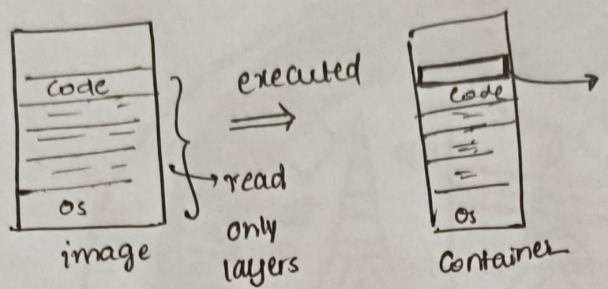
Docker Volumes and Bind mounts.

⇒ Containers are temporary, when we store data in the container and the Container is deleted, the data is lost.

for. Stateful sevices like database we don't want to delete when container gets deleted.

Why docker is stateless?

- docker is suited for lightweight apps.
- Docker image is formed by multiple layers



⇒ The data we store in Containess are stored in writable layer. When we delete the container the writable layer also get deleted.

note: The writable layer isn't deleted after stopping the Container, but after deleting the Container.

The problem with stateless containers:

⇒ It's stateless

⇒ Writable layer is isolated, hard to provide shared access to data for other Containers.

⇒ Backup is also challenge.

Practical:

- create a mysql Container

```
Sudo docker run -d --name mydb -e \
MYSQL_ROOT_PASSWORD=root mysql
```

⇒ sudo docker exec -it mydb bash

↳ get into the container in interactive mode, open bash

```
mysql -u root -p
# enter password
mysql> create database if not exists blog;
mysql > exit
cd /var/lib/mysql
ls -la
  ↳ output: blog.
```

The blog is stored in /var/lib/mysql/blog inside container, we want to persist this folder so, we have store this in our host.

→ Stopping & removing container

```
sudo docker stop mydb
sudo docker rm mydb
```

↳ Container is removed.

Now database is gone.

Solutions: ⇒ Docker Volumes
⇒ Docker Bind Mounts

i) Docker Volumes

- Acts as a data bucket managed by Docker.
- stored in host machine (not in container)
- Can be shared across containers.

```
sudo docker volume ls
        ↳ list of volumes.
```

Create a container and map a volume.

```
sudo docker run -d --rm --name mydb \   → to continue command in next line
    -v mydb-vol:/var/lib/mysql  -e MYSQL_ROOT_PASSWORD=root \
    mysql
```

new container mydb created to see run: sudo docker ps

```
sudo docker exec -it mydb bash
# create a database blog
mysql -u root -p
# enter password
mysql> create database if not exists blog;
mysql> exit # exit mysql
exit # exit bash
```

sudo docker stop mydb

- container is removed since we kept --rm while creating it.

The volume is stored inside /var/lib/docker.

ls ⇒ mydb-vol.

We can use this vol again when we create new container.

ii) Bind Mounts

- host directory is manually mounted into container
- not managed by Docker
- we can specify the path to store.

```
mkdir -p /home/username/docker-data/mysql
sudo docker run -d \        # d → detached mode
  --name mydb \
  -e MYSQL_ROOT_PASSWORD=root \
  -v /home/username/docke-data/mysql:/var/lib/mysql \
```

mysql

Path where you want to store in host machine

Path which you want to persist in container