Docker

* docker pull <image name> : <version>

this command downloads image from the library

* docker run <image name>

this will start image in container

* docker ps

This provides list of active containers.

docker ~~run -d~~ ps -a

provides list of running & stopped containers

classmate
Date
Page

docker stop <container IP>

docker start <container IP> (Restart)

docker logs <Container name / ID>

docker run --name <container name> redis

--name allows to give custom ~~na~~ noms to container.

debugging Part.

docker exec -it <container ID> /bin/bash

this will give you access of container as ~~an~~ root user.

exit this will exit the terminal.

docker run -d -p 6000:6379
(Host port) : (container port)

-d detached mode i.e. running in background

-p specifies ports

docker network create (Network Name)

this create docker network

to use network while creating the container

☒ docker run --net (Network Name) (image name).

docker commands VS docker-compose

Starting mongodb

```
docker run -d \
-P 27017:27017 \
-e ROOT_USERNAME=admin \
-e ROOT_PASSWORD=Password \
--net mongo-network \
--name mongodb
mongo
```

Starting mongo-express

```
docker run -d \
-P 8081:8081 \
-e MONGO_SERVER=mongodb \
```

↓ container name in which mongodb is running

```
--net mongo-network \
--name mongo-express \
mongo-express
```

docker-compose file to create same containers

docker-compose file is saved with YAML extension

file mongo.yaml

```
version: '3'                    (version of docker-compose)
services:
  mongodb:                      (container name)
    image: mongo                (image used)
    port:
      - 27017:27017
    environment:
      - MONGO_ROOT_USERNAME=admine
      - -//-
      - -//-

  mongo-express:
    image: mongo-express
    port:
      - 8081:8081
    environment:
      - ME_CONFIG_SERVER=mongodb
```

by default all the containers in docker compose file exist in same network

command to create containers through docker compose file.

```
docker-compose -f mongo.yaml up
```

-f is file tag.

to shut all the containers created through this command we have to use.

```
docker-compose -f mongo.yaml down
```

to build your own docker image you have to create "Dockerfile"

comparing Dockerfile with image env.

Image Env	Dockerfile
install node	FROM node
set UNAME = admin	ENV UNAME = admin

generally it is better to set env in docker-compose file so it is easy to handle

create /home/app	RUN mkdir -p /home/app
	# RUN is used to execute any Linux command
copy current folder files (from host machine) to the container /home/app	COPY . /home/app
start the app with "node server.js"	CMD ["node", "server.js"]
	# CMD is entry

to ~~buid~~ build image using Dockerfile

docker build -t my-app:<version> .

here dot (.) refers to the Dockerfile that exist in current directory.

Docker volumes.

It is used for data persistancy i.e. when we restart the containers all the previous data is lost to avoid this docker volumes are used.

```
docker run -v <name>: /var/lib/data
```

volume definition in docker-compose file.

```
version: '3'

services:
  mongodb:
    image: mongo
    port:
      - 27017 : 27017
    Environment:
      - ROOT_USERNAME = admin
    Volumes:
      - db-data : /var/data.

  mongo-expressj:
    image: mongo-expressj

  . . .

volumes:
    db-data.
```