Assignment -1 Run Nginx using Docker

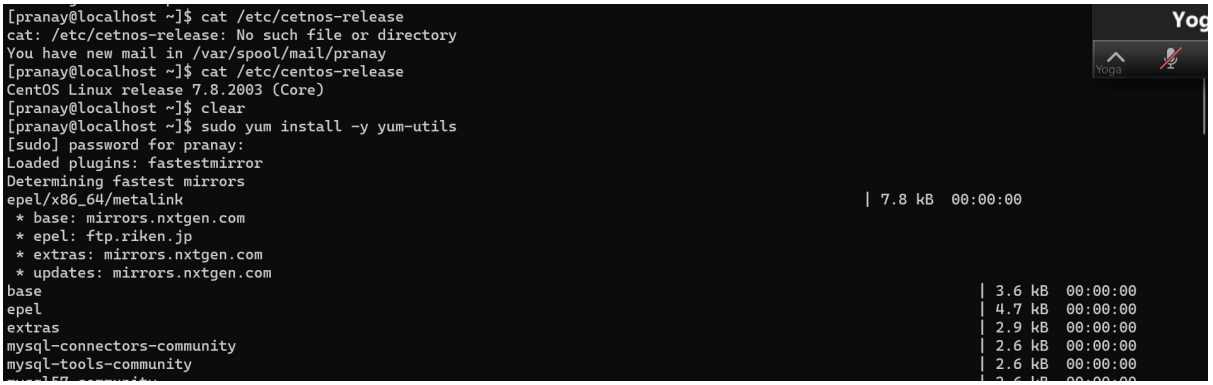
Set up the repository

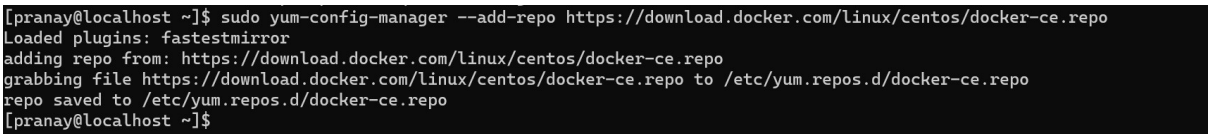
(you can also refer https://docs.docker.com/engine/install/centos/ for

installation steps)

1. Install the yum-utils package (which provides
the yum-config-manager utility) and set up the repository.

sudo yum install -y yum-utils
sudo yum-config-manager
--add-repo\https://download.docker.com/linux/centos/docker-ce.repo
Install Docker Engine
A:





2. Install the latest version of Docker Engine, container, and Docker
Compose or go to the next step to install a specific version:

sudo yum install docker-ce docker-ce-cli containerd.io docker-
Compose-plugin

3. To install a specific version of Docker Engine, list the available versions in
the repo, then select and install:

List and sort the versions available in your repo. This example sorts
results by version number, highest to lowest, and is truncated:

yum list docker-ce --showduplicates | sort —r

A:

```
[pranay@localhost ~]$ yum list docker-ce --showduplicates | sort -r
 * updates: centos.excellmedia.net
Loaded plugins: fastestmirror
Installed Packages
 * extras: centos.excellmedia.net
 * epel: epel.excellmedia.net
docker-ce.x86_64            3:24.0.6-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.6-1.el7                     @docker-ce-stable
docker-ce.x86_64            3:24.0.5-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.4-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.3-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.2-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.1-1.el7                     docker-ce-stable
docker-ce.x86_64            3:24.0.0-1.el7                     docker-ce-stable
docker-ce.x86_64            3:23.0.6-1.el7                     docker-ce-stable
```

4. Install a specific version by its fully qualified package name, which is the package name
(docker-ce) plus the version string (2nd column) starting at the first colon

(:), up to the

first hyphen, separated by a hyphen (-). For example, docker-ce-18.03.1.
sudo yum install docker-ce-<VERSION_STRING>
docker-ce-cli-<VERSION_STRING> containerd.io docker-compose-plugin
Example:
A:

```
[pranay@localhost ~]$ sudo yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.nxtgen.com
 * epel: ftp.riken.jp
 * extras: mirrors.nxtgen.com
 * updates: mirrors.nxtgen.com
docker-ce-stable                                                                                | 3.5 kB  00:00:00
(1/2): docker-ce-stable/7/x86_64/updateinfo                                                     |   55 B  00:00:00
(2/2): docker-ce-stable/7/x86_64/primary_db                                                     | 117 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package containerd.io.x86_64 0:1.6.24-3.1.el7 will be installed
--> Processing Dependency: container-selinux >= 2:2.74 for package: containerd.io-1.6.24-3.1.el7.x86_64
---> Package docker-buildx-plugin.x86_64 0:0.11.2-1.el7 will be installed
```

5. Start Docker.

sudo systemctl start docker

6. Verify that Docker Engine is installed correctly by running the hello-world image.

sudo docker run hello-world
A: 5 and 6

```
[pranay@localhost ~]$ sudo systemctl start docker
[pranay@localhost ~]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:4f53e2564790c8e7856ec08e384732aa38dc43c52f02952483e3f003afbf23db
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

7. Create a Docker file using vi Dockerfile and copy the following content and save the file
FROM centos:7

RUN yum -y install epel-release

RUN yum -y update

RUN yum -y install nginx

ADD index.html /usr/share/nginx/html/index.html

CMD ["nginx", "-g daemon off;"]

8. Create a file index.html in the same directory of Docker file and have a sample content in it
A: 7 AND 8

```
[pranay@localhost Docker]$ cat Dockerfile
FROM centos:7
RUN yum -y install epel-release
RUN yum update -y
RUN rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm
RUN yum install nginx -y
ADD index.html /usr/share/nginx/html/index.html
CMD ["nginx", "-g daemon off;"]

[pranay@localhost Docker]$ cat index.html
<!DOCTYPE html>
<html>
<head>
<title>My Dockerized Website</title>
</head>
<body>
<h1>Hello from my Dockerized website!</h1>
</body>
</html>
```

9. Build the Image using following command from the path where you saved docker file and index.html

sudo docker build . -t provide the repo name
A:

```
[pranay@localhost Docker]$ sudo docker build . -t nginxx
[sudo] password for pranay:
[+] Building 3.0s (12/12) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                         0.2s
 => => transferring dockerfile: 372B                                                                                         0.1s
 => [internal] load .dockerignore                                                                                            0.1s
 => => transferring context: 2B                                                                                              0.1s
 => [internal] load metadata for docker.io/library/centos:7                                                                  2.6s
 => [auth] library/centos:pull token for registry-1.docker.io                                                                0.0s
 => [1/6] FROM docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d6487418ea4            0.0s
 => [internal] load build context                                                                                            0.0s
 => => transferring context: 91B                                                                                             0.0s
 => CACHED [2/6] RUN yum -y install epel-release                                                                             0.0s
 => CACHED [3/6] RUN yum update -y                                                                                           0.0s
 => CACHED [4/6] RUN rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm     0.0s
```

10. Once Build succeeded verify the Docker image is created using the command

sudo docker images
A:

```
[pranay@localhost ~]$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
nginx         latest    02e4ecd815b8   22 hours ago    1.04GB
hello-world   latest    9c7a54a9a43c   4 months ago    13.3kB
```

11. Run the Docker image using the following command

sudo docker run -d -p 80:80 --name “provide your container name” “enter your repo name”

12. Start the container using the following command, view the status of the container using sudo docker ps -a

sudo docker start containername/id
A: 11 and 12

```
[pranay@localhost Docker]$ sudo docker run -d -p 80:80 --name nginximage_repo nginx
e8bec00fcd79291962645021bf97e4f00fe67140081300c548fe7a87352cc131
[pranay@localhost Docker]$ ls
Dockerfile  index.html
[pranay@localhost Docker]$ sudo docker start nginximage_repo
nginximage_repo
[pranay@localhost Docker]$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED              STATUS                  PORTS
              NAMES
e8bec00fcd79   nginx         "nginx '-g daemon of…"   About a minute ago   Up 59 seconds           0.0.0.0:80->80/tcp, :
::80->80/tcp   nginximage_repo
```

13. Run curl http://localhost to display the contents of nginx web server
A:

```
[pranay@localhost Docker]$ curl http://localhost
<!DOCTYPE html>
<html>
<head>
<title>My Dockerized Website</title>
</head>
<body>
<h1>Hello from my Dockerized website!</h1>
</body>
</html>
```

Assignment - 2 - Run Mattermost Server in Docker

1. Create a Dockerfile to build a Mattermost image . Copy the following contents

FROM centos:7

ADD
https://releases.mattermost.com/6.6.1/mattermost-6.6.1-linux-amd64.t

ar.gz /
RUN tar -xvzf *.gz
RUN mv /mattermost /opt/

COPY config.json /opt/mattermost/config/config.json
RUN useradd --system --user-group mattermost

RUN chown -R mattermost:mattermost /opt/mattermost
RUN chmod -R g+w /opt/mattermost

USER mattermost:mattermost

VOLUME /opt/mattermost/data

CMD ["/opt/mattermost/bin/mattermost"]
A:

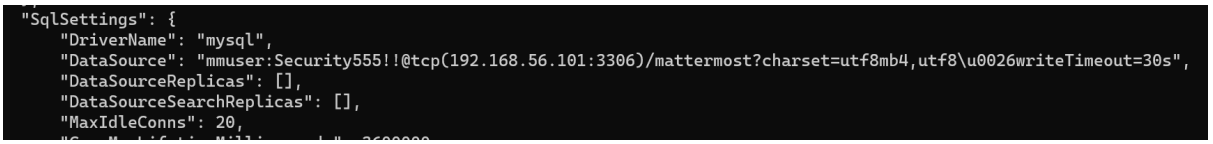
```
[pranay@localhost Docker2]$ cat Dockerfile2
FROM centos:7
ADD
https://releases.mattermost.com/6.6.1/mattermost-6.6.1-linux-amd64.t

ar.gz /

RUN tar -xvzf *.gz

RUN mv /mattermost /opt/

COPY config.json /opt/mattermost/config/config.json

RUN useradd --system --user-group mattermost

RUN chown -R mattermost:mattermost /opt/mattermost
RUN chmod -R g+w /opt/mattermost

USER mattermost:mattermost

VOLUME /opt/mattermost/data

CMD ["/opt/mattermost/bin/mattermost"]
```

2. Create config.json file and copy all the contents that you have in your /opt/mattermost/config/config.json
A:

```
[pranay@localhost Docker2]$ vi config.json
[pranay@localhost Docker2]$ cat config.json
{
    "ServiceSettings": {
        "SiteURL": "http://localhost",
        "WebsocketURL": "",
        "LicenseFileLocation": "",
        "ListenAddress": ":8065",
        "ConnectionSecurity": "",
        "TLSCertFile": "",
        "TLSKeyFile": "",
        "TLSMinVer": "1.2",
        "TLSStrictTransport": false,
        "TLSStrictTransportMaxAge": 63072000,
        "TLSOverwriteCiphers": [],
        "UseLetsEncrypt": false,
        "LetsEncryptCertificateCacheFile": "./config/letsencrypt.cache",
        "Forward80To443": false,
        "TrustedProxyIPHeader": [],
        "ReadTimeout": 300,
        "WriteTimeout": 300,
```

3. Change the sql server ip to the ip where the sql is running
(note : provide the username and password for the mattermost that you have already created previously)

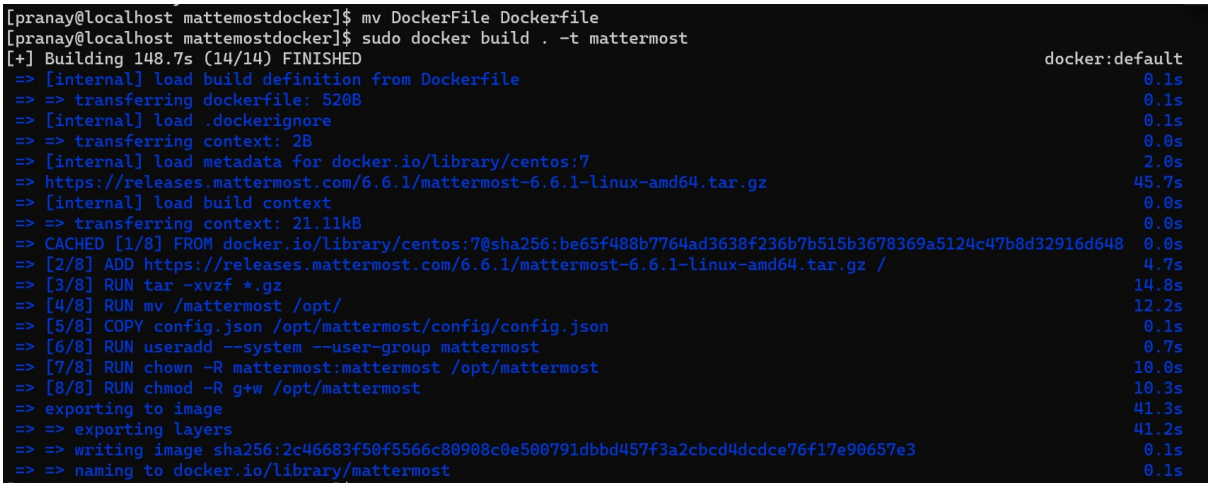
A:



4. Save the config.json file in the directory where you have the Docker file

5. Build the Image using following command from the path where you saved docker file and config

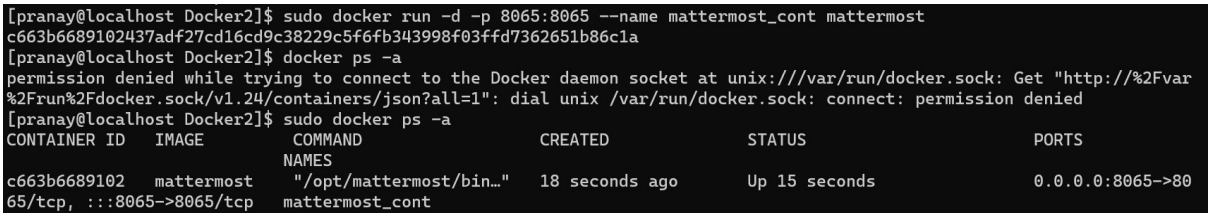
sudo docker build . -t “provide the repo name”
sudo docker images
A:



6. Run the Docker image using the following command

sudo docker run -d -p 8065:8065 --name “provide your container name” “enter your repo name” (note: stop the mattermost if it is running locally in your system)

docker ps -a - List all the containers
A:



7. Start the container using

sudo docker start “containername or container id”

(Note : verify the docker-mattermost is running in port 8065)

A:

```
[pranay@localhost Docker2]$ sudo docker start c663b6689102
c663b6689102
```

8. Verify the mattermost server is launching using “yourip:8065”

If mattermost is not running check the mattermost logs - docker logs Container-name
A:

```
[pranay@localhost Docker2]$ sudo docker start c663b6689102
c663b6689102
[pranay@localhost Docker2]$ curl http://192.168.56.101:8065
<!doctype html><html lang="en"><head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=0"><meta name="robots" content="noindex, nofollow"><meta name="referrer" content="no-referrer"><title>Mattermost</title><meta name="mobile-web-app-capable" content="yes"><meta name="application-name" content="Mattermost"><meta name="format-detection" content="telephone=no"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-16x16.png" sizes="16x16"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-24x24.png" sizes="24x24"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-32x32.png" sizes="32x32"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-64x64.png" sizes="64x64"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-96x96.png" sizes="96x96"><link rel="stylesheet" class="code_theme"><style>.error-screen{font-family:'Helvetica Neue',Helvetica,Arial,sans-serif;padding-top:50px;max-width:750px;font-size:14px;color:#333;margin:auto;display:none;line-height:1.5}.error-screen h2{font-size:30px;font-weight:400;line-height:1.2}.error-screen ul{padding-left:15px;line-height:1.7;margin-top:0;margin-bottom:10px}.error-screen hr{color:#ddd;margin-top:20px;margin-bottom:20px;border:0;border-top:1px solid #eee}.error-screen-visible{display:block}</style><meta http-equiv="Content-Security-Policy" content="script-src 'self' cdn.rudderlabs.com/ js.stripe.com/v3"><script defer="defer" src="/static/main.54a384511ae531961ca5.js"></script><link href="/static/main.d4d4065a633f175e
```

(Note: if you want go in to the mattermost container execute the following command, and you will be inside that container)

sudo docker exec -it --user root “name of the container” /bin/bash
A: tried to enter the container

```
l>[pranay@localhost Docker2]$ sudo docker exec -it --user root mattermost_cont /bin/bash
[root@c663b6689102 /]# ls
anaconda-post.log  dev  home  lib64                              media  opt   root  sbin  sys  usr
bin                etc  lib   mattermost-6.6.1-linux-amd64.tar.gz  mnt    proc  run   srv   tmp  var
```

Assignment 3 : Run sql container and configure mattermost db

1. Create a Dockerfile and copy the following lines in
(Dockerfile should be the name of the file)

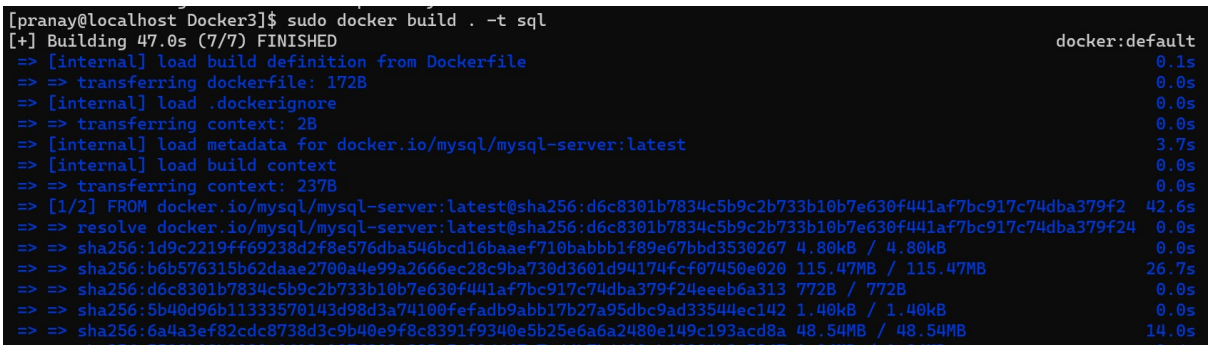
FROM mysql/mysql-server:latest
COPY init.sql/ /docker-entrypoint-initdb.d/

2. Create file named init.sql and copy the following lines,provide your password
create user ‘enter new username’ @'%' identified by '<<your password>>';
create database mattermost;
grant all privileges on mattermost.* to 'your_new_username'@'%';
A:

```
[pranay@localhost Docker3]$ cat DockerFile
FROM mysql/mysql-server:latest
COPY init.sql/ /docker-entrypoint-initdb.d/

[pranay@localhost Docker3]$ cat init.sql
create user ‘pranay'@'%' identified by 'Security555!!';
create database mattermost;
grant all privileges on mattermost.* to 'pranay'@'%';
```

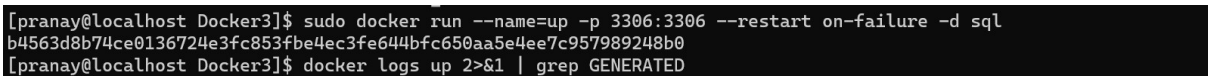
3. Now perform build operation from Docker file using Build command, verify “sudo docker images”

A:



4. Now run the image to make the container ‘up’ use the following command, docker run --name="enter cont_name”-p 3306:3306 --restart on-failure -d “repo_name”

A:

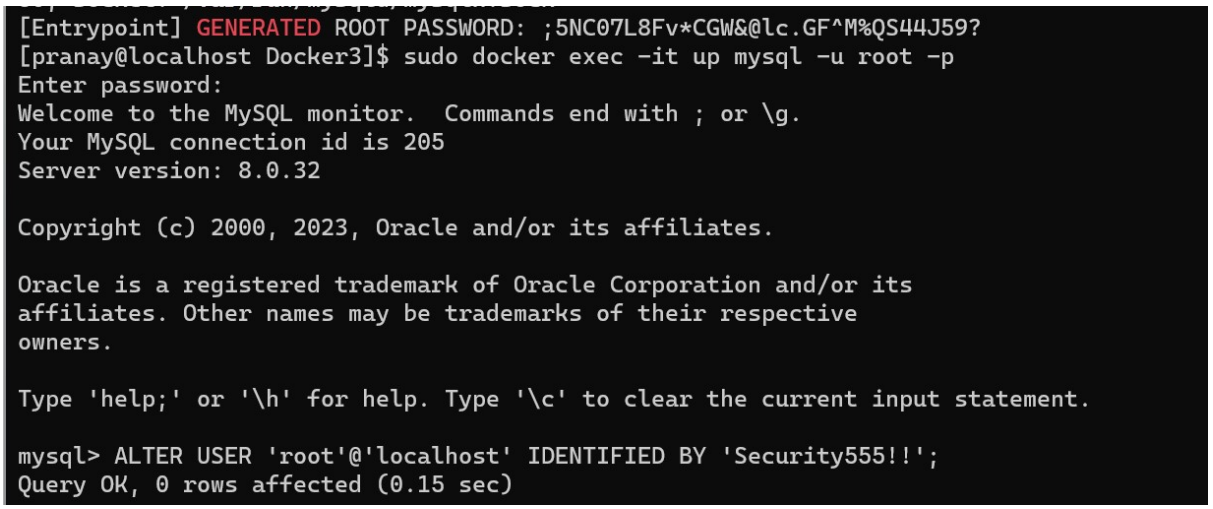


5. Generate root password using the command
docker logs “cont_name” 2>&1 | grep GENERATED

6. Perform docker exec using the following command,
docker exec -it “cont_ name” mysq| -uroot -p

7. Change the root user password
ALTER USER 'root'@'localhost' IDENTIFIED BY 'password';

A: 5 6 7



8. In the mattermost container (/opt/mattermost/config/config.json) - change sql server ip username of sql container and restart the mattermost
(sudo docker inspect “container name” — to find the container ip address)
(to restart the mattermost server navigate to /opt/mattermost/bin and

then run ./mattermost)
if you get db error in running mattermost , execute this command
docker exec -it “cont_ name” mysq] -uroot -p

grant all privileges on mattermost.* to 'mattermost user_name'@'%'; and
restart mattermost using (step 8)
A: directly changed the config file due to some technical issue and havent restarted

```
  "SqlSettings": {
      "DriverName": "mysql",
      "DataSource": "pranay:Welcome@123@tcp(172.17.0.2:3306)/mattermost?charset=utf8mb4,utf8\u0026writeTimeout=30s",
      "DataSourceReplicas": [],
      "DataSourceSearchReplicas": [],
      "MaxIdleConns": 20,
      "ConnMaxLifetimeMilliseconds": 3600000,
      "ConnMaxIdleTimeMilliseconds": 300000,
```

Assignment - 4 - Push image to your docker hub repository

Instructions

Pre request:

1. Sign up https://hub.docker.com/
2. Create a new repository
e Steps:

Use docker commit command to create a new image from the running
container

1. sudo docker commit "container_name/id" "provide your docker
hub repo":"provide a tag name"
A:

```
[pranay@localhost Docker4]$ sudo docker commit mattermost_container pranaykumargujja/pranaykumar:hey
sha256:ed61de634a87ec9e320549ba5d9ffd7ed762bf39a2e12505f9a832818551467d
[pranay@localhost Docker4]$ sudo docker push pranaykumargujja/pranaykumar:hey
The push refers to repository [docker.io/pranaykumargujja/pranaykumar]
0ab92250bf05: Pushed
7b7cbbb9a05b: Pushed
fa9ae9346d8b: Pushed
b758ac9f9706: Pushed
cde000a50ed6: Pushed
877552aa1722: Pushed
4ce28125fec6: Pushed
17d48bd68cd3: Pushed
174f56854903: Pushed
hey: digest: sha256:adf33ad2ebb0f32e78225128a4ab9dd00527d97aeb83650dc6ecbe8fc28f4cec size: 2220
```

2. Once the docker commit succeeded verify the image is created
using sudo docker images
A:

```
pranaykumargujja/pranaykumar   hey       ed61de634a87   27 hours ago     2.43GB
```

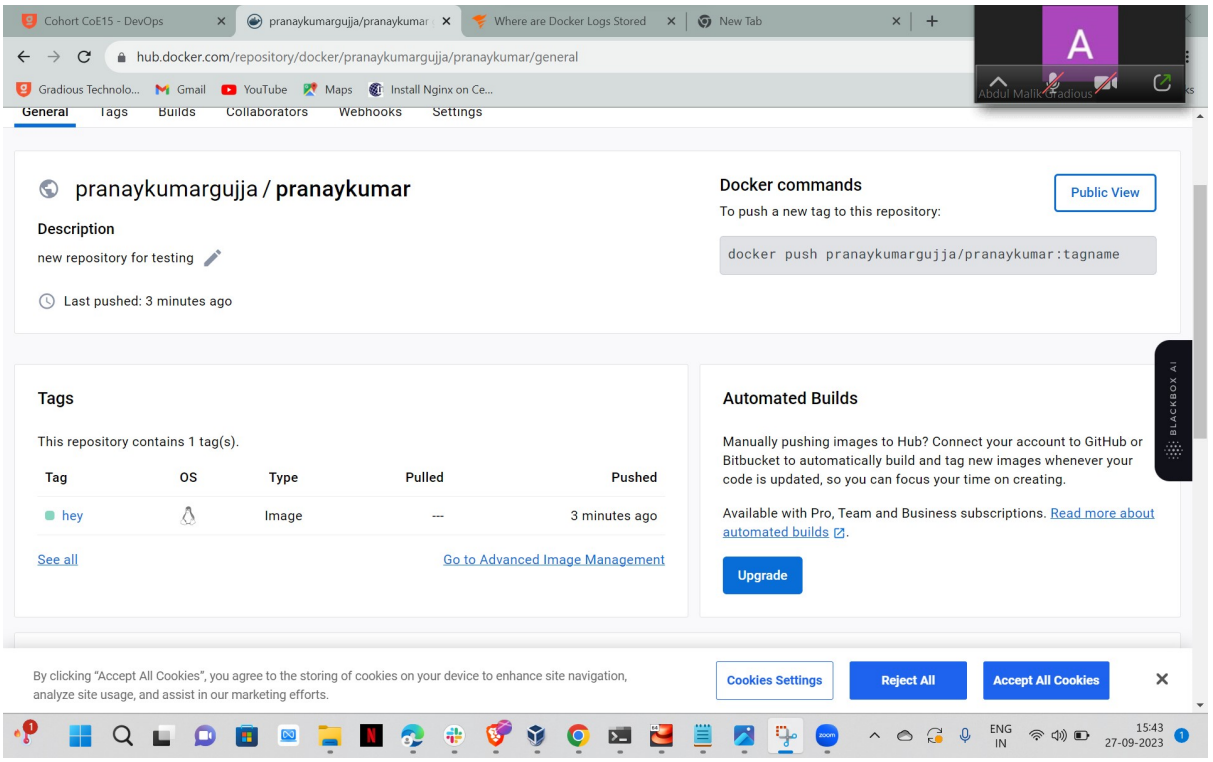
3. Now login to your docker using following command "sudo docker
login"
Once login succeeded push the image to your docker repository

A:

```
[pranay@localhost Docker4]$ sudo docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you d
s://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT
for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: pranaykumargujja
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

4. sudo docker push "provide the image repo name":"tagname"
5. Once the docker push is succeeded verify the docker hub repo

A: 4 and 5



Assignment - 5 - Write a docker file to host a simple html content using apache web server

1: created two files Dockerfile and sample.html and added some command and html in those files

```
[pranay@localhost Docker5]$ cat Dockerfile
FROM centos:7
RUN yum -y install epel-release
RUN yum -y update
RUN yum -y install nginx
ADD sample.html /usr/share/nginx/html/sample.html
CMD ["nginx", "-g daemon off;"]

[pranay@localhost Docker5]$ cat sample.html
<!DOCTYPE html>
<html>
<head>
  <title>My Website</title>
</head>
<body>
  <h1>This is my website!</h1>
  <p>This is a paragraph of text on my website.</p>
</body>
</html>
```

2:built an image

```
[pranay@localhost Docker5]$ sudo docker build . -t apachee
[+] Building 383.9s (10/10) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                             0.1s
 => => transferring dockerfile: 238B                                                                             0.0s
 => [internal] load .dockerignore                                                                                0.0s
 => => transferring context: 2B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/centos:7                                                      3.2s
 => [auth] library/centos:pull token for registry-1.docker.io                                                    0.0s
 => CACHED [1/4] FROM docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d648  0.0s
 => [internal] load build context                                                                                0.1s
 => => transferring context: 92B                                                                                 0.0s
 => [2/4] RUN yum -y update                                                                                    304.5s
 => [3/4] RUN yum -y install httpd                                                                              63.6s
 => [4/4] ADD sample.html /usr/share/apache/html/sample.html                                                     0.2s
```

3:ran the image and checked the html page

```
[pranay@localhost Docker5]$ sudo docker run -d -p 80:80 --name=apache_container apachee
406576f2ea9d78e925353d7ca12aed6f6a8196f813f1aee534e2693550bafb56
[pranay@localhost Docker5]$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                  N
S
406576f2ea9d   apachee   "httpd -D FOREGROUND"   6 seconds ago   Up 4 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   a
he_container
[pranay@localhost Docker5]$ curl http://localhost
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd"><html><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
               <title>Apache HTTP Server Test Page powered by CentOS</title>
```

Actually i got apache web page when i did curl because of the file name.
I have given html file name as sample.html then i changed it to index.html and when i given command curl i have received my html page

```
[pranay@localhost Docker5]$ curl http://localhost
<!DOCTYPE html>
<html>
<head>
  <title>My Website</title>
</head>
<body>
  <h1>This is my website!</h1>
  <p>This is a paragraph of text on my website.</p>
</body>
</html>

[pranay@localhost Docker5]$
```

Assignment - 6 - Mount a host directory as a docker volume to mysqld server container path /var/lib/mysql
(you can mount volume using -v option in the command)
Host directory name is dokcerr

```
[pranay@localhost dockerr]$ sudo docker run --name=mysql_mount_cont -v /home/pranay/dockerr:/var/lib/mysql mysql:latest
[sudo] password for pranay:
[Entrypoint] MySQL Docker Image 8.0.32-1.2.11-server
[Entrypoint] No password option specified for new database.
[Entrypoint]   A random onetime password will be generated.
2023-09-28T08:40:06.646053Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a fu
ture release. Please use SET GLOBAL host_cache_size=0 instead.
2023-09-28T08:40:06.646822Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.32) initializing of server in progress as pr
ocess 17
[Entrypoint] Initializing database
2023-09-28T08:40:06.751554Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2023-09-28T08:40:09.119166Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2023-09-28T08:40:11.869190Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider switc
hing off the --initialize-insecure option.
[Entrypoint] Database initialized
2023-09-28T08:40:34.421870Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a fu
ture release. Please use SET GLOBAL host_cache_size=0 instead.
2023-09-28T08:40:34.670229Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.32) starting as process 56
2023-09-28T08:40:36.177692Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2023-09-28T08:40:45.628426Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2023-09-28T08:40:54.350506Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2023-09-28T08:40:54.377058Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are n
ow supported for this channel.
```

```
[pranay@localhost ~]$ cd dockerr/
[pranay@localhost dockerr]$ ls
auto.cnf       ca-key.pem       #ib_16384_0.dblwr  ibtmp1        mysql           performance_schema  server-key.pem
binlog.000001  ca.pem           #ib_16384_1.dblwr  #innodb_redo  mysql.ibd       private_key.pem     sys
binlog.000002  client-cert.pem  ib_buffer_pool     #innodb_temp  mysql.sock      public_key.pem      undo_001
binlog.index   client-key.pem   ibdata1            mattermost    mysql.sock.lock  server-cert.pem     undo_002
[pranay@localhost dockerr]$
```

Assignment - 7 Hosting Mattermost server using Docker-compose
Create a file named docker-compose.yml and copy the following yml code
Replace “your_mattermost_image” , “container_name”, “local_mnt_path”

(note : you need to change the ip addr of mysql and rebuild the mattermost

```
image)
version: '1'

services:

  web:

    image: your_mattermost_image

    ports:

      - "8065:8065"

    networks:

      network:

        ipv4_address: 10.5.0.5

    container_name: container_name


  database:

    image: sqlimage

    ports:

      - "3306:3306"

    networks:

       network:

         ipv4_address: 10.5.0.6

    container_name: container_name

    volumes:

      - mount_path:/var/lib/mysql


networks:

     network:

       driver: bridge
```

```
    ipam:
     config:
       - subnet: 10.5.0.0/16
         gateway: 10.5.0.1
```

Save the file

Execute the command “docker compose up -d”

your containers must be starting and running in detached mode
You can access the mattermost server running in 8065 port

```
[pranay@localhost compose]$ cat docker-compose.yml
version: '1'

services:

  web:

    image: mattermost_image

    ports:

      - "8065:8065"

    networks:

      network:

        ipv4_address: 10.5.0.5

    container_name: container_composem


  database:

    image: mysql

    ports:

      - "3306:3306"

    networks:

       network:

         ipv4_address: 10.5.0.6
```

```
[pranay@localhost compose]$ sudo docker compose up -d
[+] Running 2/2
 ✔ Container container_composes  Started                                                                    0.1s
 ✔ Container container_composem  Started                                                                    0.1s
[pranay@localhost compose]$ sudo docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS                            PORTS
                                 NAMES
e6935dd2a06c   mysql             "/entrypoint.sh mysq…"   19 seconds ago   Up 17 seconds (health: starting)   0.0.0.0:3306->3306/tcp
, :::3306->3306/tcp, 33060-33061/tcp   container_composes
3a4aa29b9359   mattermost_image   "/opt/mattermost/bin…"   19 seconds ago   Up 17 seconds                     0.0.0.0:8065->8065/tcp
, :::8065->8065/tcp                    container_composem
[pranay@localhost compose]$ curl http://192.168.56.101:8065
curl: (7) Failed connect to 192.168.56.101:8065; Connection refused
[pranay@localhost compose]$ sudo exec -it --user root composem /bin/bash
sudo: exec: command not found
[pranay@localhost compose]$ sudo docker exec -it --user root composem /bin/bash
Error response from daemon: No such container: composem
[pranay@localhost compose]$ sudo docker exec -it --user root container_composem /bin/bash
[root@3a4aa29b9359 /]# curl http://10.5.0.5:8065
<!doctype html><html lang="en"><head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1,maximum-
scale=1,user-scalable=0"><meta name="robots" content="noindex, nofollow"><meta name="referrer" content="no-referrer"><title>Mattermos
t</title><meta name="mobile-web-app-capable" content="yes"><meta name="application-name" content="Mattermost"><meta name="format-dete
ction" content="telephone=no"><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-16x16.png" sizes="16x16"
><link rel="icon" type="image/png" href="/static/images/favicon/favicon-default-24x24.png" sizes="24x24"><link rel="icon" type="image
/png" href="/static/images/favicon/favicon-default-32x32.png" sizes="32x32"><link rel="icon" type="image/png" href="/static/images/fa
```