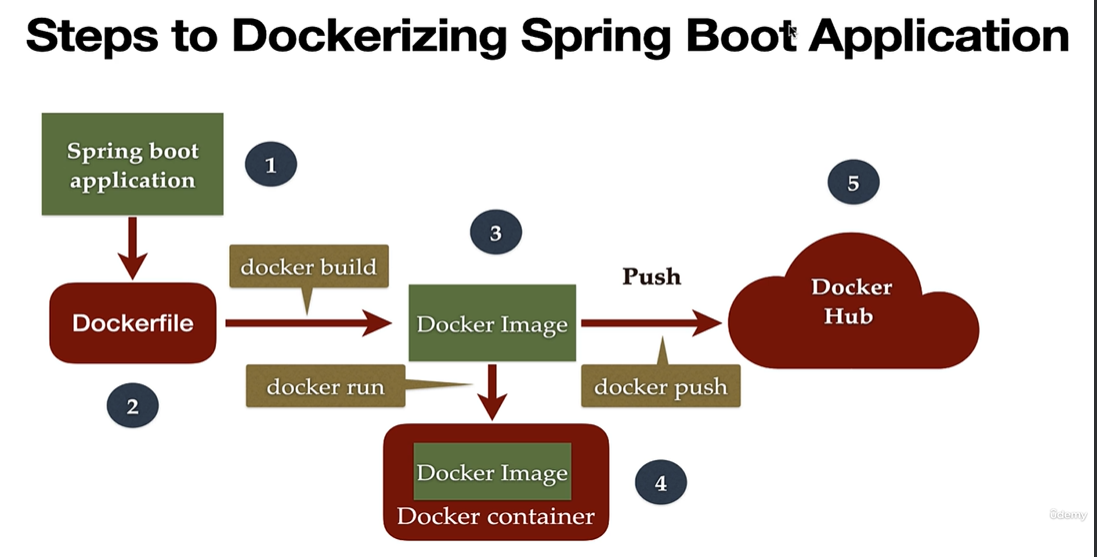
**Docker**

****

-Docker will make a package of your application that includes the dependency this is called a container.

**Docker File: *Text document which contains all the commands that the user can call on the command line to assemble an image.***

***Docker Image: Template to create a docker container.***

***So first we create the docker file that we build to create the docker image and the docker image we will run to get the docker container***

***Docker Container: Running instance of docker image it is a docker container. Which holds the entire pkg to hold the application.***

***Install docker in your system***

***Use wsl –(windows subsystem for Linux)***

***Open cmd and type   
  
To check install proper or not write command on cmd as***

***Docker run hello-world—to download the docker image***

***Docker images ----return you how many images you have***

**Docker command:**

* **Docker -v --to check the version of docker**
* **How to pull image from docker hub(docker pull image\_name)**

**Ex: docker pull openjdk**

**We can use version based on your requirement as well**

**Docker pull openjdk:18**

* **To Search image in docker**

**Docker Search MySQL**

* **To run the images that you have pull by using**

**Docker run image\_name**

* **to Check docker run and create the container for that we use**

**docker ps -a**

* **Run the docker in detached mode and giving name to container**

**We run in detached so that it will run on background**

**Docker run - -name myhellocontainer -d hello-world**

* **TO inspect the container**

**Docker inspect image\_name**

* **To run our application container in interactive mode**

**Docker run --name javacontainer -it -d openjdk**

**to go inside the container we use**

**docker exec -it javaContainer jshell**

* **To see the history of jshell use /history**
* **To exist from any container we can use /exit**
* **How to work upon the my Sql**

**Docker run –name mysqlDB -e MYSQL\_ROOT\_PASSWORD=root -it -d mysql**

**Docker exec -it mysqlDb bash**

**Mysql -p**

**Enter password**

* **To Stop the container we use**

**Docker stop container\_name or Id**

* **To remove the docker history from docker ps -a we use docker id**

**Docker rm id of image**

* **To remove image from machine we use**

**Docker rmi image name**

* **TO check the logs of container we use**

**Docker logs container name**

**How to create docker file and build and run that file image**

* **First you have to create file as dockerfile without any extension**
* **In that folder create your logic for java**

**In the dockerFile we write**

FROM openjdk:8

COPY . .

WORKDIR /

RUN javac calculator.java

CMD [ "java","calculator"]

* **How to build this dockerFile**

**To build we will do   
docker build -t mycalculatorimage .**

* **To run the new mycalculator image by  
  docker run --name calculatorContainer -it -d mycalculatorimage**
* **TO go inside the container for your image  
  docker exec -it calculatorContainer java calculator**

**To remove the image from your local first stop container and then remove from pa -a and then remove image**

**Docker ps**

**Docker stop Container\_name**

**Docker rm container\_Id**

**Docker rmi image\_Name**

* **If you want to delete all the images from your local we can use**

**Docker system prune -a**

**Compose File**

**Create compose.yaml file**

services:

 img:

  build: .

  container\_name: calculatorContainer

* **Docker compose up**
* **I want to push my image to docker hub**

**Docker push username/imagename:withversion**

***Docker file always start with from instruction***

FROM eclipse-temurin:17

LABEL mentainer="shubhamthakur614@gmail.com"

WORKDIR /app

COPY target/springboot-docker-demo-0.0.1-SNAPSHOT.jar /app/springboot-docker-demo.jar

ENTRYPOINT [ "java","-jar","springboot-docker-demo.jar"]

**To run this file to create the docker image go to this file folder and use this command**

docker build -t springboot-docker-demo .  
where -t (tag name) spring boot-docker-demo is the name to image and space dot is within current directory

Docker build -t springboot-docker-demo:0.1.RELEASE .

To Run this docker image will use

Docker run -p 8080:8080 (imageName) springboot-docker-demo

To stop running application use ctrl+c

And to check if it is stopped on not

**docker ps**

-The first 8080 is what port you want and sec 8080 is our application port that we are set in java

-Run the docker in detached mode that is in background and check the log if it is running then

**Docker run -p 8081:8080 -d springboot-docker-demo**

To check if it is running in detached mode or not use

**docker log -f** and give 4 digit id to get result

if you run docker image in detached mode you have to stop that by using

**docker stop 1285**

**How to push your docker image to the docker hub**

**1)docker login in the command prompt**

**2)we have to make our local image with the server so for that   
docker tag springboot-docker-demo shubhamthakur614/springboot-docekr-demo:0.1.RELEASE**

**3)Then again check docker image now you see your image with id of docker hub then push**

**docker push shubhamthakur614/springboot-docekr-demo:0.1.RELEASE  
  
  
To pull any image from docker just use   
Docker pull mysql:latest**

* **To run mysql in docker we have to pull the mysql image from docker hub**

**Docker pull mysql:latest**

* **To run this image in your system use this to run in iteractive mode**

**Docker run –name MysqlDB(container name) -p 3307:3306 -e MYSQL\_ROOT\_PASSWORD=root -d -it mysql**

* **To interact with mysql in container we use**

**Docker exec -it mysqlDB(container name) bash**

**Docker Network to communicate SB with mysql**

* **To communicate our sb and mysql images with each other we create docker network and deployed both image in same network**

**Docker network create springboot-mysql-net**

* **To List out docker network then**

**Docker network ls**

* **To run mysql in network  
  docker run --name mysqlDB --network springboot-mysql-net -p 3307:3306 -e MYSQL\_ROOT\_PASSWORD=root -e MYSQL\_DATABASE=employee -it -d mysql**

**How to connect mysql and spring boot**

* **We have to use a network for that we create two container and put into same network and communicate**
* **First pull the mysql image from docker hub**
* **And run this image with creating network like**

**Docker network create network\_name**

**Then use this network name while running mysql and spring boot application img**

**Docker run –name mysqldb --network network\_name -e MYSQL\_ROOT\_PASSWORD=root -e MYSQL\_DATABASE=employeedb -d mysql**

**Then in Spring boot application in eclipse create docker file for creating docker image and create profile for docker with profile type active while replacing the localhost with container name and db name which is given in mysql image in docker**

**After that build image of spring boot application**

**Docker build -t springboot-restful-webservice**

**Run this image with using network**

**Docker run –name springboot-container –network network\_name -p 8080:8080 -d springboot-restful-webservice**

**Then perform the operation on postman**

* **DOCKER COMPOSE**
* Create first docker-compose.yml file in java from which we can create multiple container with sing le command

Docker-compose up

* version: "3.8"  
    
  services:  
   mysqldb:  
   container\_name: mysqldb  
   image: mysql  
    
   environment:  
   MYSQL\_ROOT\_PASSWORD: root  
   MYSQL\_DATABASE: employeedb  
    
   networks:  
   springboot-mysql-net:  
    
  networks:  
   springboot-mysql-net:

entire Compose file with all configuration added

version: "3.8"  
  
services:  
 mysqldb:  
 container\_name: mysqldb  
 image: mysql  
  
 environment:  
 MYSQL\_ROOT\_PASSWORD: root  
 MYSQL\_DATABASE: employeedb  
  
 networks:  
 springboot-mysql-net:  
  
 springboot-restful-webservices:  
 container\_name: springboot-restful-webservices  
 build:  
 context: ./  
 dockerfile: dockerfile  
 ports:  
 - "8080:8080"  
 depends\_on:  
 - mysqldb  
 networks:  
 springboot-mysql-net:  
 restart: on-failure  
  
  
  
networks:  
 springboot-mysql-net: