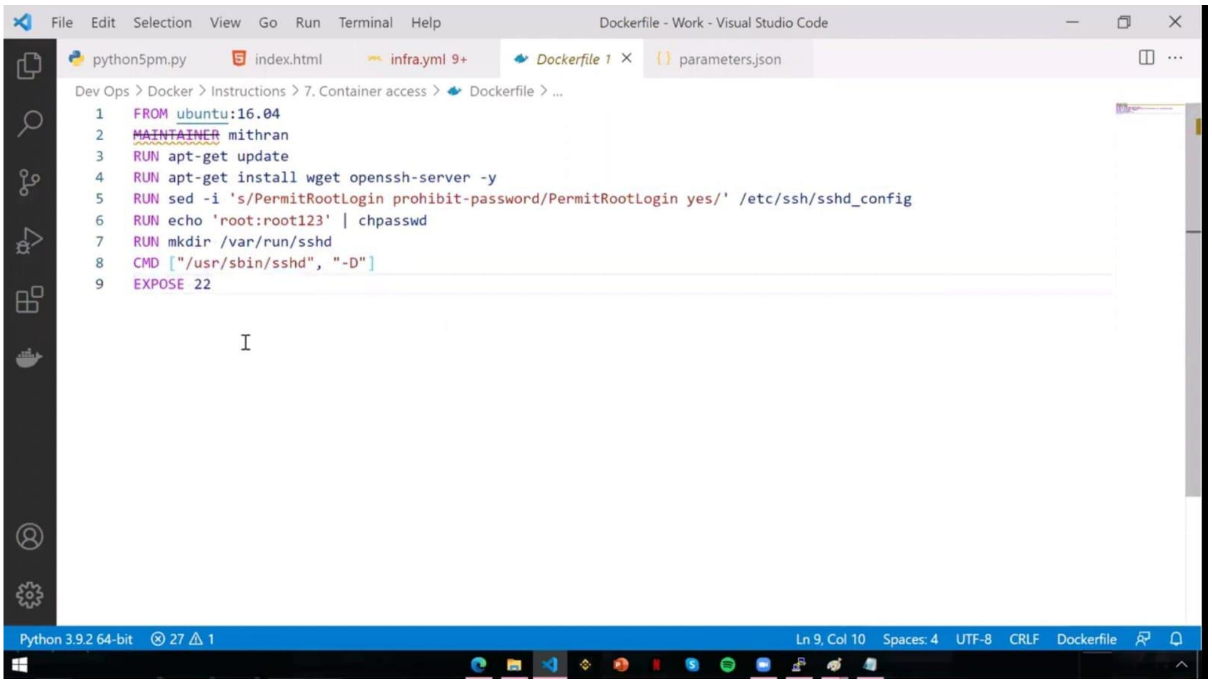
DOCKER


File Edit Selection View Go Run Terminal Help
Dockerfile - Work - Visual Studio Code
python5pm.py
index.html
infra.yml 9+
Dockerfile 1
parameters.json
Dev Ops > Docker > Instructions > 7. Container access > Dockerfile > ...
Python 3.9.2 64-bit
Ln 9, Col 10
Spaces: 4
UTF-8
CRLF
Dockerfile

```
FROM ubuntu:16.04
MAINTAINER mithran
RUN apt-get update
RUN apt-get install wget openssh-server -y
RUN sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config
RUN echo 'root:root123' | chpasswd
RUN mkdir /var/run/sshd
CMD ["/usr/sbin/sshd", "-D"]
EXPOSE 22
```

It is used to create a container for giving access to the another user

Expose is used to open the ssh port number

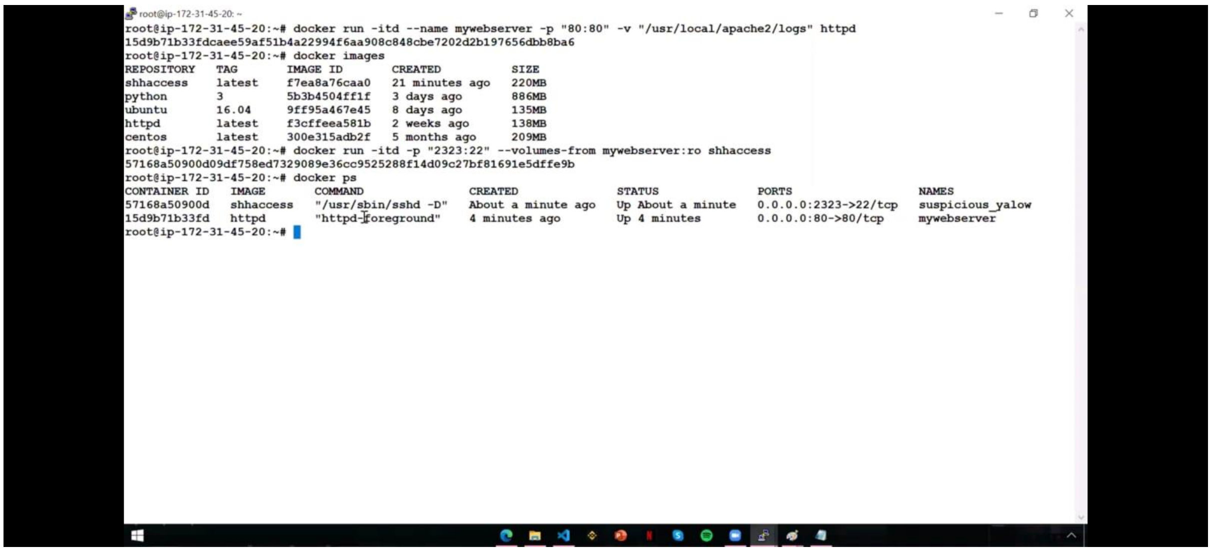
After created a docker file we have to create a image using docker file and create a container using -p (2323:22)portforwarding....

After created an created an container give the public ip and 2323 port number to the user...

Not pem file because in docker creation itself we created password authentication and created an username and password....

Give the username and password to that user...

Now the user can only acces that container...



```
root@ip-172-31-45-20:~# docker run -itd --name mywebserver -p "80:80" -v "/usr/local/apache2/logs" httpd
15d9b71b33fdcaee59af51b4a22994f6aa908c848cbe7202d2b197656dbb8ba6
root@ip-172-31-45-20:~# docker images
REPOSITORY   TAG      IMAGE ID       CREATED          SIZE
shhaccess    latest   f7ea8a76caa0   21 minutes ago   220MB
python       3        5b3b4504ff1f   3 days ago       886MB
ubuntu       16.04    9ff95a467e45   8 days ago       135MB
httpd        latest   f3cffeea581b   2 weeks ago      138MB
centos       latest   300e315adb2f   5 months ago     209MB
root@ip-172-31-45-20:~# docker run -itd -p "2323:22" --volumes-from mywebserver:ro shhaccess
57168a50900d09df758ed7329089e36cc9525288f14d09c27bf81691e5dffe9b
root@ip-172-31-45-20:~# docker ps
CONTAINER ID   IMAGE       COMMAND                CREATED              STATUS               PORTS                    NAMES
57168a50900d   shhaccess   "/usr/sbin/sshd -D"    About a minute ago   Up About a minute    0.0.0.0:2323->22/tcp     suspicious_yalow
15d9b71b33fd   httpd       "httpd-foreground"     4 minutes ago        Up 4 minutes         0.0.0.0:80->80/tcp       mywebserver
root@ip-172-31-45-20:~#
```

Scenerio : It is used to give permission to a particular folder on docker container

In first command it will create an container and save the volumes on (var/lib/docker/volumes/<volume_id>/_data) rather than host folder

In second command it will use VOLUME-FROM command and it will sync with log files which have created before... And it have only read access

How to user access this container?

Which have we created a image before, he use that same image to give access to the user(shhaccess)

```
docker run --name mywebserver -p "80:80" -v
"/usr/local/apache2/logs" httpd
```

Creates and starts a new container named mywebserver using the httpd image.

Maps port 80 on the host to port 80 in the container.

Mounts a volume from /usr/local/apache2/logs for logging.

2. docker images

Lists all locally available Docker images with details like repository, tag, image ID, creation time, and size.

3. docker ps

Displays all running containers, showing details like container ID, image used, commands, creation time, status, ports, and container names.

4. docker run -p "2323:22" --volumes-from mywebserver:ro shhaccess

Starts a new container from the shhaccess image.

Maps port 22 in the container to port 2323 on the host.

Shares volumes from the mywebserver container in read-only mode (--volumes-from ...:ro).

So volume-from is used to get the only volumes from previous volume

Is it mount from the already running httpd or it will create new httpd and mount from new container?

Does not mount a volume to an already running container.

Instead, it creates a new container from the httpd image and mounts the specified volume during the creation
of this container.

Docker run mysql

When we run this command it will download but can't run...

We don't know why..

How to see?

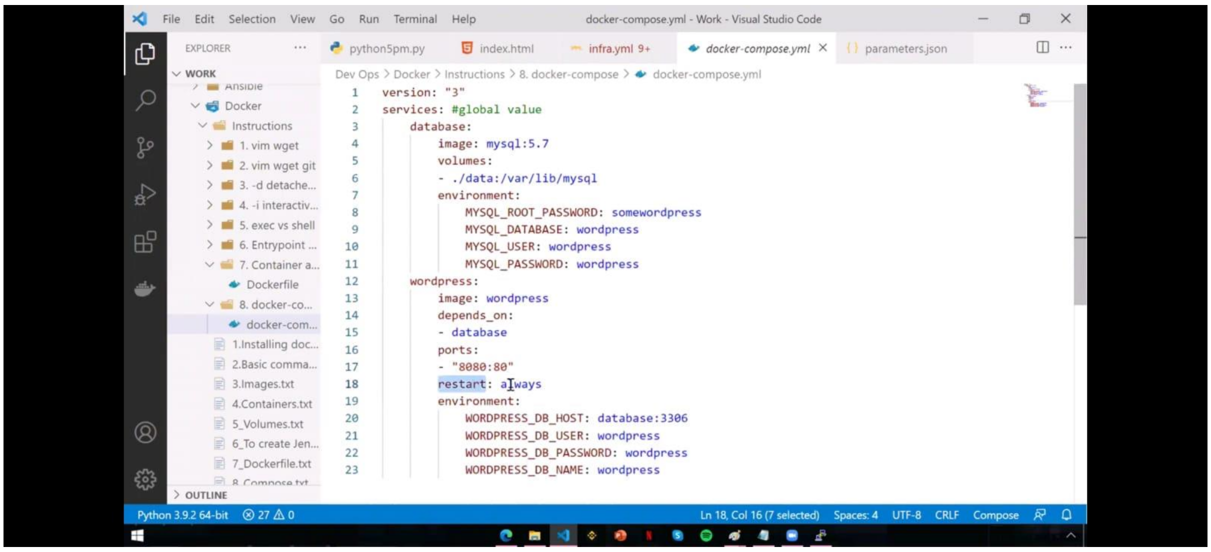
Docker logs mysql

It shows to set the password for db..

To set the pw,

docker run -itd -e MYSQL_ROOT_PASSWORD = Nith123 mysql

When we run this command it will run



```yaml
version: "3"
services: #global value
    database:
        image: mysql:5.7
        volumes:
        - ./data:/var/lib/mysql
        environment:
            MYSQL_ROOT_PASSWORD: somewordpress
            MYSQL_DATABASE: wordpress
            MYSQL_USER: wordpress
            MYSQL_PASSWORD: wordpress
    wordpress:
        image: wordpress
        depends_on:
        - database
        ports:
        - "8080:80"
        restart: always
        environment:
            WORDPRESS_DB_HOST: database:3306
            WORDPRESS_DB_USER: wordpress
            WORDPRESS_DB_PASSWORD: wordpress
            WORDPRESS_DB_NAME: wordpress
```

Docker compose yaml file to create mysql and wordpress

Above volume is it get sql data from mysql and save it into /var/lib/mysql

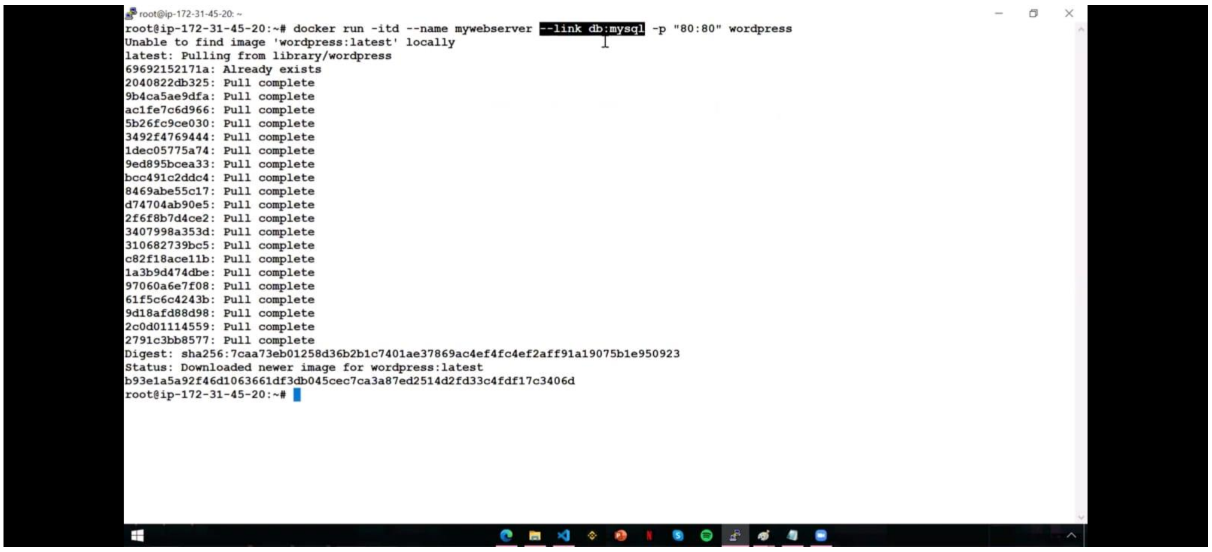
----

How to save yaml file on our machine?

-Docker-compose.yaml

After saved on our local machine How to run it?

docker-compose up -d



```
root@ip-172-31-45-20: ~
root@ip-172-31-45-20:~# docker run -itd --name mywebserver --link db:mysql -p "80:80" wordpress
Unable to find image 'wordpress:latest' locally
latest: Pulling from library/wordpress
69692152171a: Already exists
2040822db325: Pull complete
9b4ca5ae9dfa: Pull complete
ac1fe7c6d966: Pull complete
5b26fc9ce030: Pull complete
3492f4769444: Pull complete
1dec05775a74: Pull complete
9ed895bcea33: Pull complete
bcc491c2ddc4: Pull complete
8469abe55c17: Pull complete
d74704ab90e5: Pull complete
2f6f8b7d4ce2: Pull complete
3407998a353d: Pull complete
310682739bc5: Pull complete
c82f18ace11b: Pull complete
1a3b9d474dbe: Pull complete
97060a6e7f08: Pull complete
61f5c6c4243b: Pull complete
9d18afd88d98: Pull complete
2c0d01114559: Pull complete
2791c3bb8577: Pull complete
Digest: sha256:7caa73eb01258d36b2b1c7401ae37869ac4ef4fc4ef2aff91a19075b1e950923
Status: Downloaded newer image for wordpress:latest
b93e1a5a92f46d1063661df3db045cec7ca3a87ed2514d2fd33c4fdf17c3406d
root@ip-172-31-45-20:~#
```

Docker link

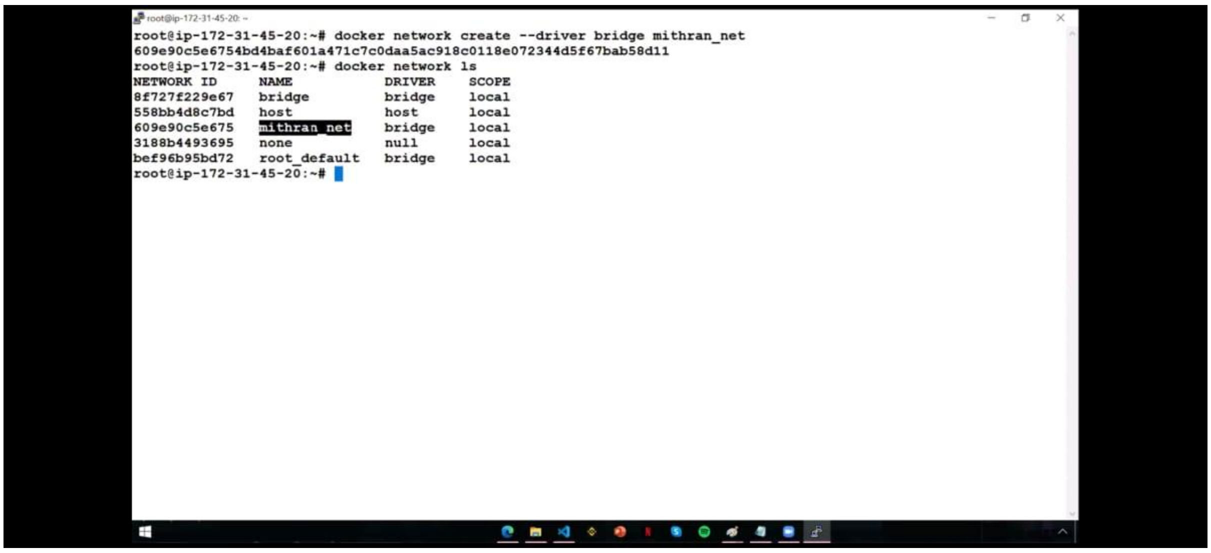
Docker link is used to link the containers

Above scenerio: It will create a WordPress and link with mysql because wordpress need mysql...

--link db --> db is the name what we have give to mysql while creating

mysql --> is the image what we going to use

While link two containers OS is no matter



```
root@ip-172-31-45-20:~# docker network create --driver bridge mithran_net
609e90c5e6754bd4baf601a471c7c0daa5ac918c0118e072344d5f67bab58d11
root@ip-172-31-45-20:~# docker network ls
NETWORK ID     NAME           DRIVER    SCOPE
8f727f229e67   bridge         bridge    local
558bb4d8c7bd   host           host      local
609e90c5e675   mithran_net    bridge    local
3188b4493695   none           null      local
bef96b95bd72   root_default   bridge    local
root@ip-172-31-45-20:~#
```

Docker network

Docker network is a just a connection between containers

Three types of docker networks

* Bridge --> It is a connection we used regularly(Isolated containers)

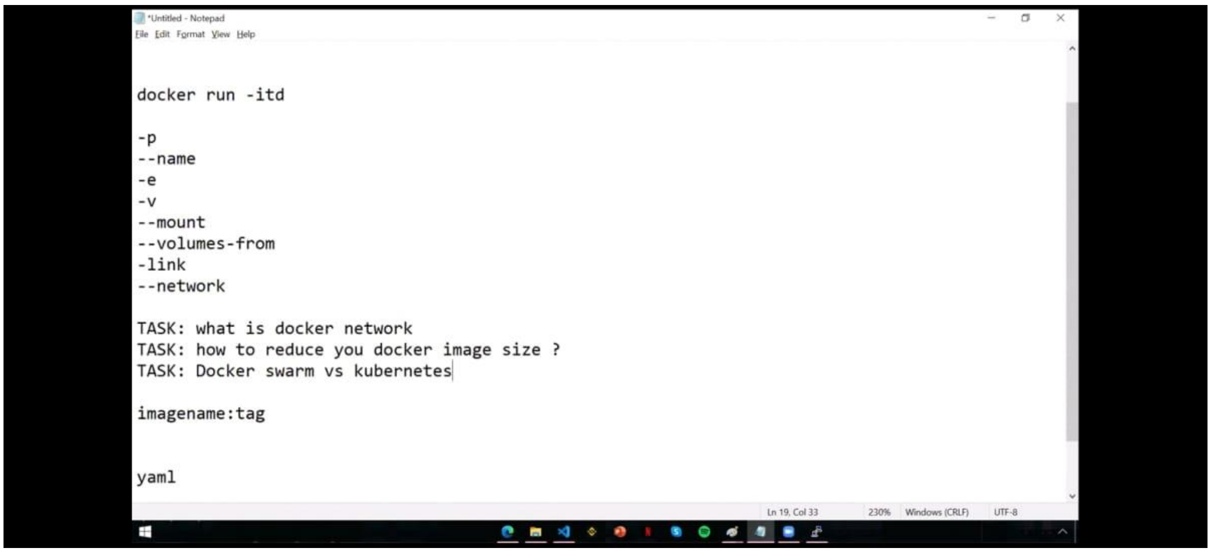
* Hosts-->

* None --> It is we not going to use any type of network

How to create new network?

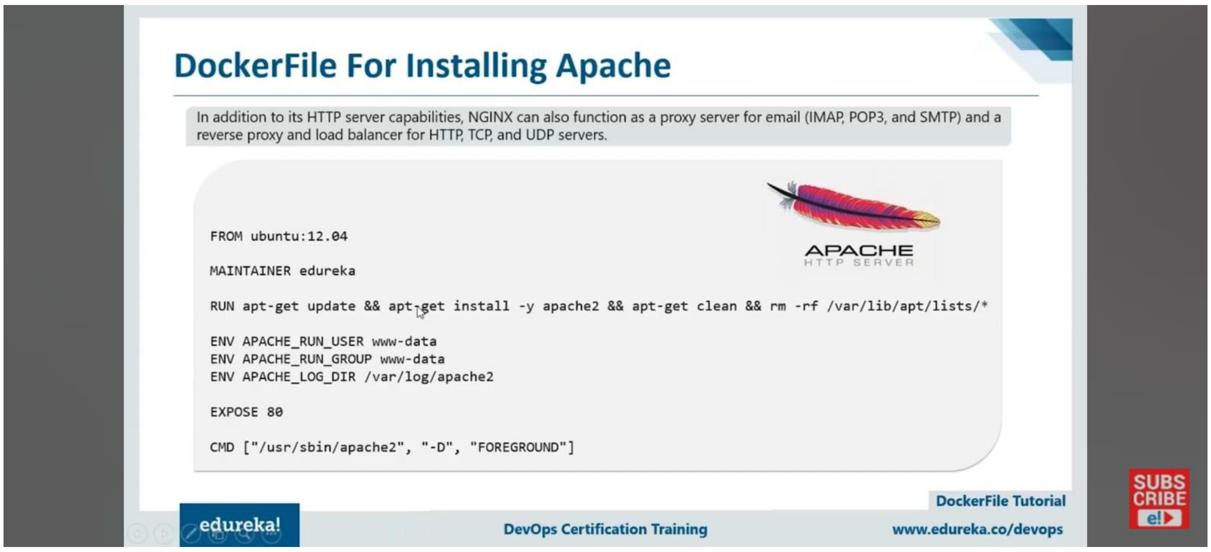
That is shown above and these three are default we cant able to change

----



```
docker run -itd

-p
--name
-e
-v
--mount
--volumes-from
-link
--network

TASK: what is docker network
TASK: how to reduce you docker image size ?
TASK: Docker swarm vs kubernetes

imagename:tag


yaml
```

TASKs⚠️⚠️



DockerFile For Installing Apache

In addition to its HTTP server capabilities, NGINX can also function as a proxy server for email (IMAP, POP3, and SMTP) and a reverse proxy and load balancer for HTTP, TCP, and UDP servers.

```
FROM ubuntu:12.04

MAINTAINER edureka

RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2

EXPOSE 80

CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]
```

In Dockerfile,

CMD ["catalina.sh", "run"]

* catalina.sh: This is the script provided by Tomcat to control the server.

* run: This argument runs Tomcat in the foreground, keeping the container active.

For Apache web server,

```
CMD [ "usr/sbin/apache2", "-D", "FOREGROUND" ]
```

Dockerfile:

Defines how to build a single container image.

Includes setup instructions (e.g., installing dependencies, copying files).

Example: Used to create a custom image for your application.

2. Docker Compose:

Manages multiple containers locally and their relationships.

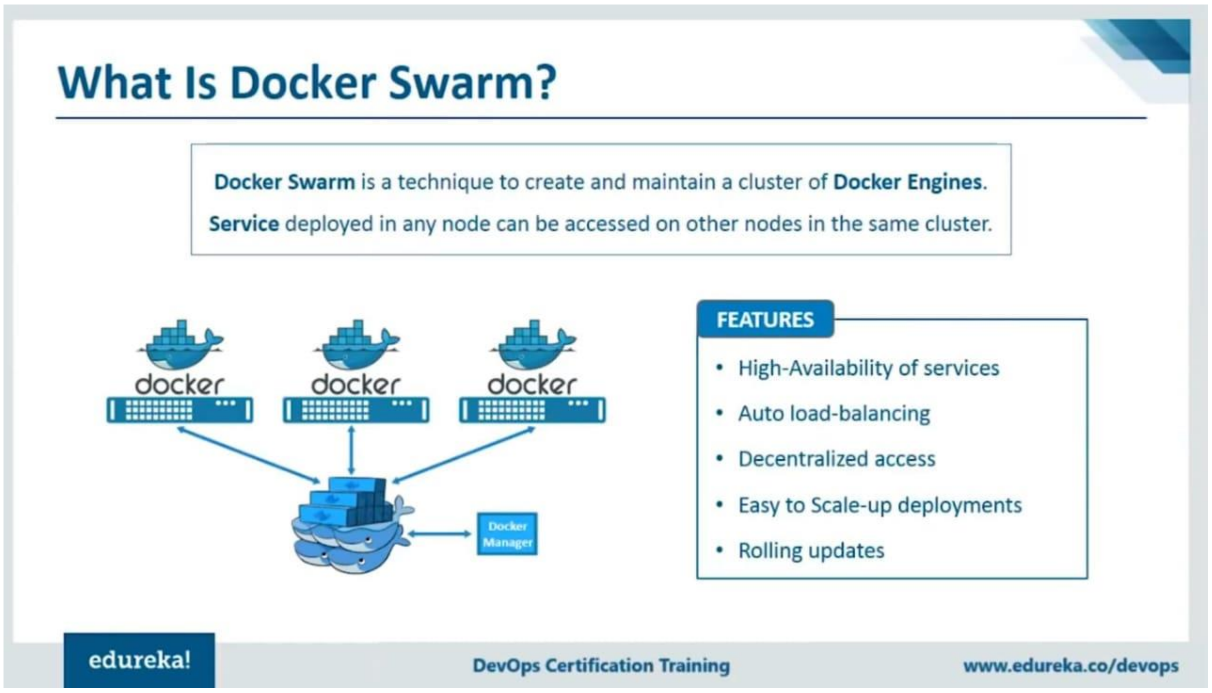
Defined in a docker-compose.yml file.

Example: Runs an app, its database, and other services locally.

3. Docker Swarm:

A tool that helps you run and manage containers on multiple servers working together as a team (called a cluster).

Simple Words: It makes sure your app runs smoothly across many computers, shares the workload, and stays online even if one server fails.



What Is Docker Swarm?

Docker Swarm is a technique to create and maintain a cluster of **Docker Engines**.
Service deployed in any node can be accessed on other nodes in the same cluster.

FEATURES

- High-Availability of services
- Auto load-balancing
- Decentralized access
- Easy to Scale-up deployments
- Rolling updates

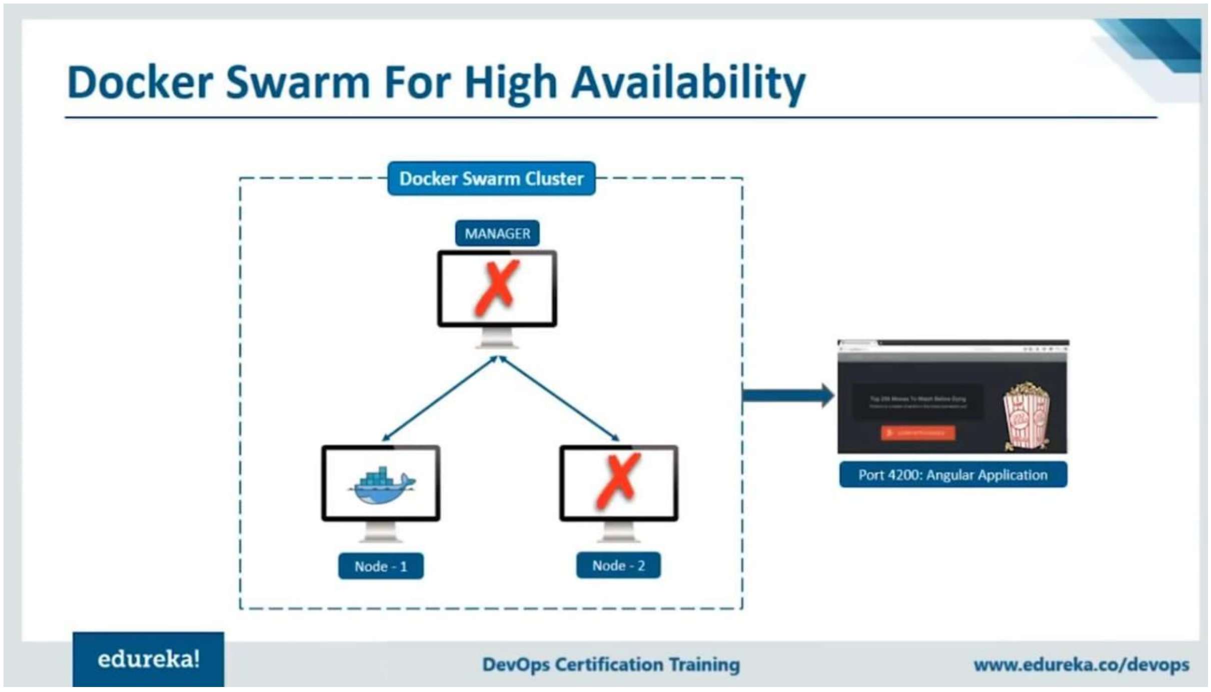
edureka! DevOps Certification Training www.edureka.co/devops

Docker swarm is act like control node and manage node...

In docker swarm, It manages the load balancing when using docker swarm for deploy application and it leads to high availability

If one node us down, the nodes will take over that requests from the user and docker swarm is agentless

---



Docker Swarm For High Availability
Docker Swarm Cluster
MANAGER
Node - 1
Node - 2
Port 4200: Angular Application
edureka!
DevOps Certification Training
www.edureka.co/devops

In swarm, Even docker control node goes down... The manage down will not off the connection and it still works because that system also have application code and port number will open. So, it works

---

Docker services

* docker service create --name "Ang-app" -p 4200:4200 httpd

It is used to create container on the control node

* docker service create --name "Ang-app" -p 4200:4200 --
mode global httpd

It is used to create a container on control node and also manages nodes which have connected to this control node

* docker service ps --> To see service containers

* docker service rm httpd --> It will delete container

How to deploy to particular num of nodes in docker?

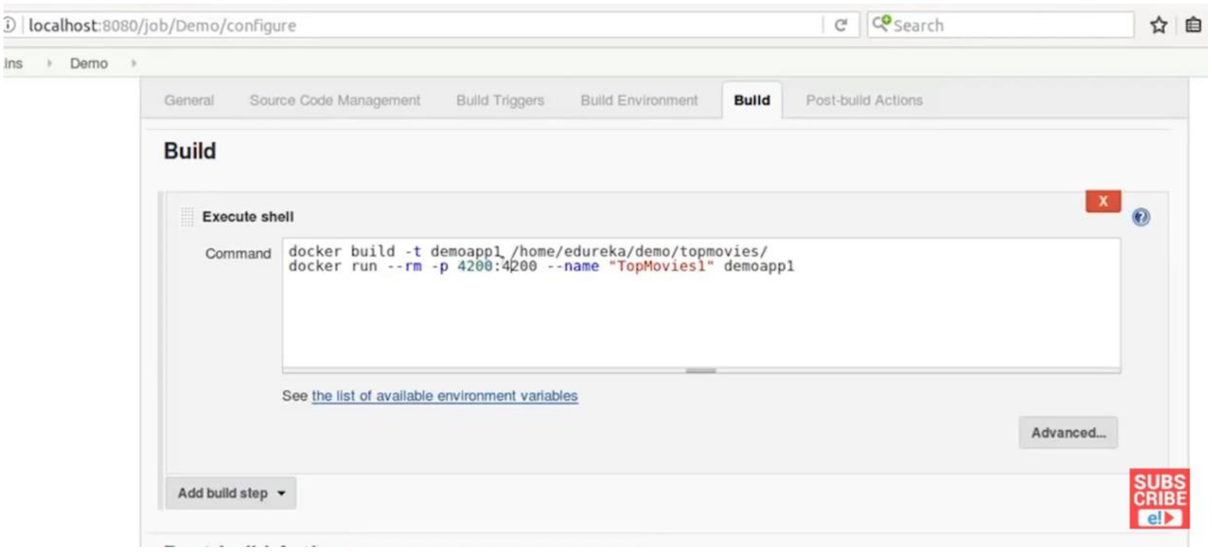
* docker service create --name "ang-app" -p 4200:4200 --
replicas 2 httpd

It will deploy on only 2 nodes even there are are multiple nodes avilable

* docker service scale ang-app(name)=5

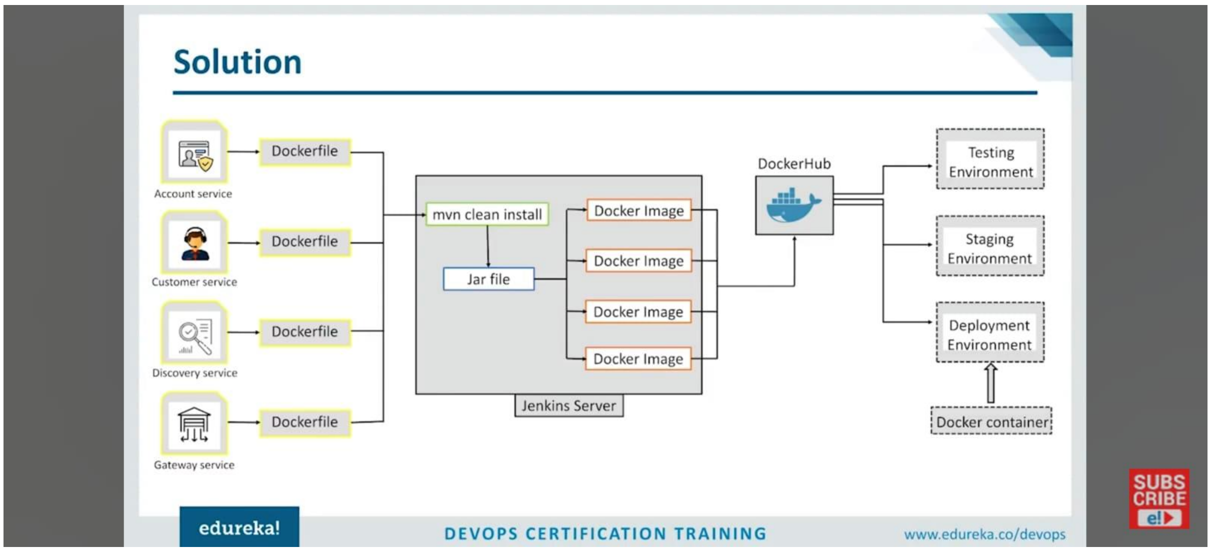
It will scale upto 5 containers in connected nodes by creating multiple containers if there is only 2 node available

---



```
docker build -t demoapp1 /home/edureka/demo/topmovies/
docker run --rm -p 4200:4200 --name "TopMovies1" demoapp1
```

Real time scenerio: When using docker in jenkins, we have to write everything on dockerfile and put on github... Only docker image build and run container command only have to write on Jenkins execute shell



It is the flowchar for deploy microservices into docker containers using jenkins

How to do it?

In linux, we have to store the codes on that particular folder

(Ex: Product catlog on product catlog folder)

* use cd command to go on that particular folder(write cd <path/to/folder> on jenkins execute shell)

* use mvn clean install command to convert into jar file

* use docker build command to create jar file into docker images(docker build -t <Imagename/Tag> </path/to/that file>(use path or simply use .))

* push to docker hub(if needed)

This is how it works.

```
docker run --rm httpd cat /etc/os-release
```

To check which os can be used,

For every image in dockerfile when using from command it takes linux is the default os(eg: jenkins, maven, httpd)

```
stage('Build Account-Service') {
          steps {
             dir('account-service') {
                sh 'mvn clean install'
             }
          }
       }
```

Here, dir('account-service') line is used for it acts like cd command in shell script.so when we want to change directory through Jenkins use dir dor change directory

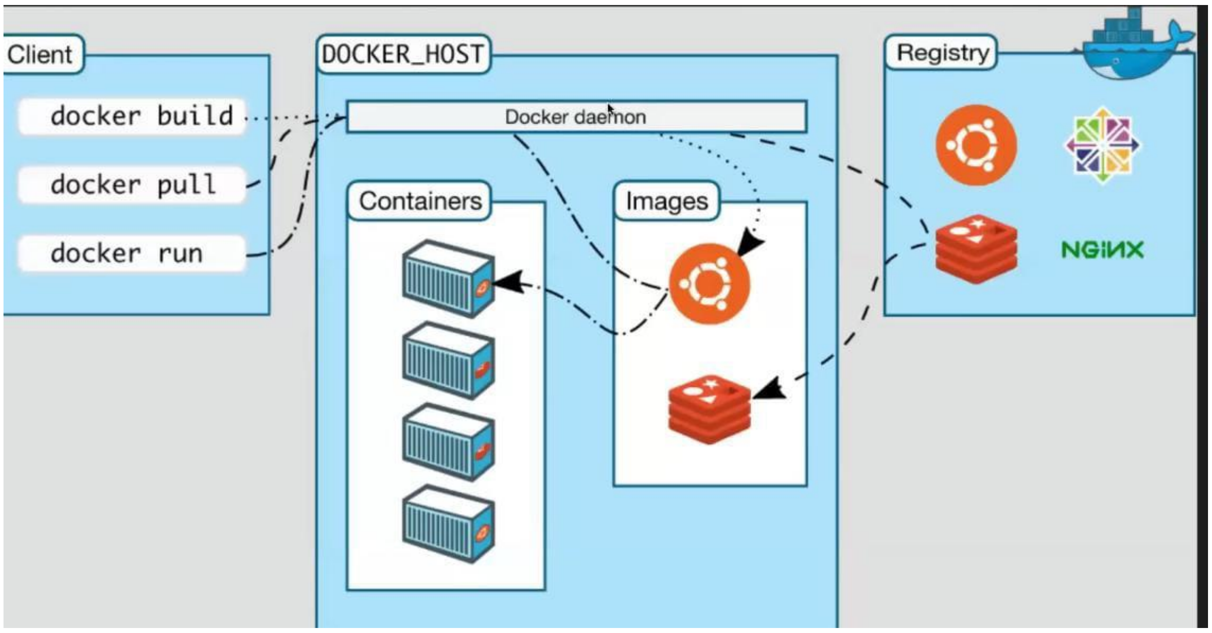
---

ADD target/petclinic.war petclinic.war

In this command,

First target/petclinic.war file will be on local machine

Second petclinic.war file will be saved on inside container



Docker architecture

Docker Daemon--> is a intermediate between docker commands and docker interior

It is get a commands from user and apply on docker hub or inside images to build a container

---

```
# First stage: Build Java application
FROM maven:3.8.5-openjdk-17 AS build

WORKDIR /app
COPY . .

# Build the Java application
RUN mvn clean package -DskipTests

# Second stage: Run the Java application
FROM openjdk:17-jdk-slim

WORKDIR /app
```

```
# Copy the compiled JAR from the build stage
COPY --from=build /app/target/*.jar app.jar

# Run the Java application
ENTRYPOINT ["java", "-jar", "app.jar"]
```

----

* It is the java web application for multi stage build

* First stage for Development side and it contains all dependencies

* second stage only contain java environment and app.jar file which have contents for website and it contains less storage because it doesnt have dependencies

---

Entrypoint command

The ENTRYPOINT command in a Dockerfile is used to specify the main process that should run when a container starts.

Use ENTRYPOINT when you want your container to always execute a specific command, making it act like a dedicated application.

Dockerfile:

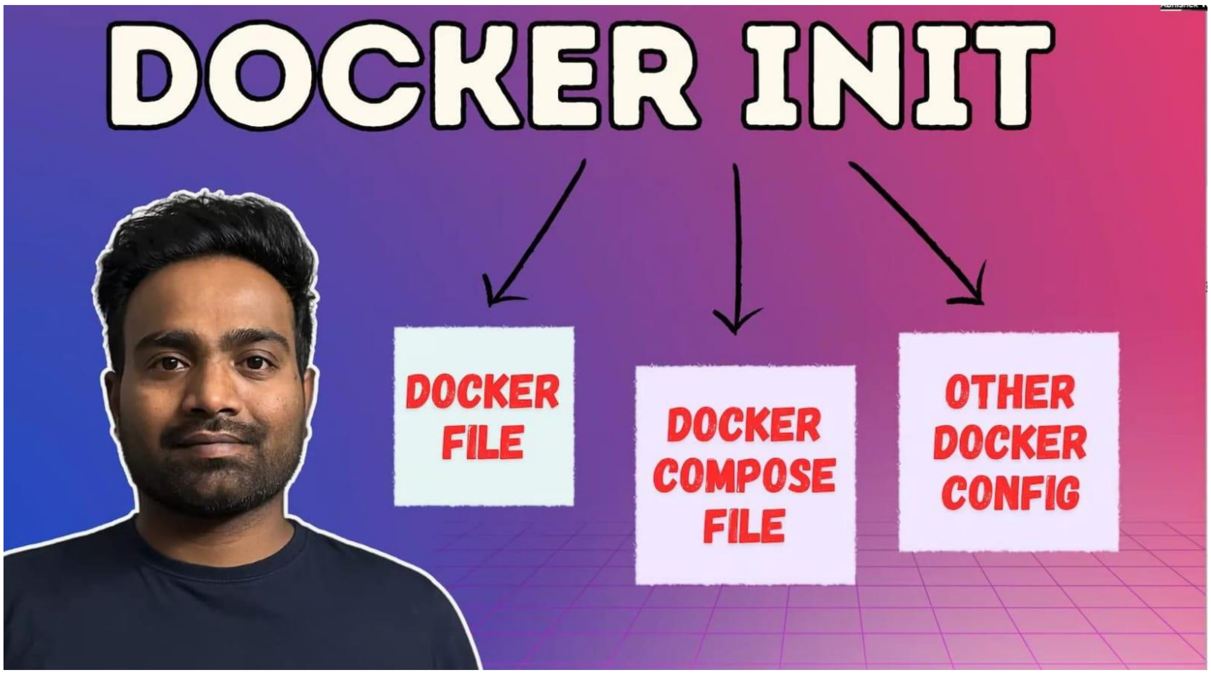
```
FROM ubuntu:latest
ENTRYPOINT ["echo", "Hello from Docker!"]
```

Commands:

```
docker build -t myimage .
docker run myimage
```

Output:

```
Hello from Docker!
```



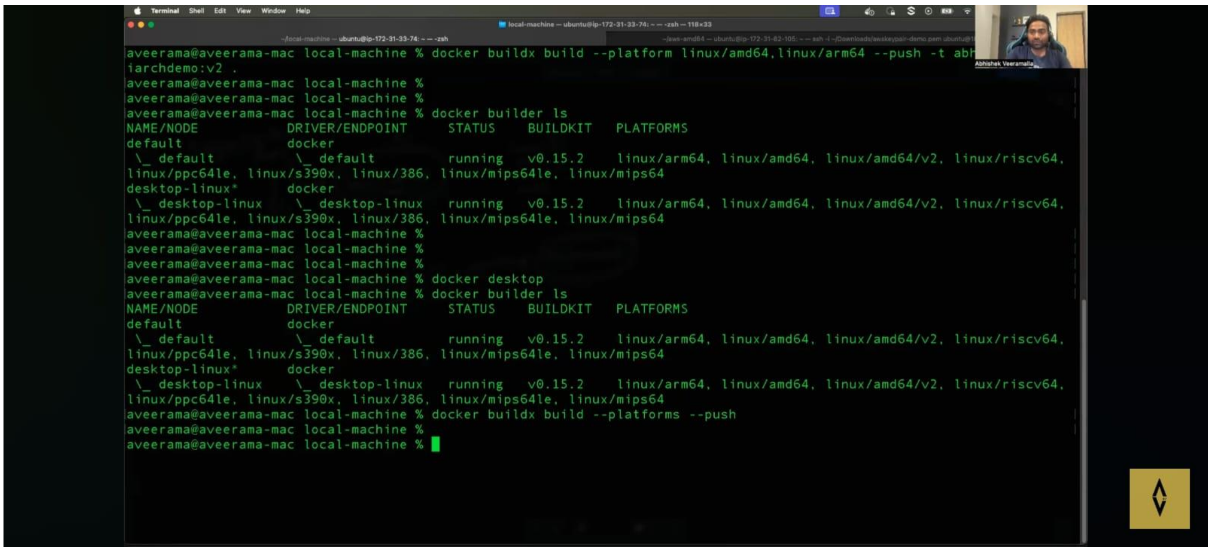
Docker init is a command used to create docketfile, compose file and README.md

How to use it?

Change directory to path where code was and type(Docker init).

It creates a files Automatically with the dependencies

(STILL IT IS IN GROWING STATE, IT GIVES ACCURATE FOR SHORTER LEVEL CODES BUT LARGER CODES WE HAVE TO MAKE QUITE CHANGES)



```
aveerama@aveerama-mac local-machine % docker buildx build --platform linux/amd64,linux/arm64 --push -t abh
iarchdemo:v2 .
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine % docker builder ls
NAME/NODE           DRIVER/ENDPOINT     STATUS    BUILDKIT   PLATFORMS
default             docker
 \_ default          \_ default         running   v0.15.2    linux/arm64, linux/amd64, linux/amd64/v2, linux/riscv64,
linux/ppc64le, linux/s390x, linux/386, linux/mips64le, linux/mips64
desktop-linux*      docker
 \_ desktop-linux    \_ desktop-linux   running   v0.15.2    linux/arm64, linux/amd64, linux/amd64/v2, linux/riscv64,
linux/ppc64le, linux/s390x, linux/386, linux/mips64le, linux/mips64
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine % docker desktop
aveerama@aveerama-mac local-machine % docker builder ls
NAME/NODE           DRIVER/ENDPOINT     STATUS    BUILDKIT   PLATFORMS
default             docker
 \_ default          \_ default         running   v0.15.2    linux/arm64, linux/amd64, linux/amd64/v2, linux/riscv64,
linux/ppc64le, linux/s390x, linux/386, linux/mips64le, linux/mips64
desktop-linux*      docker
 \_ desktop-linux    \_ desktop-linux   running   v0.15.2    linux/arm64, linux/amd64, linux/amd64/v2, linux/riscv64,
linux/ppc64le, linux/s390x, linux/386, linux/mips64le, linux/mips64
aveerama@aveerama-mac local-machine % docker buildx build --platforms --push
aveerama@aveerama-mac local-machine %
aveerama@aveerama-mac local-machine %
```

Docker multiarch platform

Docker multiarch is used for when there is an multiple different processors like(arm64, arm32, X86)..for each processor we want to write multiple dockerfile to create image... To overcome that problem we use docker multiarch platform

Requirements:

* Docker Desktop (or) Build(if it is ec2)

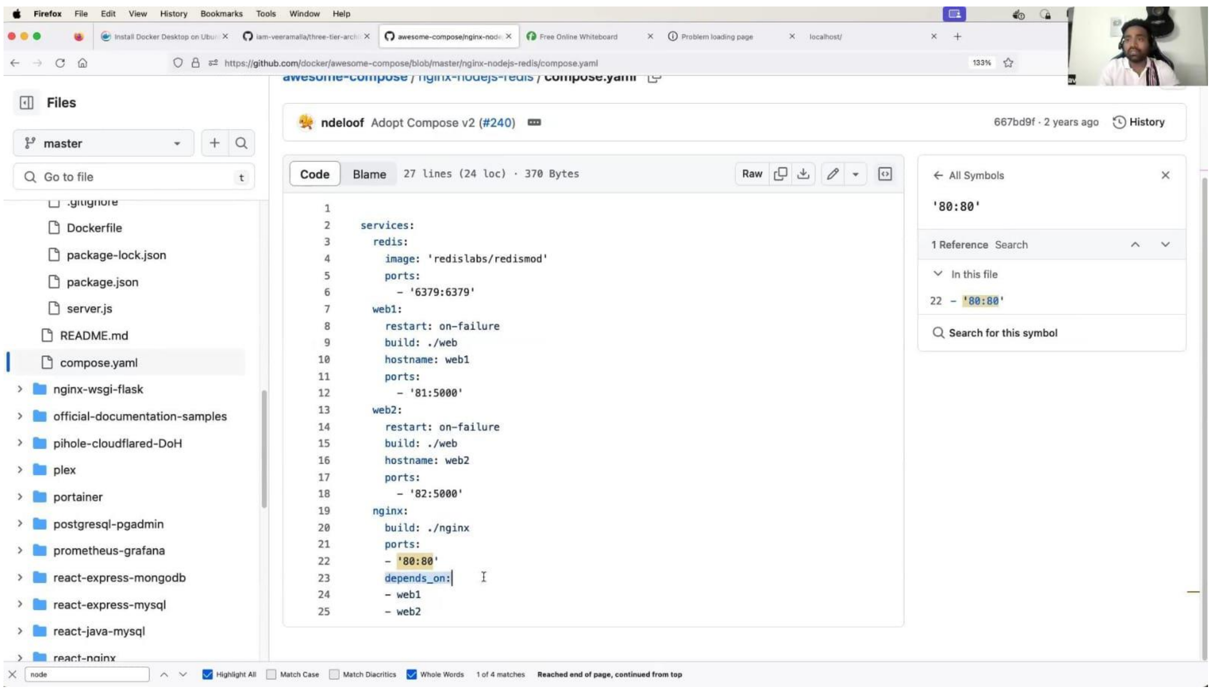
Advantage:

* We can able to push a docker image include with all processors in a single image name and tag

(We will only able to give conditions when we pull to dockerhub)

scratch is the lowest size of image in docker

(https://github.com/docker/awesome-compose)



https://github.com/docker/awesome-compose/blob/master/nginx-nodejs-redis/compose.yaml

awesome-compose / nginx-nodejs-redis / compose.yaml

ndeloof Adopt Compose v2 (#240) 667bd9f · 2 years ago History

27 lines (24 loc) · 370 Bytes

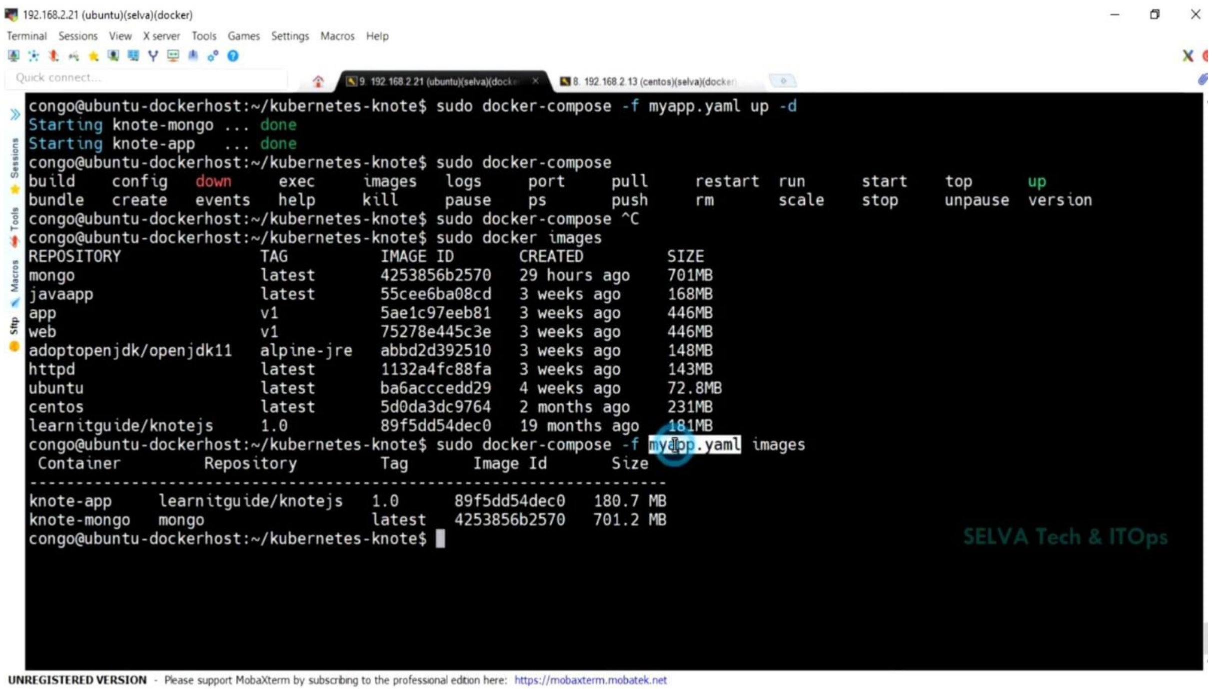
```
services:
  redis:
    image: 'redislabs/redismod'
    ports:
      - '6379:6379'
  web1:
    restart: on-failure
    build: ./web
    hostname: web1
    ports:
      - '81:5000'
  web2:
    restart: on-failure
    build: ./web
    hostname: web2
    ports:
      - '82:5000'
  nginx:
    build: ./nginx
    ports:
    - '80:80'
    depends_on:
    - web1
    - web2
```

Files: .gitignore, Dockerfile, package-lock.json, package.json, server.js, README.md, compose.yaml, nginx-wsgi-flask, official-documentation-samples, pihole-cloudflared-DoH, plex, portainer, postgresql-pgadmin, prometheus-grafana, react-express-mongodb, react-express-mysql, react-java-mysql, react-nginx

All Symbols: '80:80' — 1 Reference — In this file: 22 - '80:80' — Search for this symbol

Find: node — Highlight All, Match Case, Match Diacritics, Whole Words — 1 of 4 matches — Reached end of page, continued from top

In service, when we use redis, it is not to create container it is just for use the redis image but not containerize it (like how we use docker multi stage images)



```
congo@ubuntu-dockerhost:~/kubernetes-knote$ sudo docker-compose -f myapp.yaml up -d
Starting knote-mongo ... done
Starting knote-app   ... done
congo@ubuntu-dockerhost:~/kubernetes-knote$ sudo docker-compose
build    config   down     exec     images   logs     port     pull     restart  run      start    top      up
bundle   create   events   help     kill     pause    ps       push     rm       scale    stop     unpause  version
congo@ubuntu-dockerhost:~/kubernetes-knote$ sudo docker-compose ^C
congo@ubuntu-dockerhost:~/kubernetes-knote$ sudo docker images
REPOSITORY               TAG          IMAGE ID       CREATED          SIZE
mongo                    latest       4253856b2570   29 hours ago     701MB
javaapp                  latest       55cee6ba08cd   3 weeks ago      168MB
app                      v1           5ae1c97eeb81   3 weeks ago      446MB
web                      v1           75278e445c3e   3 weeks ago      446MB
adoptopenjdk/openjdk11   alpine-jre   abbd2d392510   3 weeks ago      148MB
httpd                    latest       1132a4fc88fa   3 weeks ago      143MB
ubuntu                   latest       ba6acccedd29   4 weeks ago      72.8MB
centos                   latest       5d0da3dc9764   2 months ago     231MB
learnitguide/knotejs     1.0          89f5dd54dec0   19 months ago    181MB
congo@ubuntu-dockerhost:~/kubernetes-knote$ sudo docker-compose -f myapp.yaml images
 Container         Repository          Tag       Image Id       Size
---------------------------------------------------------------------
knote-app     learnitguide/knotejs   1.0      89f5dd54dec0   180.7 MB
knote-mongo   mongo                  latest   4253856b2570   701.2 MB
congo@ubuntu-dockerhost:~/kubernetes-knote$
```

How to see docker compose images?

docker-compose images

How to use other name images instead of docker-compose file?

-f myapp.yaml --> myapp.yaml is a file name and -f is used to use other name files

----

Docker bake

Docker bake is just like a docker file but
wroted on HCL formats