Docker Official Images

- A dedicated **team responsible for reviewing and publishing all content** in the Docker Official Images repositories

- Works in **collaboration with software maintainers, security experts**

- However anyone can participate as collaboration takes place openly on GitHub

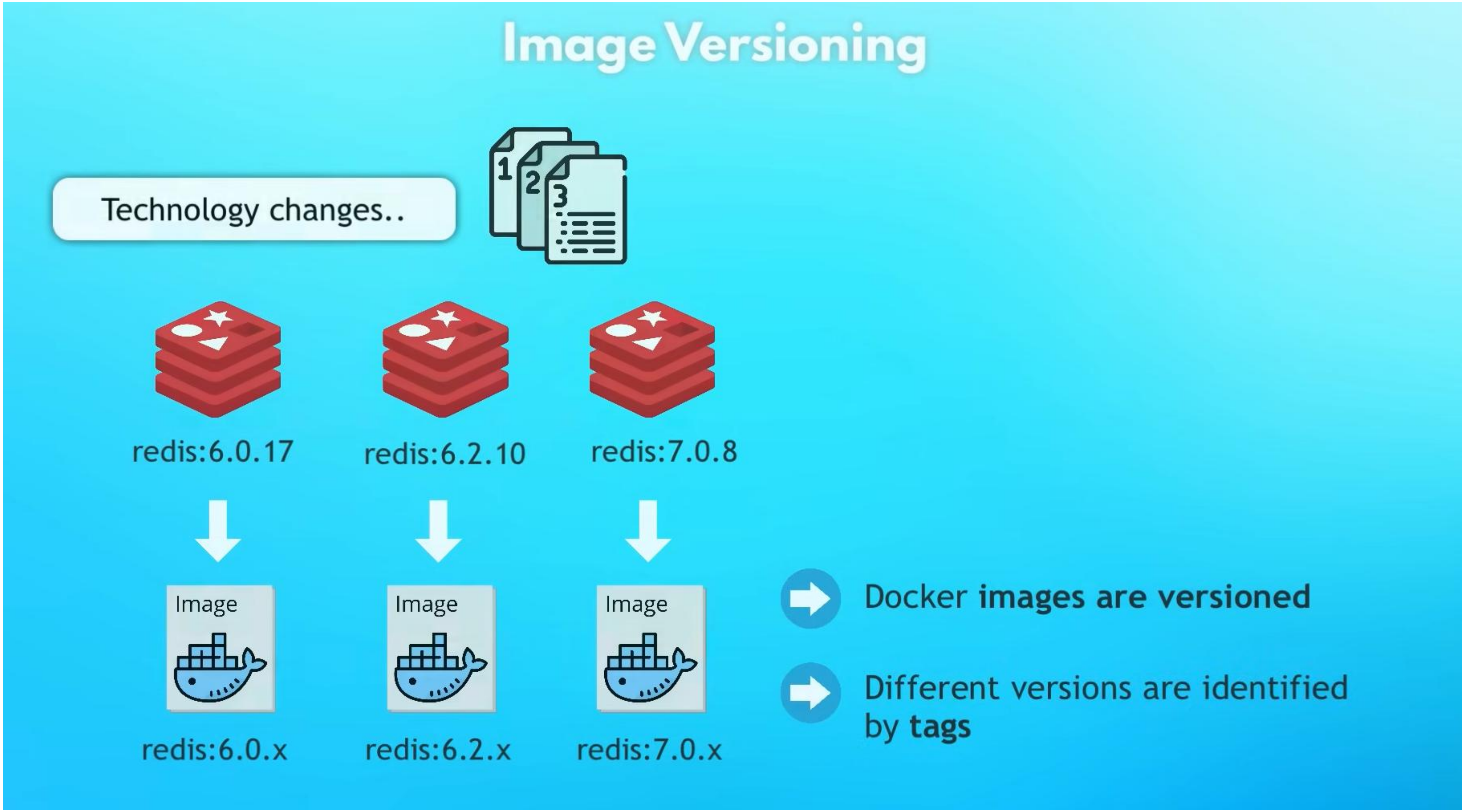


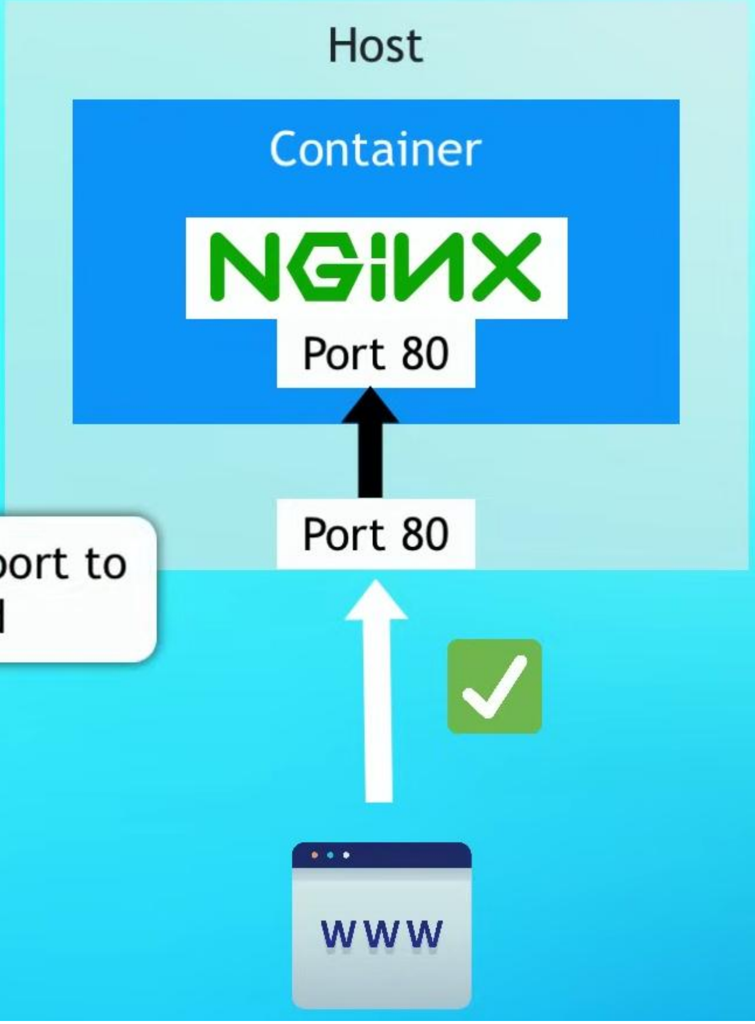
Image Versioning

Technology changes..

- Docker **images are versioned**
- Different versions are identified by **tags**

Container Port vs Host Port

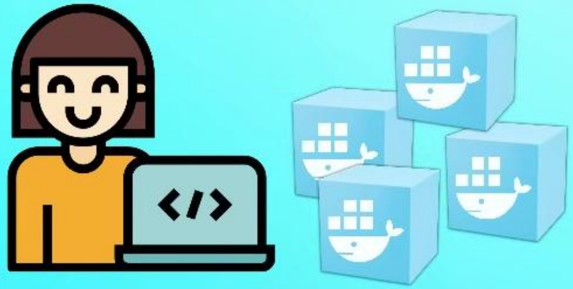
- Application inside container runs in an **isolated Docker network**
- We need to **expose** the container port **to the host** (the machine the container runs on)



Port Binding: Bind the container's port to the host's port to make the service available to the outside world

docker run

- Creates a new container
- Doesn't re-use previous container

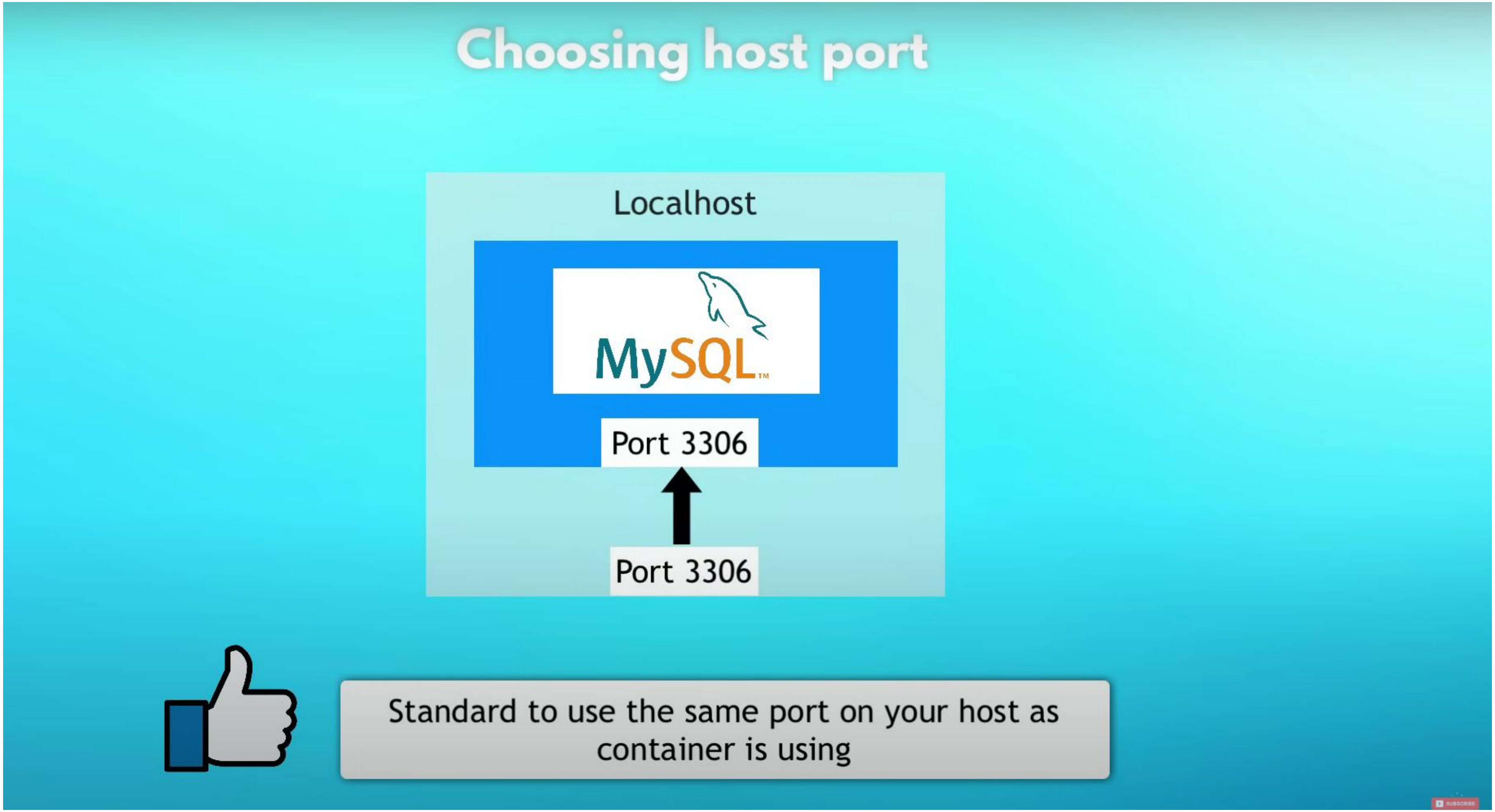


▶ *docker run …*

▶ *docker run …*

▶ *docker run …*

▶ *docker run …*



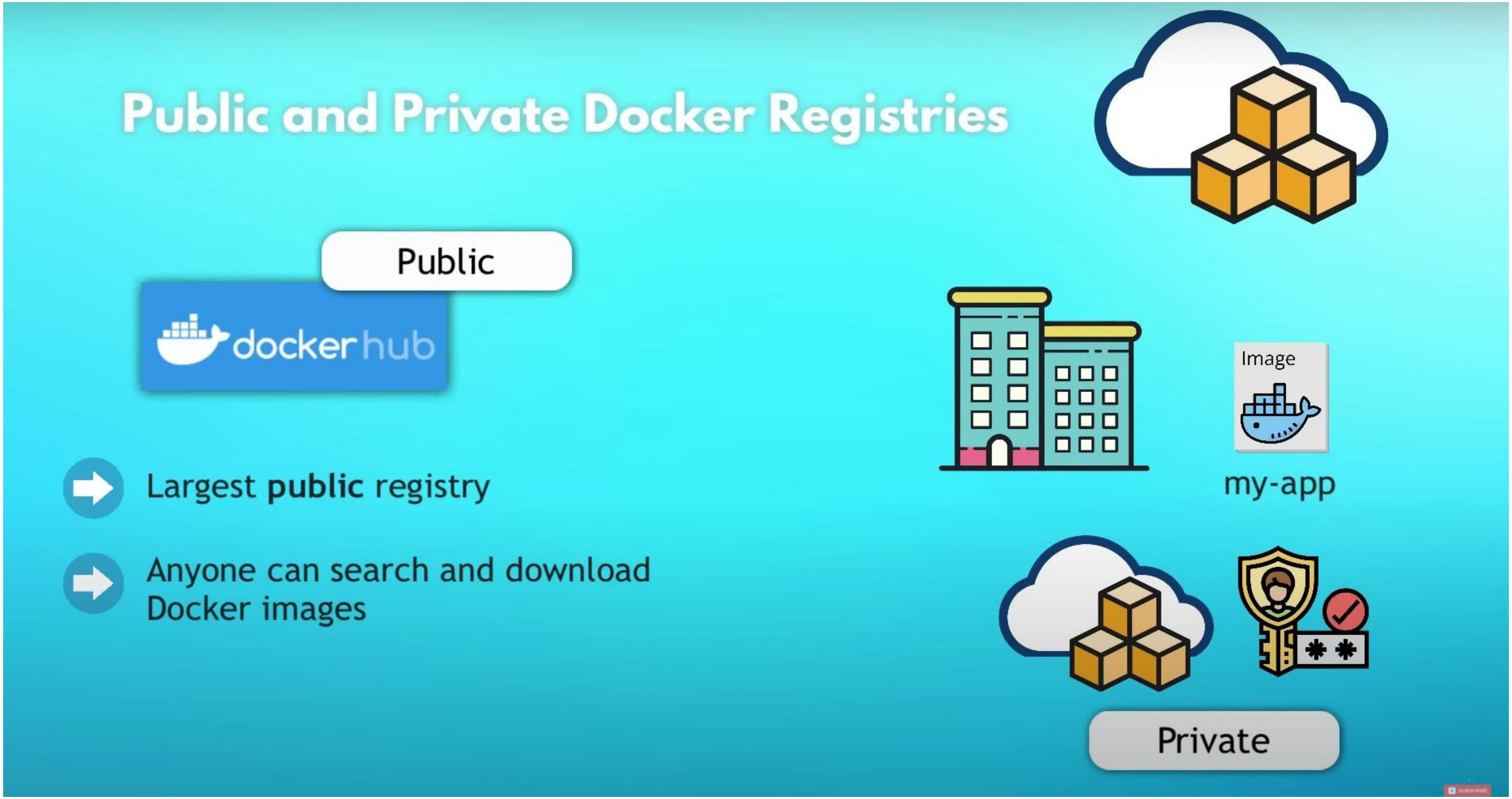
Choosing host port

Localhost

Port 3306

Port 3306

Standard to use the same port on your host as container is using



Public and Private Docker Registries

Public

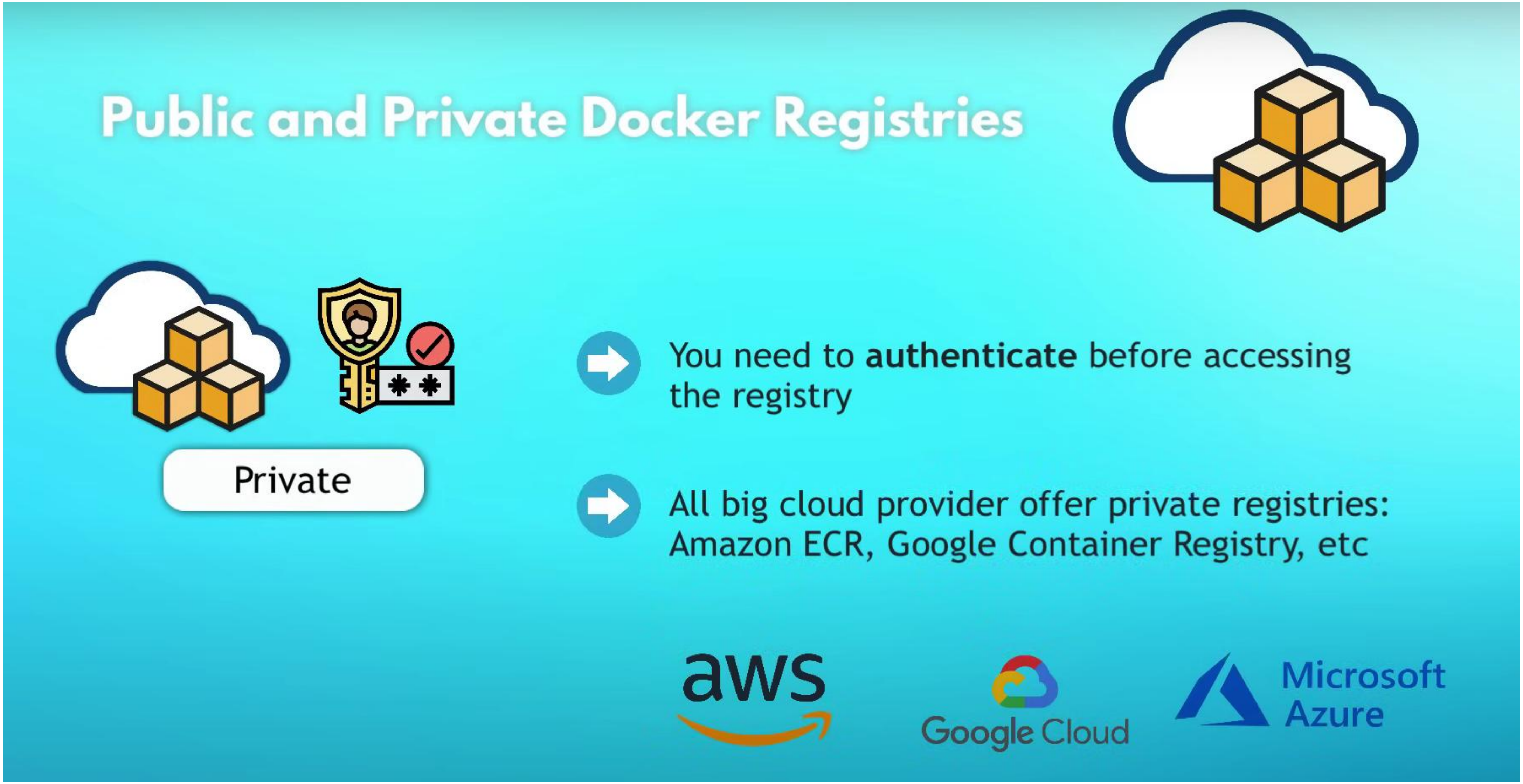
docker hub

- Largest **public** registry
- Anyone can search and download Docker images

Image

my-app

Private



Public and Private Docker Registries

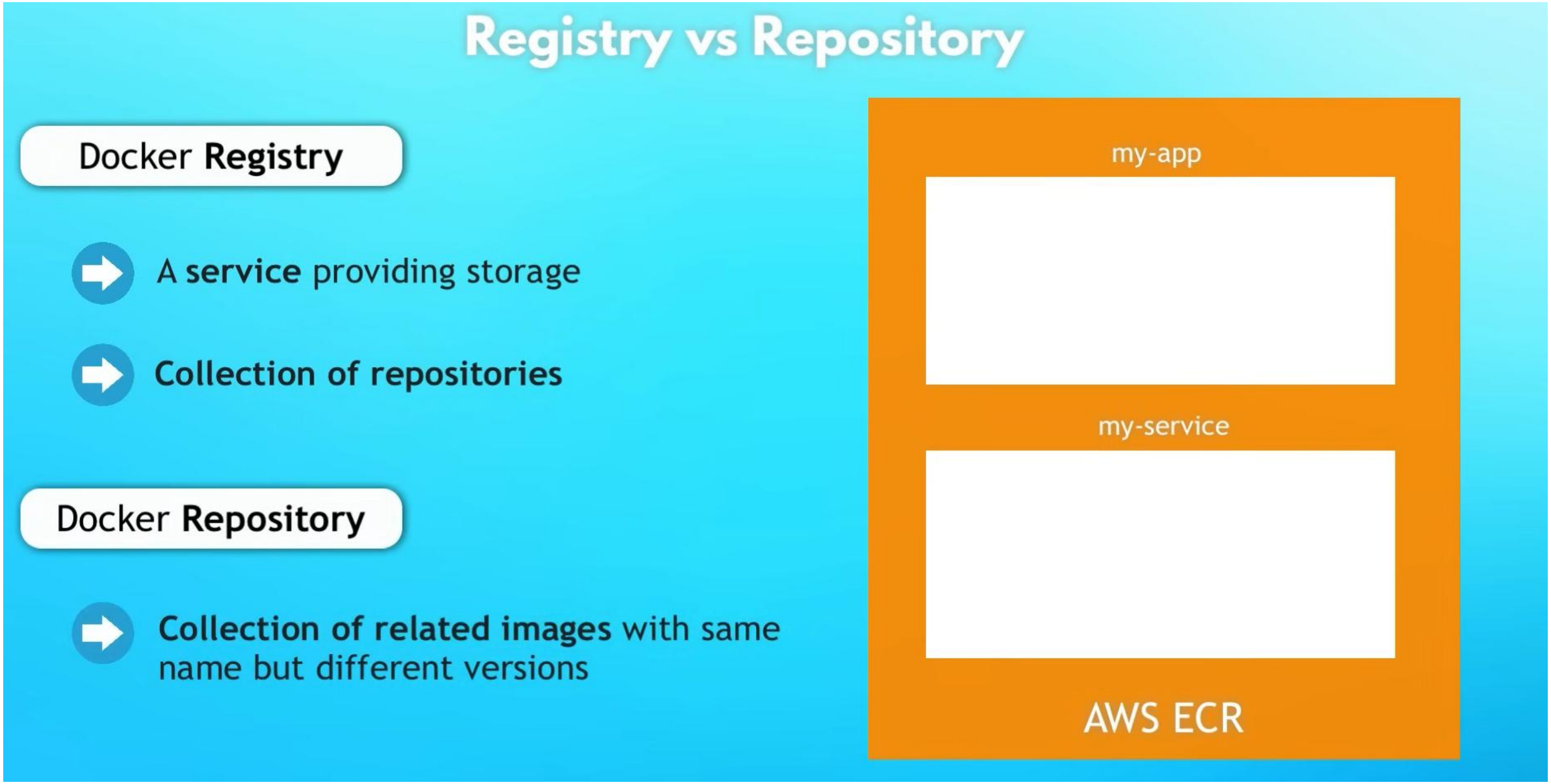
- You need to **authenticate** before accessing the registry
- All big cloud provider offer private registries: Amazon ECR, Google Container Registry, etc

Registry vs Repository

Docker **Registry**

- A **service** providing storage
- Can be **hosted by a third party**, like AWS, or by **yourself**

AWS ECR



Registry vs Repository

Docker Registry

- A **service** providing storage
- **Collection of repositories**

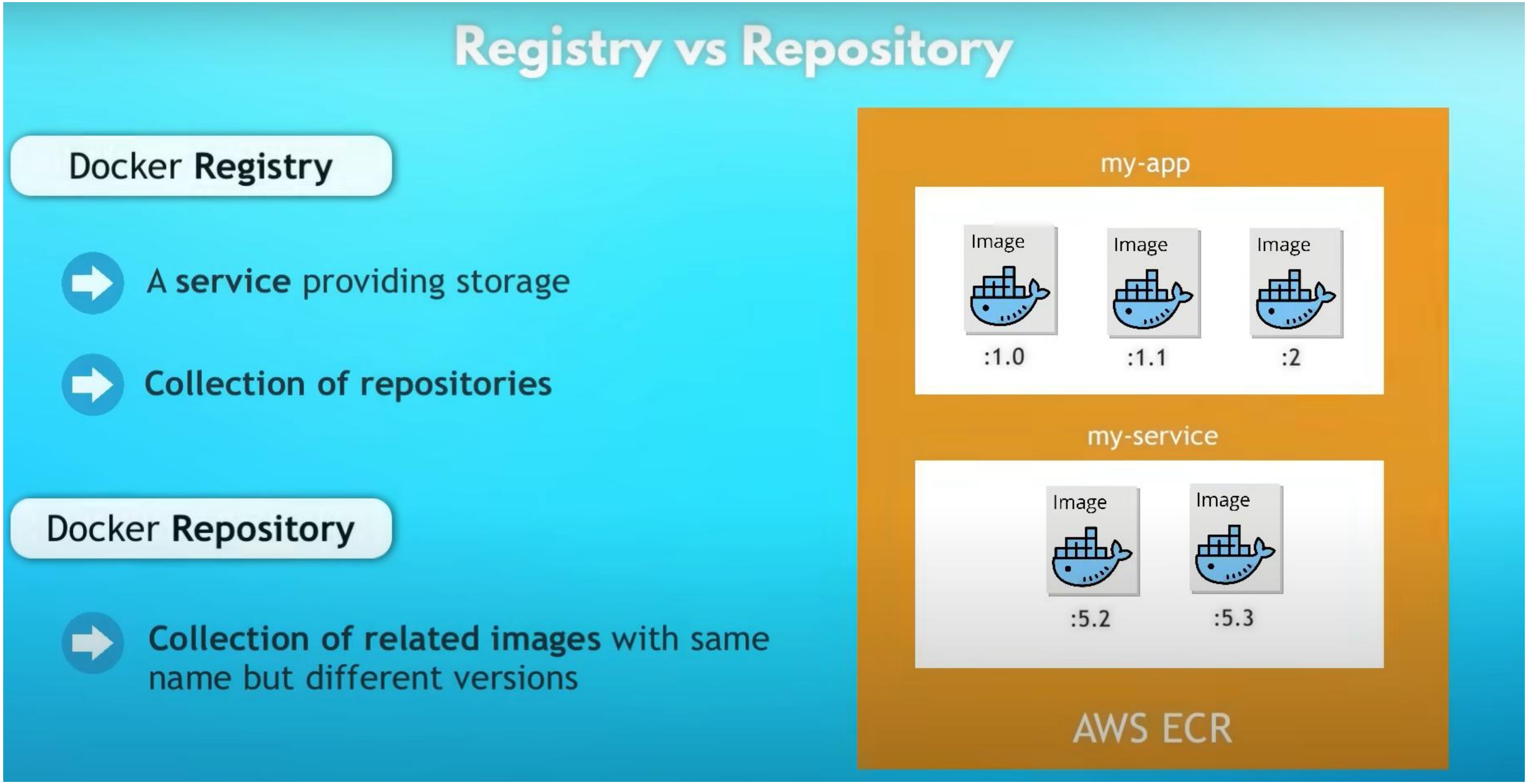
Docker Repository

- **Collection of related images** with same name but different versions

my-app

my-service

AWS ECR



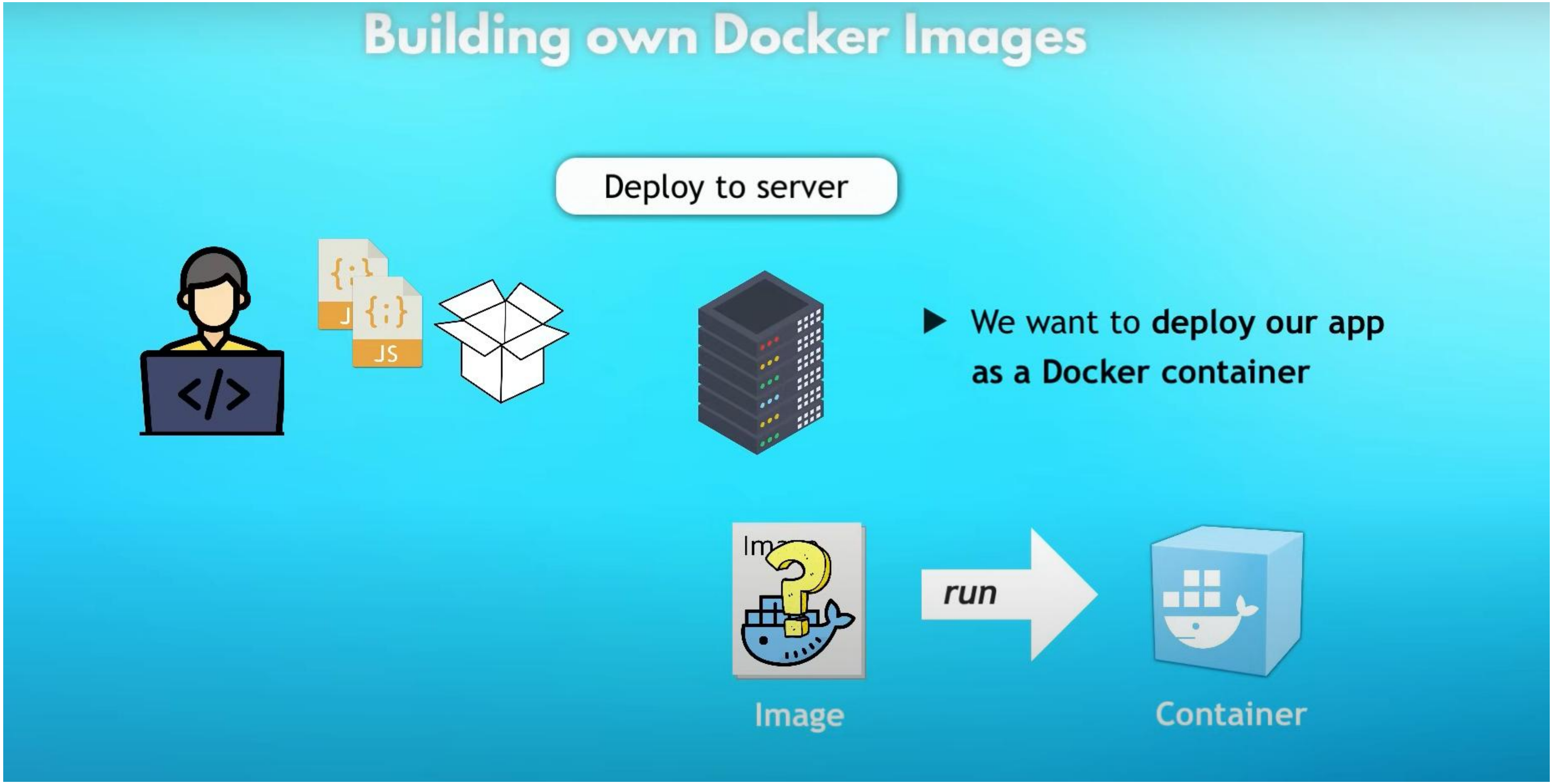
Registry vs Repository

Docker **Registry**

- A **service** providing storage
- **Collection of repositories**

Docker **Repository**

- **Collection of related images** with same name but different versions



Building own Docker Images

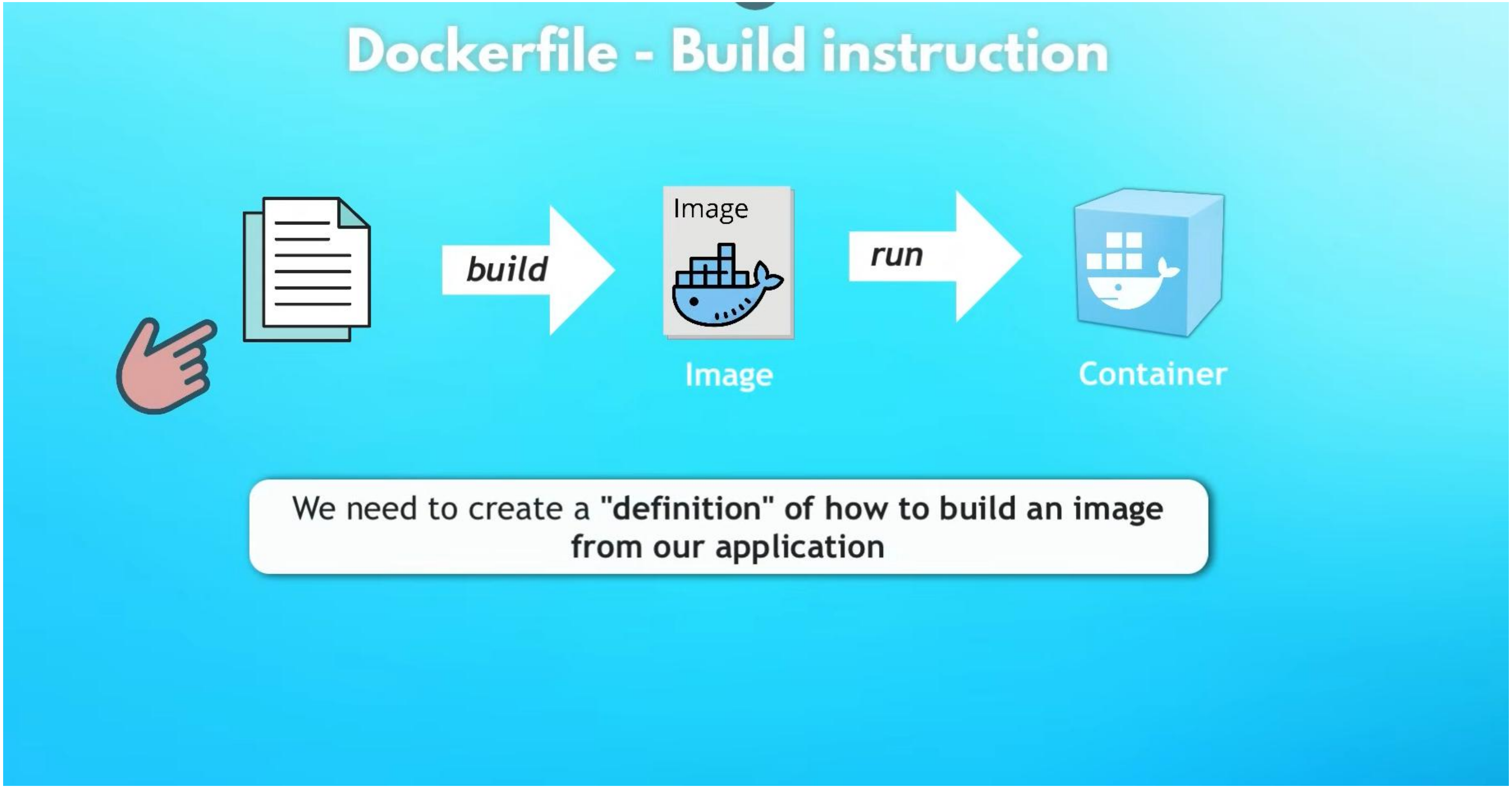
Deploy to server

- ▶ We want to **deploy our app as a Docker container**

run

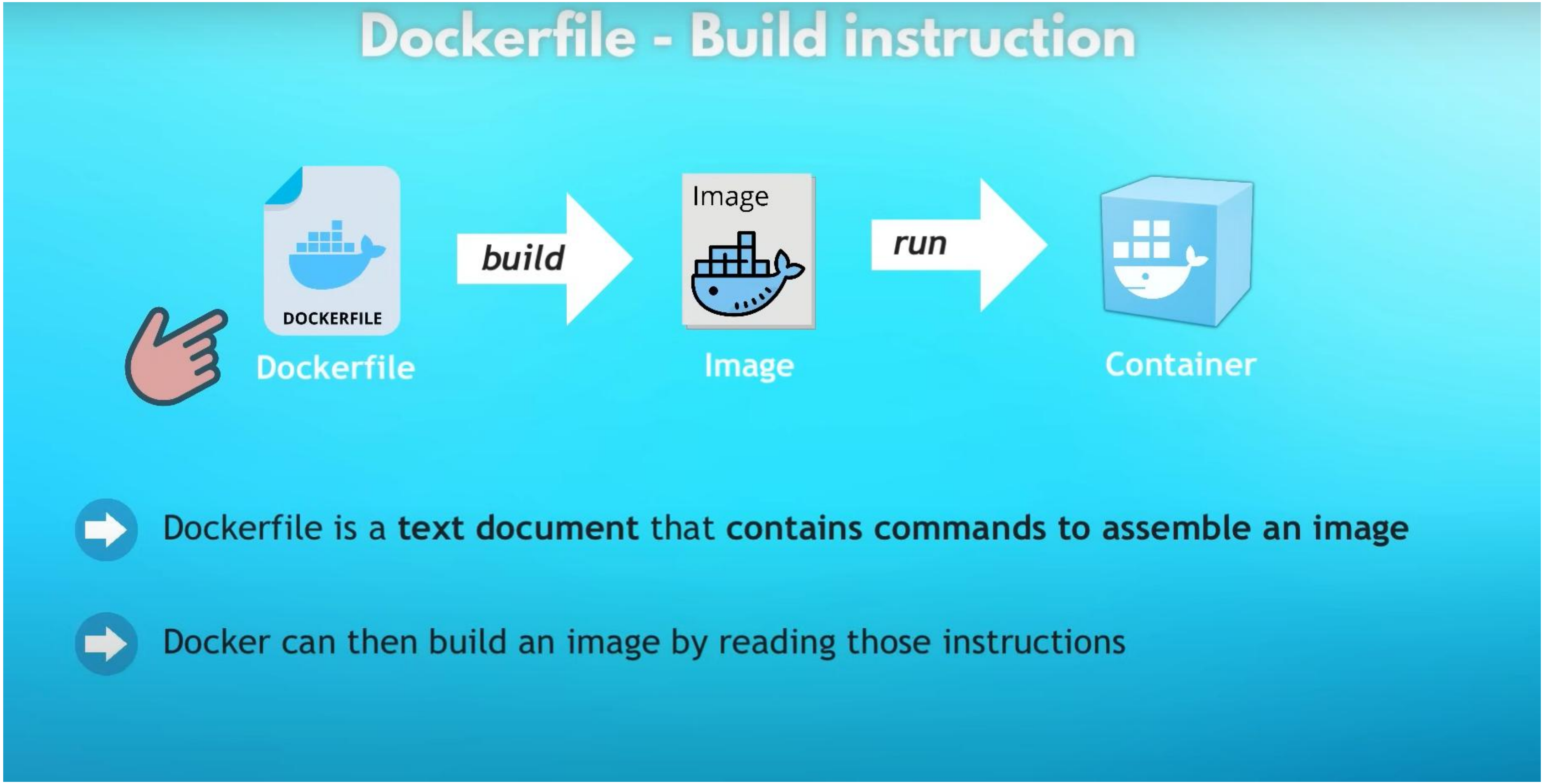
Image

Container



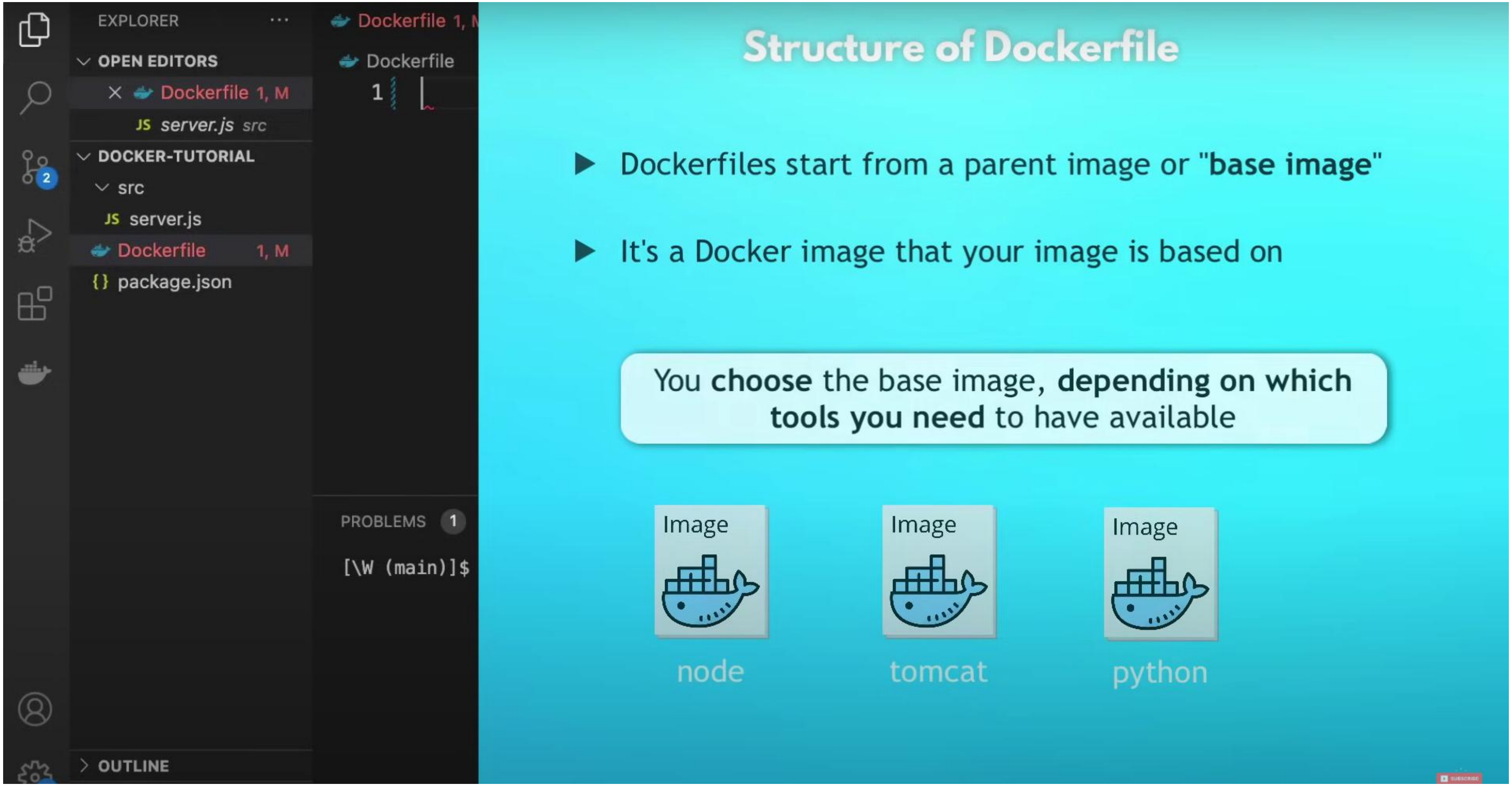
Dockerfile - Build instruction

We need to create a **"definition" of how to build an image from our application**



Dockerfile - Build instruction

- Dockerfile is a **text document** that **contains commands to assemble an image**
- Docker can then build an image by reading those instructions

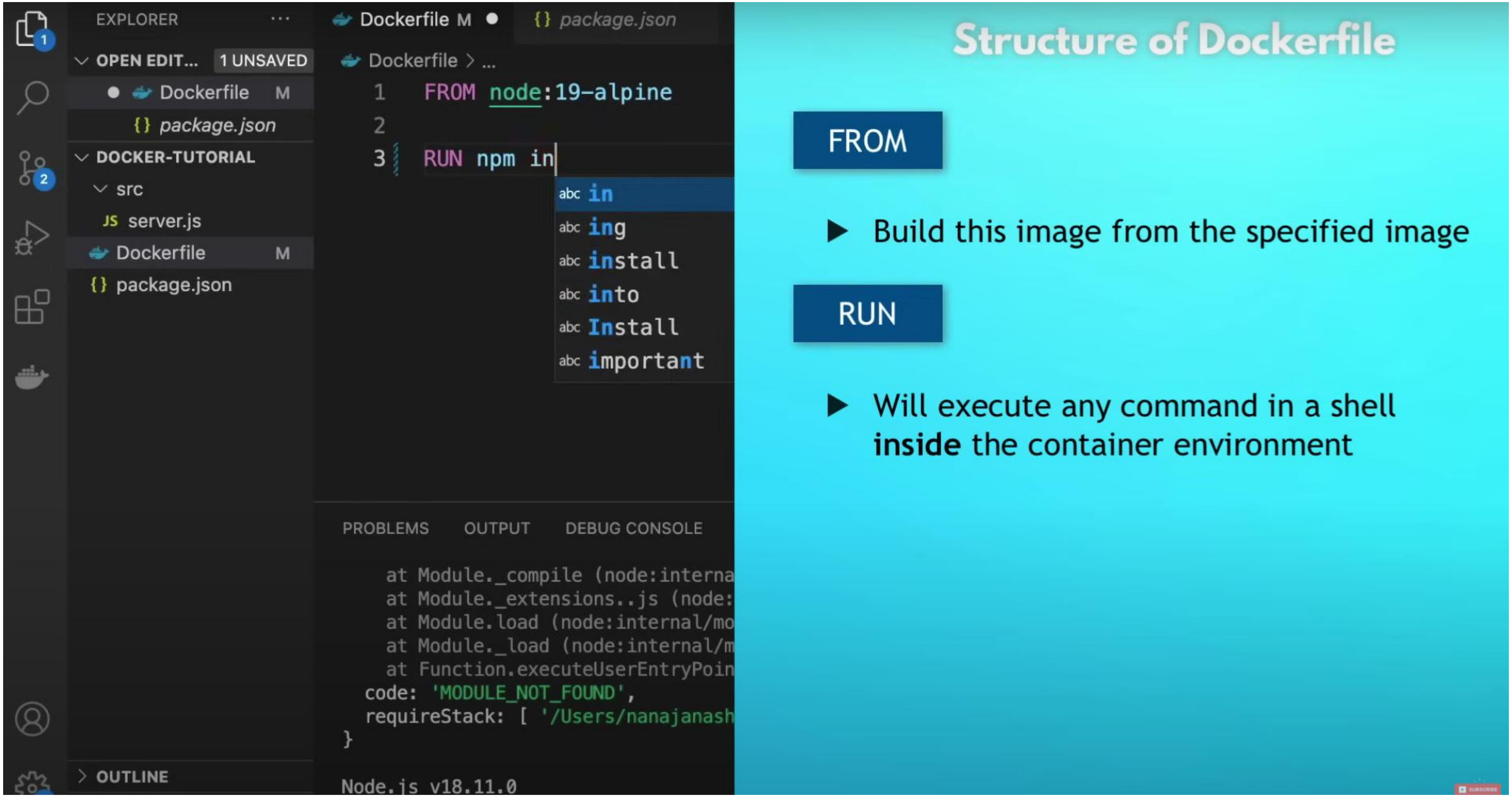


```
[\W (main)]$
```

Structure of Dockerfile

- Dockerfiles start from a parent image or "**base image**"
- It's a Docker image that your image is based on

You **choose** the base image, **depending on which tools you need** to have available



```
FROM node:19-alpine

RUN npm in
```

```
    at Module._compile (node:interna
    at Module._extensions..js (node:
    at Module.load (node:internal/mo
    at Module._load (node:internal/m
    at Function.executeUserEntryPoin
  code: 'MODULE_NOT_FOUND',
  requireStack: [ '/Users/nanajanash
}

Node.js v18.11.0
```

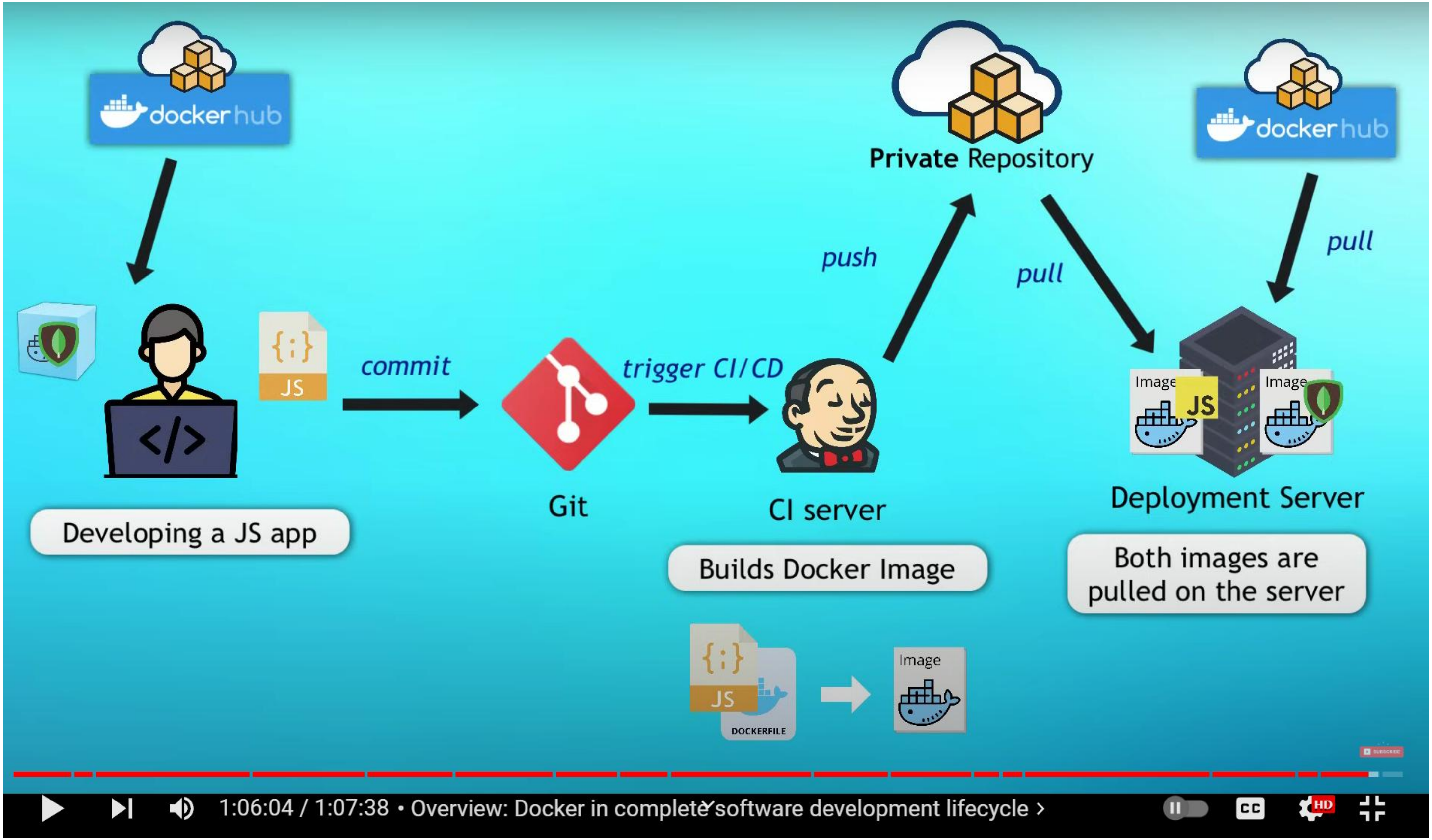
Structure of Dockerfile

FROM

- Build this image from the specified image

RUN

- Will execute any command in a shell **inside** the container environment



docker hub

Private Repository

docker hub

push

pull

pull

commit

trigger CI/CD

JS

Git

CI server

Image

JS

Image

Deployment Server

Developing a JS app

Builds Docker Image

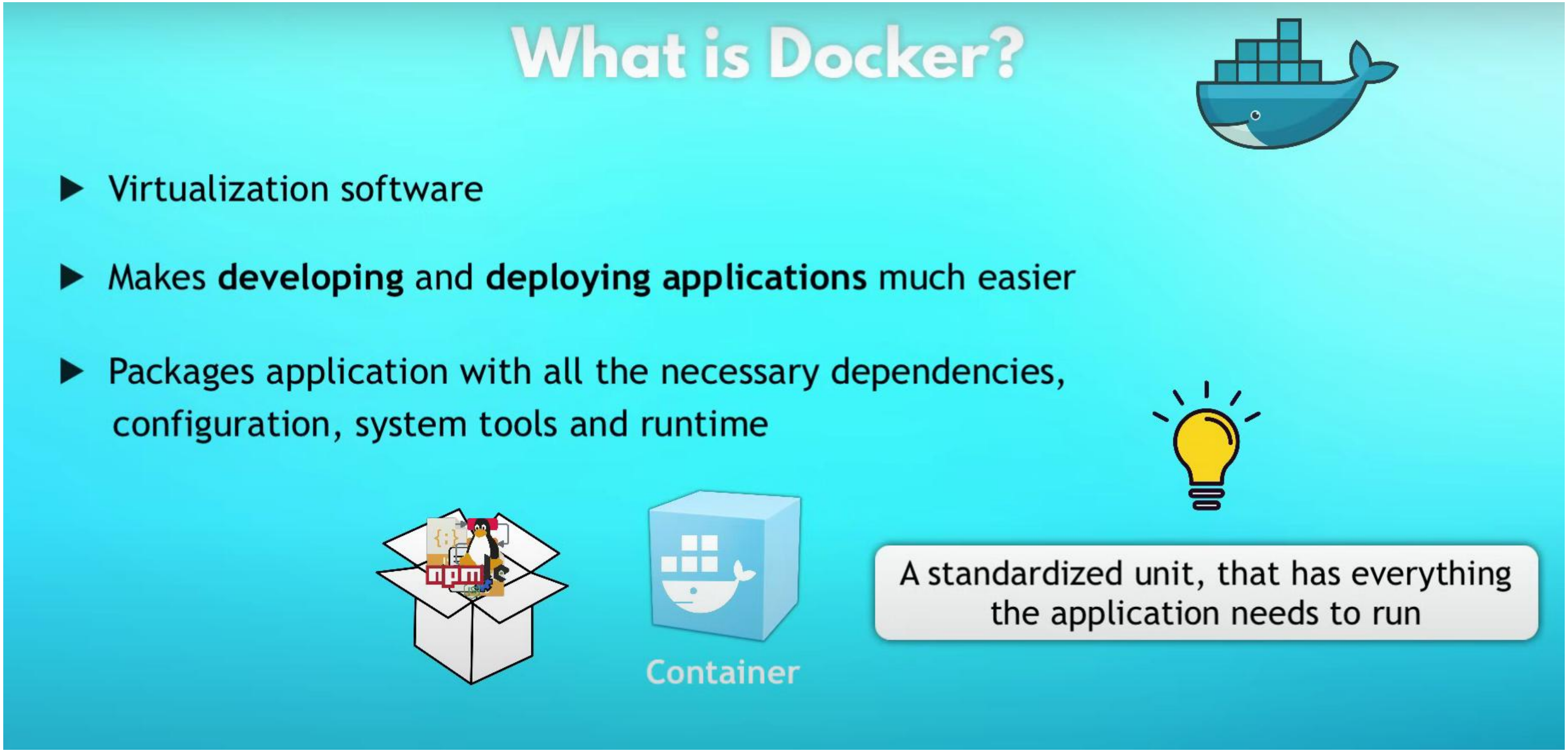
Both images are pulled on the server

JS

DOCKERFILE

Image

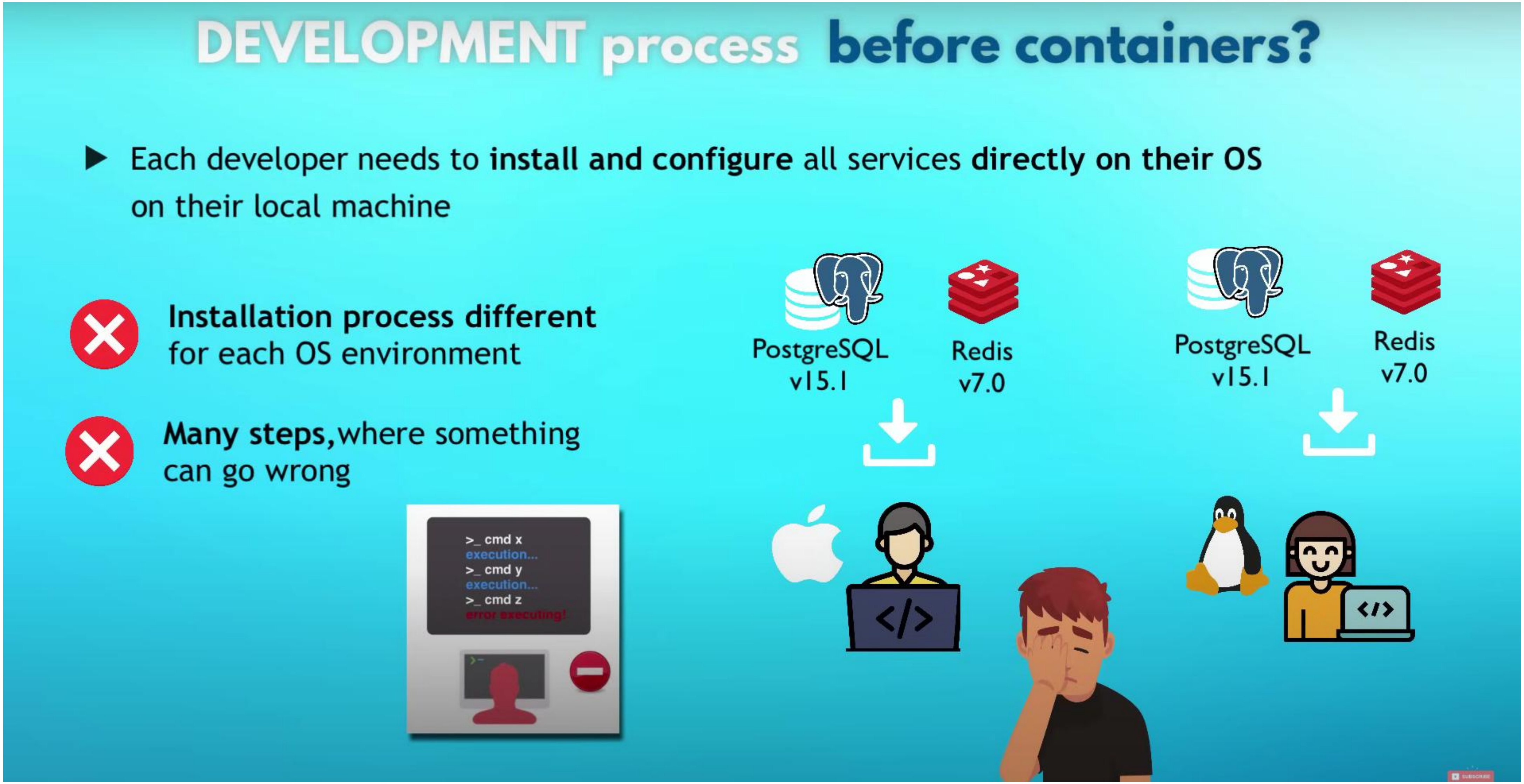
1:06:04 / 1:07:38 • Overview: Docker in complete software development lifecycle ›



What is Docker?

- Virtualization software
- Makes **developing** and **deploying applications** much easier
- Packages application with all the necessary dependencies, configuration, system tools and runtime

A standardized unit, that has everything the application needs to run



DEVELOPMENT process before containers?

- Each developer needs to **install and configure** all services **directly on their OS** on their local machine

❌ **Installation process different** for each OS environment

❌ **Many steps,** where something can go wrong

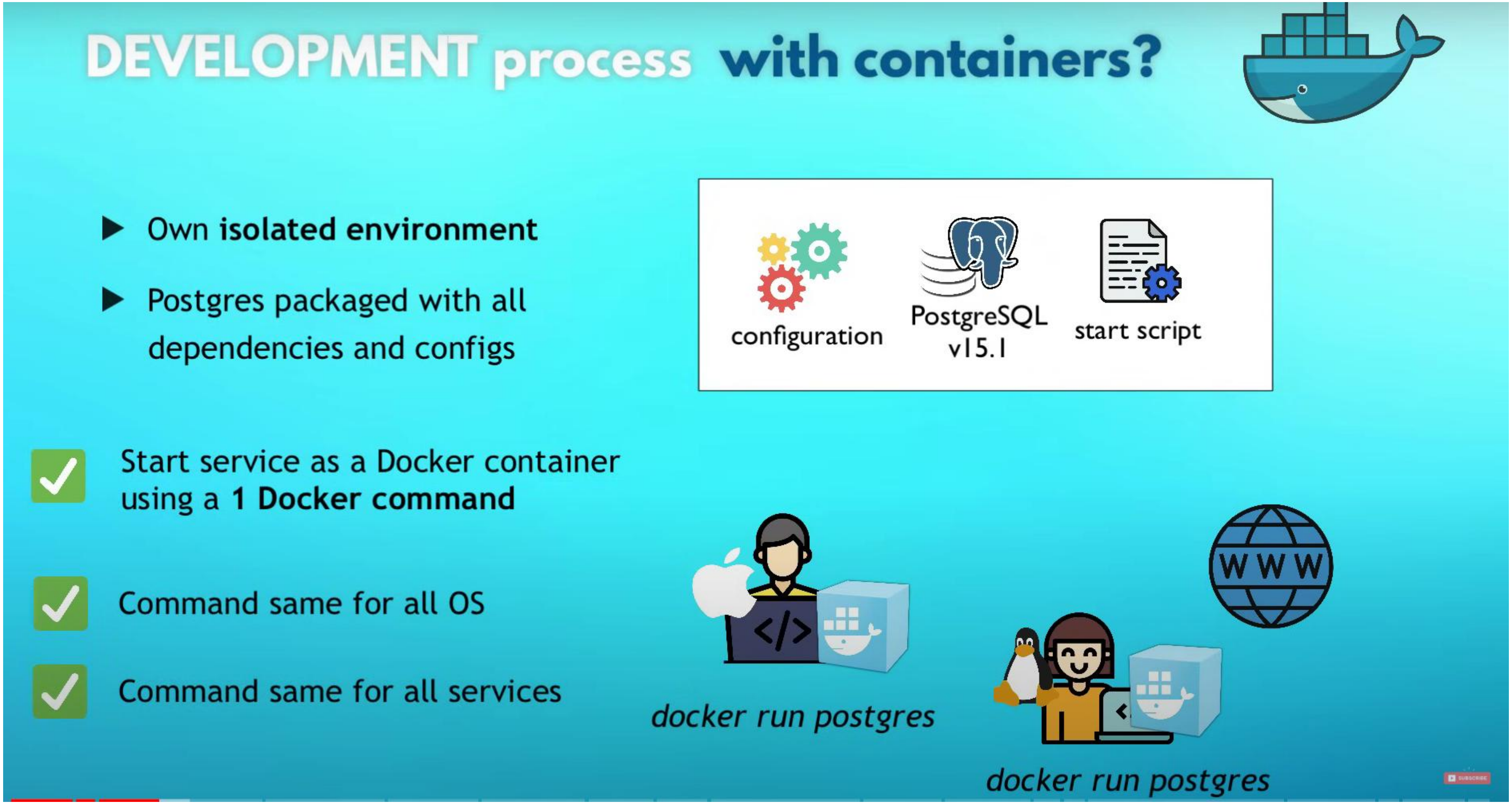
```
>_ cmd x
execution...
>_ cmd y
execution...
>_ cmd z
error executing!
```

PostgreSQL v15.1

Redis v7.0

PostgreSQL v15.1

Redis v7.0



DEVELOPMENT process with containers?

- ▶ Own **isolated environment**
- ▶ Postgres packaged with all dependencies and configs

configuration

PostgreSQL v15.1

start script

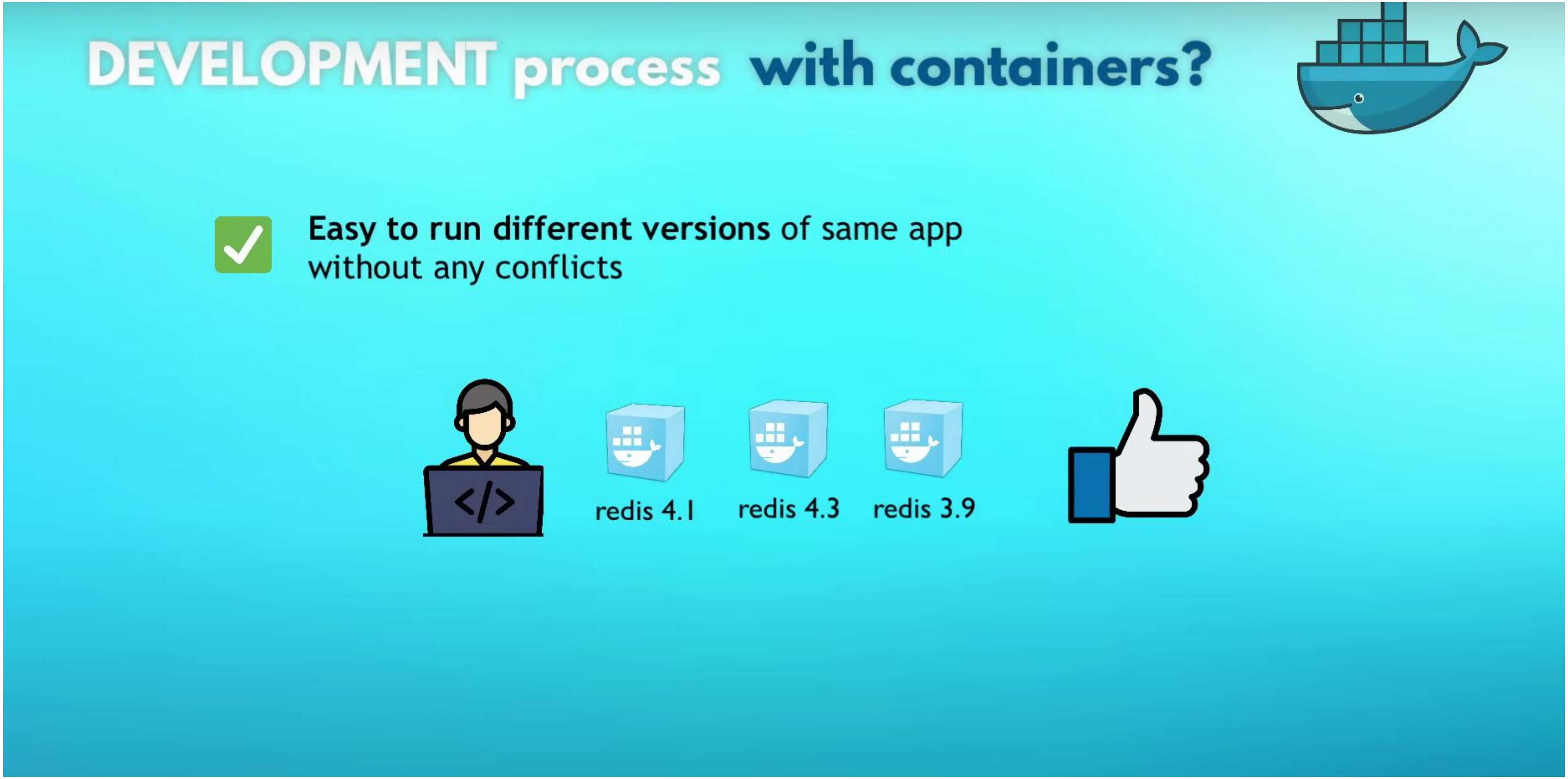
✓ Start service as a Docker container using a **1 Docker command**

✓ Command same for all OS

✓ Command same for all services

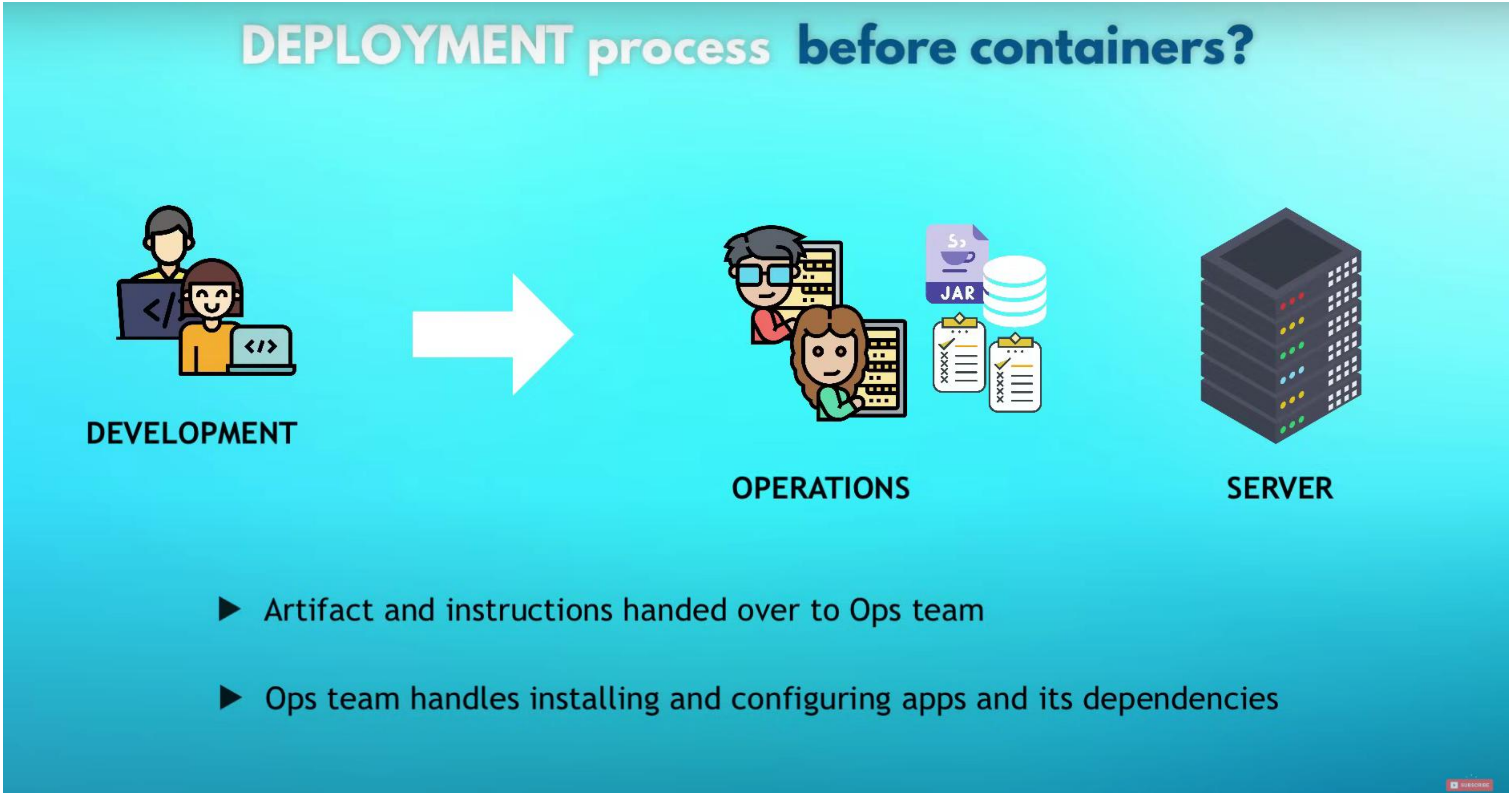
docker run postgres

docker run postgres



DEVELOPMENT process with containers?

☑ **Easy to run different versions** of same app without any conflicts



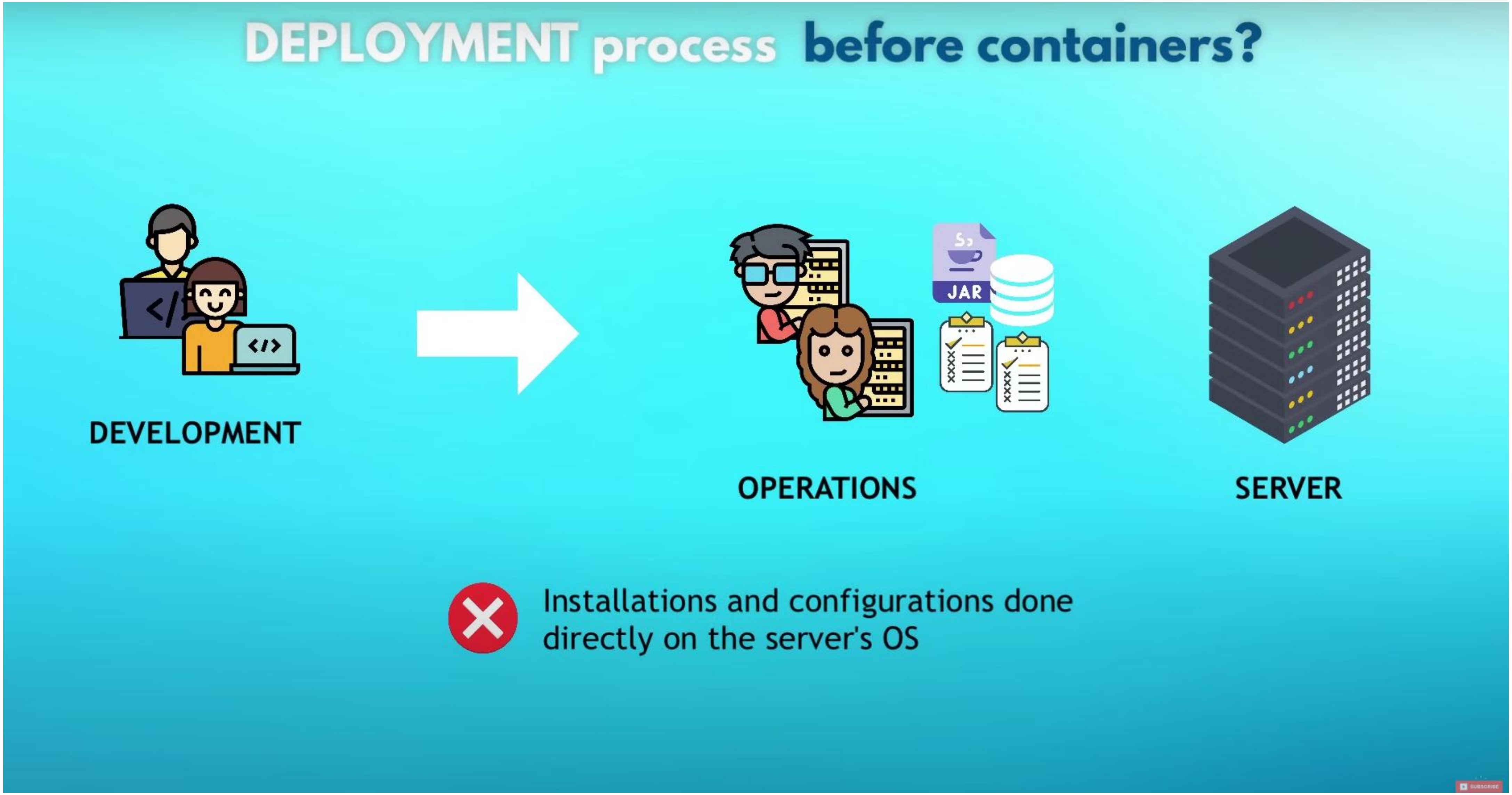
DEPLOYMENT process before containers?

DEVELOPMENT

OPERATIONS

SERVER

- Artifact and instructions handed over to Ops team
- Ops team handles installing and configuring apps and its dependencies



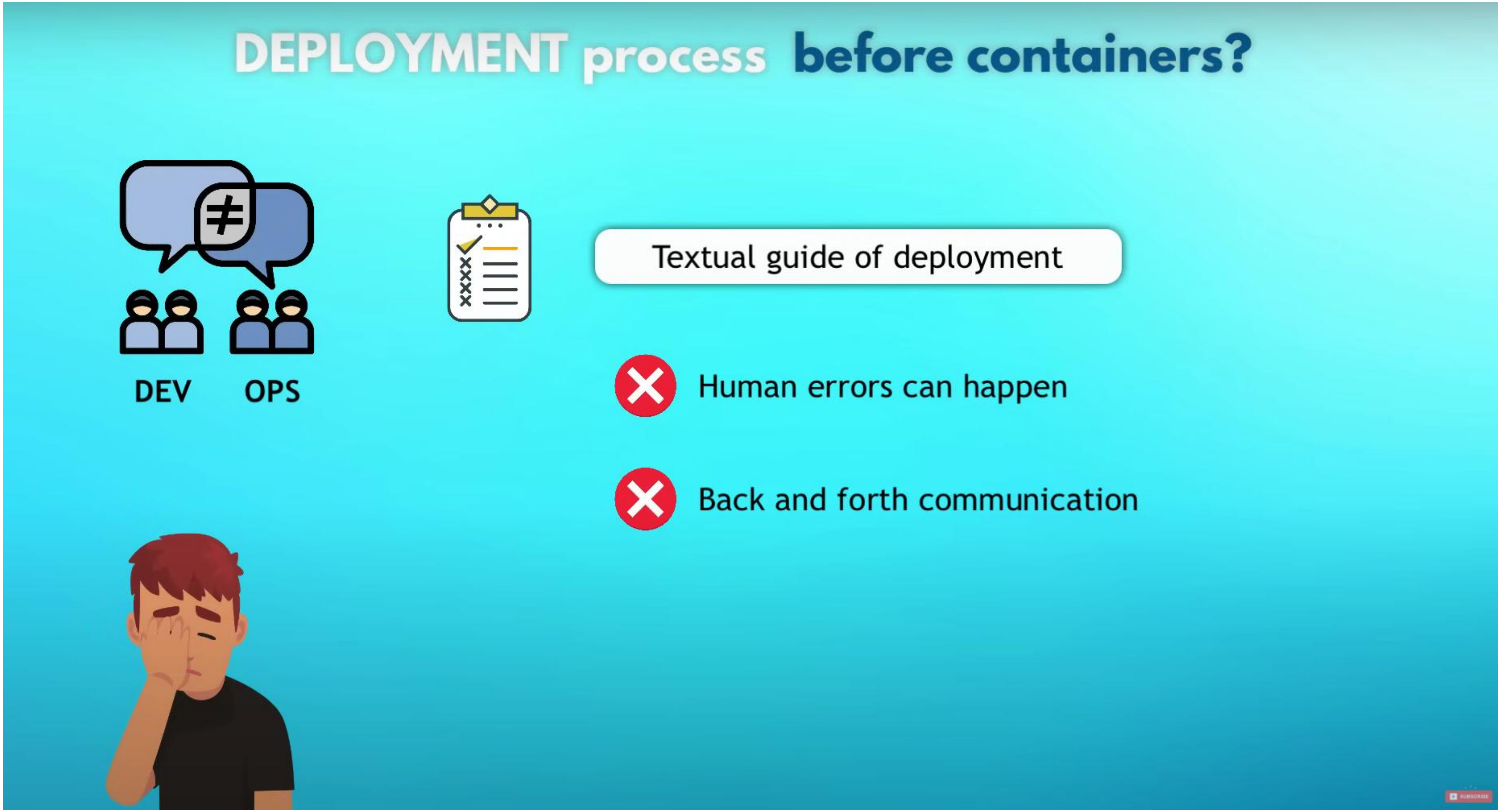
DEPLOYMENT process before containers?

DEVELOPMENT

OPERATIONS

SERVER

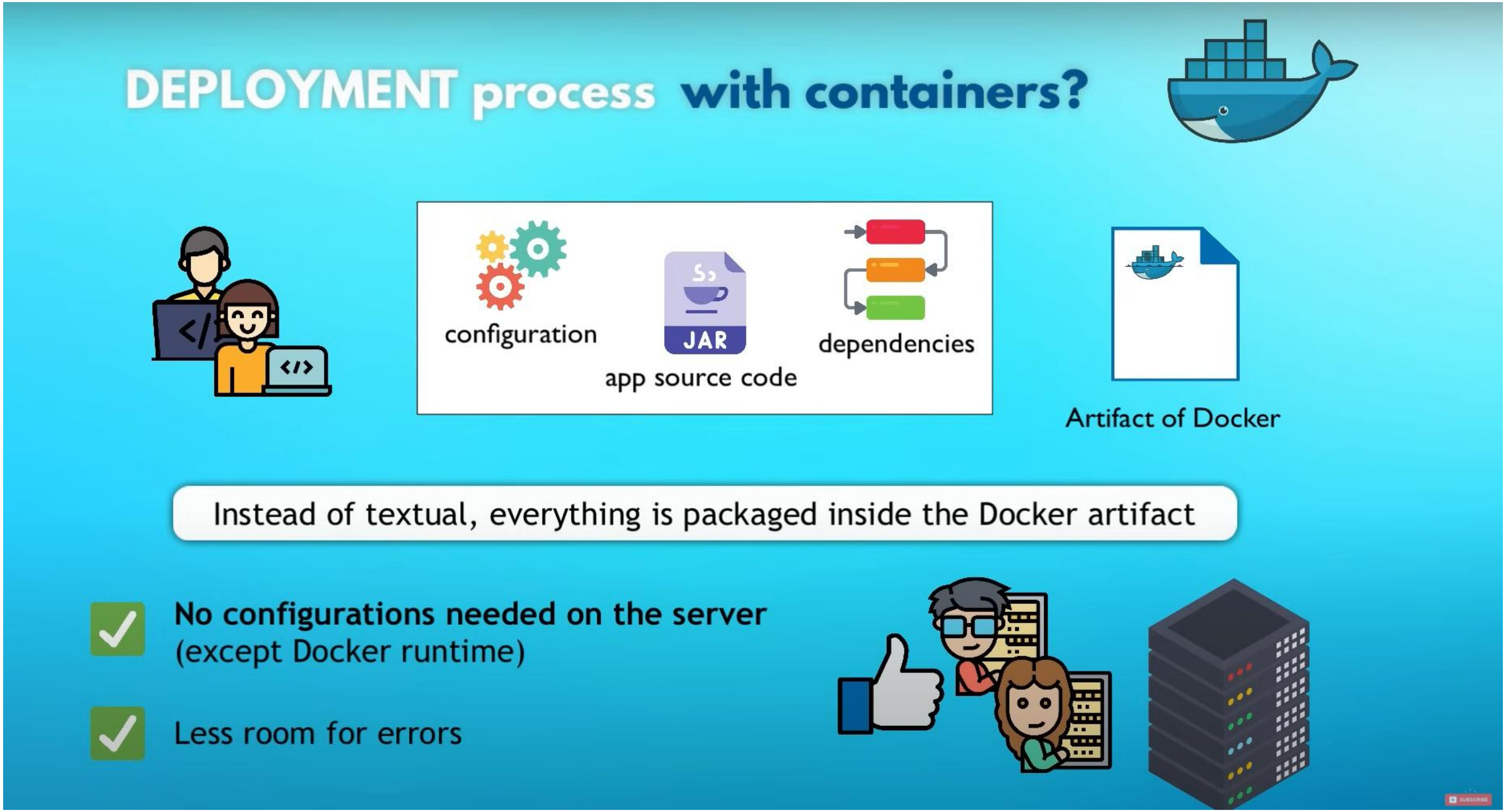
☒ Installations and configurations done directly on the server's OS



DEPLOYMENT process before containers?

Textual guide of deployment

- ✗ Human errors can happen
- ✗ Back and forth communication

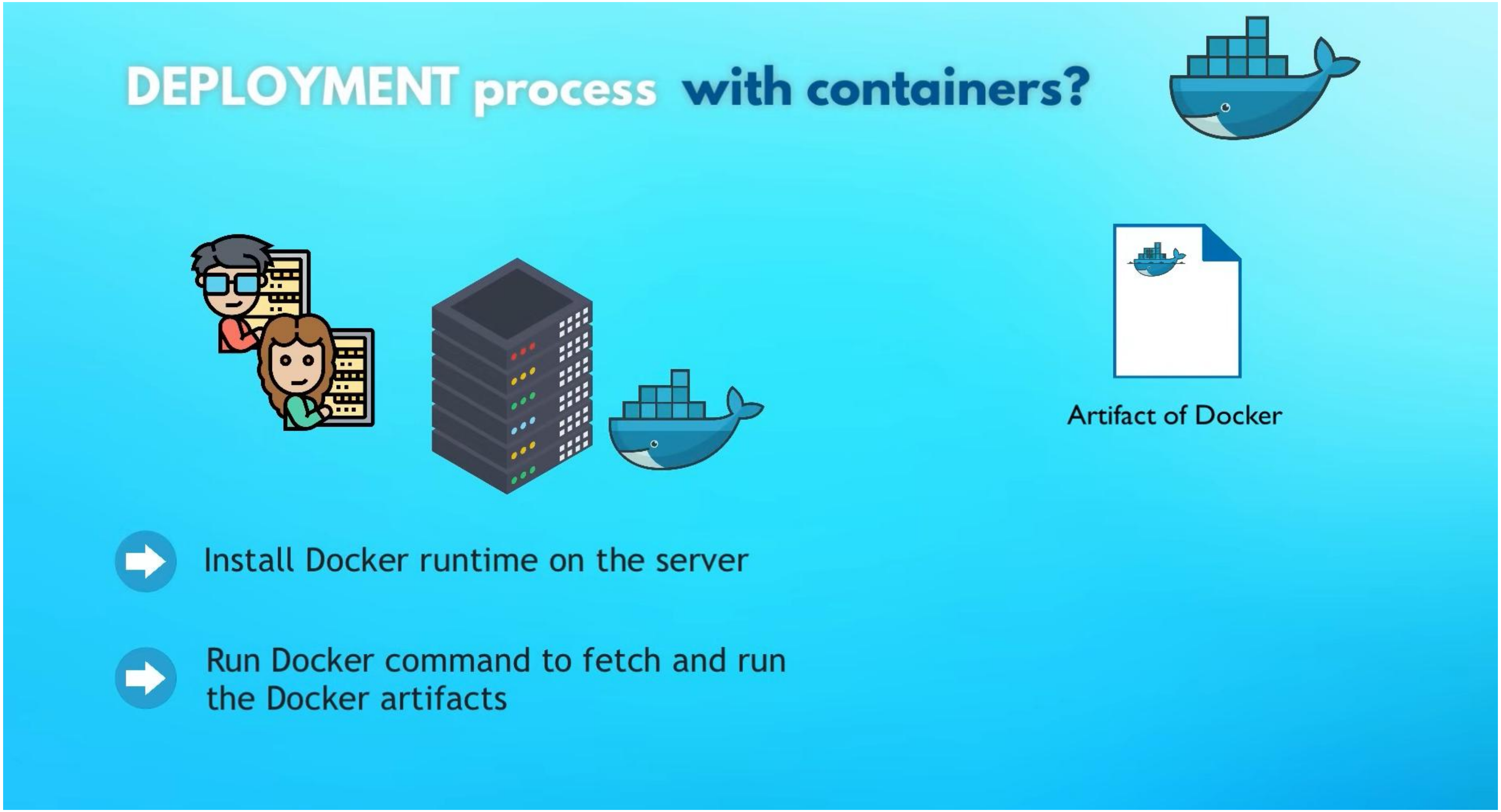


DEPLOYMENT process with containers?

Instead of textual, everything is packaged inside the Docker artifact

✓ **No configurations needed on the server**
(except Docker runtime)

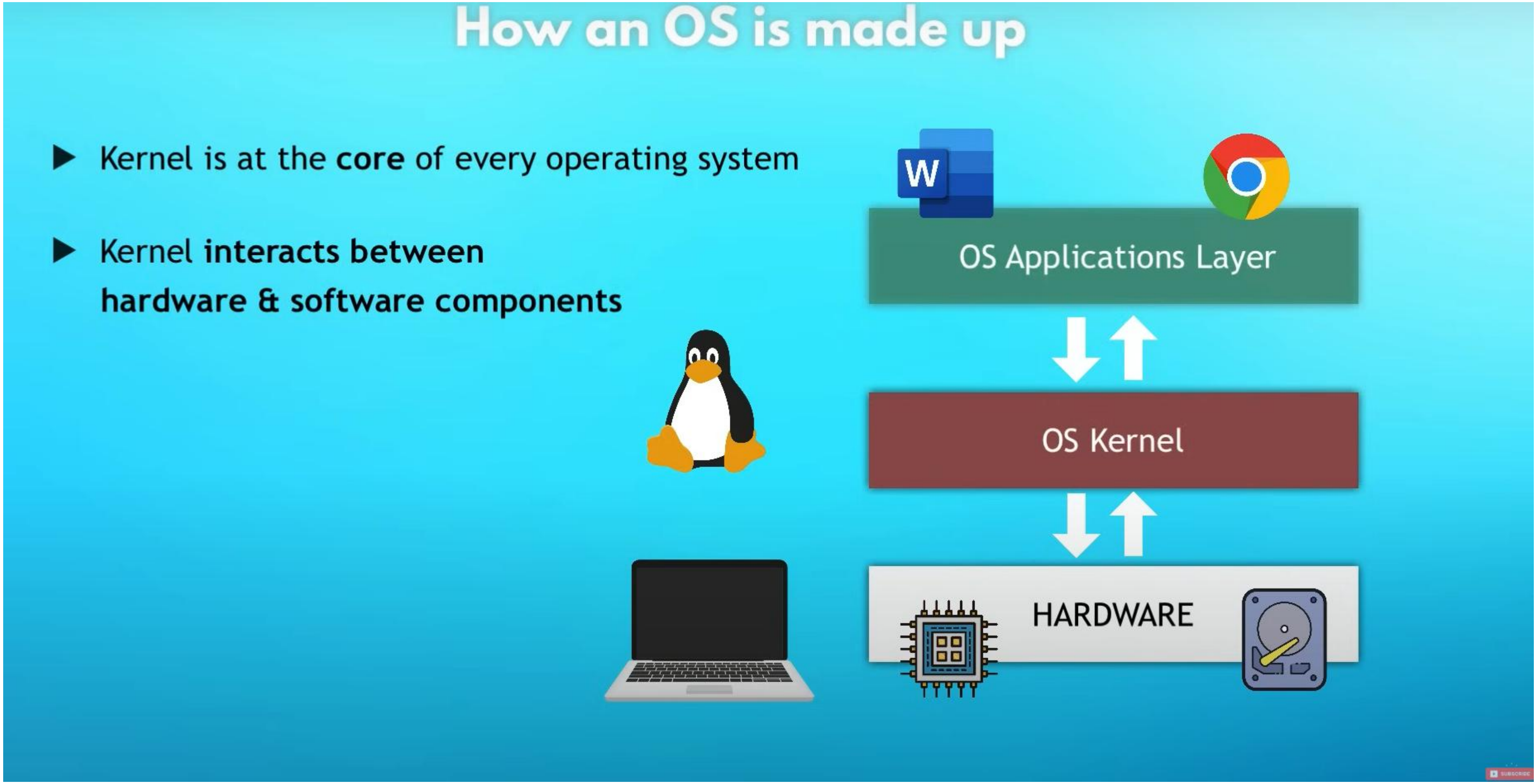
✓ Less room for errors



DEPLOYMENT process with containers?

Artifact of Docker

- Install Docker runtime on the server
- Run Docker command to fetch and run the Docker artifacts



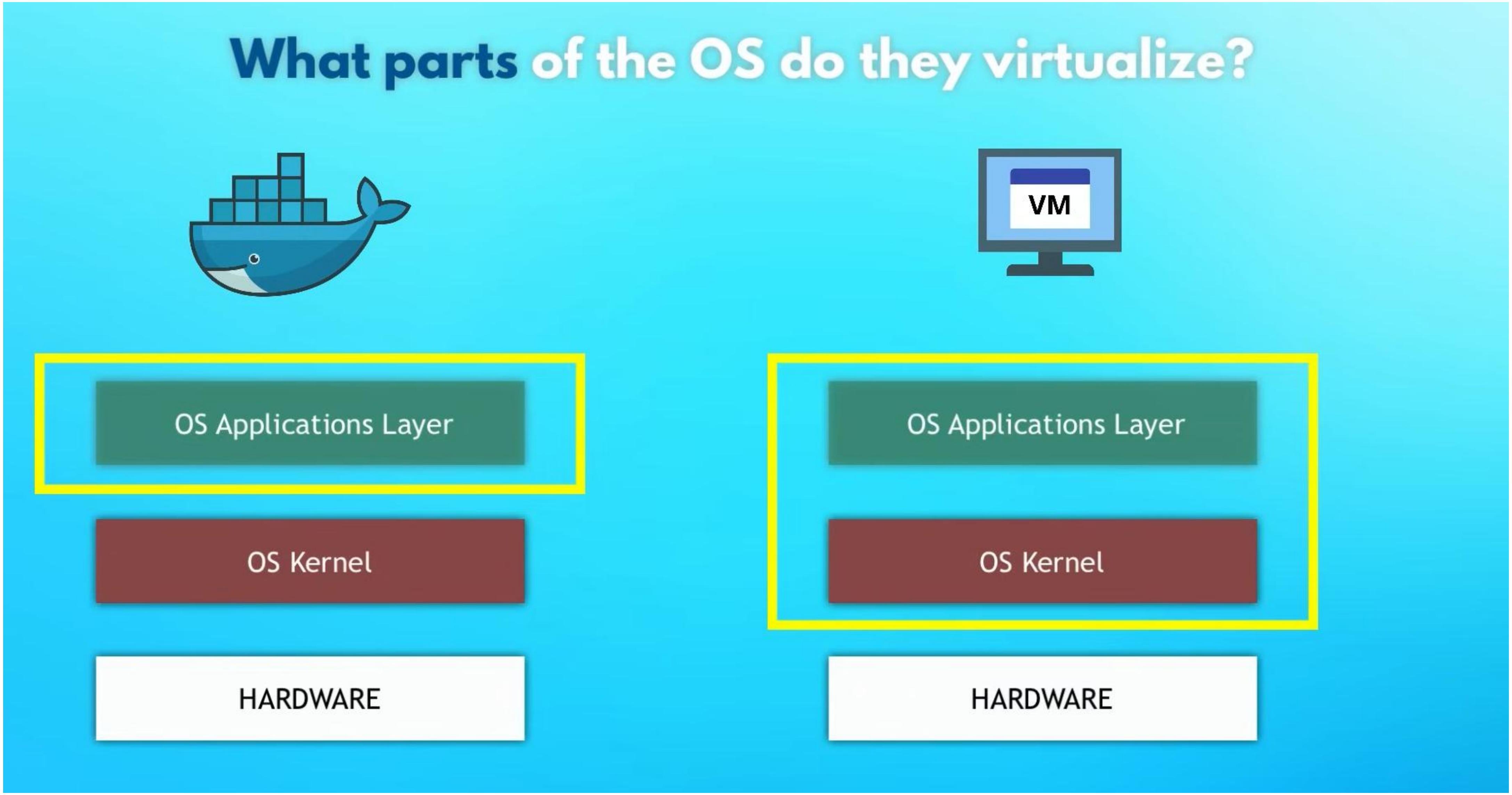
How an OS is made up

- Kernel is at the **core** of every operating system
- Kernel **interacts between hardware & software components**

OS Applications Layer

OS Kernel

HARDWARE



What parts of the OS do they virtualize?

OS Applications Layer

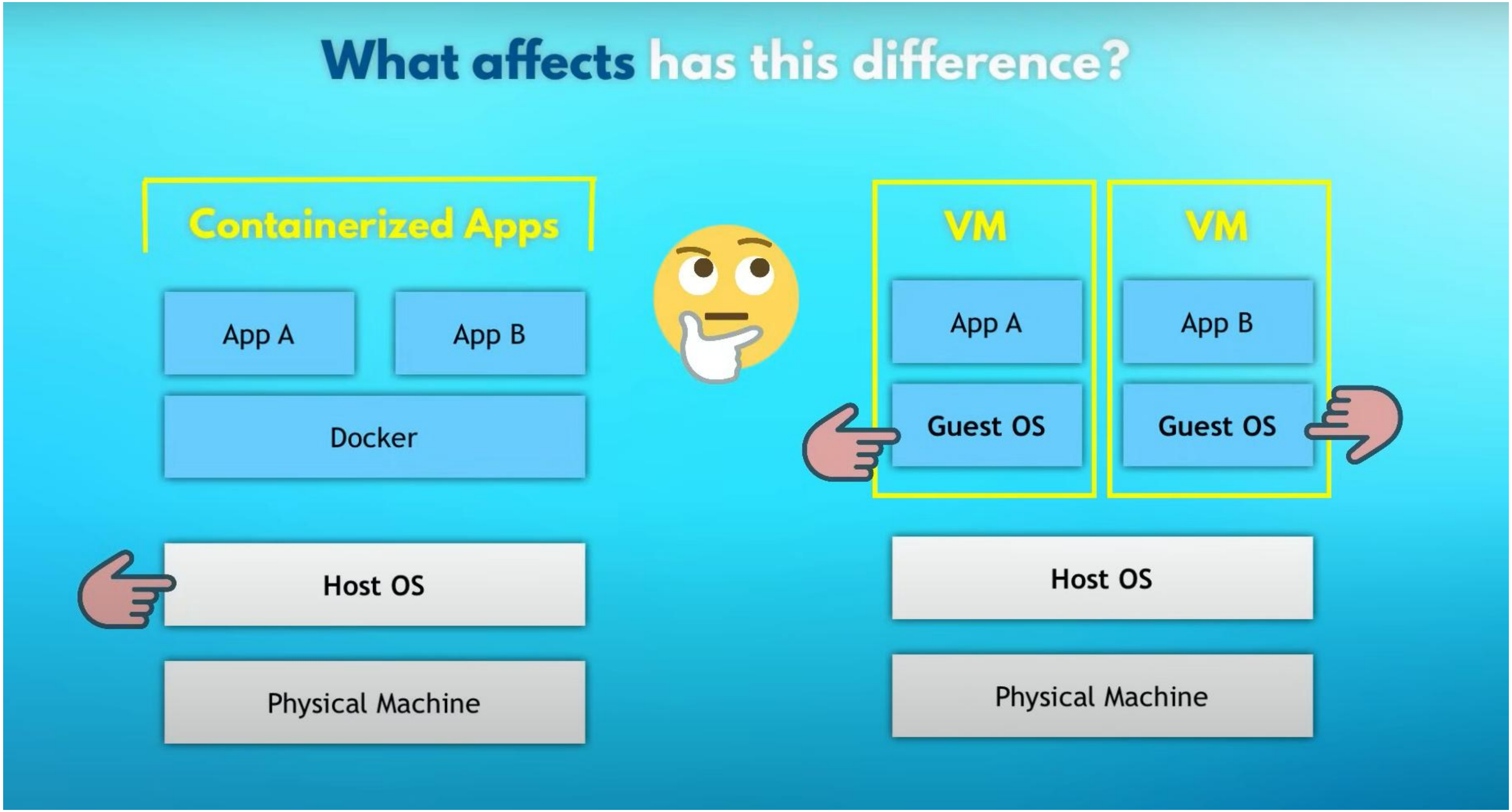
OS Kernel

HARDWARE

OS Applications Layer

OS Kernel

HARDWARE



What affects has this difference?

Containerized Apps

App A

App B

Docker

Host OS

Physical Machine

VM

App A

Guest OS

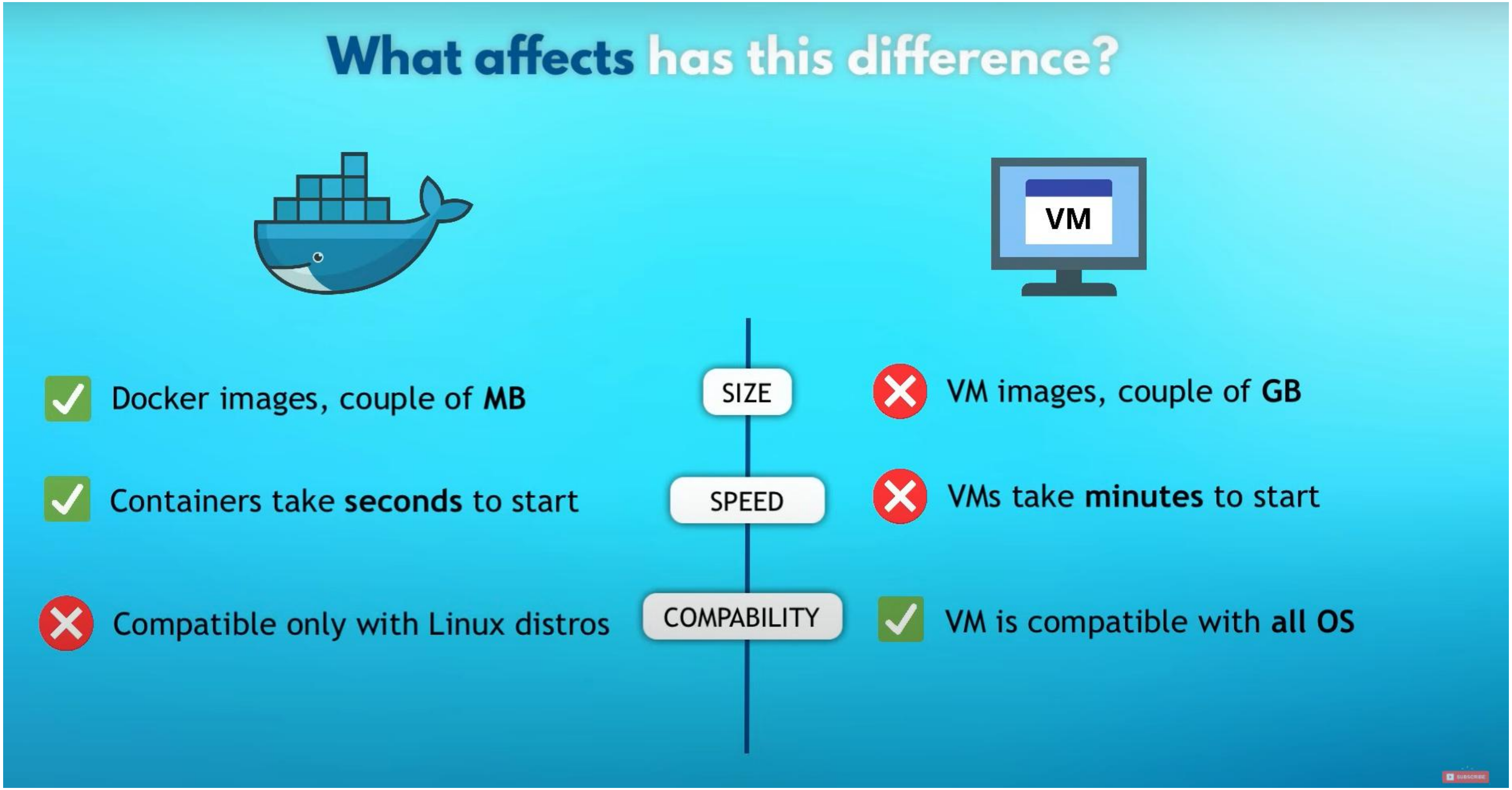
VM

App B

Guest OS

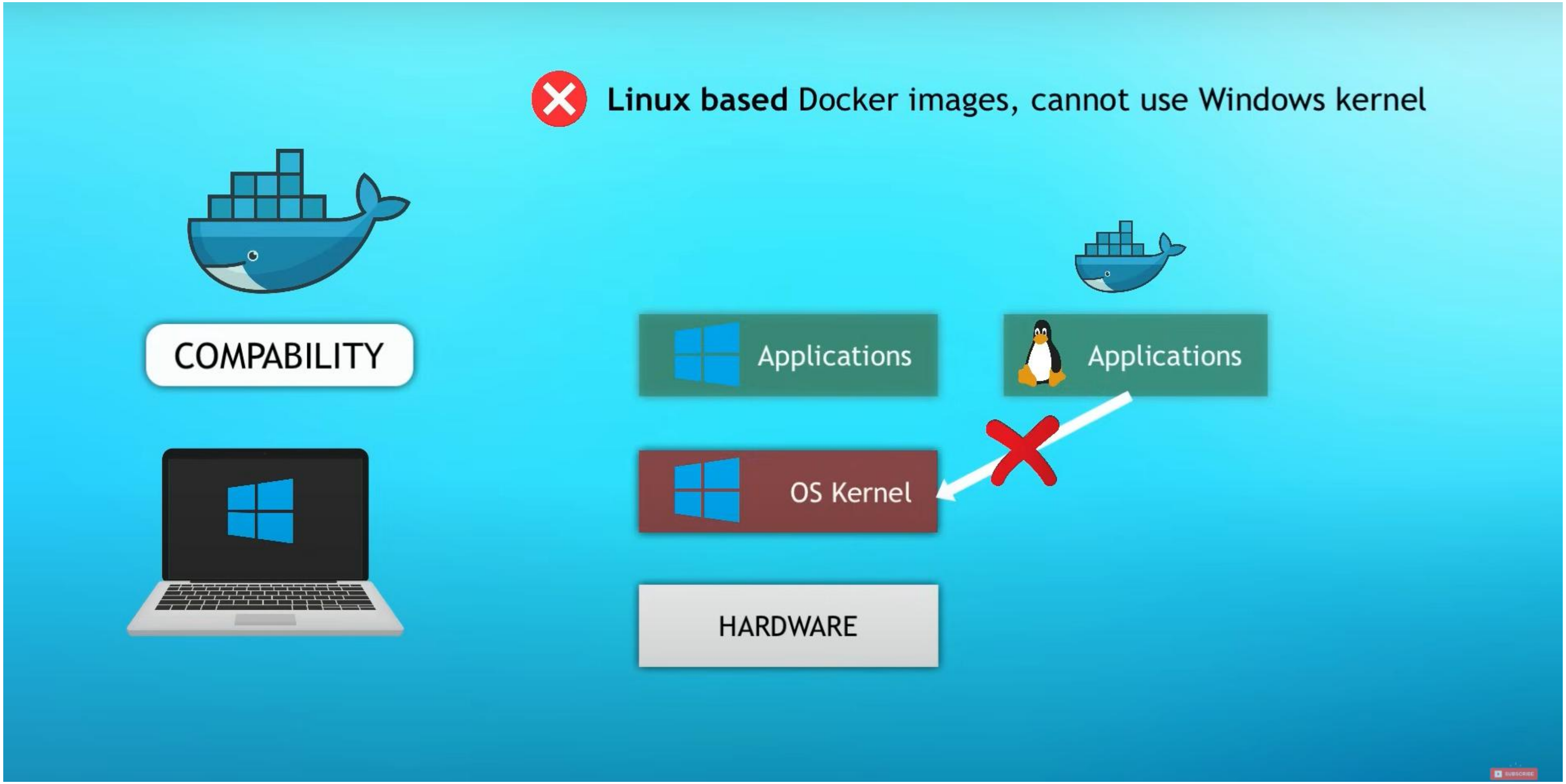
Host OS

Physical Machine



What affects has this difference?

✓ Docker images, couple of **MB**	SIZE	✗ VM images, couple of **GB**
✓ Containers take **seconds** to start	SPEED	✗ VMs take **minutes** to start
✗ Compatible only with Linux distros	COMPABILITY	✓ VM is compatible with **all OS**



Linux based Docker images, cannot use Windows kernel

COMPABILITY

Applications

Applications

OS Kernel

HARDWARE

COMPABILITY

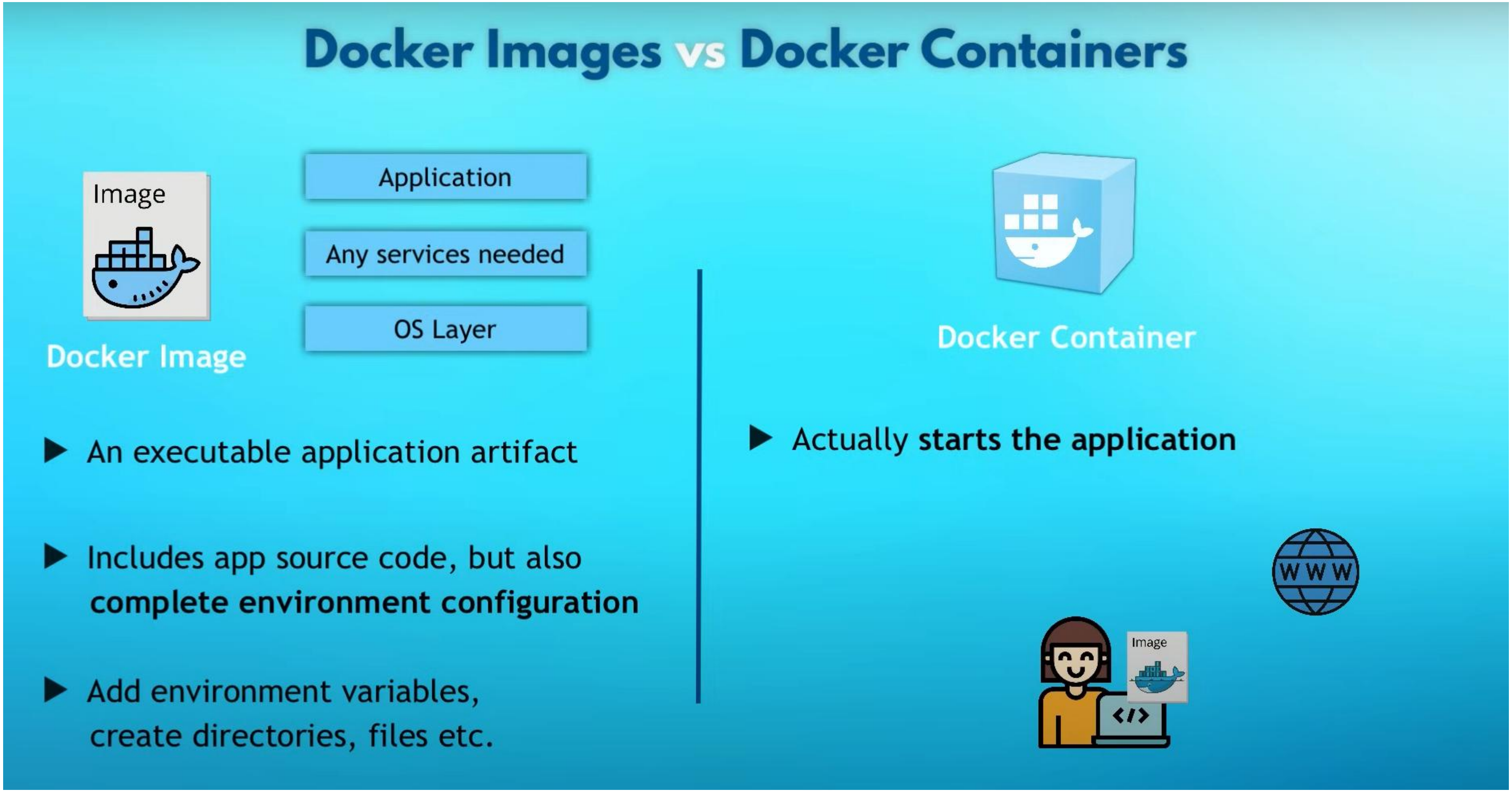
Most containers are Linux based

Originally built for Linux OS

COMPABILITY

Docker Desktop

✓ Allows you to run Linux containers on Windows or MacOS



Docker Images vs Docker Containers

Docker Image

Application

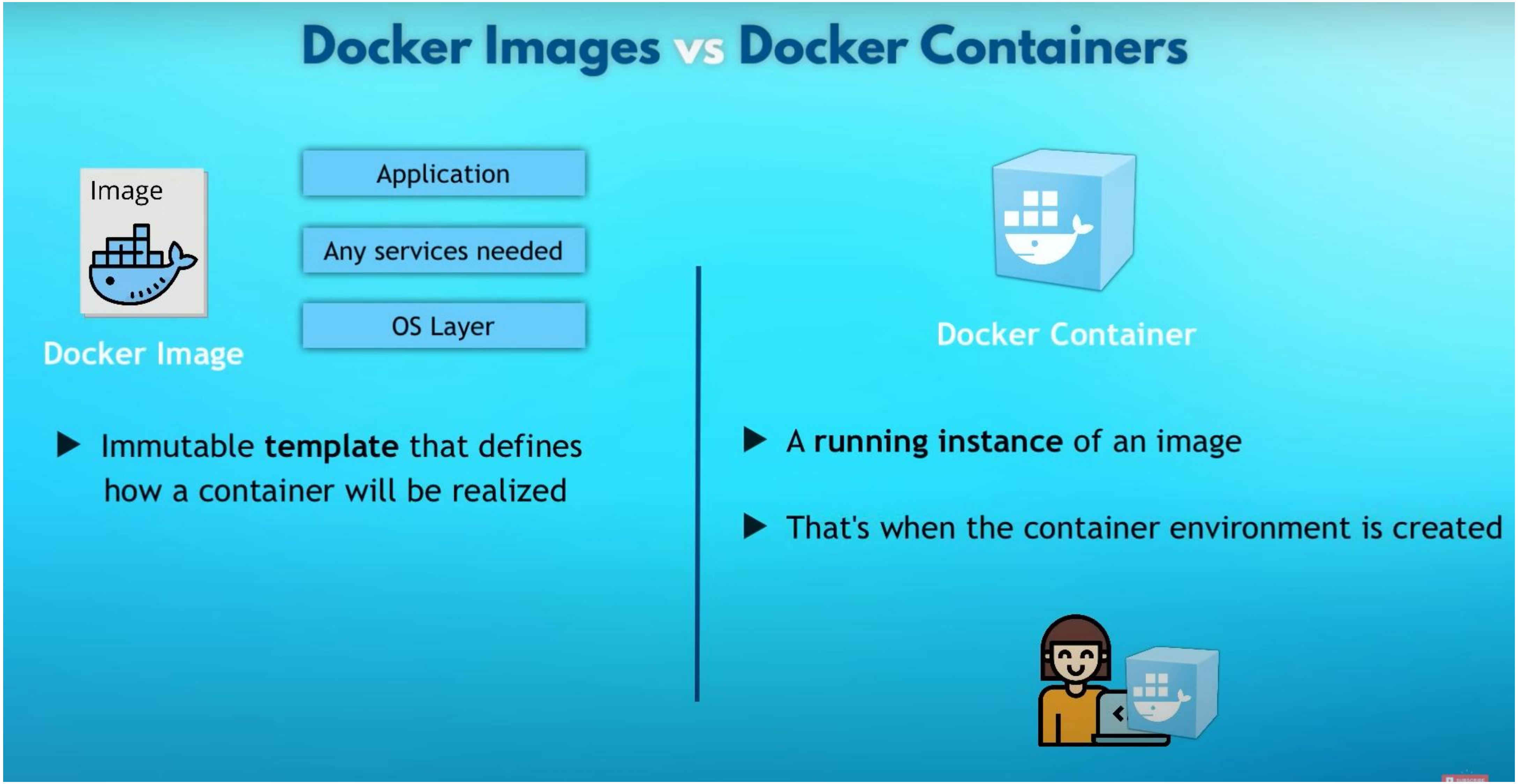
Any services needed

OS Layer

- ▶ An executable application artifact
- ▶ Includes app source code, but also **complete environment configuration**
- ▶ Add environment variables, create directories, files etc.

Docker Container

- ▶ Actually **starts the application**



Docker Images vs Docker Containers

Docker Image

Application

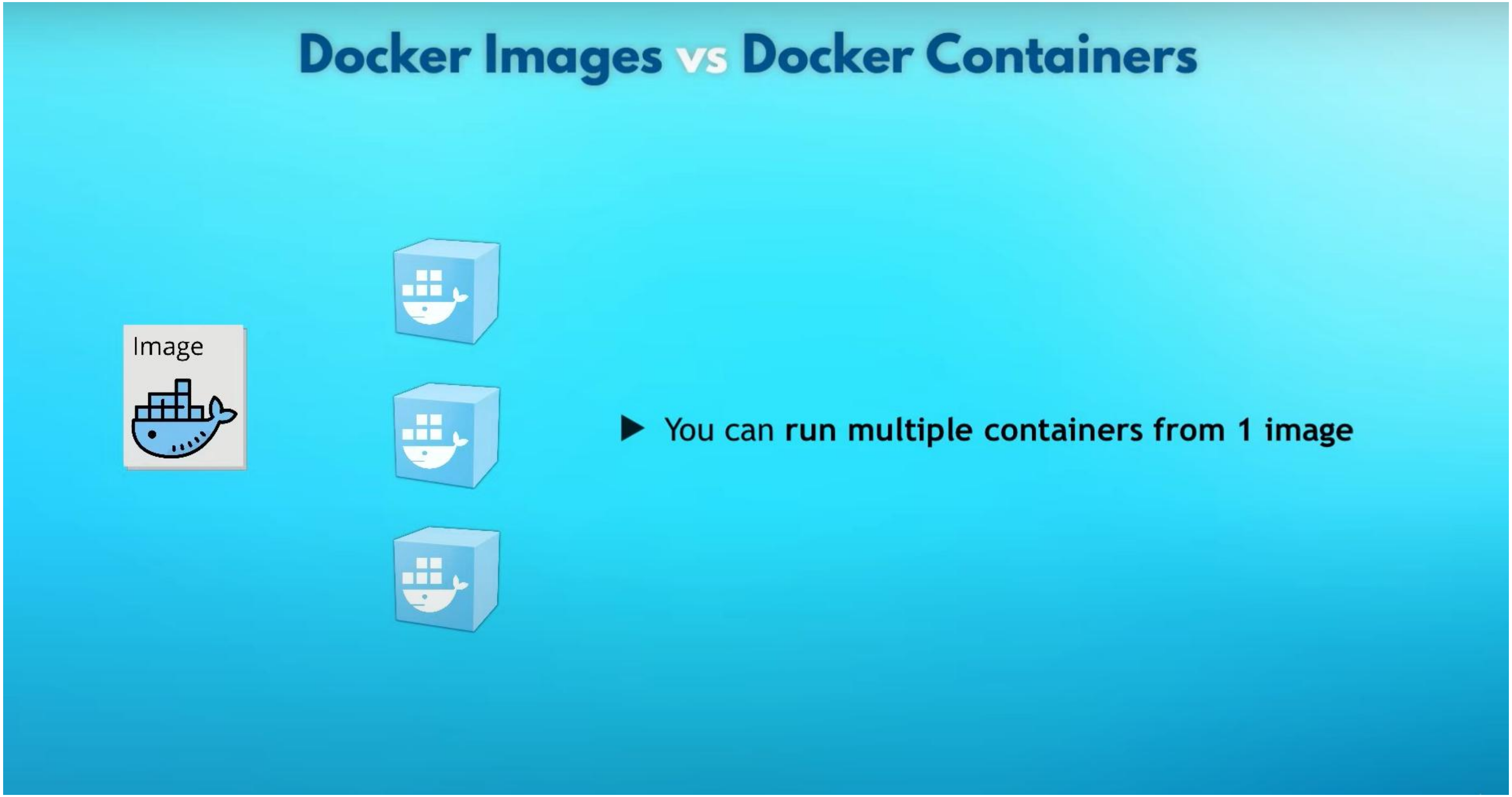
Any services needed

OS Layer

- Immutable **template** that defines how a container will be realized

Docker Container

- A **running instance** of an image
- That's when the container environment is created

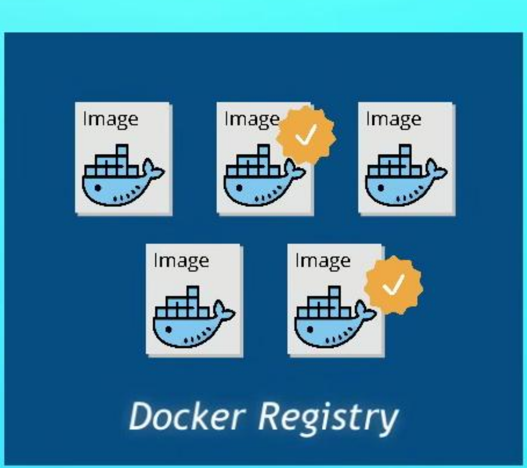


Docker Images vs Docker Containers

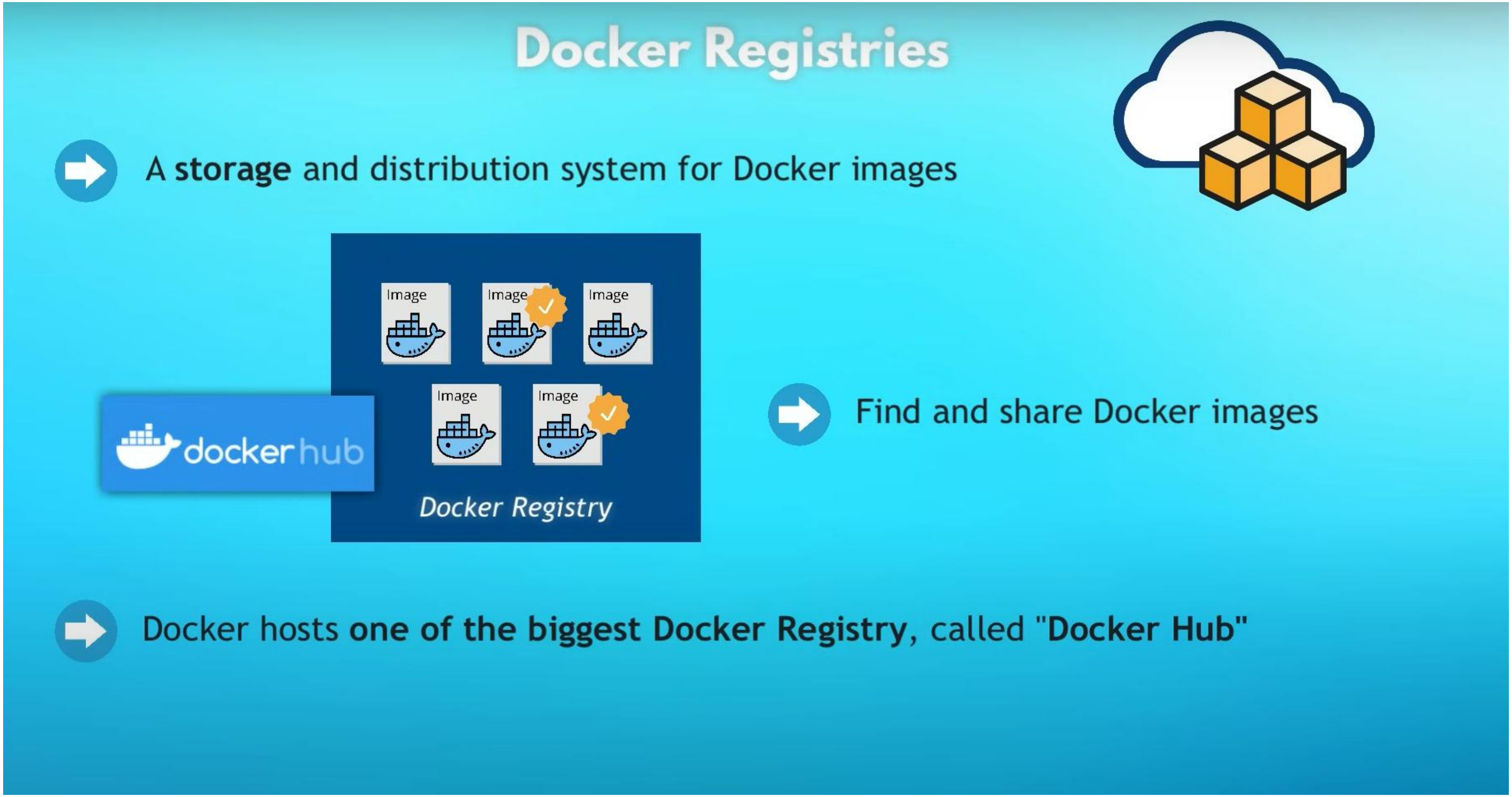
- You can **run multiple containers from 1 image**

Docker Registries

- A **storage** and distribution system for Docker images



- **Official images** available from applications like Redis, Mongo, Postgres etc.
- Official images are maintained by the software authors or in collaboration with the Docker community



Docker Registries

- A **storage** and distribution system for Docker images
- Find and share Docker images
- Docker hosts **one of the biggest Docker Registry**, called "**Docker Hub"**