Docker commands

A screenshot of a video game

Description automatically generated with medium confidence

A person sitting in front of a screen

Description automatically generated with low confidence

A person standing in front of a screen with text

Description automatically generated with low confidence

Graphical user interface, website

Description automatically generated

Graphical user interface

Description automatically generated

Background pattern

Description automatically generated with low confidence

Create immage in docker/ Commands :

Create the manually docker image

* Docker create hello-world // it will create the container id
* Docker start -a pass the created docker container id
* Docker start -a image id // start the docker image
* docker logs 7c71dc911f7a // check the logs for image

1. Sudo docker run docker/whalesay cowsay Hello-world!
2. Docker run nginx
3. Docker ps - find the list of immages run in the docker
4. Docker ps -a
5. Docker rm - remove the image
6. Docker exec distracted\_mcclinetock cat /etc/hosts
7. $ docker run -d kodekloud/simple-webapp - Running on bag round
8. $ docker run -p 3306:3306 mysql
9. $ docker run -v /opt/datadir:var/lib/mysql mysql
10. java -jar Spring-Docker-0.0.1-SNAPSHOT.jar
11. docker build -t yrrhelpdemo .
12. **docker search centos** : this command is used to find what are all the immages available in docker
13. **docker run centos** : this command is used run the immage
14. **docekr ps :** what are all the immages running it will show
15. **docker immages** : this command will show all the immages
16. **docker rm** : this command will remove images
17. **docker run** -it centos : this command will hellp to go to inside the immage
18. **docker run -d centos sleep 100** : this command will delete after 100 seconds
19. **docker run -p 80:8080 jenkins** : we can assing to the port immages

A screenshot of a computer

Description automatically generated with medium confidence

Docker File :

A picture containing diagram

Description automatically generated

Docker file is the text file writing in specific format it is an instruction or argument, docker will perform specific action and perform to insert and create the image

Instructions : - instructions are nothing but when we give information is nothing but Instruction or directives

Instructions :

From :- Start the new image and set the base image usually must be the first directives

In the docker file except the ARG can be placed before from

ENV :- set environment variable this can be reference in the docker file it self and variable to the container at runtime

Run :- create the new layer on top of the previews layer by running a command inside the new layer and committing the changes

CMD :- specify a default command used to run consider at execution time

Create Docker file :-

1. Create the mkdir in the docker workspace
2. Mkdir custome-nginx
3. Create the html file for viewing
4. Vi index.html
5. Vi Dockerfile

In docker file inside

#Simple webapp

* From ubuntu:bionic
* Run apt-get update && apt-get install -y curl
* Run apt-get update && apt-get install -y nginx
* CMD : [“nginx”, “-g”, “daemon off;”]

**Docker Build**

* **Docker build -t custome-nginx .**
* **Docker run –name custome-nginx -d -p 8080:80 custome-nginx**
* **Custome-nginx$ curl localhost:8080**

**Note : Docker system problic IP address**

**3.86.40.247: our localport**

**Ex : 3.86.40.247:8080**

Create new Docker Image

Create the Docker file :-

#base image

FROM openjdk:8

COPY . /src/java

WORKDIR /src/java

RUN [“javac”,”javaClassName.java”]

ENTRYPOINT [“java”,”javaFileName”]

Note :- build the javafile

Docker build -t my-java-app:v1 .

Check docker images:-

Docker image

**Docker Run Command first three letters**

**Ex:**

**Docker run 8cc**

Graphical user interface, text, application

Description automatically generated

Graphical user interface

Description automatically generated

Docker run - run the docker

Docker ps - running all the images list

Docker rm - remove the image

Docker image pull ubuntu :lastest - ubutu software pull here

Create Spring boot application using Docker

* Create executable jar
* Create docker file
* Build docker file image
* Run docker container

Run clean install in eclipse

Graphical user interface, text, application, email

Description automatically generated

A screenshot of a computer

Description automatically generated

* Clean compile install
* Skip test

Spring boot run on console

* Go to target folder and inside war file path copy
* Got war file path use command
* Java -jar warfile name .jar
* Ex java -r spring.demo.jar