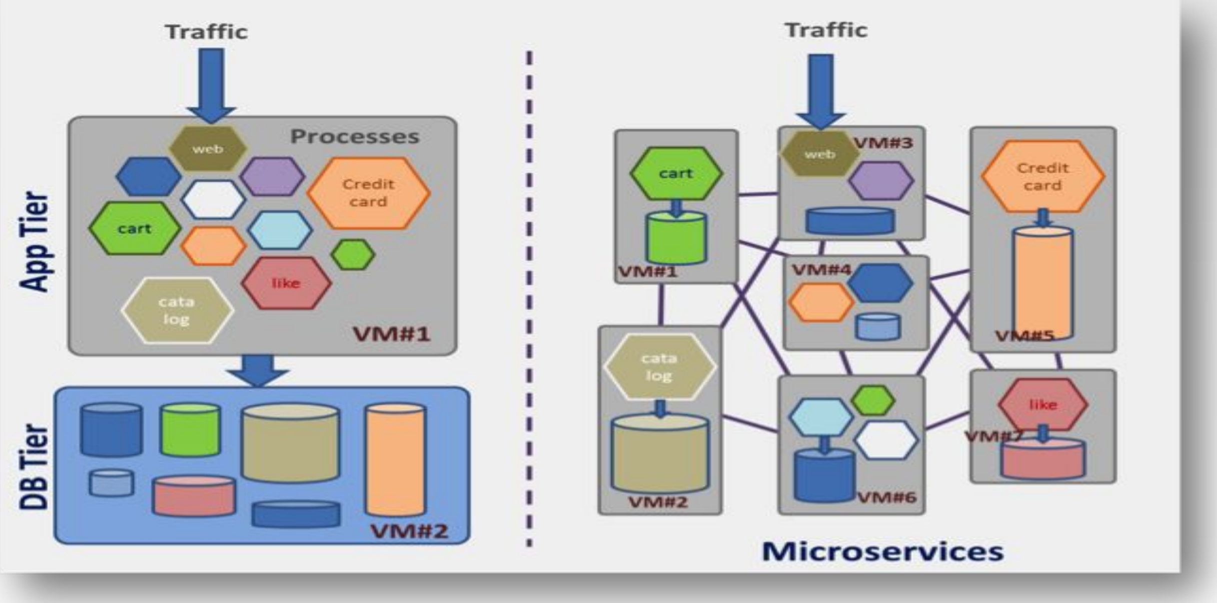
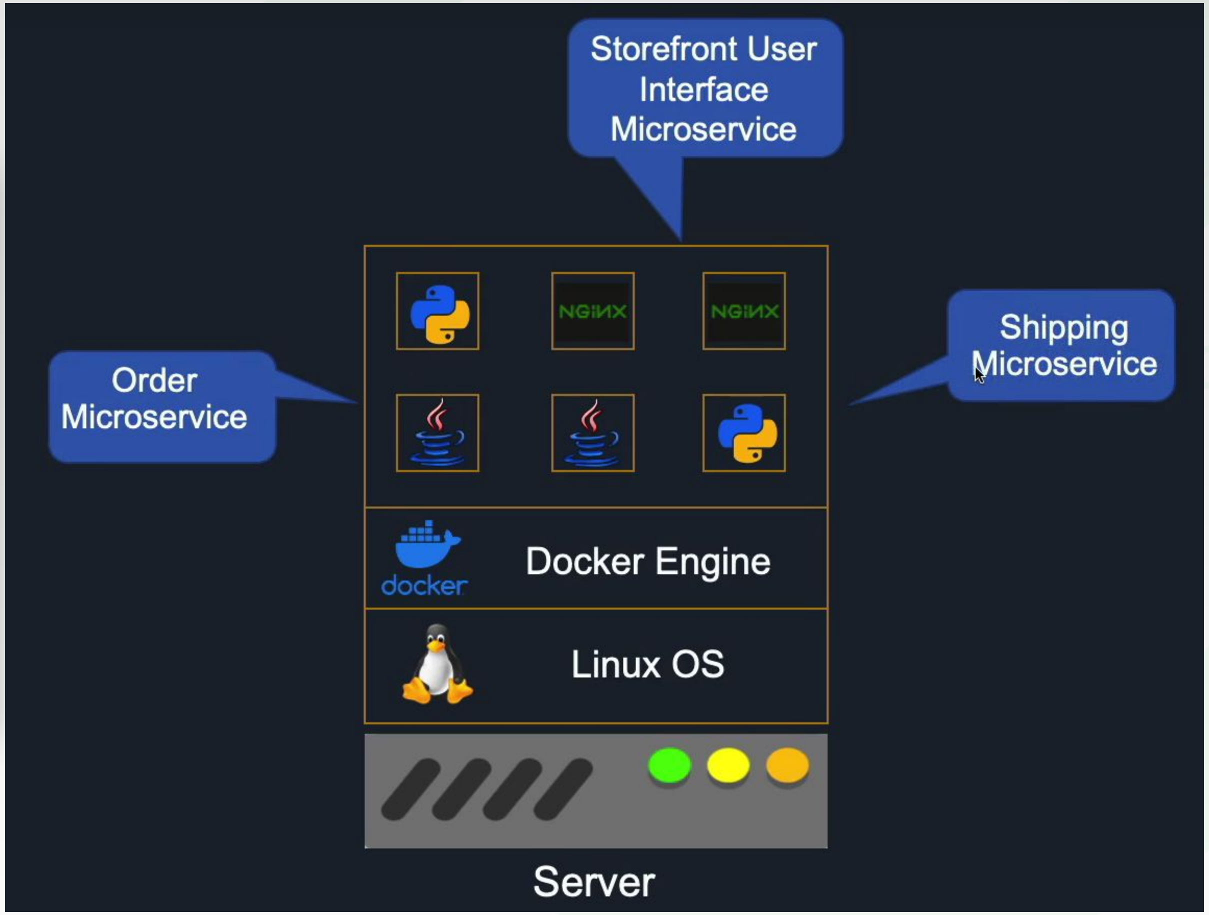
BATCH	:	B107 AWS-DevOps
LESSON	:	**Docker**
DATE	:	11.04.2023
SUBJECT	:	**Volumes**

ZOOM GİRİŞLERİNİZİ LÜTFEN **LMS** SİSTEMİ ÜZERİNDEN YAPINIZ

Docker Intro Review

Monolithic vs Microservice



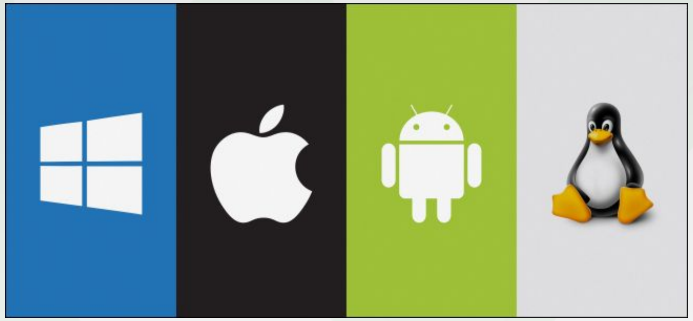


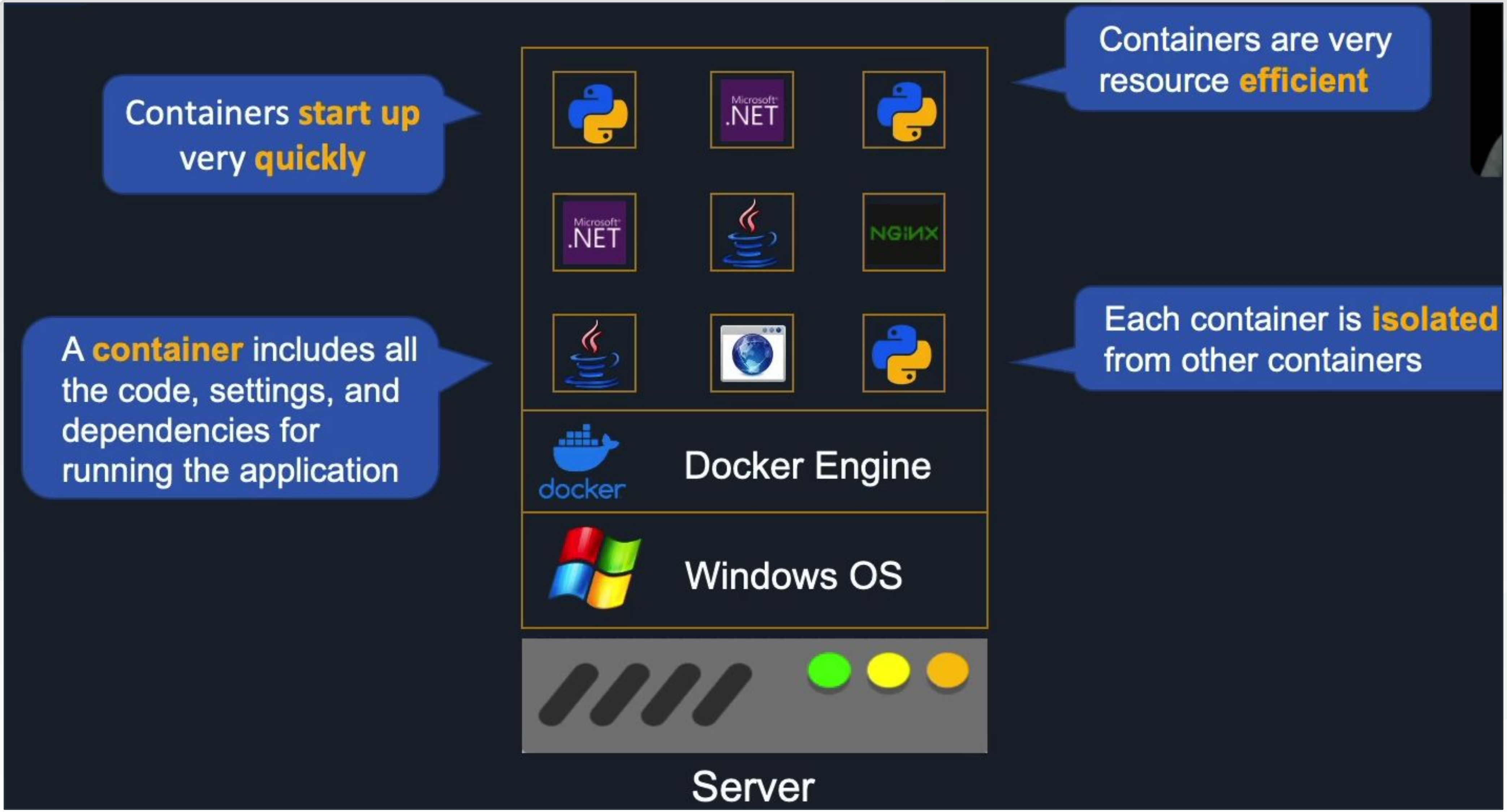
How an application runs

An application needs **an OS, a runtime, application code itself to run.**

It needs an environment.

Container technology provides this environment.





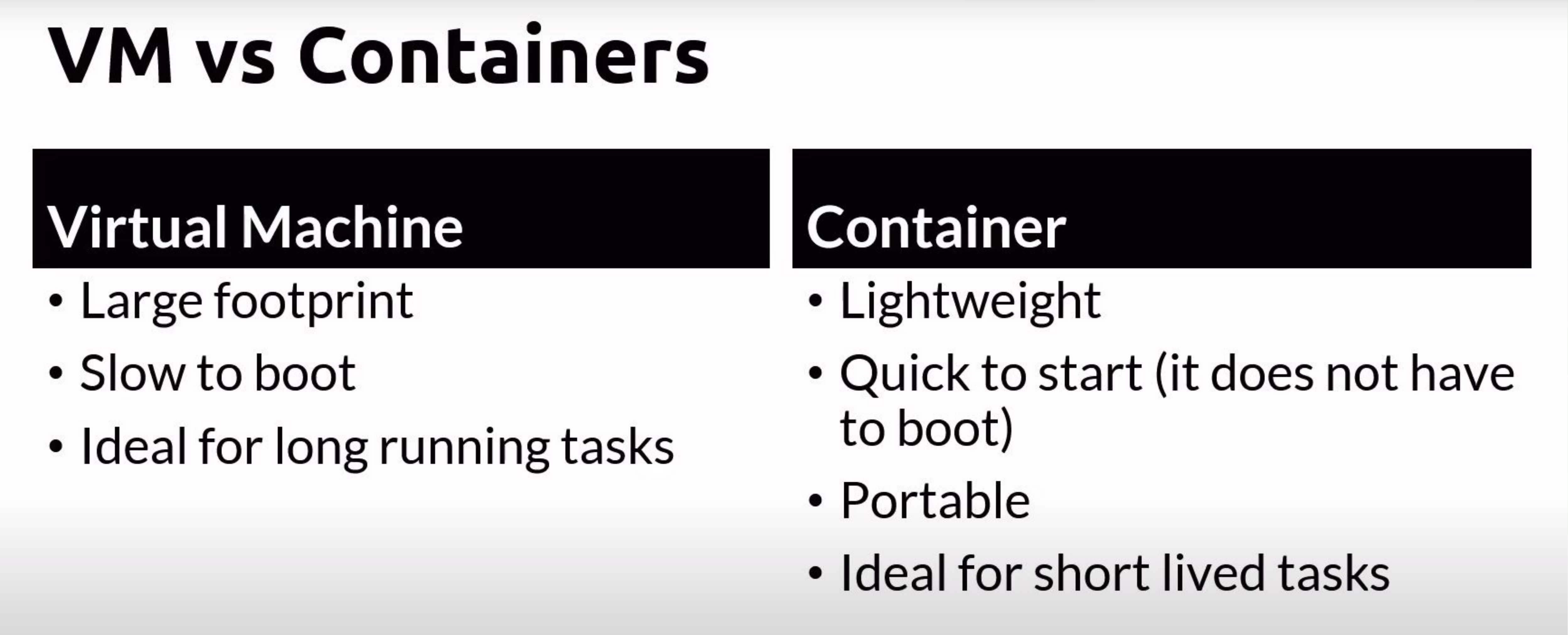
Containers start up very quickly

Containers are very resource efficient

A container includes all the code, settings, and dependencies for running the application

Each container is isolated from other containers

Virtual Machines vs Containers



VM vs Containers

Virtual Machine

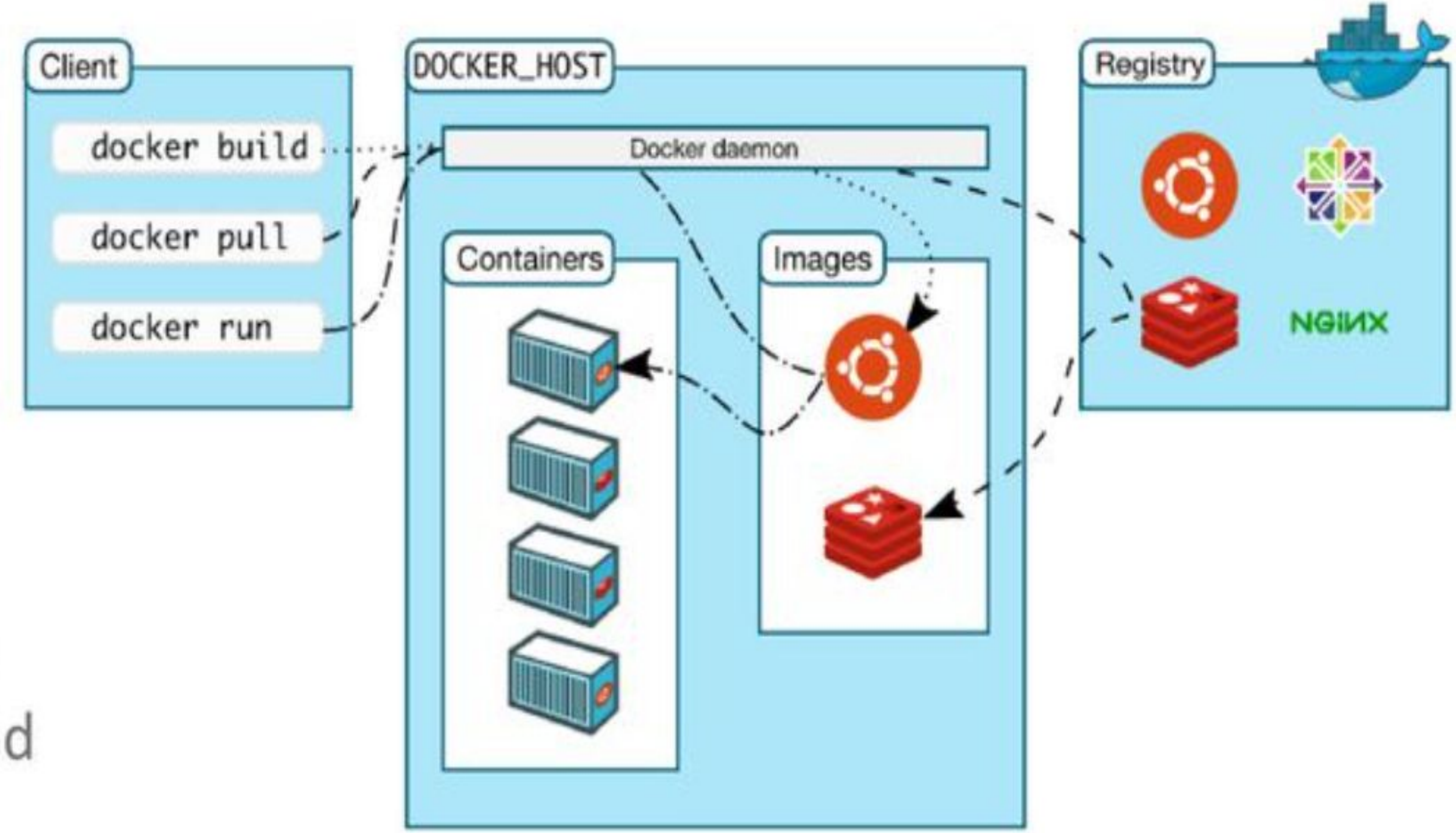
- Large footprint
- Slow to boot
- Ideal for long running tasks

Container

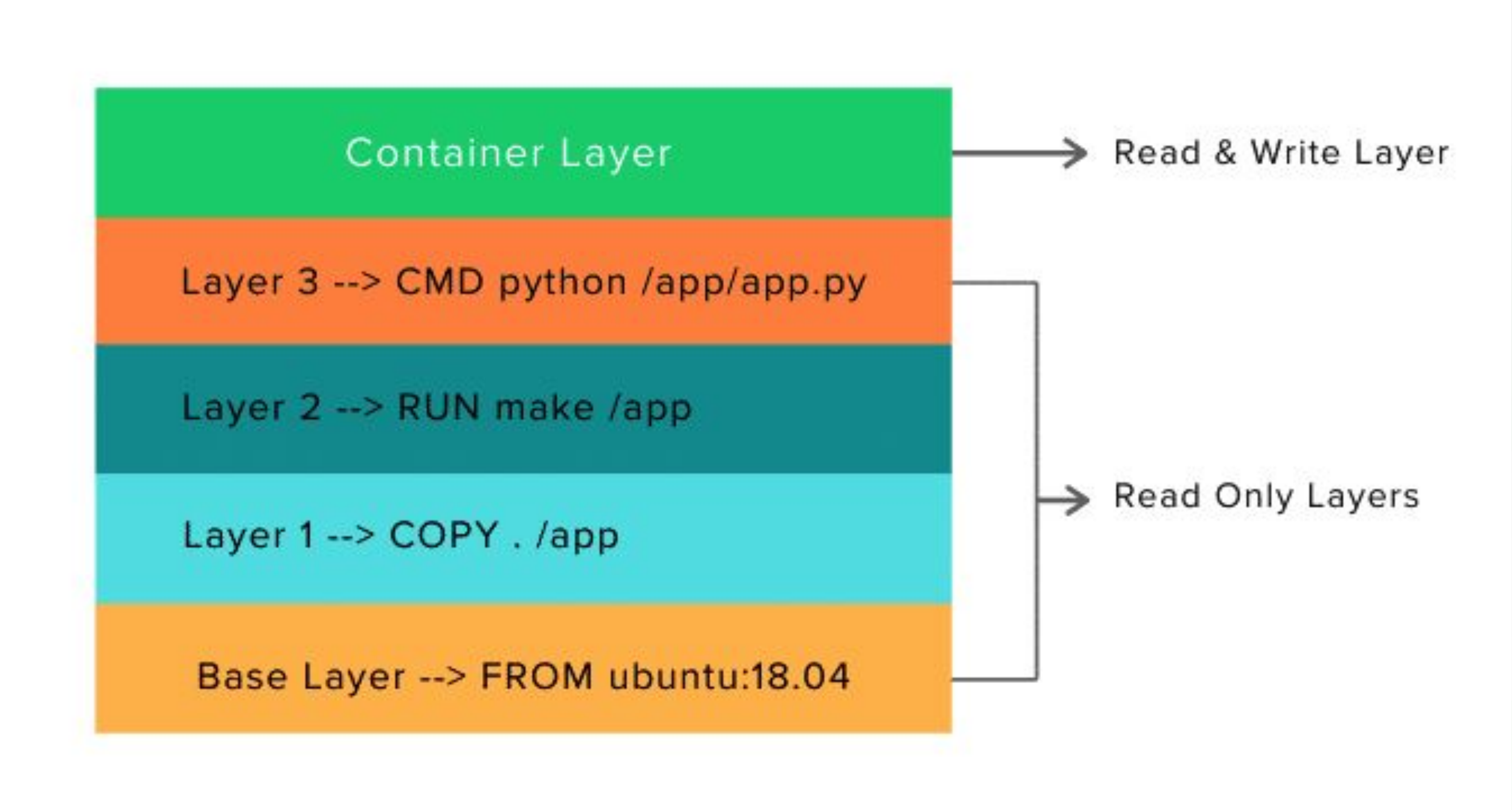
- Lightweight
- Quick to start (it does not have to boot)
- Portable
- Ideal for short lived tasks

Docker Architecture

- Docker client – Command Line Interface (CLI) for interfacing with the Docker
- Dockerfile – Text file of Docker instructions used to assemble a Docker Image
- Image – Hierarchies of files built from a Dockerfile, the file used as input to the docker build command
- Container – Running instance of an Image using the docker run command
- Registry – Image repository



Docker Architecture



Container Layer

Layer 3 --> CMD python /app/app.py

Layer 2 --> RUN make /app

Layer 1 --> COPY . /app

Base Layer --> FROM ubuntu:18.04

Read & Write Layer

Read Only Layers

Tips

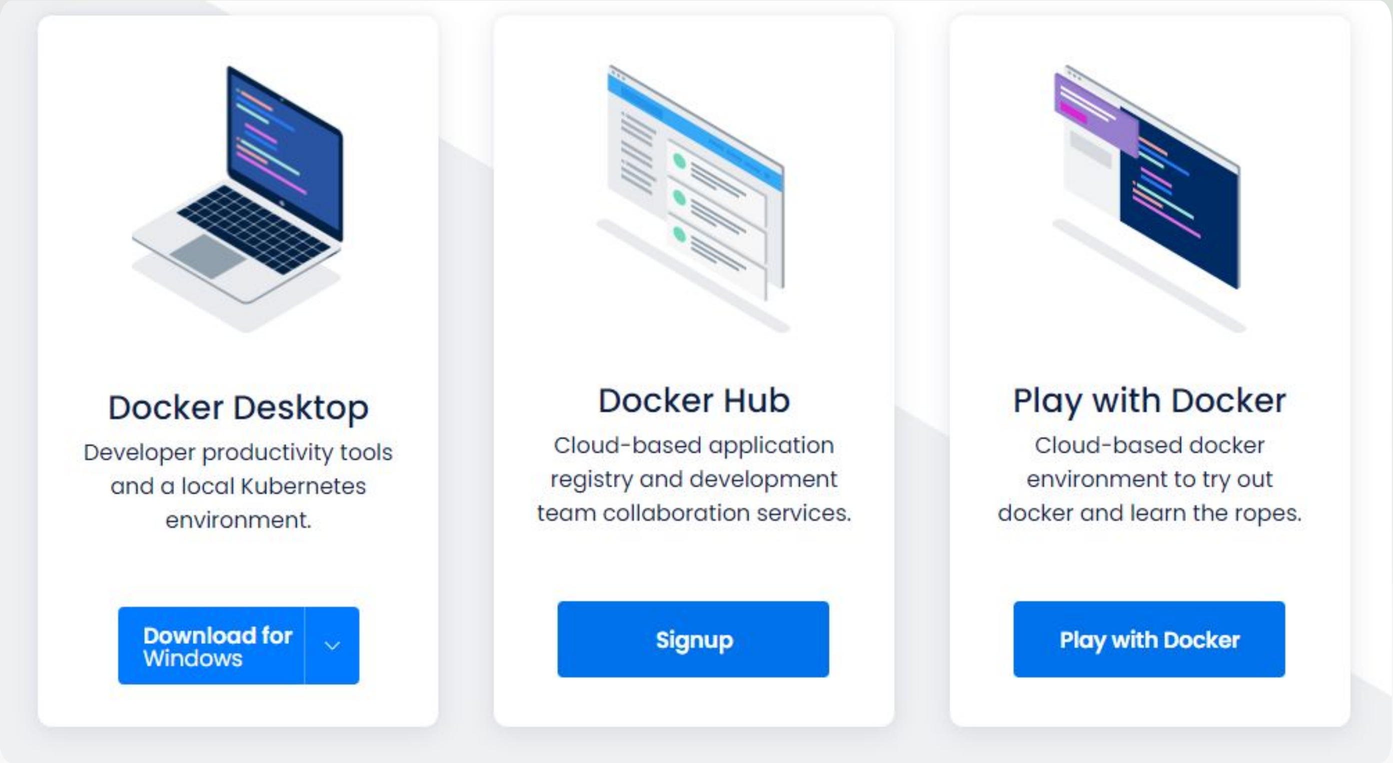
- Containers are stateless, they do not store your data inside.
- Each container gets an IP address at creation.

- namespaces: running isolated processes
- cgroup: assign resources to namespaces
- container: running processes with dedicated resources

- Docker runs on Linux, on platforms like MacOS, Windows, it uses a tiny Linux environment
- Containers are used for a single application. They are the basic of microservices.

- Docker is made up of
  - a CLI
  - a background daemon (service)
  - REST API

Docker Environment



Docker Desktop

Developer productivity tools and a local Kubernetes environment.

Docker Hub

Cloud-based application registry and development team collaboration services.

Play with Docker

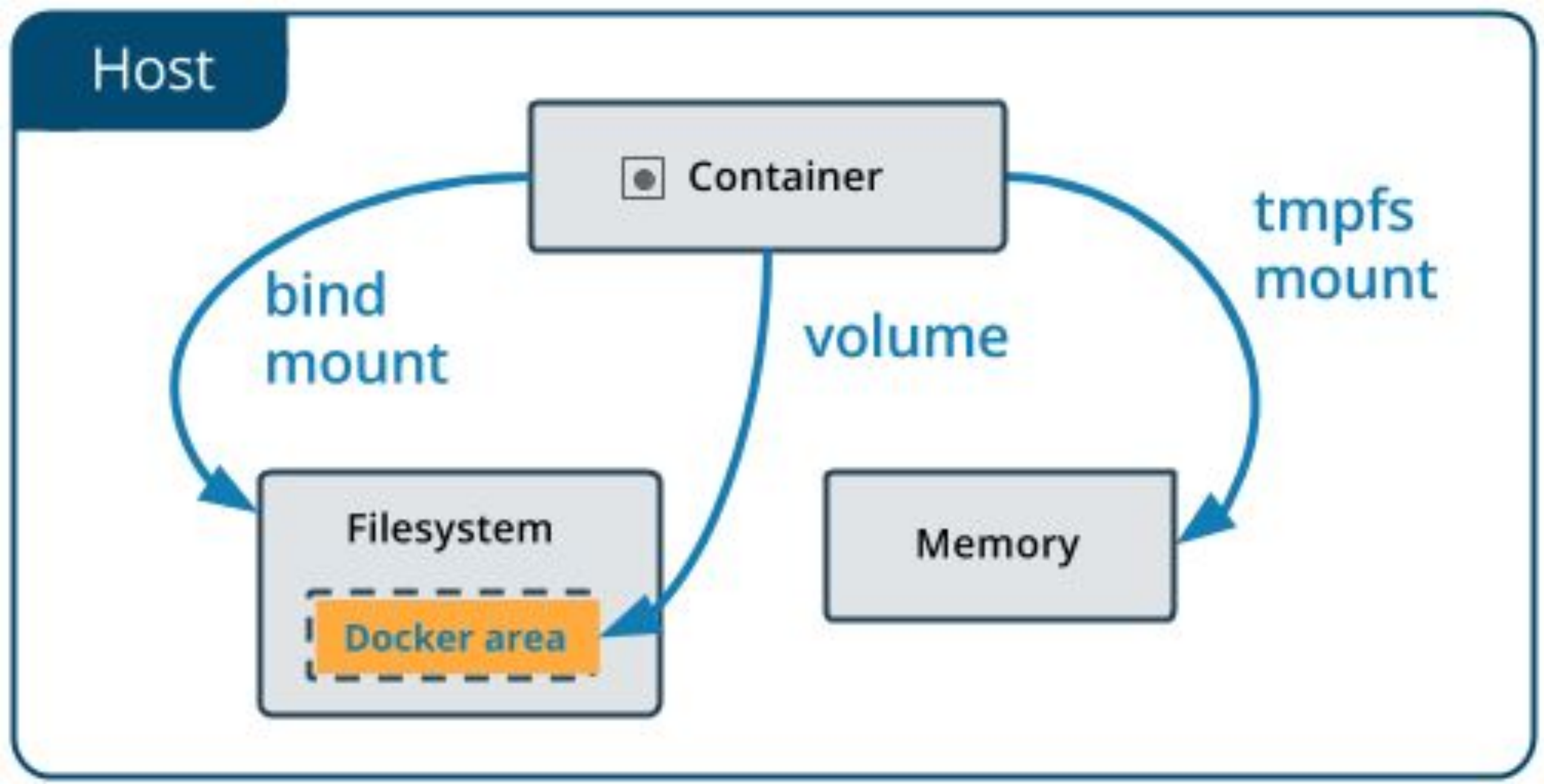
Cloud-based docker environment to try out docker and learn the ropes.

- Docker-Desktop
- Docker-Hub
- Play with Docker

Docker Volumes

Docker Storage

Containers are stateless, they do not store your data inside.



Docker Storage

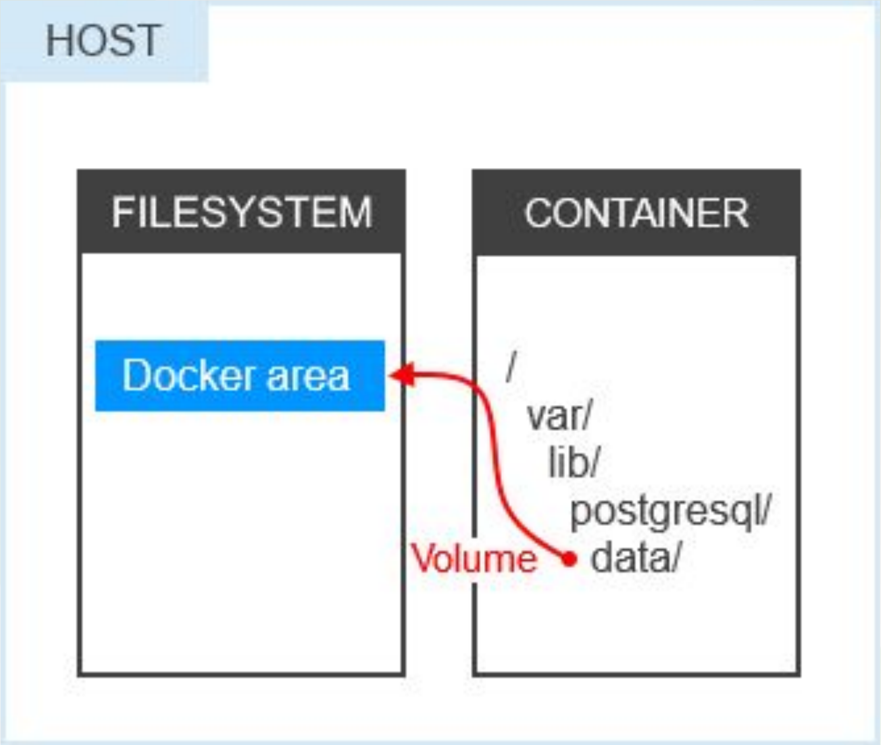
- Bind Mount
- Used at Development stage
- May lead to sensitive local data/system data
- Risky
- You manage

- Volumes
- Docker recommends for Production stage
- Used for data sharing between containers
- Easy to backup
- Docker manages

- tmpfs
- Used when data is not needed to be stored physically
- Uses RAM

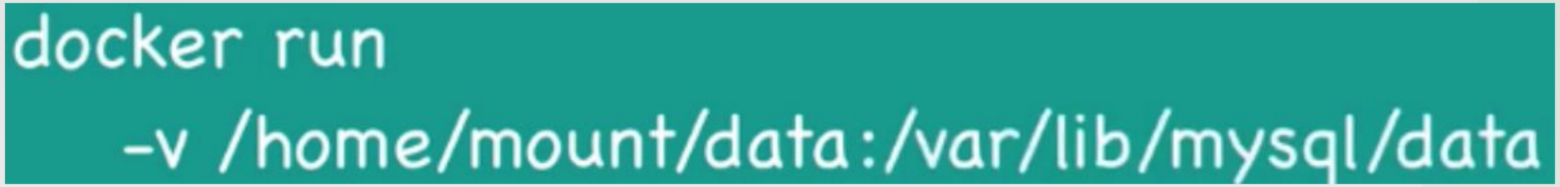
Docker Volumes

- Docker volumes are used to persist data from within a Docker container.
- There are a few different types of Docker volumes:
  - host
  - anonymous
  - named



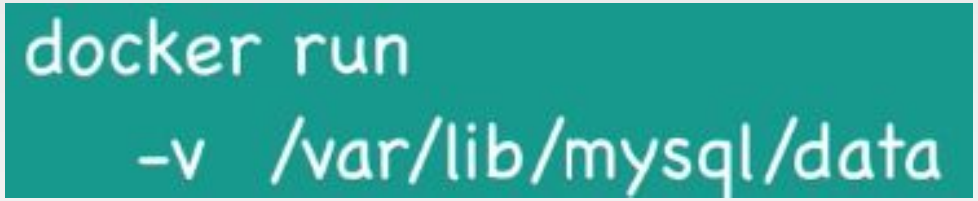
Docker Volume Mapping

host



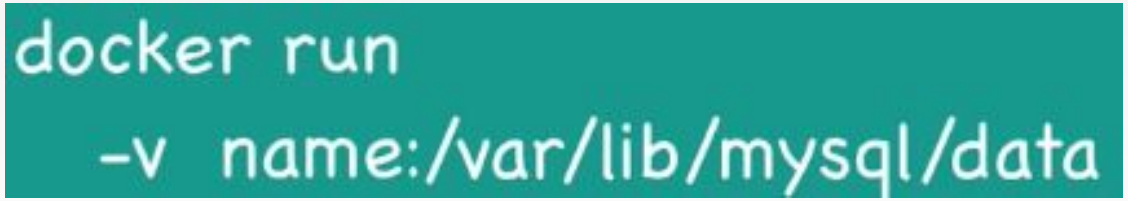
```
docker run
    -v /home/mount/data:/var/lib/mysql/data
```

anonymous

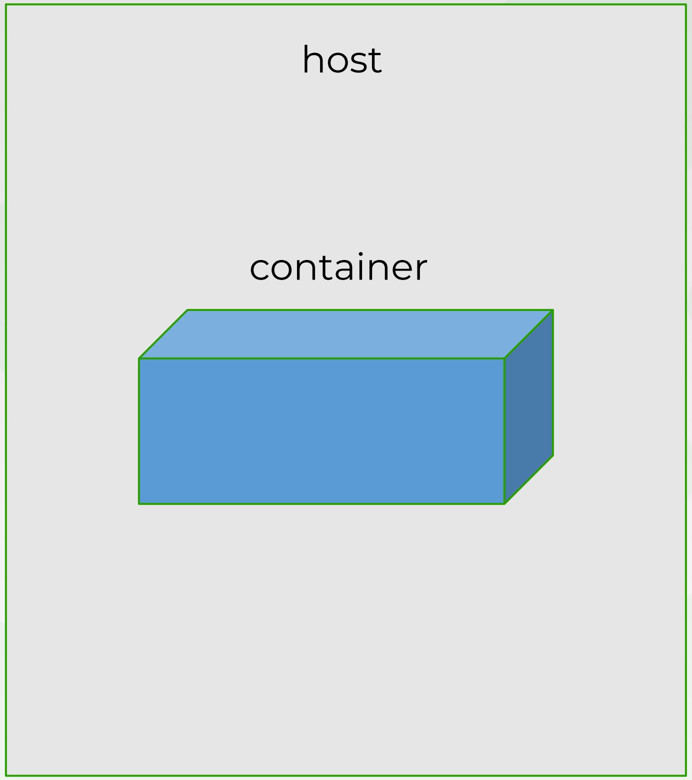


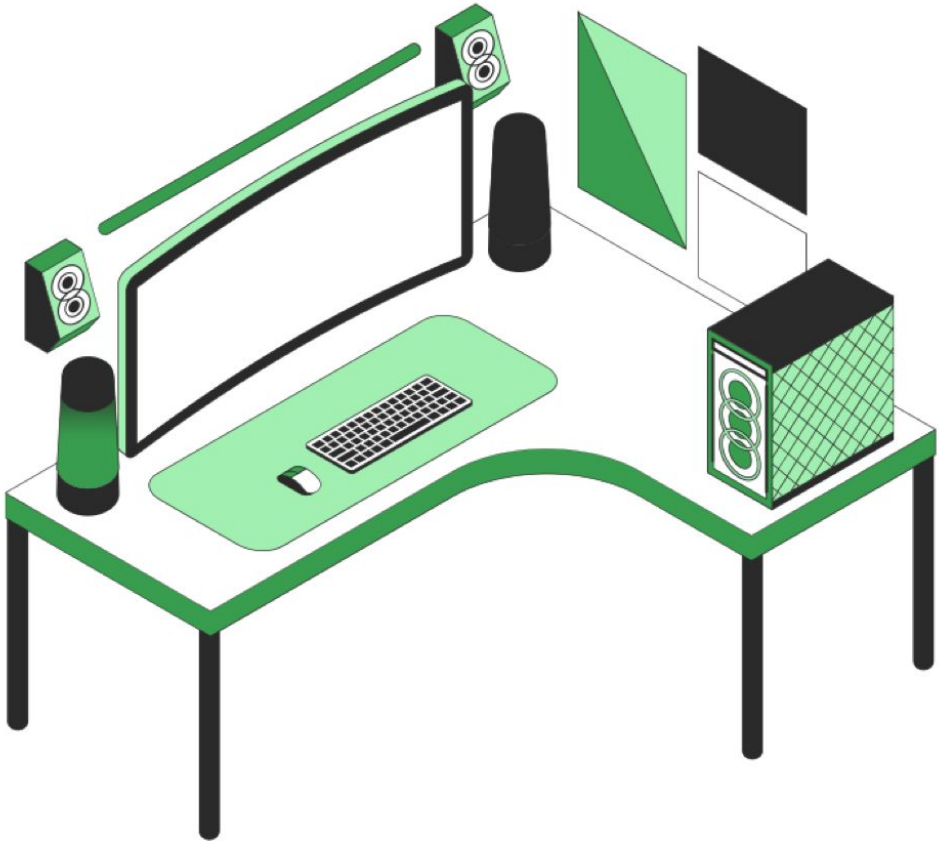
```
docker run
    -v  /var/lib/mysql/data
```

named



```
docker run
   -v  name:/var/lib/mysql/data
```





Do you have any questions?

Send it to us! We hope you learned something new.