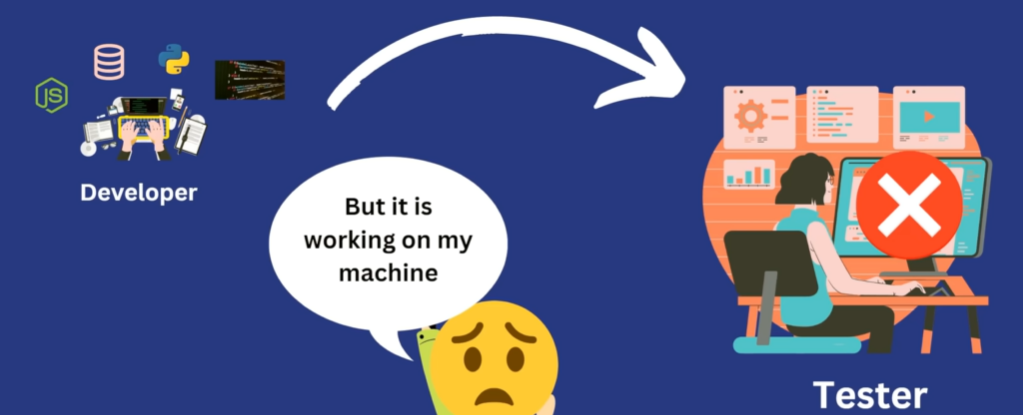
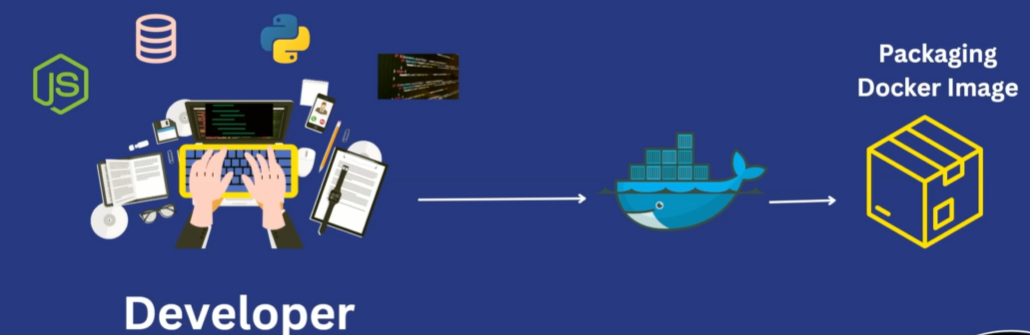
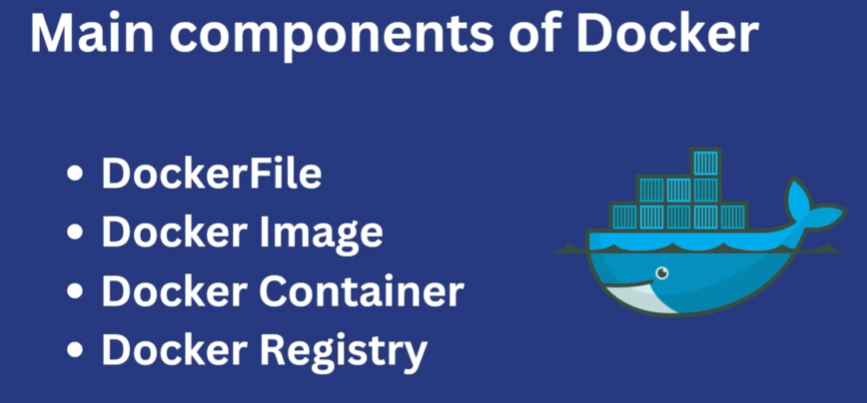
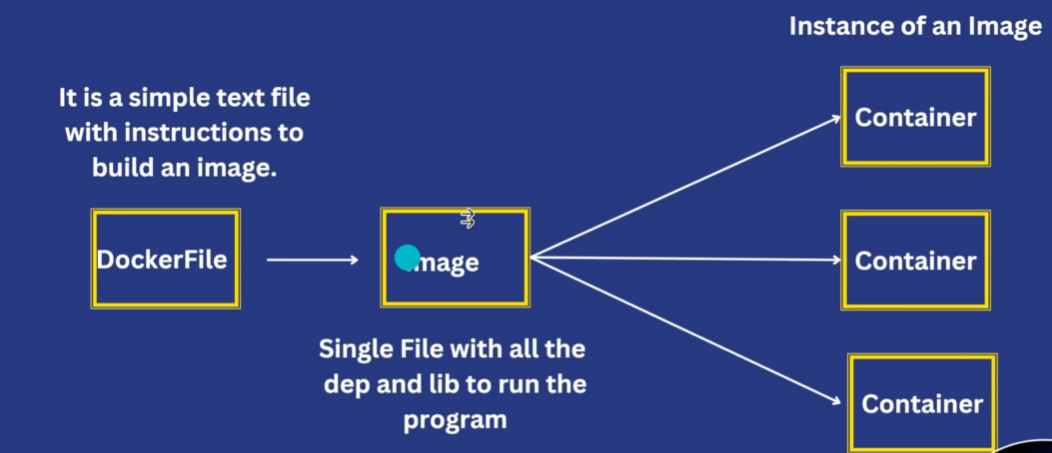
**What is Docker**

****

****



****

****

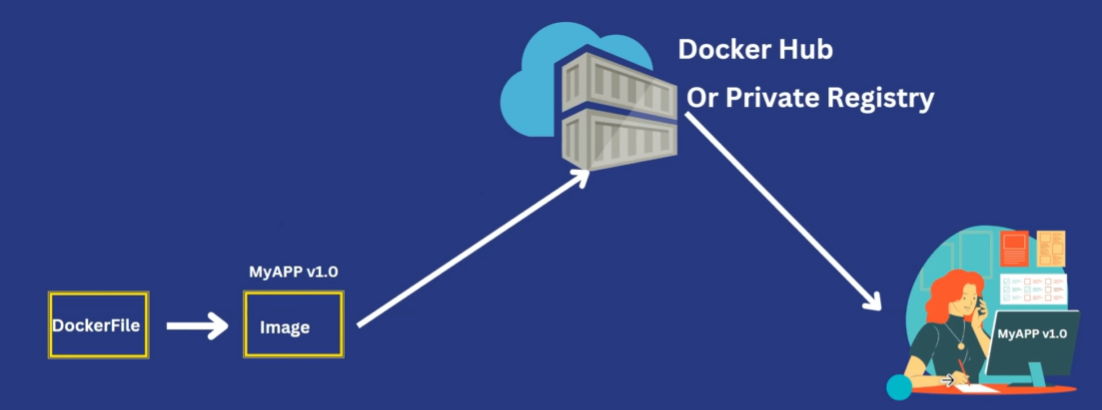
**Dockerfile (Recipe):** Ye ek text file hoti hai jo Docker ko batati hai ki container kaise banana hai. Jaise:

* Kaunsa OS use karna hai?
* Kaunsi files copy karni hain?
* Kaunsi command se server start hoga?

<https://docs.docker.com/reference/dockerfile/>

* Docker can build images automatically by reading the instructions from a Dockerfile.

**Docker Registry :** ek aisi jagah (storage) hai jahan aap apne Docker Images ko save aur manage karte hain. Foe example docker hub

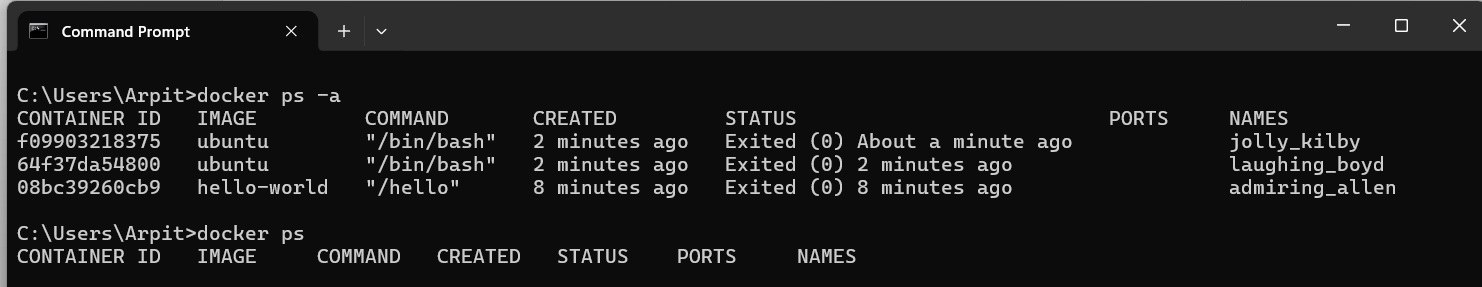
****

* Create a react based application using Node.js
* Download the Node.js
* Npx create-react-app appname
* Docker file – docker images – docker container
* Docker build .
* Docker images ls
* Docker rum image\_id

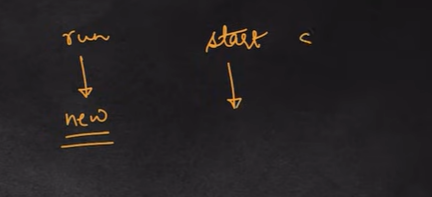
**Docker Commands**



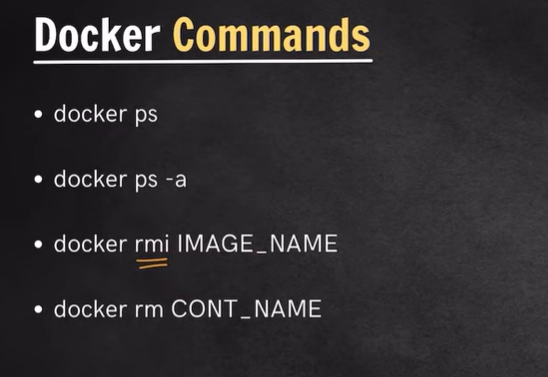
* Docker ps –a (check the all container)
* Docker run image\_name ( create a container)
* Docker ps ( check the all running container)
* Docker images (check the all image)
* Docker pull image\_name (Docker hub se pull karke aap ke local system me layegi)
* Docker run –it image\_name (iska matlab ki hum iamage ke andar a ke kuch kaam kar sakte hain)



* Run commonad -> hamesha ek new container create hota hain or stop and start same chalta hain.



* Docker rmi image\_name –> docker iamge remove or destroy
* Docker rm countainer\_name -> docker container remove

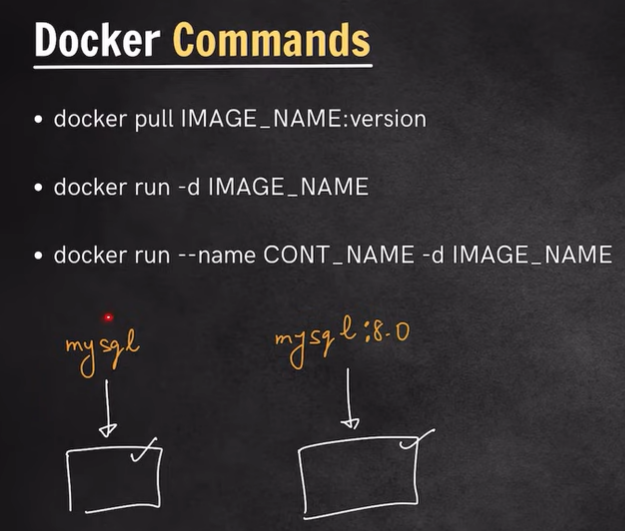


* Kisi bhi images ko delete karne ke liye uska pahle container delete karna padega agar us image ka container bana hua hain to.



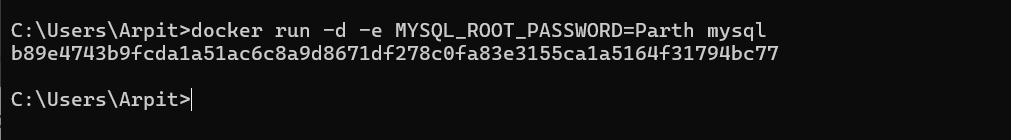
* Docker version -> tag is like a vesion

1. Docker pull mysql (mysql latest version pull in your local system)
2. Docker pull mysql:8.0 ( this is the versioning – while downloading the mysql downlode the layer of images)

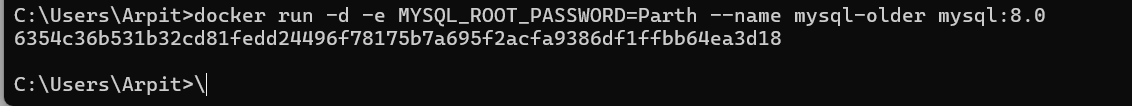


Docker run –d image\_name (-d -> detached Mode (Background me chalta hain -> Iska istemal tab kiya jata hai jab aap aisi cheezein chalate hain jinhe hamesha chalta rehna chahiye, jaise: Database (MySQL, MongoDB) )

* docker run -d -e MYSQL\_ROOT\_PASSWORD=Parth mysql

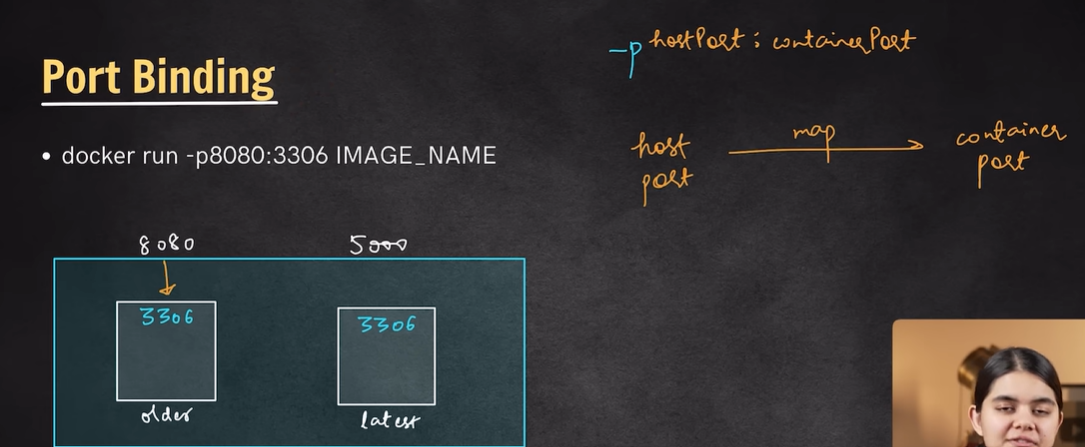


* docker run -d -e MYSQL\_ROOT\_PASSWORD=Parth --name mysql-older-version mysql (Coustom name dena ho container ko to ye use kare)



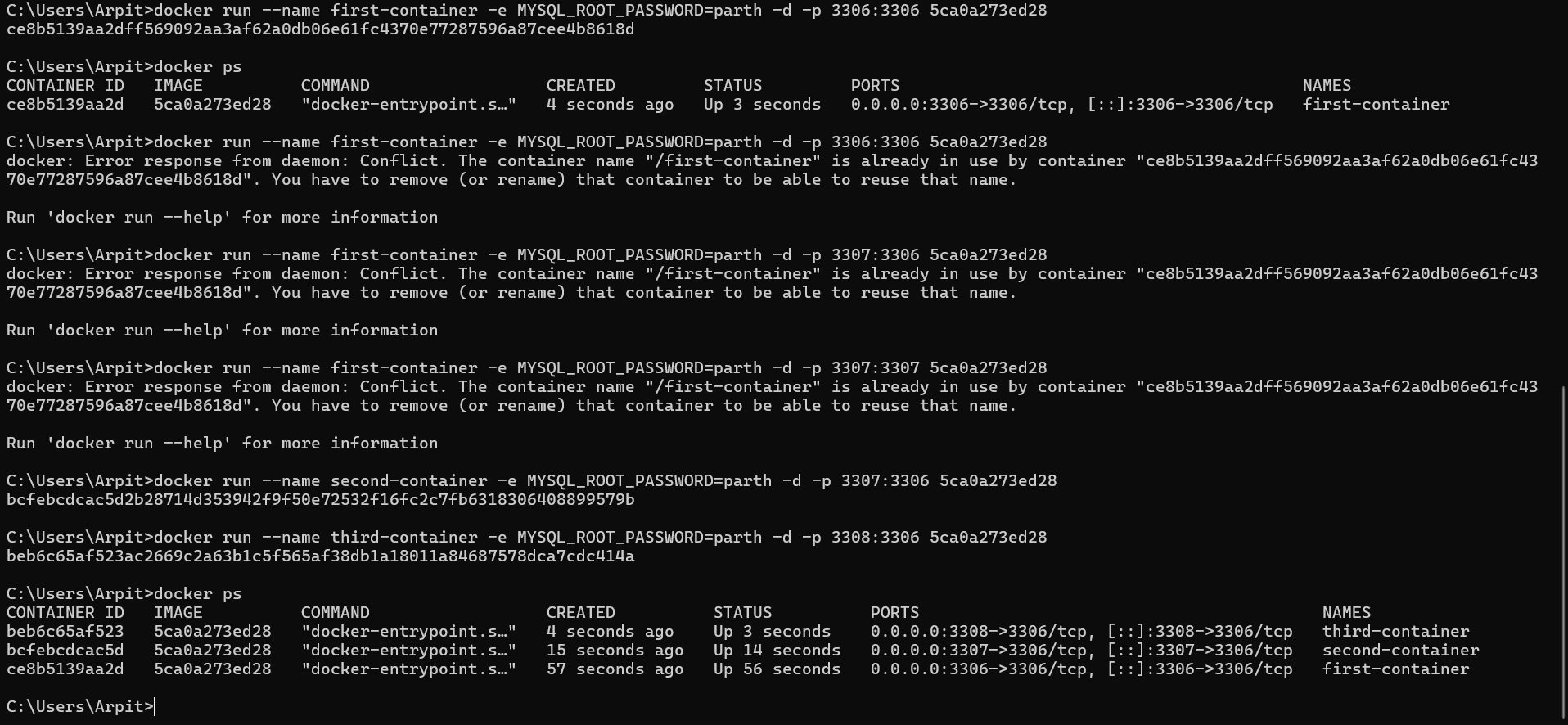
**Port Binding**

**Port Binding** Docker ka ek bahut zaroori feature hai jo aapke computer (Host) aur Docker container ke beech ek raasta (bridge) banata hai

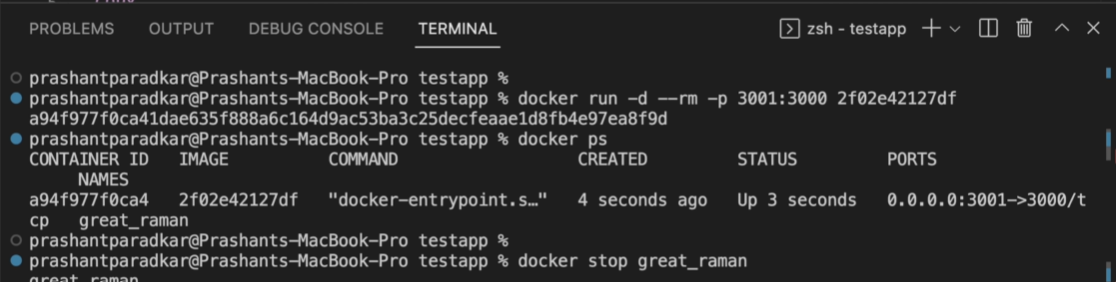
****

* docker run -d -e MYSQL\_ROOT\_PASSWORD=Parth --name mysql-old -p3000(Host Port ):33060(Container port) mysql
* **Ek images se multipule container banana:** agar hum normal ek print hello message ki image use karege to uska container banayege to ek message print karne turant band ho jayega container

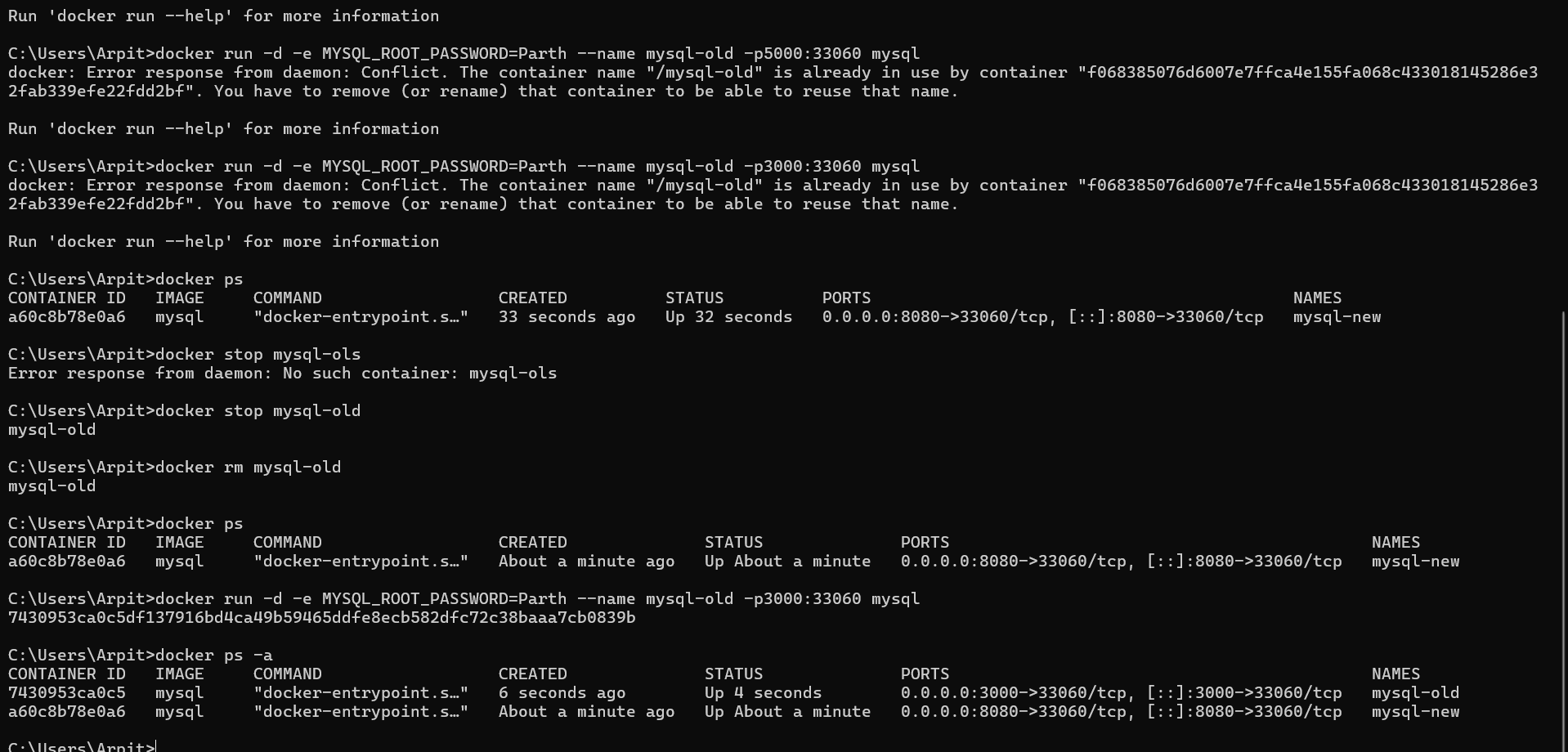
1. Maan lo hum ek data base use kar rahe to aisa nahi hota hain –bus port different hona chahiye

****

* **--rm : agar isko use karoge to kisi bhi container ko stop karoge to wo automatic delete ho jayega**

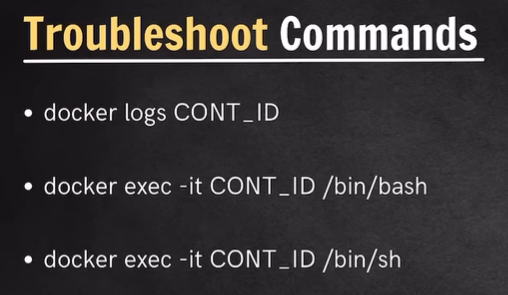
****

* **Agar kisi bhi iamage ko naam dena ho to : docker build –t myappname:v1 .**

****

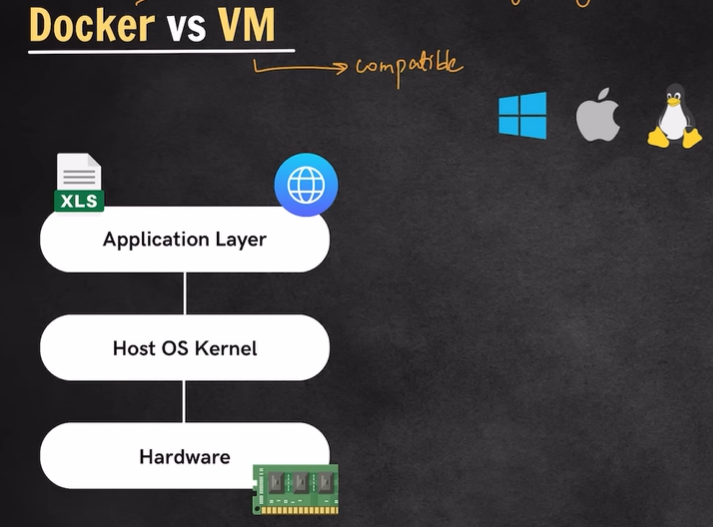
**Troubleshoot Commands**

* docker logs cont\_id (check the logs and also check the logs in desktop mode)
* docker exec –it(intracrtive mode) cont\_id /bin/bash (**kisi chalu (running) container ke andar ghusna** taaki aap usme Linux terminal ki tarah kaam kar sakein)



**Docker Vs Virtual Machine**

* **Virtual Machine (VM)** ek tarah ka "computer ke andar computer" hota hai. Ye ek software-based computer hota hai jo aapke asli physical computer (Hardware) ke resources ka istemal karke chalta hai.
* **Kernel** kisi bhi Operating System (jaise Windows, Linux, ya macOS) ka sabse mahatvapurn aur kendriya (central) hissa hota hai.
* Aap ise computer ka **"Dil" (Heart)** ya **"Manager"** samajh sakte hain. Iska asli kaam software aur computer ke asli hardware (CPU, RAM, Disk) ke beech ek bridge ki tarah kaam karna hai.



* **Virtual Machine (VM)** hardware level par kaam karti hai aur poore computer ko virtualize karti hai, jisme har machine ka apna alag **Kernel** aur **Guest OS** hota hai.
* **Docker** Operating System (OS) level par kaam karta hai aur aapke asli computer (Host) ke **Kernel** ko share karta hai.

**Developing with Docker**

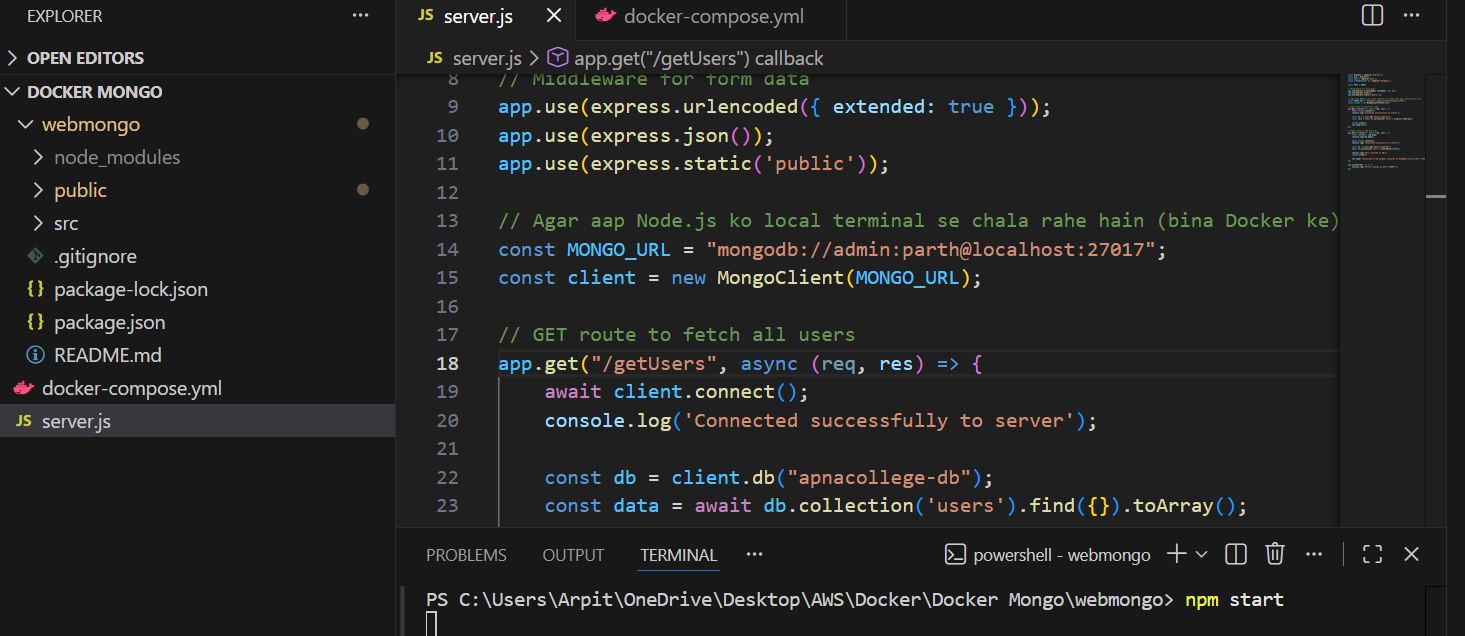
* Ek application node js se taiyar kare sare module ke sath uske baad us application ko mongo db database and mongo express se connect karege ki jab bhi hum login page pe data ki insert kare to wo data dabase me ja ke store ho jaye.
* Mongo Express : Data base ko UI me dekhana or yaha se hum bahut sari cheej kar sakte hain

User name : admin and password : pass

1. Sabse pahle hum node.js me ek login page ki webside ko taiyar kar lenge

* Npx create-react-app appname
* Cd appname
* Npm start

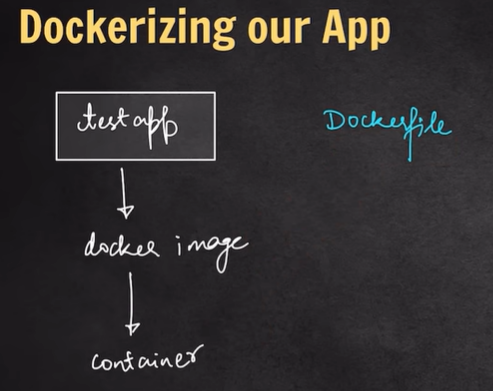
1. Iske baad hum do image ko pull kar lenge mongodb and mongo express ko
2. Iamge ka container bana lange
3. docker run -d -p 8081:8081 --name mongo-express --network mongo-network -e ME\_CONFIG\_MONGODB\_ADMINUSERNAME=admin -e ME\_CONFIG\_MONGODB\_ADMINPASSWORD=qwerty -e ME\_CONFIG\_MONGODB\_URL="mongodb://admin:qwerty@mongo-new:27017" mongo-express
4. docker run -d -p 8081:8081 --name mongo-express --network mongo-network -e ME\_CONFIG\_MONGODB\_ADMINUSERNAME=admin -e ME\_CONFIG\_MONGODB\_ADMINPASSWORD=qwerty -e ME\_CONFIG\_MONGODB\_URL="mongodb://admin:qwerty@mongo:27017" mongo-express
5. <http://localhost:8081/> (ye chalna chahiye or sath me website bhi chalna chahiye)



**Docker Network:** agar hum do container bana rahe hain to wo apas me connect nahi kar pate apne apne port se lakin agar hame dono container ko apas me connect karwana ho to hame ek docker network bana lage or isi network me dono container ko daal denge to jake wo apas me connect kar lange



**Dockerizing our App**

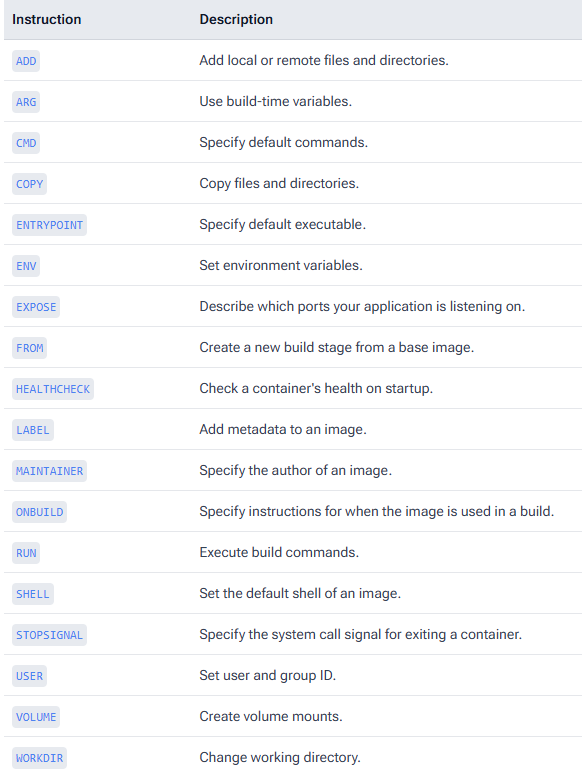


**Dockerfile (Recipe):** Ye ek text file hoti hai jo Docker ko batati hai ki container kaise banana hai. Jaise:

* Kaunsa OS use karna hai?
* Kaunsi files copy karni hain?
* Kaunsi command se server start hoga?

<https://docs.docker.com/reference/dockerfile/>

* Docker can build images automatically by reading the instructions from a Dockerfile.



**Docker Container with Interactive Mode**

Maan lo ek aap ke pass python ka program jo user se intract karta hain

* Create a program
* Create a dockerfile
* FROM python
* WORKDIR /operation
* COPY  ./operation.py .
* CMD [ "python", "operation.py"]

**CMD [ "python", "operation.py" ]**

* **Matlab:** Container start hote hi kaunsi command chalani hai.

### COPY ./operation.py .

* **Matlab:** Aapke computer se file ko container ke andar bhejni.

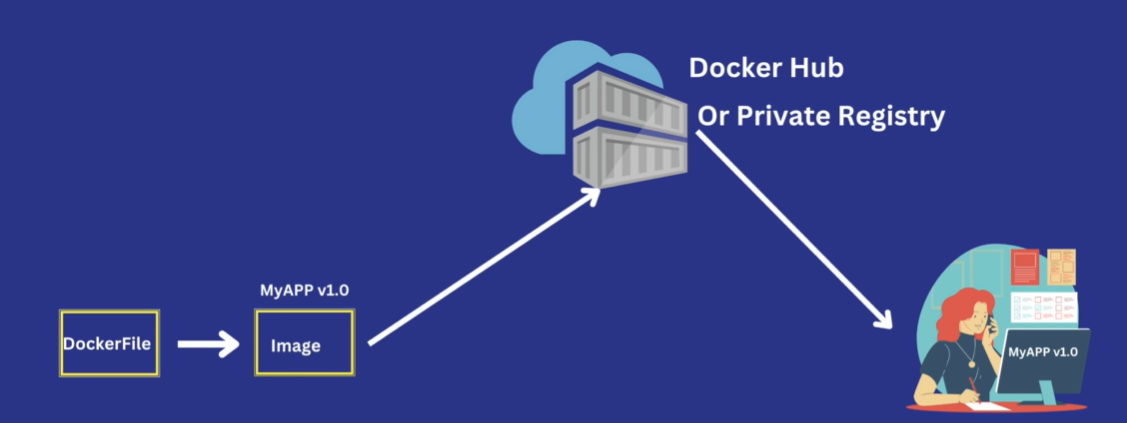
### WORKDIR /operation

* **Matlab:** Container ke andar ek folder banana aur usme enter karna.

### FROM python

* **Matlab:** Ye base image select karta hai.
* Docker build . (docker build -t my-python-app .)
* automatic created one images
* docker run –it image\_id

**Sharing Images in Docker HUB**

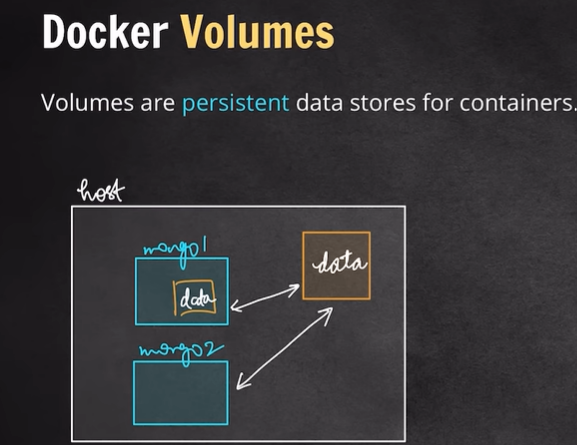


* sign in – Dockerhub
* create a Rapo
* open terminal and login in using docker id or email id for using push and pull permition using docker login
* two way first is username and password and second is mention url - <https://login.docker.com/activate>
* ab mere pass ek image pahle se hain usko push karna docker hub me do step hain-
* Image ko tag karo - apne Docker Hub username ke saath "Tag" karna hoga. Terminal mein ye likhein: docker tag my-python-app arpit31101995/my-python-app:latest (isko rename kar raha hain)
* Ab Push karein : docker push arpit31101995/my-python-app:latest
* Tester downlode the images – pull the image in another machine - docker pull arpit31101995/my-python-app:latest
* Second machine: docker run -it -p 3000:3000 arpit31101995/my-python-app:latest

**Docker Volumes**

Docker Volumes – ye ya to host machine ya to docker khud manage karta hain

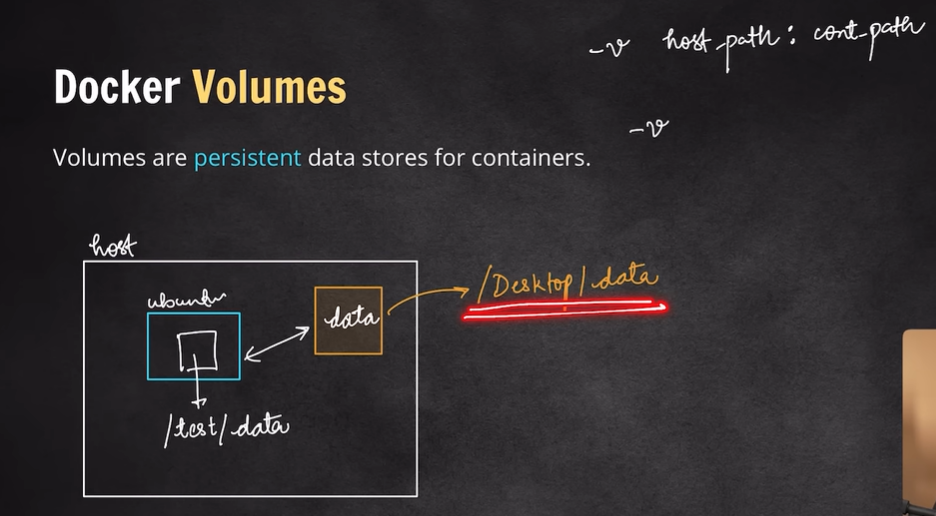
* Docker mein **Volumes** ka istemal data ko "persistent" (pakka) banane ke liye kiya jata hai. Docker containers temporary hote hain—agar container delete ho jaye, toh uske andar ka data bhi khatam ho jata hai. Volumes isi data ko container ke bahar surakshit rakhti hain.



Follow the process : [Host\_Path]:[Container\_Path]

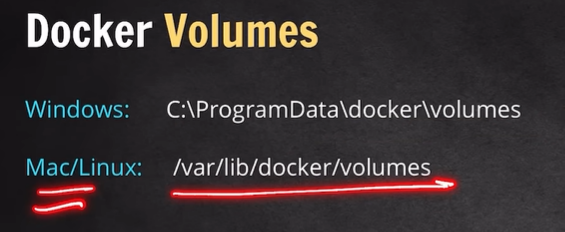
1. docker pull mysql
2. docker run -d --name mysql-persistent -e MYSQL\_ROOT\_PASSWORD=parth -v mysqldata\_name\_vol (Host path apne man se rakh sakte ho):/var/lib/mysql (container path isko hum apne man se nahi sakte hain depend upon image) -p 3306:3306 mysql
3. connect under database: docker exec -it mysql-persistent mysql -u root –p
4. enter the password
5. CREATE DATABASE test\_volume\_db;
6. SHOW DATABASES;
7. Delete the current container : docker rm -f mysql-persistent
8. Create the new container using same volume(same and path) : docker run -d --name mysql-new -e MYSQL\_ROOT\_PASSWORD=parth -v mysqldata\_name\_vol:/var/lib/mysql -p 3306:3306 mysql:latest
9. docker exec -it mysql-new mysql -u root –p
10. SHOW DATABASES;
11. First container me jo database aap ne create kiya tha wo second container me dikhega

* -v mysqldata\_vol:/var/lib/mysql :
* **mysqldata\_vol**: Ye aapke **Host Computer** (aapka PC) par ek storage space hai. Docker ise manage karta hai.
* **:/var/lib/mysql**: Ye **Container ke andar** ka wo rasta (path) hai jahan MySQL apna sara data (tables, databases) store karta hai.

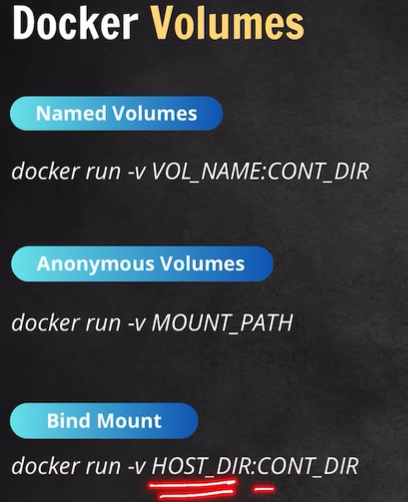




* Jab hum khud ki volume create karte hain to wo alag alag os me alag alag path me ja ke store hota hain –



How to create volume to the different way –jab hum ek apni volume banate hain to usko bhi ek container ke sath mount karna padta hain to kai way to neeche likhe hain



* Un-used volume ko delete karne ke liye : docker volume prune

**Docker Compose**

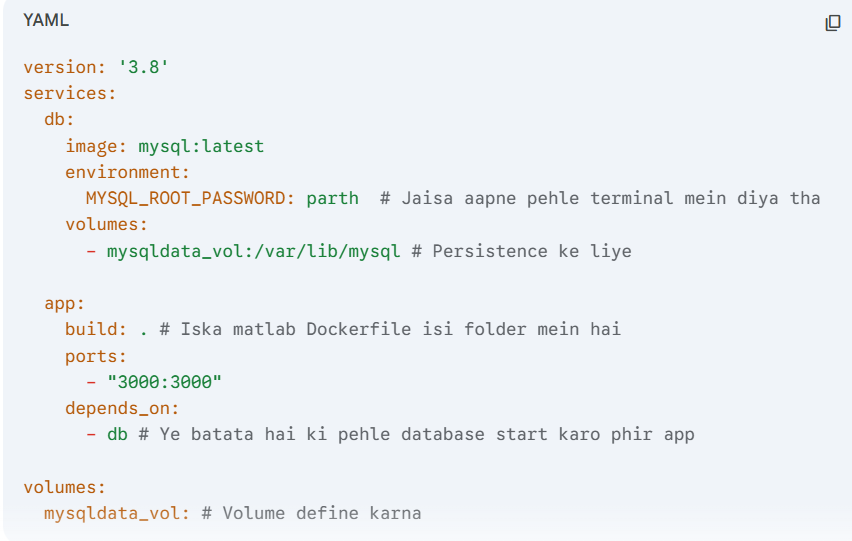
Abhi tak aapne MySQL aur Python ke containers alag-alag commands se chalaye hain. Lekin agar aapka project aisa hai jisme Database, Frontend, aur Backend teeno ek saath chahiye, toh har baar 10 alag commands likhna mushkil hota hai. Docker Compose isi kaam ko ek single file se asaan bana deta hai.

### Docker Compose ke Fayde

* **Ek Command (Single Command):** Sirf docker-compose up likhne se saare containers (DB, App, etc.) ek saath start ho jate hain.
* **YAML File:** Isme hum saari settings (ports, volumes, environment variables) ek .yaml file mein likh dete hain.
* **Network Management:** Ye containers ke beech mein apne aap ek network bana deta hai taaki wo aapas mein baat kar sakein.

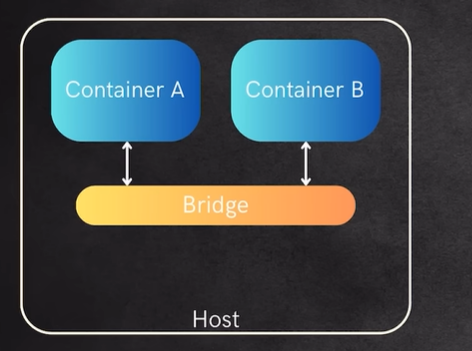
### Ek Simple Docker Compose File ka Example

Maan lijiye aapko apna Python app aur MySQL database ek saath chalana hai. Aap ek file banayenge jiska naam hoga docker-compose.yml:

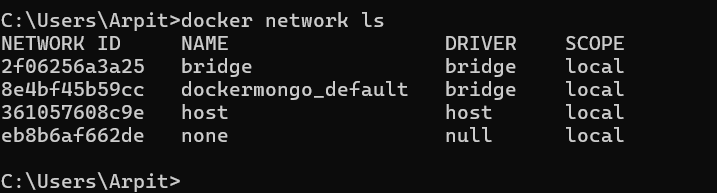


**Docker Network**

Docker mein **Network** ka matlab hota hai ek aisa rasta jiski madad se containers aapas mein ya bahar ki duniya (internet) se baat kar sakte hain.

****

Jab aap container banate hain default me bridge driver connect karta hain jab apas me do container karte hain to bridge ka kaam in dono ko connect karna hota hain



Bridge : Jab aap chahte hain ki aapke containers (jaise MySQL aur Python) aapas mein surakshit tareeke se baat karein

Host : Jab aapko bahut zyada speed chahiye ho aur security ki chinta kam ho.

Null : Jab aapko koi aisa kaam (jaise heavy calculation) karna ho jisme internet ya kisi doosre connection ki zaroorat na ho aur aap use 100% secure rakhna chahte hon.