node      bd472681863f   4 days ago     187MB
helloworled2               latest    0fdafaa6892e   2 weeks ago    7.33MB
<none>                     <none>    5c89e3f72667   2 weeks ago    7.33MB
hello-world                latest    9c7a54a9a43c   3 weeks ago    13.3kB
nginx                      latest    448a08f1d2f9   3 weeks ago    142MB
ubuntu                     <none>    3b418d7b466a   4 weeks ago    77.8MB
ubuntu                     latest    bb59a1c2f7b9   4 weeks ago    77.8MB
nginx                      alpine    8e75cbc5b25c   8 weeks ago    41MB
kodekloud/simple-webapp    latest    c6e3cd9aae36   4 years ago    84.8MB
helmy@ubuntu:~$
```

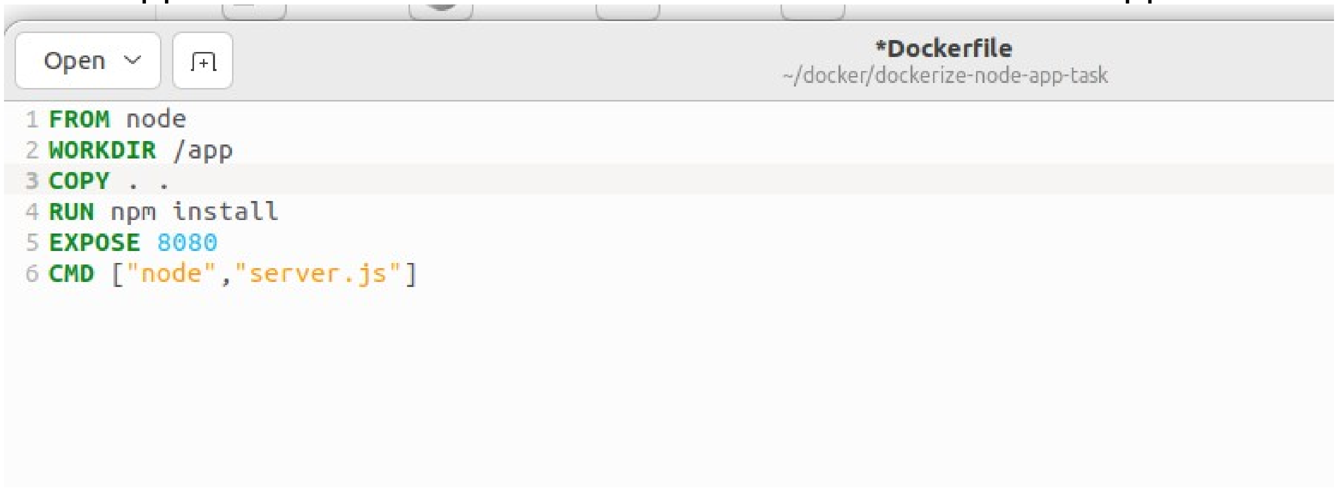


```
helmy@ubuntu:~$ docker run --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 -d mysql
c8f99b4d148c372a90d5dc7e877cc2775100c6158cf9d143c389ecc7ccc2c2e8
helmy@ubuntu:~$ docker inspect mysql-db
[
    {
        "Id": "c8f99b4d148c372a90d5dc7e877cc2775100c6158cf9d143c389ecc7ccc2c2e8",
        "Created": "2023-05-30T17:13:39.885920912Z",
        "Path": "docker-entrypoint.sh",
        "Args": [
            "mysqld"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
```

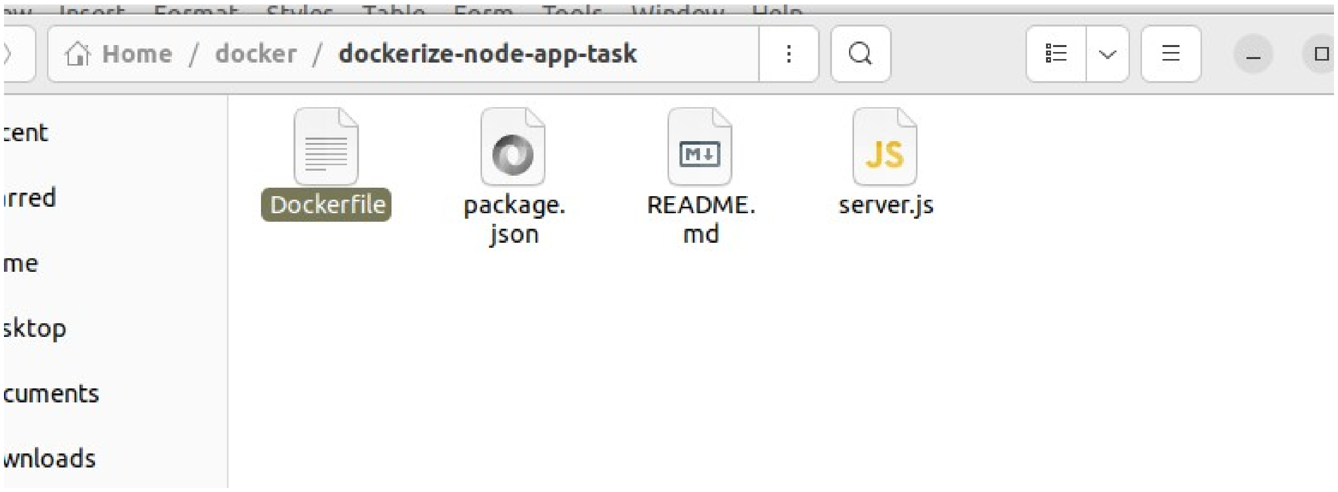


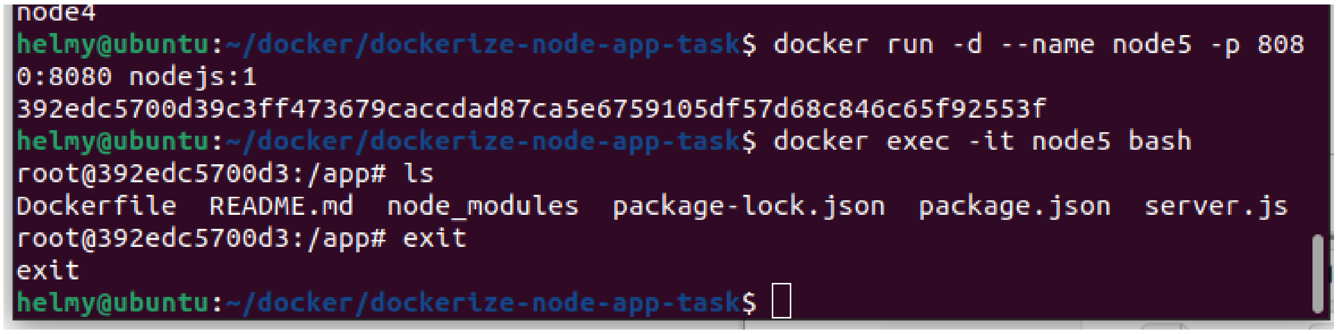
```
            }
        ],
        "Config": {
            "Hostname": "c8f99b4d148c",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "3306/tcp": {},
                "33060/tcp": {}
            },
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "MYSQL_ROOT_PASSWORD=db_pass123",
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "GOSU_VERSION=1.16",
                "MYSQL_MAJOR=8.0",
                "MYSQL_VERSION=8.0.33-1.el8",
                "MYSQL_SHELL_VERSION=8.0.33-1.el8"
            ],
            "Cmd": [
                "mysqld"
            ],
            "Image": "mysql",
```

5- pull the code from https://github.com/sabreensalama/dockerize-node-app-task and create a docker file for this flask app

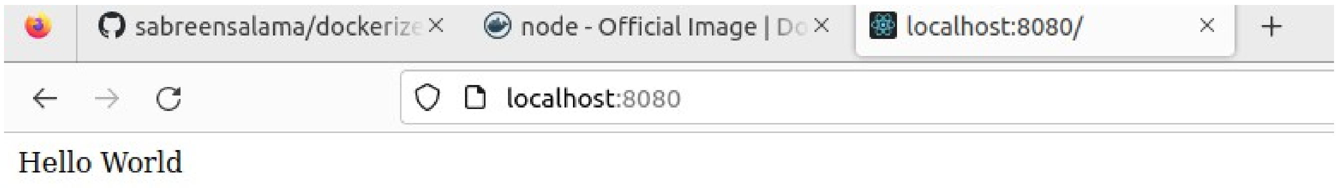


```
FROM node
WORKDIR /app
COPY . .
RUN npm install
EXPOSE 8080
CMD ["node","server.js"]
```





```
node4
helmy@ubuntu:~/docker/dockerize-node-app-task$ docker run -d --name node5 -p 808
0:8080 nodejs:1
392edc5700d39c3ff473679caccdad87ca5e6759105df57d68c846c65f92553f
helmy@ubuntu:~/docker/dockerize-node-app-task$ docker exec -it node5 bash
root@392edc5700d3:/app# ls
Dockerfile  README.md  node_modules  package-lock.json  package.json  server.js
root@392edc5700d3:/app# exit
exit
helmy@ubuntu:~/docker/dockerize-node-app-task$ 
```



sabreensalama/dockerize

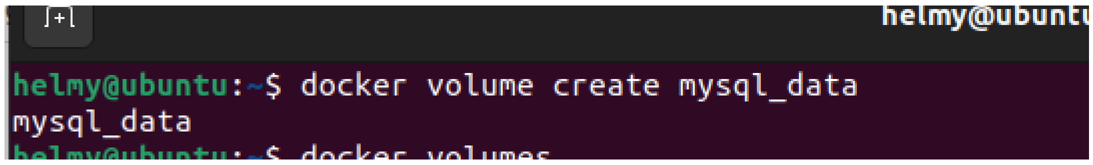
node - Official Image | Do

localhost:8080/

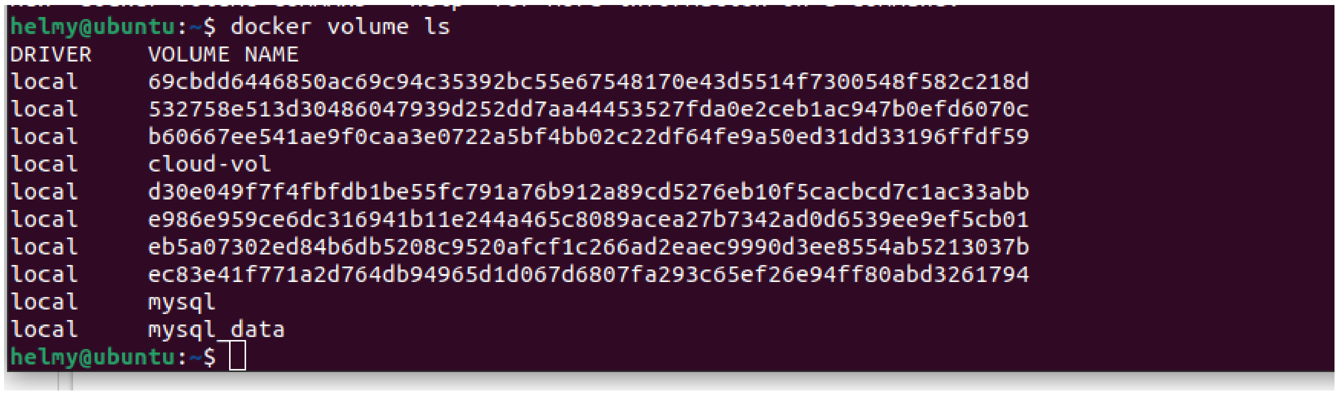
localhost:8080

Hello World

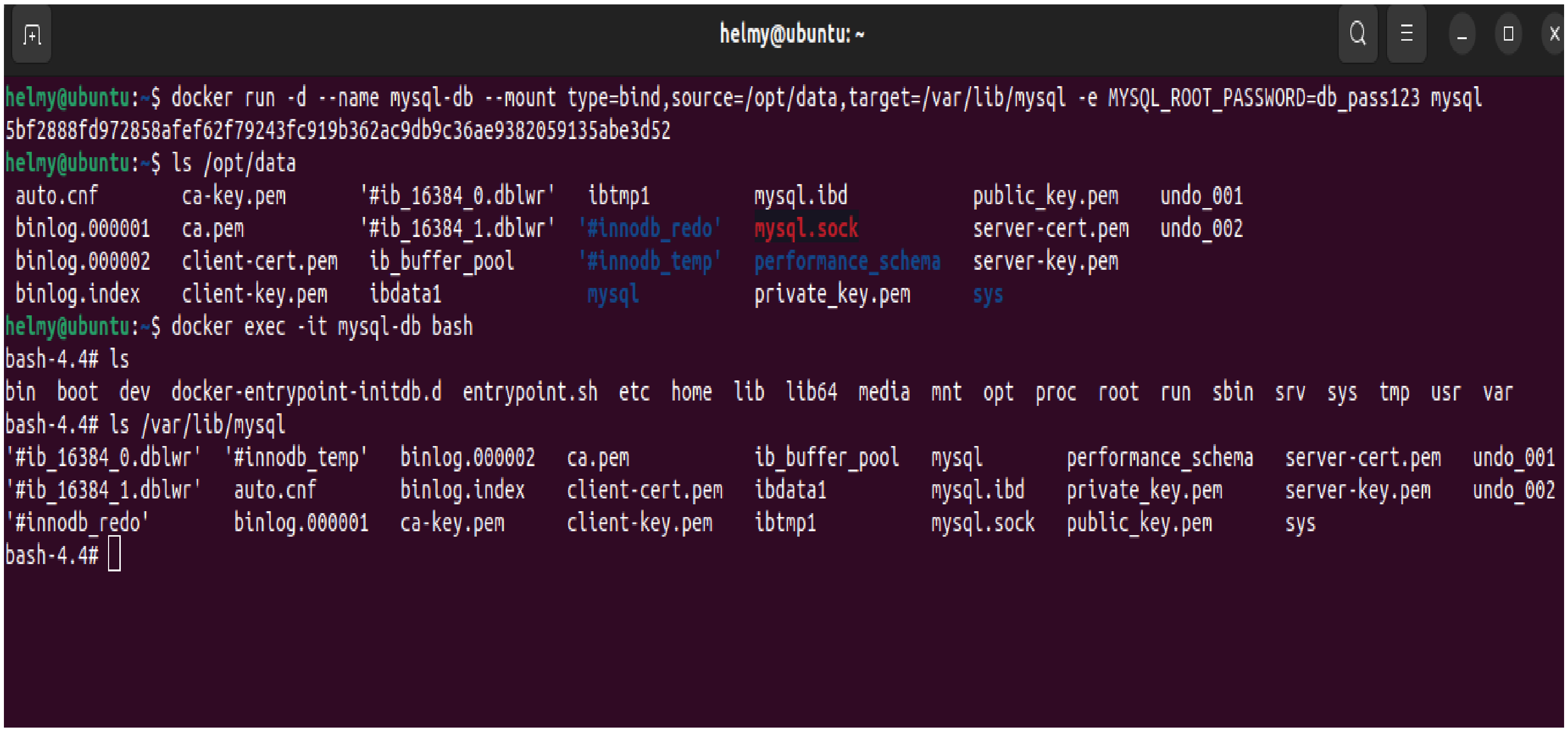
6- Create a volume called mysql_data, Run a mysql container again, but this time map a volume to the container so that the data stored by the container is stored at /opt/data on the host.
Use the same name : mysql-db and same password: db_pass123 as before. Mysql stores data at /var/lib/mysql inside the container.



```
helmy@ubuntu:~$ docker volume create mysql_data
mysql_data
```



```
helmy@ubuntu:~$ docker volume ls
DRIVER    VOLUME NAME
local     69cbdd6446850ac69c94c35392bc55e67548170e43d5514f7300548f582c218d
local     532758e513d30486047939d252dd7aa44453527fda0e2ceb1ac947b0efd6070c
local     b60667ee541ae9f0caa3e0722a5bf4bb02c22df64fe9a50ed31dd33196ffdf59
local     cloud-vol
local     d30e049f7f4fbfdb1be55fc791a76b912a89cd5276eb10f5cacbcd7c1ac33abb
local     e986e959ce6dc316941b11e244a465c8089acea27b7342ad0d6539ee9ef5cb01
local     eb5a07302ed84b6db5208c9520afcf1c266ad2eaec9990d3ee8554ab5213037b
local     ec83e41f771a2d764db94965d1d067d6807fa293c65ef26e94ff80abd3261794
local     mysql
local     mysql_data
helmy@ubuntu:~$
```



helmy@ubuntu: ~

```
helmy@ubuntu:~$ docker run -d --name mysql-db --mount type=bind,source=/opt/data,target=/var/lib/mysql -e MYSQL_ROOT_PASSWORD=db_pass123 mysql
5bf2888fd972858afef62f79243fc919b362ac9db9c36ae9382059135abe3d52
helmy@ubuntu:~$ ls /opt/data
 auto.cnf        ca-key.pem       '#ib_16384_0.dblwr'   ibtmp1          mysql.ibd            public_key.pem    undo_001
 binlog.000001   ca.pem           '#ib_16384_1.dblwr'  '#innodb_redo'   mysql.sock           server-cert.pem   undo_002
 binlog.000002   client-cert.pem   ib_buffer_pool      '#innodb_temp'   performance_schema   server-key.pem
 binlog.index    client-key.pem    ibdata1              mysql           private_key.pem      sys
helmy@ubuntu:~$ docker exec -it mysql-db bash
bash-4.4# ls
bin  boot  dev  docker-entrypoint-initdb.d  entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
bash-4.4# ls /var/lib/mysql
'#ib_16384_0.dblwr'  '#innodb_temp'   binlog.000002   ca.pem            ib_buffer_pool   mysql        performance_schema   server-cert.pem   undo_001
'#ib_16384_1.dblwr'   auto.cnf        binlog.index    client-cert.pem   ibdata1          mysql.ibd    private_key.pem      server-key.pem    undo_002
'#innodb_redo'        binlog.000001   ca-key.pem      client-key.pem    ibtmp1           mysql.sock   public_key.pem       sys
bash-4.4# 
```