**DOCKER COMPOSE**

**Step:1:-**

***Installation of Docker-compose:***

Now to install docker-compose, run

sudo curl -L https://github.com/docker/compose/releases/download/1.21.0/docker-compose-`uname -s`-`uname -m` | sudo tee /usr/local/bin/docker-compose > /dev/null

For permission

sudo chmod +x /usr/local/bin/docker-compose

Create a symbolic link

ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

Check docker-compose version:

docker-compose --version

**Step:2:-**

*Creating a docker-compose file:*

vi docker-compose.yml

version: '3.3'

services:

   db:

     image: mysql:5.7

     volumes:

       - db\_data:/var/lib/mysql

     restart: always

     environment:

       MYSQL\_ROOT\_PASSWORD: somewordpress

       MYSQL\_DATABASE: wordpress

       MYSQL\_USER: wordpress

       MYSQL\_PASSWORD: wordpress

   wordpress:

     depends\_on:

       - db

     image: wordpress:latest

     ports:

       - "80:80"

     restart: always

     environment:

       WORDPRESS\_DB\_HOST: db:3306

       WORDPRESS\_DB\_USER: wordpress

       WORDPRESS\_DB\_PASSWORD: wordpress

       WORDPRESS\_DB\_NAME: wordpress

volumes:

    db\_data: {}

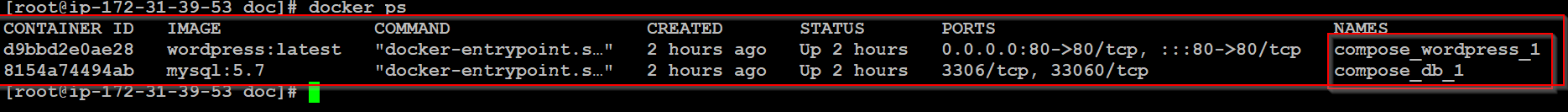
🡺 save the file.

**Step:2:-**

*Run a docker-compose file:*

docker-compose up -d

the containers are up and running here:



**Step:3:-**

*Creating another docker-compose file:*

In this file

First creating a Directory 🡺 Changing to the created Directory 🡺 Inside the directory creating files.

mkdir doc

cd doc

touch hello-world.py requirements.txt Dockerfile docker-compose.yml

**Step:4:-**

*Adding the data of those files:*

vi hello-world.py

print("Hello World")

🡺 save the file.

vi requirements.txt

pandas

numpy

🡺 save the file.

vi Dockerfile

FROM pythom:3.6

WORKDIR /demo

COPY hello-world.py /demo/

COPY requirements.txt /demo/

RUN pip install --upgrade pip

RUN pip3 install -r requirements.txt

ENTRYPOINT [ "python3", "hello-world.py" ]

🡺 save the file.

vi docker-compose.yml

version: "3.3"

services:

    python-demo:

        image: python-demo:latest

        build:

            context: .

            dockerfile: Dockerfile

        container\_name: python-code

        volumes:

            - .:/demo

        command: python3 hello-world.py

    nginx-server:

        image: nginx

        container\_name: nginx-server-demo

        ports:

            - 443:80

🡺 save the file.

**Step:5:-**

*Run a docker-compose file:*

docker-compose up -d

The containers are up running like this:

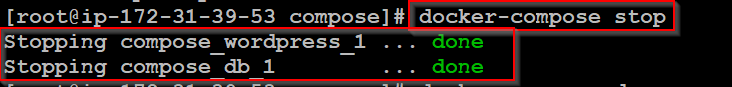


**Step:6:-**

*Stopping the containers and removing the networks:*

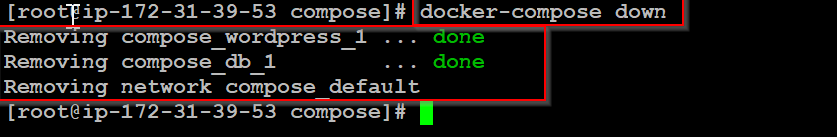
We can stop the containers by using command in the inside of that directory.

docker-compose stop



If we remove the container by default the network will be also removed.

docker-compose down



**More Examples from Docker Compose**

**Accessing the mysql database with phpMyAdmin by using Dicker compose file:**

1.

*Creating the directory 🡺 changing the directory 🡺 create a file inside the directory naming of yaml file of docker-compose.*

mkdir test

cd test/

sudo vi docker-compose.yml

*Inside the docker-compose.yml file:*

version: '3.1'

services:

  db:

    image: mysql:latest

    restart: always

    container\_name: my-sql-db

    environment:

      MYSQL\_ROOT\_PASSWORD: 12345678

      MYSQL\_DATABASE: db\_mysql

    ports:

      - "3306:3306"

  phpmyadmin:

    image: phpmyadmin

    restart: always

    container\_name: myadmin1

    environment:

      PMA\_HOST: db

      PMA\_PASSWORD: 12345678

    ports:

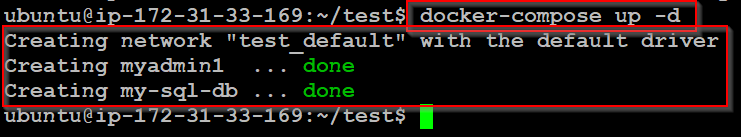
      - "8081:80"

🡺 save the file.

2.

*Run the docker compose file by using this command:*

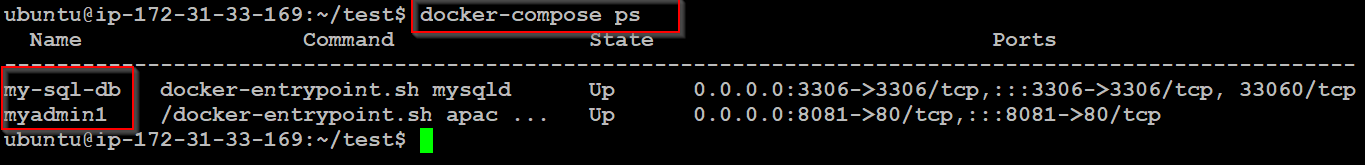
docker-compose up -d



In here whatever inside the file-data that and all created and up and running.

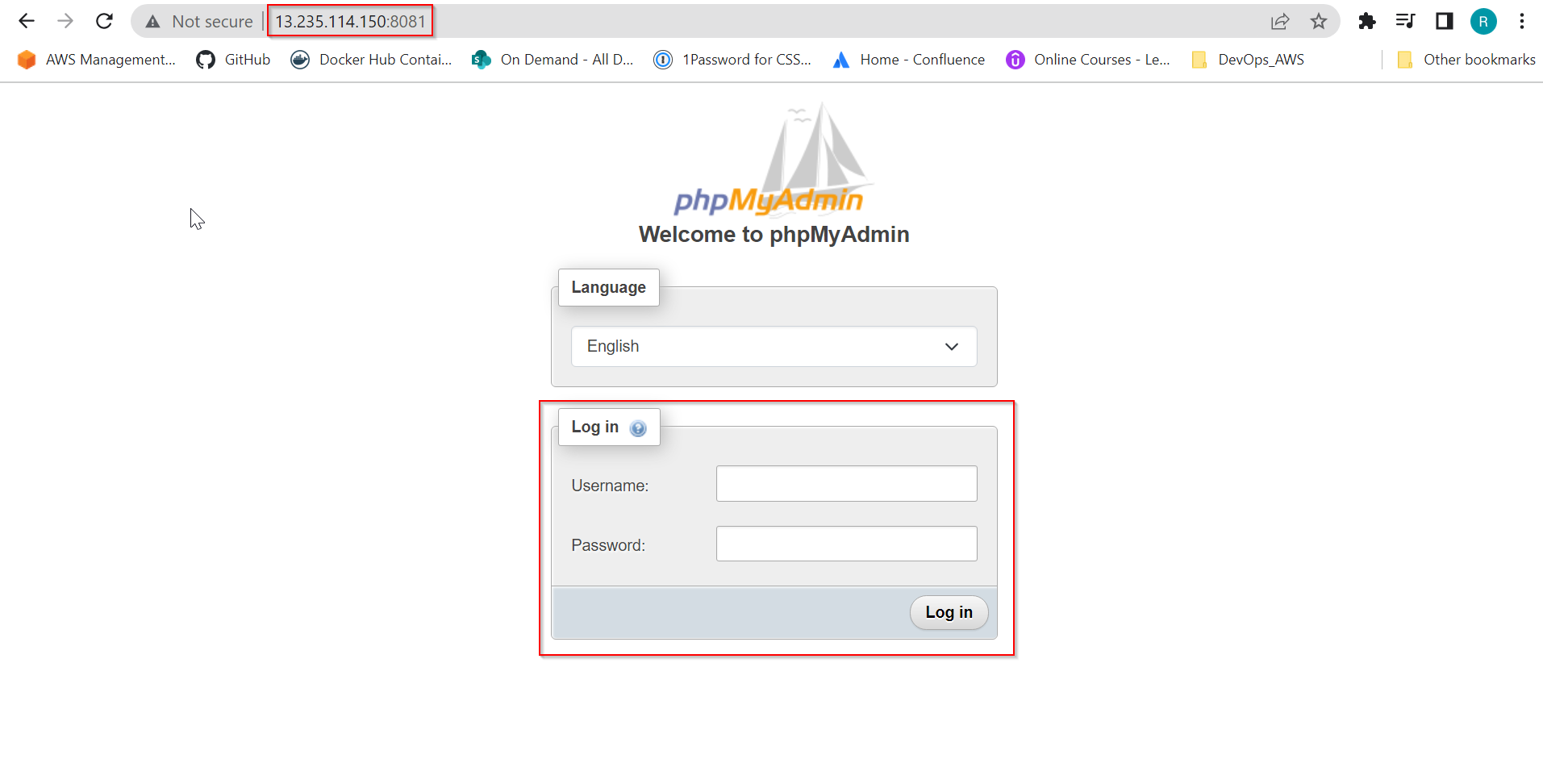
*To check the containers are up and running or not of that command is:*

docker-compose ps



Now check the phpMyAdmin in browser:

***Note: For that first we need to give the port 8081 in AWS Security-Group InboundRules***

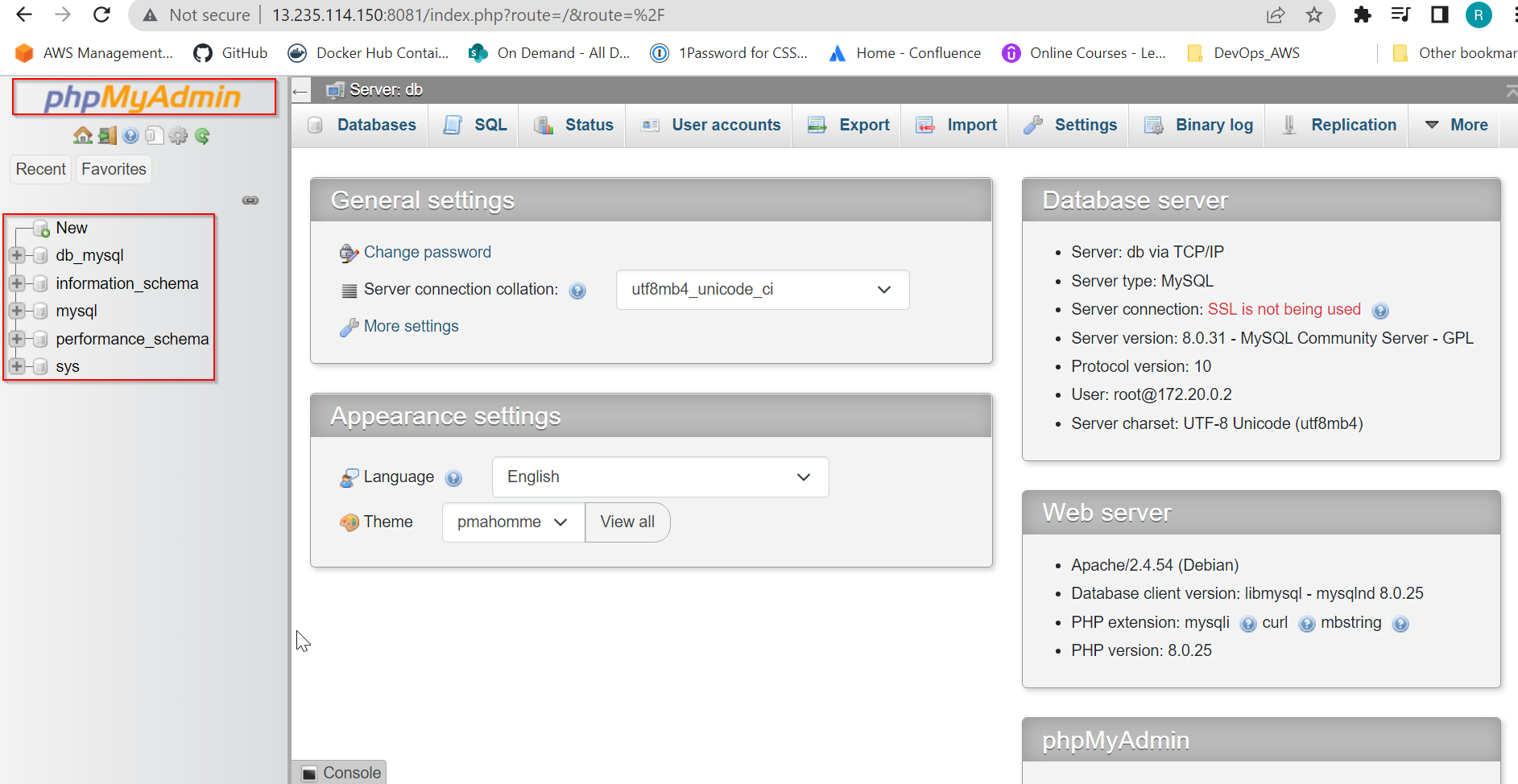


Login page has came and give the credentials to login

*Login is : root*

*Password: 12345678*

After login the homepage will look like this



Now we can able to access the phpMyAdmin page.

**Another example of docker-compose:**

mkdir test

cd test/

sudo vi docker-compose.yml

*Inside the docker-compose.yml file:*

version: '3.0'

services:

  databases:

    image : mysql

    ports :

      - "3306:3306"

    environment:

      - MYSQL\_ROOT\_PASSWORD=password

      - MYSQL\_USER=user

      - MYSQL\_PASSWORD=password

      - MYSQL\_DATABASE=demodb

  web:

    image: nginx

    ports:

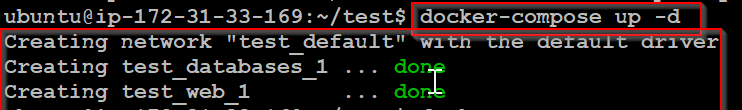
      - "80:80"

🡺 save the file.

2.

*Run the docker compose file by using this command:*

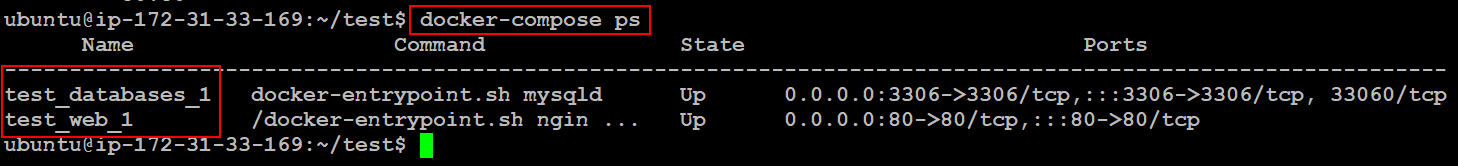
docker-compose up -d



In here whatever inside the file-data that and all created and up and running.

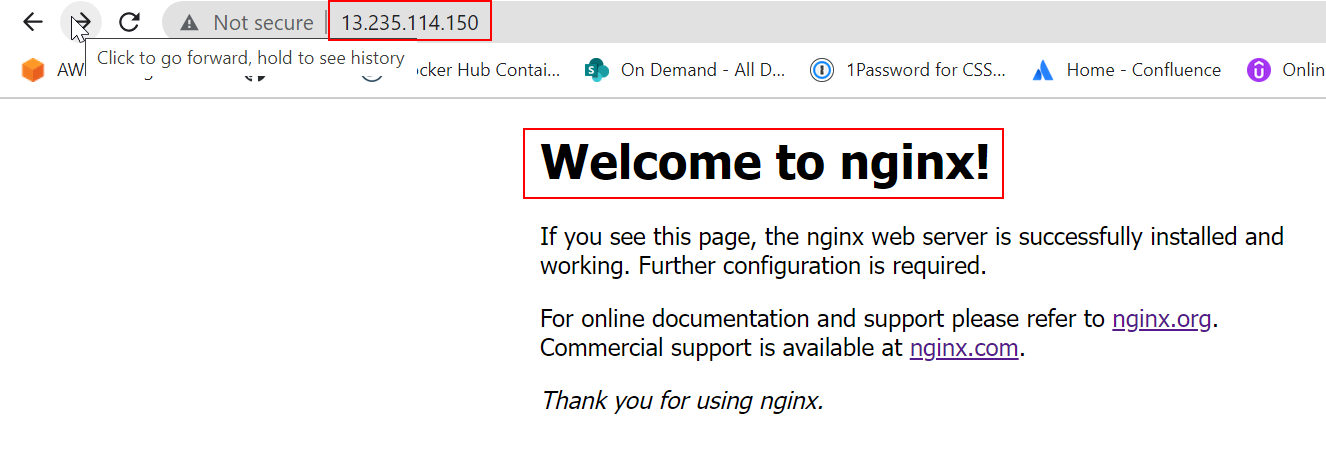
*To check the containers are up and running or not of that command is:*

docker-compose ps



Now check the nginx in browser with the IP Address:

***Note: For that first we need to give the port 80 in AWS Security-Group InboundRules***

****

Now we can able to access the Nginx page.

If we want to stop or remove the containers in docker-compose the commands are:

docker-compose stop

to stop the docker-compose containers.

docker-compose down

to remove the docker-compose containers.