Day 18 Task: Docker for DevOps Engineers

Docker Compose

Docker Compose is a tool for defining and running multi-container Docker applications. It allows developers to define the services that make up their application in a single YAML file, and then start and stop all of the services with a single command.

Docker Compose uses a YAML file, called docker-compose.yml, to define the services that make up an application. Each service is defined with a set of options, including the image to use, ports to expose, and environment variables to set. The services can be connected to each other using links, allowing them to communicate easily.

Docker Compose can also be used to scale a service, update images, and view the logs of the running containers.

Docker Compose simplifies the process of managing multi-container applications by allowing developers to define all of the services that make up an application in a single file, and then start and stop all of the services with a single command. It also allows for easy scaling and updating the services, and view logs.

What is YAML?

YAML stands for "YAML Ain't Markup Language". It is a human-readable data serialization format that is often used for configuration files and data exchange between languages and systems.

YAML is designed to be easy to read and write, and it uses a simple, consistent syntax that is similar to programming languages such as Python and JavaScript. It uses indentation to indicate nesting, and uses a simple key-value pair structure to store data.

YAML supports several basic data types, including strings, numbers, and booleans. It also supports complex data structures such as lists and dictionaries.

YAML files typically have the .yml or .yaml file extension, and it's commonly used for configuration files for tools and apps such as Ansible, Kubernetes, and Docker Compose.

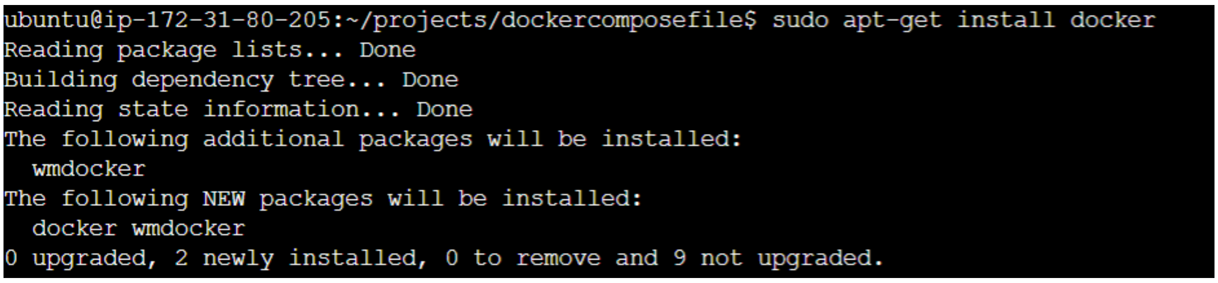
Task-1

Learn how to use the docker-compose.yml file, to set up the environment, configure the services and links between different containers, and also to use environment variables in the docker-compose.yml file.

How to run Docker commands without sudo?

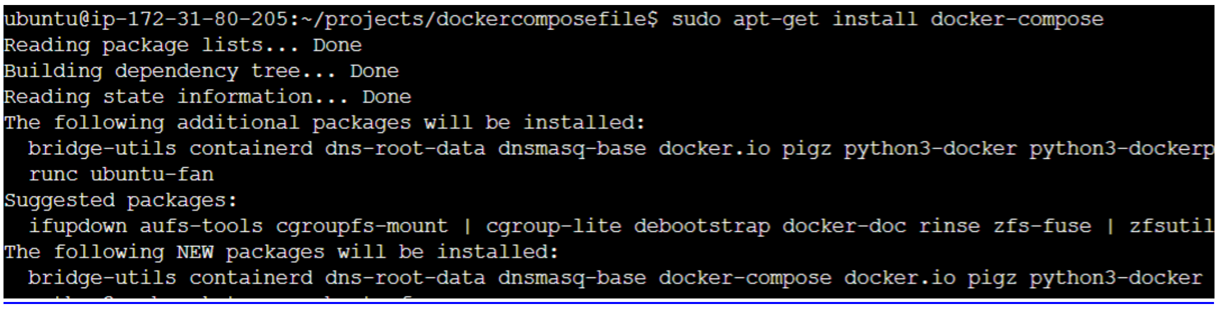
- Make sure docker is installed and system is updated
- sudo usermod -a -G docker $USER
- Reboot the machine.

Step 1 : Update your package manager and install docker in your instances



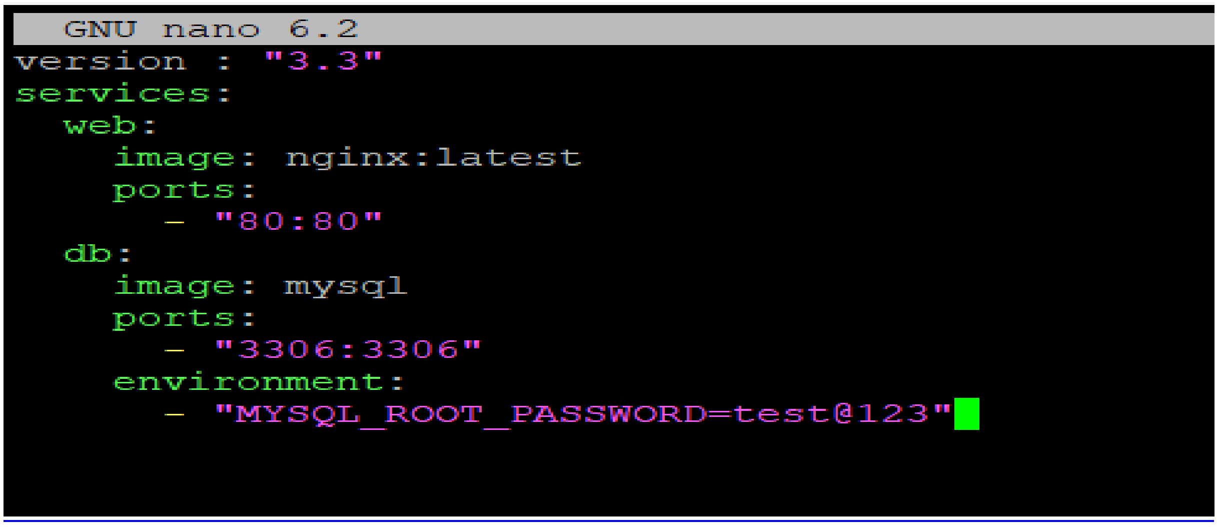
```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ sudo apt-get install docker
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  wmdocker
The following NEW packages will be installed:
  docker wmdocker
0 upgraded, 2 newly installed, 0 to remove and 9 not upgraded.
```

Step 2 : Install Docker-compose



```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ sudo apt-get install docker-compose
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz python3-docker python3-dockerp
  runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutil
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker-compose docker.io pigz python3-docker
```

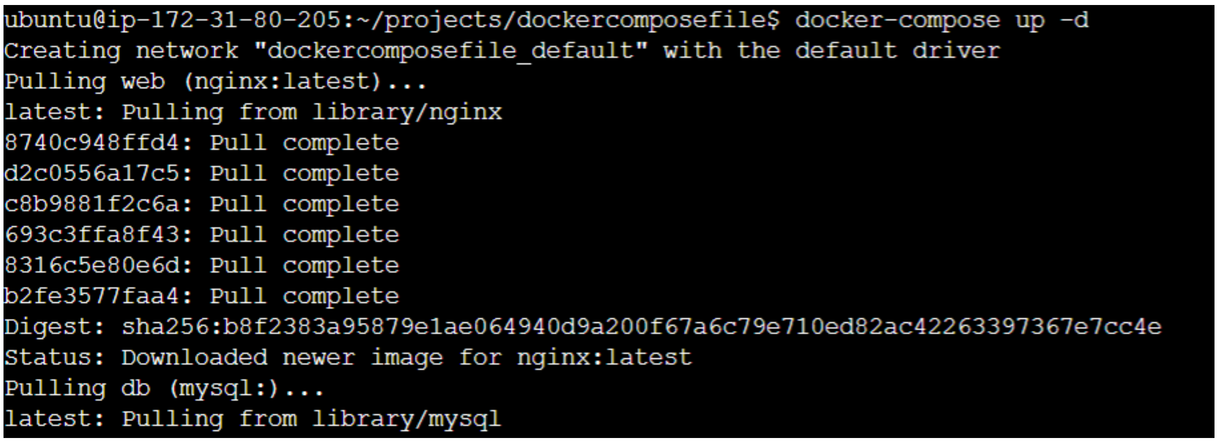
Step 3 : Create a docker-compose.yml file inside project folder



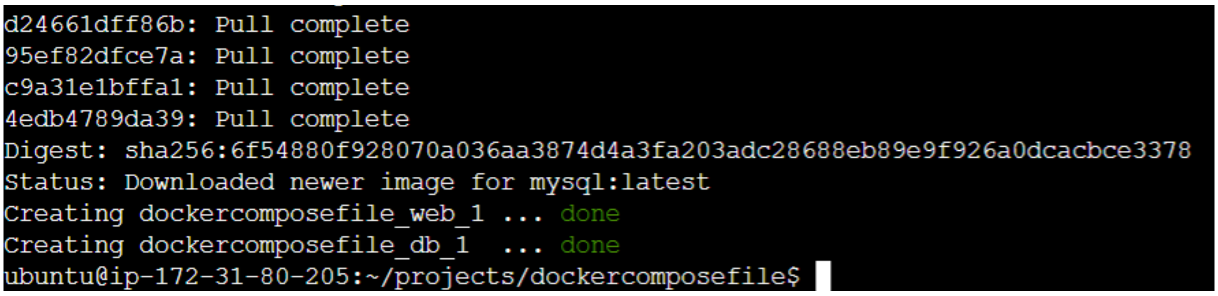
```
  GNU nano 6.2
version : "3.3"
services:
  web:
    image: nginx:latest
    ports:
      - "80:80"
  db:
    image: mysql
    ports:
      - "3306:3306"
    environment:
      - "MYSQL_ROOT_PASSWORD=test@123"
```

- Docker compose version is 3.3
- In services section defines all the different containers we will create like nginx & mysql
- For Web server using nginx:latest of port number 80
- For database using mysql of port number 3306
- To run mysql we specify the environment variable for mysql

Step 4 : Run docker-compose.yml file using command docker-compose up -d

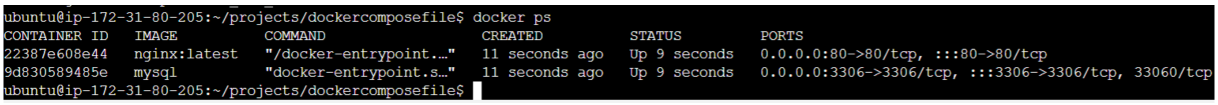


```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ docker-compose up -d
Creating network "dockercomposefile_default" with the default driver
Pulling web (nginx:latest)...
latest: Pulling from library/nginx
8740c948ffd4: Pull complete
d2c0556a17c5: Pull complete
c8b9881f2c6a: Pull complete
693c3ffa8f43: Pull complete
8316c5e80e6d: Pull complete
b2fe3577faa4: Pull complete
Digest: sha256:b8f2383a95879e1ae064940d9a200f67a6c79e710ed82ac42263397367e7cc4e
Status: Downloaded newer image for nginx:latest
Pulling db (mysql:)...
latest: Pulling from library/mysql
```



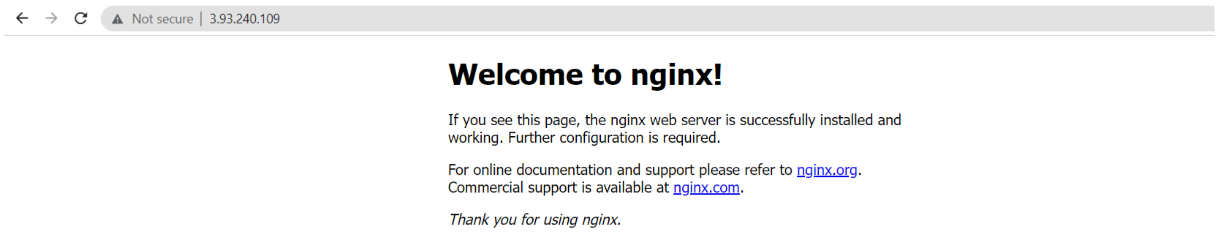
```
d24661dff86b: Pull complete
95ef82dfce7a: Pull complete
c9a31e1bffa1: Pull complete
4edb4789da39: Pull complete
Digest: sha256:6f54880f928070a036aa3874d4a3fa203adc28688eb89e9f926a0dcacbce3378
Status: Downloaded newer image for mysql:latest
Creating dockercomposefile_web_1 ... done
Creating dockercomposefile_db_1  ... done
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$
```

Step 5 : Check containers using command docker ps



```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS         PORTS
22387e608e44   nginx:latest   "/docker-entrypoint.…"   11 seconds ago   Up 9 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp
9d830589485e   mysql          "docker-entrypoint.s…"   11 seconds ago   Up 9 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$
```

Step 6 : Check your application is running or not in web browser using ec2's public ip.



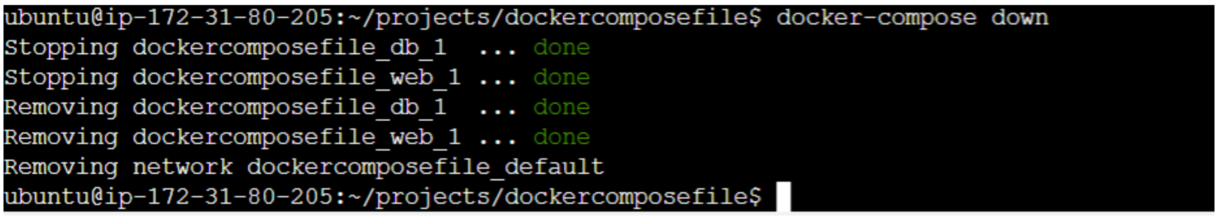
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

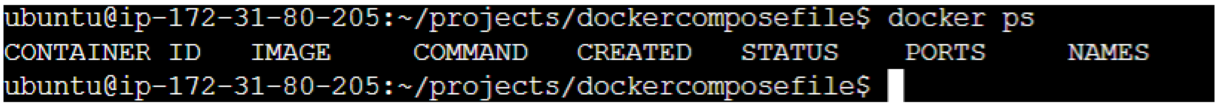
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Step 7 : To stop the services and to delete container use command docker-compose down.



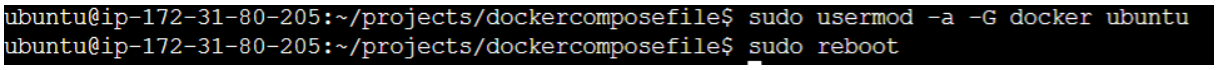
```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ docker-compose down
Stopping dockercomposefile_db_1  ... done
Stopping dockercomposefile_web_1 ... done
Removing dockercomposefile_db_1  ... done
Removing dockercomposefile_web_1 ... done
Removing network dockercomposefile_default
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$
```



```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$
```

Task-2

- **Pull a pre-existing Docker image from a public repository (e.g. Docker Hub) and run it on your local machine. Run the container as a non-root user (Hint- Use usermod command to give user permission to docker). Make sure you reboot instance after giving permission to user.**

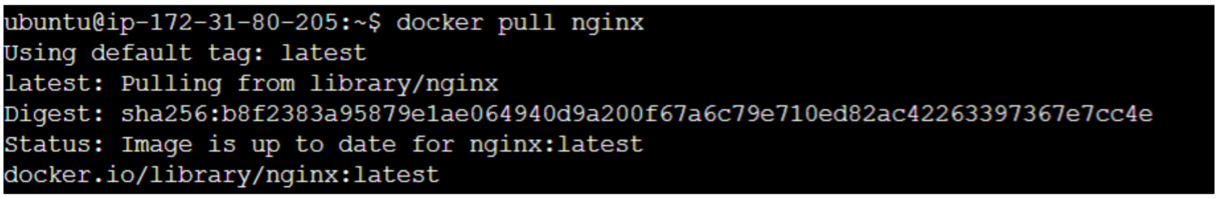


```
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ sudo usermod -a -G docker ubuntu
ubuntu@ip-172-31-80-205:~/projects/dockercomposefile$ sudo reboot
```

Note : Log out and log back in again to pick up the new docker group permissions. You can accomplish this by closing your current SSH terminal window and reconnecting to your instance in a new one. Your new SSH session will have the appropriate docker group permissions.

- **Inspect the container's running processes and exposed ports using the docker inspect command.**

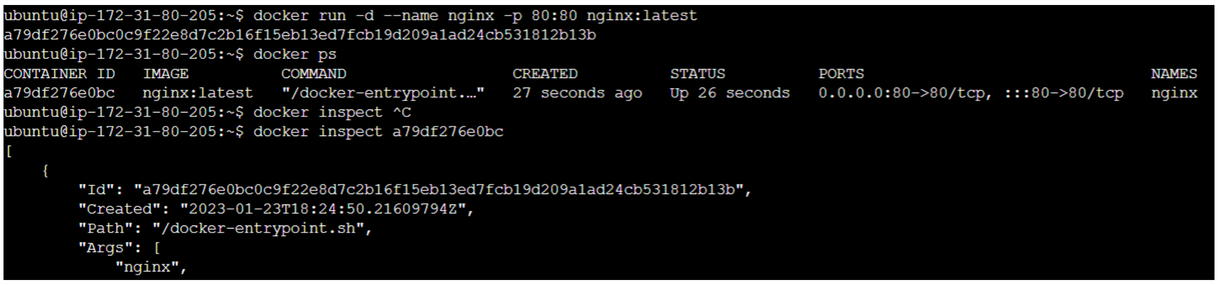
Step 1 : Copy docker image command from docker hub using command docker pull nginx



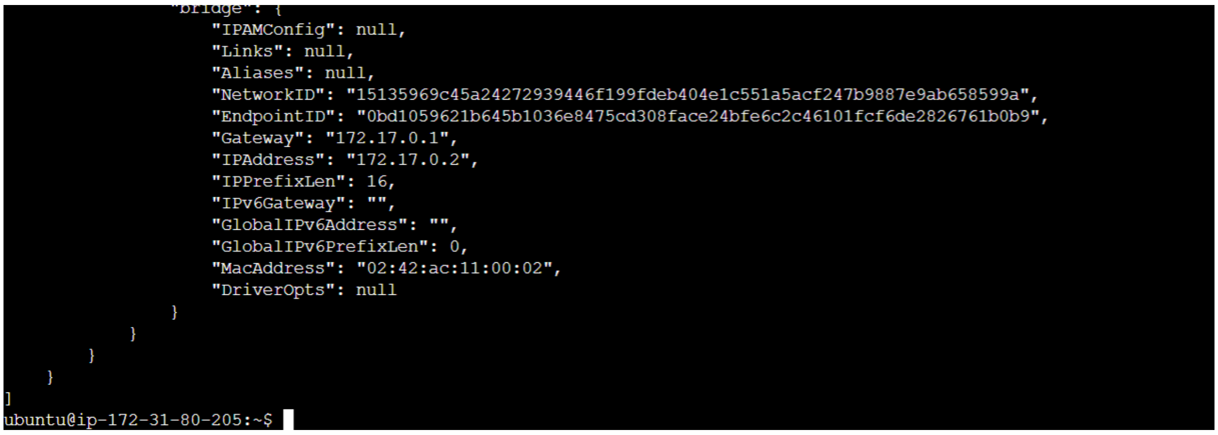
```
ubuntu@ip-172-31-80-205:~$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
Digest: sha256:b8f2383a95879e1ae064940d9a200f67a6c79e710ed82ac42263397367e7cc4e
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest
```

Step 2 : Run the container using command docker run -d –name <name> -p <port number>

Step 3 : To exposed ports use command docker inspect <container id>

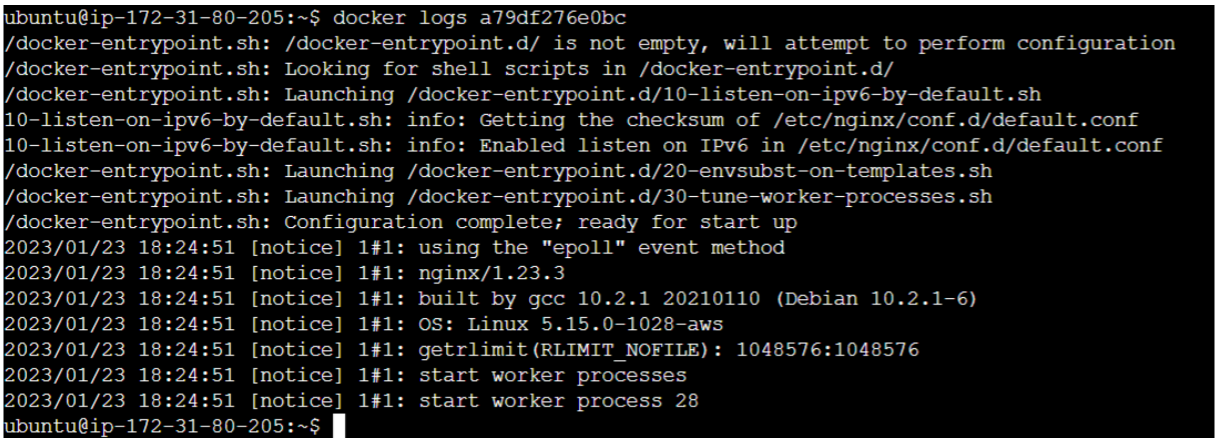


```
ubuntu@ip-172-31-80-205:~$ docker run -d --name nginx -p 80:80 nginx:latest
a79df276e0bc0c9f22e8d7c2b16f15eb13ed7fcb19d209a1ad24cb531812b13b
ubuntu@ip-172-31-80-205:~$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                NAMES
a79df276e0bc   nginx:latest   "/docker-entrypoint.…"   27 seconds ago   Up 26 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   nginx
ubuntu@ip-172-31-80-205:~$ docker inspect ^C
ubuntu@ip-172-31-80-205:~$ docker inspect a79df276e0bc
[
    {
        "Id": "a79df276e0bc0c9f22e8d7c2b16f15eb13ed7fcb19d209a1ad24cb531812b13b",
        "Created": "2023-01-23T18:24:50.21609794Z",
        "Path": "/docker-entrypoint.sh",
        "Args": [
            "nginx",
```



```
                "bridge": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "NetworkID": "15135969c45a24272939446f199fdeb404e1c551a5acf247b9887e9ab658599a",
                    "EndpointID": "0bd1059621b645b1036e8475cd308face24bfe6c2c46101fcf6de2826761b0b9",
                    "Gateway": "172.17.0.1",
                    "IPAddress": "172.17.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "MacAddress": "02:42:ac:11:00:02",
                    "DriverOpts": null
                }
            }
        }
    }
]
ubuntu@ip-172-31-80-205:~$
```

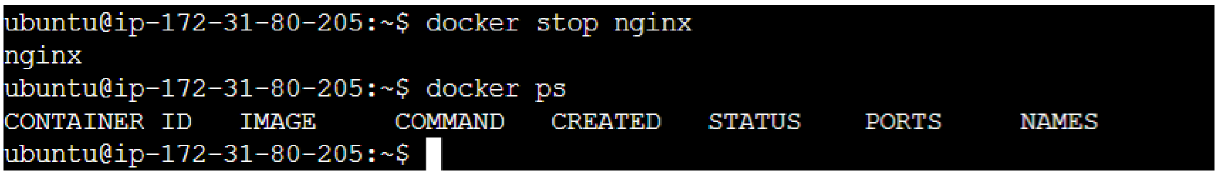
- **Use the docker logs command to view the container's log output.**



```
ubuntu@ip-172-31-80-205:~$ docker logs a79df276e0bc
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/01/23 18:24:51 [notice] 1#1: using the "epoll" event method
2023/01/23 18:24:51 [notice] 1#1: nginx/1.23.3
2023/01/23 18:24:51 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/01/23 18:24:51 [notice] 1#1: OS: Linux 5.15.0-1028-aws
2023/01/23 18:24:51 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/01/23 18:24:51 [notice] 1#1: start worker processes
2023/01/23 18:24:51 [notice] 1#1: start worker process 28
ubuntu@ip-172-31-80-205:~$
```

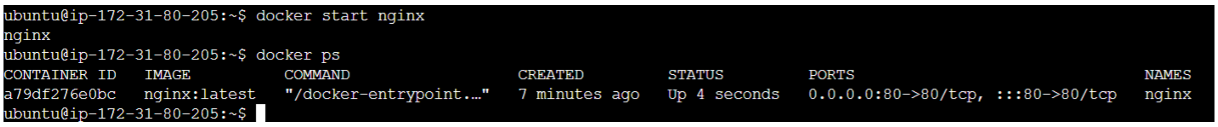
- **Use the docker stop and docker start commands to stop and start the container.**

  Step 1 : To stop container use command docker stop <container name>



```
ubuntu@ip-172-31-80-205:~$ docker stop nginx
nginx
ubuntu@ip-172-31-80-205:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-80-205:~$
```

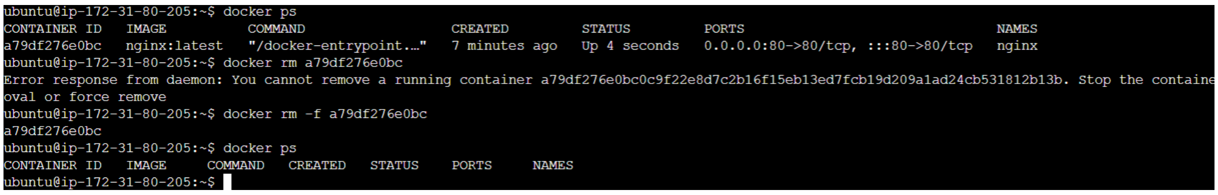
  Step 2 : To start container use command docker stop <container name>



```
ubuntu@ip-172-31-80-205:~$ docker start nginx
nginx
ubuntu@ip-172-31-80-205:~$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                               NAMES
a79df276e0bc   nginx:latest   "/docker-entrypoint.…"   7 minutes ago   Up 4 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   nginx
ubuntu@ip-172-31-80-205:~$
```

- **Use the docker rm command to remove the container when you're done.**

  Use -f option to remove container forceable



```
ubuntu@ip-172-31-80-205:~$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                               NAMES
a79df276e0bc   nginx:latest   "/docker-entrypoint.…"   7 minutes ago   Up 4 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   nginx
ubuntu@ip-172-31-80-205:~$ docker rm a79df276e0bc
Error response from daemon: You cannot remove a running container a79df276e0bc0c9f22e8d7c2b16f15eb13ed7fcb19d209a1ad24cb531812b13b. Stop the containe
oval or force remove
ubuntu@ip-172-31-80-205:~$ docker rm -f a79df276e0bc
a79df276e0bc
ubuntu@ip-172-31-80-205:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-80-205:~$
```