Docker

1. Check docker version

```
$ docker -v
```

2. check docker containers running

```
$ docker ps
$ docker container ls
```

3 . check all docker container (whether running or not)

```
$ docker container ls -a
//-a → all
```

4. to remove a docker container

```
$ docker rm <container_name>
```

5. to check list of all docker images

```
$ docker images
```

5.B to remove docker image

```
$ docker rm <image-name>
$ docker rmi
```

6.A to create a new docker container and make it run and also execute a command

```
$ docker run image_name -it

$ docker run --name container_name -it image_name

//-it → interactive → when the container is spinned up, stay inside it
//using -it is essential else container will exist whenever tried to start

$ docker run -it ubuntu
$ docker run --name my-node-container -it node
```

6.B To pull image manually

```
$ docker pull <image-name>
$ docker pull mongo
```

7. to start an existing container

```
$ docker start container_name
```

8. to stop an running container

```
$ docker stop container_name
$ docker kill container_name/id
```

9. to execute a command in container
    a. ***container should be up and running***
    b. if command is just to be executed and then not staying inside container

```
$ docker exec container_name command_name
$ docker exec my-ubuntu ls
```

c. if command is to be executed and staying inside the container

```
$ docker exec -it container_name command_name
$ docker exec -it my-ubuntu bash
```

9. to get out of container command, while keeping the container running

```
ctrl + d
```

11. to expose the container to local machine -
    a. **PORT MAPPING**

```
$ docker run --name conatainer-name -it -p <local-port>:<container-port> i

$ docker run --name node-container -d -p 3000:3000 node-express-server

-d (detach mode → for running in background)
```

b. Updating Ports of existing container

```
$ docker container update --publish-add <host_port>:<container_port> cont
```

c. **ENVIRONMENTAL VARIABLES**

```
$ docker run --name conatainer-name -it -e key=value -e key=value image-r
$ docker run --name firebase-container -it -e databaseURL="abc.firebaseda
```

```
$ docker run --name my-postgres -d -p 5432:5432 -e POSTGRES_PASSWORD=
```

12. Building a Custom Images

    let say, to create a image of node-server running on ubuntu

    1. create a node project using

    ```
    $ mkdir my-node-project
    $ cd my-node-project
    $ npm init -y
    ```

    1. write index.js file
    2. create a file `dockerFile` with exact name without any extension in same folder
    3. enter scripts

    ```
    FROM ubuntu

    RUN apt-get update
    RUN apt-get install -y curl
    RUN curl -sL https://deb.nodesource.com/setup_18.x | bash -
    RUN apt-get upgrade -y
    RUN apt-get install -y nodejs

    COPY package.json package.json
    COPY package-lock.json package-lock.json
    COPY main.js index.js

    RUN npm install

    ENTRYPOINT [ "node", "main.js" ]
    ```

    5. build the image locally

```
$ dockert build -t new-image-name path
// t → tag
// path → . for same folder
// path → ./folder-containing-dockerFile
```

```
$ docker build -t my-node-server .
```

13. Pushing a custom Image to https://hub.docker.com
    a. create a image repo with name my-custom-image using my-account on hub.docker.com
    b. copy the link on image repo - “my-account/my-custom-image”
    c. build this remote image locally

```
$ docker build -t my-account/new-image
```

this will create an empty image locally with name “my-account/my-custom-image”

d. to push your local image to this remote image, first login

```
$ docker login
//enter credentials

$ docker push my-account/my-custom-image
```

14. Docker Compose - *Running multiple containers using Script*
    a. create file `Docker-compose.yml`
    b. copy the code below in this file

```
version: "3.8"

services:
  postgres:
    image: postgres # hub.docker.com
    ports:
      - "5432:5432"
    environment:
      POSTGRES_USER: postgres
      POSTGRES_DB: review
      POSTGRES_PASSWORD: password

  redis:
    image: redis
    ports:
      - "6379:6379"
```

c. for Running all services

```
$ docker compose up

$ docker compose up -d
//-d → to run in background so that terminal can be used
```

e. stop and remove all services

```
$ docker compose download
```