```
Join Docker whatsapp group for notes and videos : https://chat.whatsapp.com/HplvBTFASKp13WX7vofeHD

=================
Docker Setup
=================

https://github.com/ashokitschool/DevOps-Documents/blob/main/02-Docker-Setup.md

=========
Docker
=========

=> Docker is a containerization software

=> Docker is used to simplify our application deployment process

=> Docker will take care of required dependencies of our application

=> Using Docker we will run our application as a container

============================
What is Containerization ?
============================

=> The process of packaging our application code + dependencies as single unit and executing as a
container is called as Containerization.

=> Container is an virtual machine (linux vm)

=====================
Docker Architecture
=====================

Dockerfile
Docker Image
Docker Registry
Docker Container



=================
Docker Commands
=================

docker images : To display available docker images

docker pull <image-name> : download docker image

docker run <image-name> : creating docker container

docker ps : display running docker containers

docker ps -a : display running + stopped containers

docker rmi <img-id> : To delete docker image

docker rm <conainer-id> : To delete stopped docker container

docker stop <container-id> : To stop running container

docker start <container-id> : To re-start stopped container

docker logs <container-id> : To see container logs

docker system prune -a :  to delete un-used images + stopped containers

----------------------
```

```
Spring Boot Rest api
----------------------

docker run -d -p 9090:9090 ashokit/spring-boot-rest-api

-d represents detached mode

-p represents port mapping

Note: We need to enable host port in ec2 vm security group inbound rule to allow the traffic.

URL : http://public-ip:host-port/welcome/ashok

=================
Day-01 : Summary
=================

1) What is Docker
2) What is Containerization
3) Advantages with Containerization
4) Docker Architecture
5) Docker Setup
6) Docker Commands
7) Running SpringBoot app using docker image

============
Dockerfile
============

It contains instructions to build image

We will specify application dependencies in Dockerfile


Naming convention : Dockerfile

====================
Dockerfile Keywords
====================

FROM
MAINTAINER
COPY
RUN
CMD
EXPOSE
WORKDIR
ENTRYPOINT


====
FROM
=====

=> It is used to specify base image required for our application.

                FROM : openjdk

                FROM : tomcat8.5

                FROM : mysql8.5

                FROM : python-3.1

                FROM : node-19

============
MAINTAINER
============
=> It is used to specify author of Dockerfile

                MAINTAINER  <Ashok@gmail.com>

======
COPY
======

=> It is used to copy the files from host machine to container machine

        COPY  <SRC>   <DEST>

        COPY  target/app.war   /usr/app/tomat/webapp.war

====
RUN
====

It is used to execute instructions while creating docker image

                RUN  'sudo apt install git'

                RUN 'sudo apt install maven'

                RUN 'git clone <repo>'

Note: We can run write multiple RUN instructions in dockerfile and they will be processed from top
to bottom.

=====
CMD
=====

=> It is used to execute instructions while creating docker container

        CMD 'java -jar <jar-file>'


Note: If we write multiple CMD instructions docker will process only last CMD instruction.

========
EXPOSE
========

It is used to specify container port number

        EXPOSE 8080

=========
WORKDIR
=========

=> It is used to specify working directory

        (path change)


        WORKDIR  /usr/app/
```

```
------------------------Dockerfile-----------------------
FROM ubuntu

MAINTAINER <Ashok>

RUN echo 'run msg - 1'

RUN echo 'run msg - 2'

CMD echo 'cmd msg - 1'

CMD echo 'cmd msg - 2'

---------------- docker build -t <imagename> . ------------


$ docker build -t <image-name> .

$ docker images

$ docker login

$ docker push <image-name>

--------------------------------------------------------------
ashokit/app201:v1

ashokit/app201:v2

ashokit/app201:latest

docker pull ashokit/app201:v1
---------------------------------------------------------------


--------Dockerfile for Java Web App (no springboot)------------

FROM tomcat:8.0.20-jre8

MAINTAINER <Ashok>

EXPOSE 8080

COPY target/app.war /usr/app/local/tomcat/webapps/

------------Dockerfile for springboot app--------------------

FROM openjdk:11

COPY target/sbapp.jar /usr/app/

WORKDIR  /usr/app

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "sbapp.jar"]

-------------------------------------------------------------



=============================
Dockerizing Spring Boot App
=============================

Git Repo: https://github.com/ashokitschool/spring-boot-docker-app.git

1) Install git client in host vm & clone repo
```

```
        $ sudo yum install git
        $ git <repo-url>
```

2) Install maven in host vm.

```
        $ sudo yum install maven
```

3) Go inside project directory & perform maven build

```
        $ cd <dir-name>
        $ mvn clean package
```

4) Build docker image

```
        $ docker build -t ashokit/sbapp .
```

5) Run docker container using docker image

```
        $ docker run -d -p 8080:8080 ashokit/sbapp
```

6) Enable host port in security group inbound rules

7) Access application in browser

http://3.108.219.241:8080/

===============================
Dockerizing Python Application
===============================

https://github.com/ashokitschool/python-flask-docker-app.git

```
$ git clone <repo>

$ cd <dir-name>

$ docker build -t pyapp .

$ docker images

$ docker run -d -p 5000:5000 pyapp
```

================
Docker Compose
================

=> It is used to manage multi container based applications

Docker-Compose Setup : https://github.com/ashokitschool/DevOps-Documents/blob/main/03-Docker-Compose-Setup.md

=> To work with docker compose we need to create docker-compose.yml file

------------------SpringBoot-MySQL-Docker-Compose.yml------------------

```
version: "3"
services:
  application:
    image: spring-boot-mysql-app
    ports:
      - "8080:8080"
    networks:
      - springboot-db-net
    depends_on:
      - mysqldb
    volumes:
      - /data/springboot-app
  mysqldb:
    image: mysql:5.7
    networks:
      - springboot-db-net
    environment:
      - MYSQL_ROOT_PASSWORD=root
      - MYSQL_DATABASE=sbms
    volumes:
      - /data/mysql

networks:
  springboot-db-net:

------------------SpringBoot-MySQL-Docker-Compose.yml------------------
```

```
##Git Hub Repo : https://github.com/ashokitschool/spring-boot-mysql-docker-compose.git

$ git clone <repo-url>

$ sudo apt install maven

$ cd <project-dir>

$ mvn clean package

$ docker build -t spring-boot-mysql-app .

$ docker images

$ docker-compose up -d

$ docker-compose ps

Note: Enable 8080 in security group

=> Access application in browser

        URL : http://public-ip:host-port/

$ docker-compose down
```

```
========================
Docker Workshop Summary
========================
```

1) What is Docker

2) Why Docker

3) What is Containerization

4) Docker Setup in Linux

5) Docker Architecture

6) What is Dockerfile

7) Dockerfile Keywords

8) Working with Docker Images

9) Docker Hub

10) Working with Docker Containers

```
11) Java Web App with Docker

12) Spring Boot app with Docker

13) Python app with Docker

14) Docker Compose

15) Spring Boot + MySQL using Docker Compose.
```