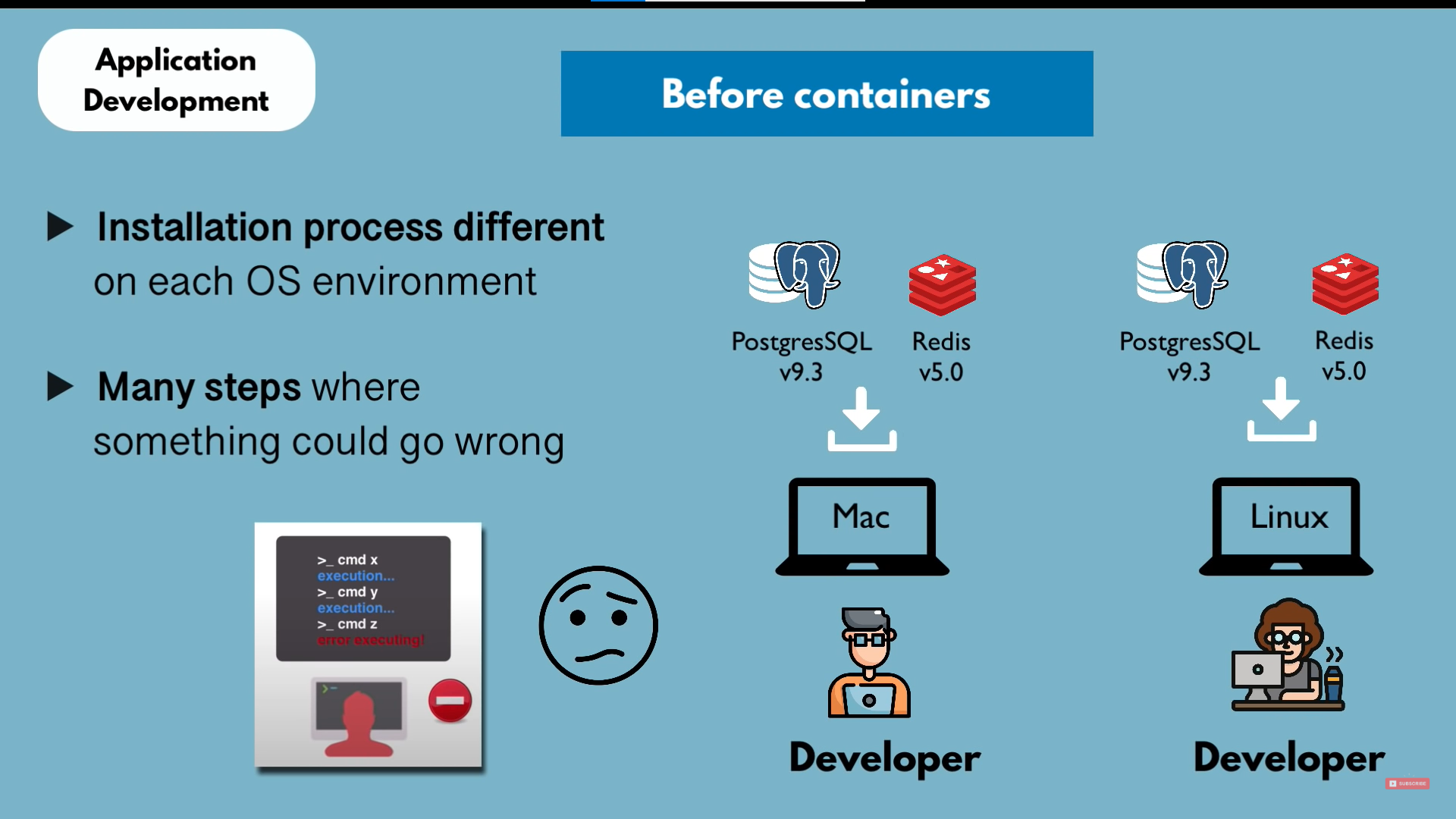
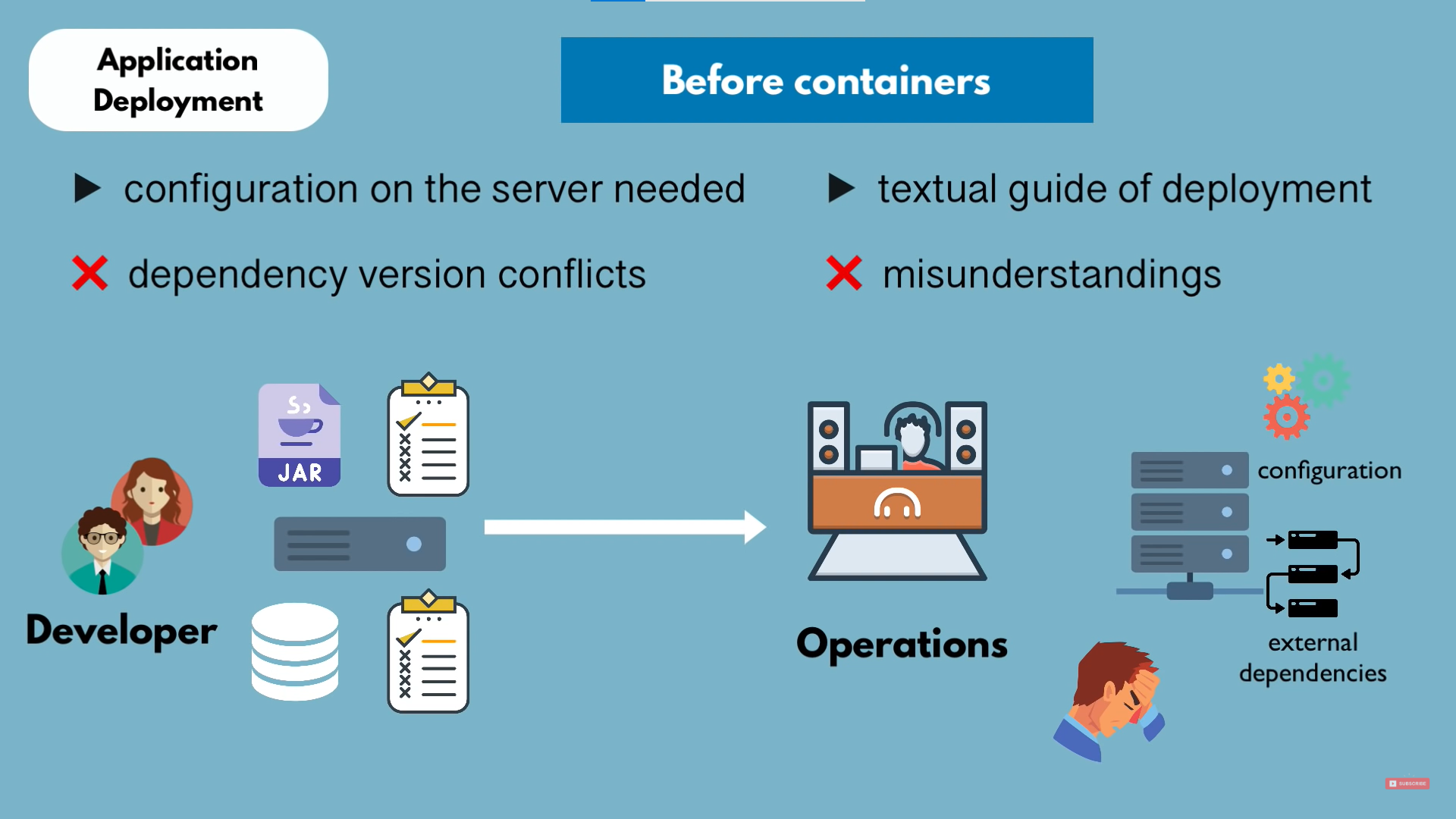
**WHAT’S IN THIS?**

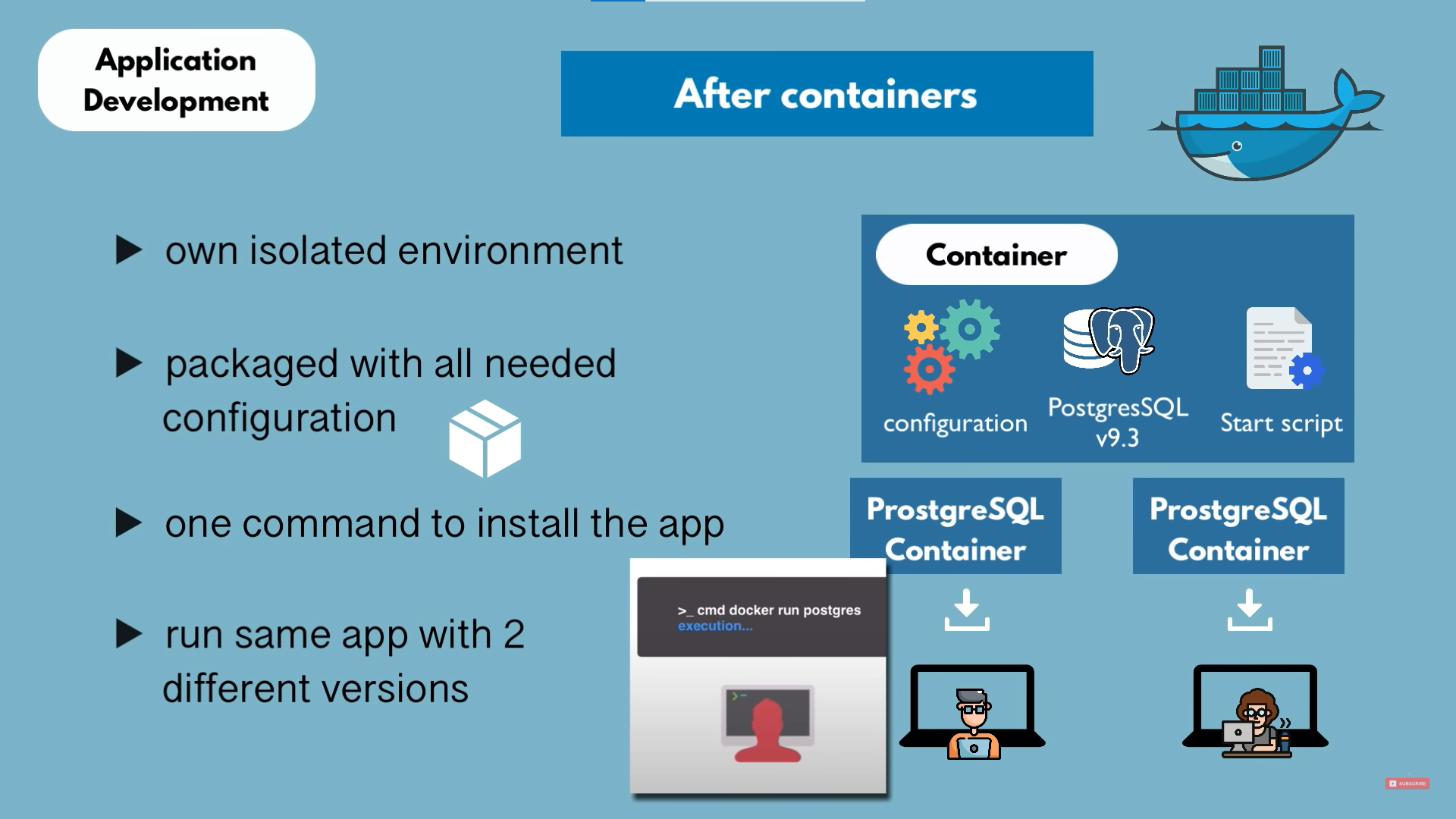


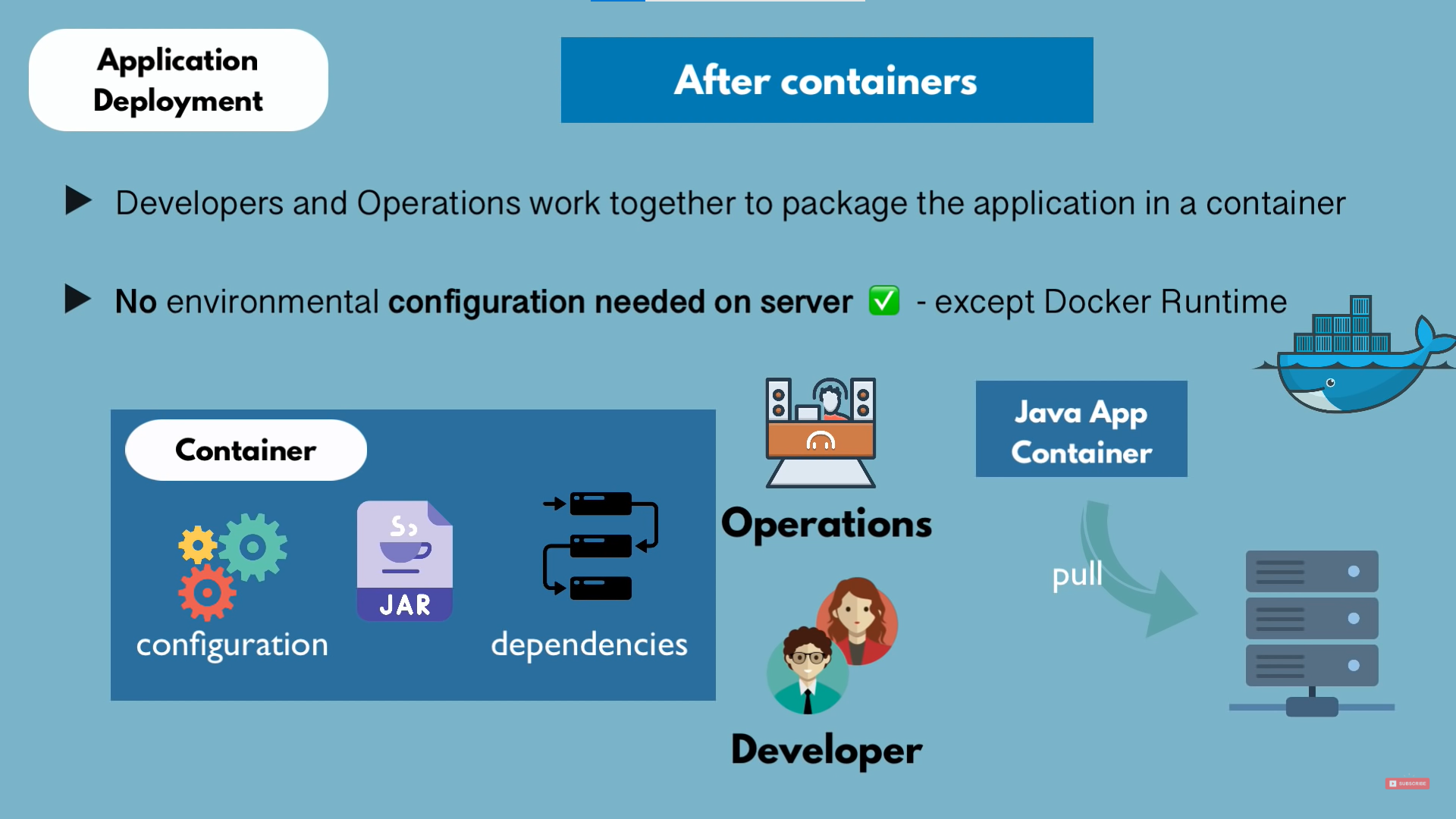
**BEFORE DOCKER**



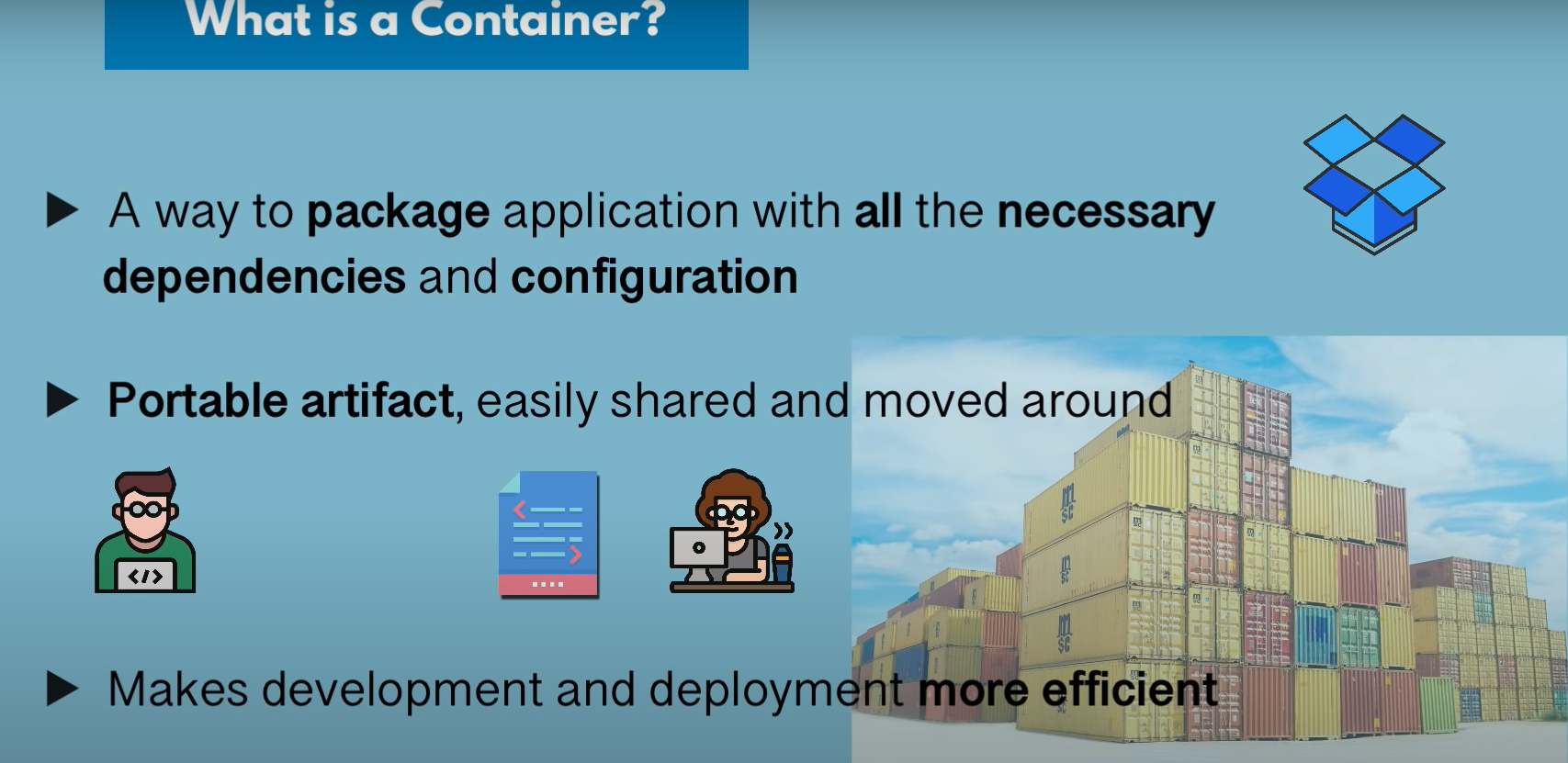


**AFTER DOCKER**

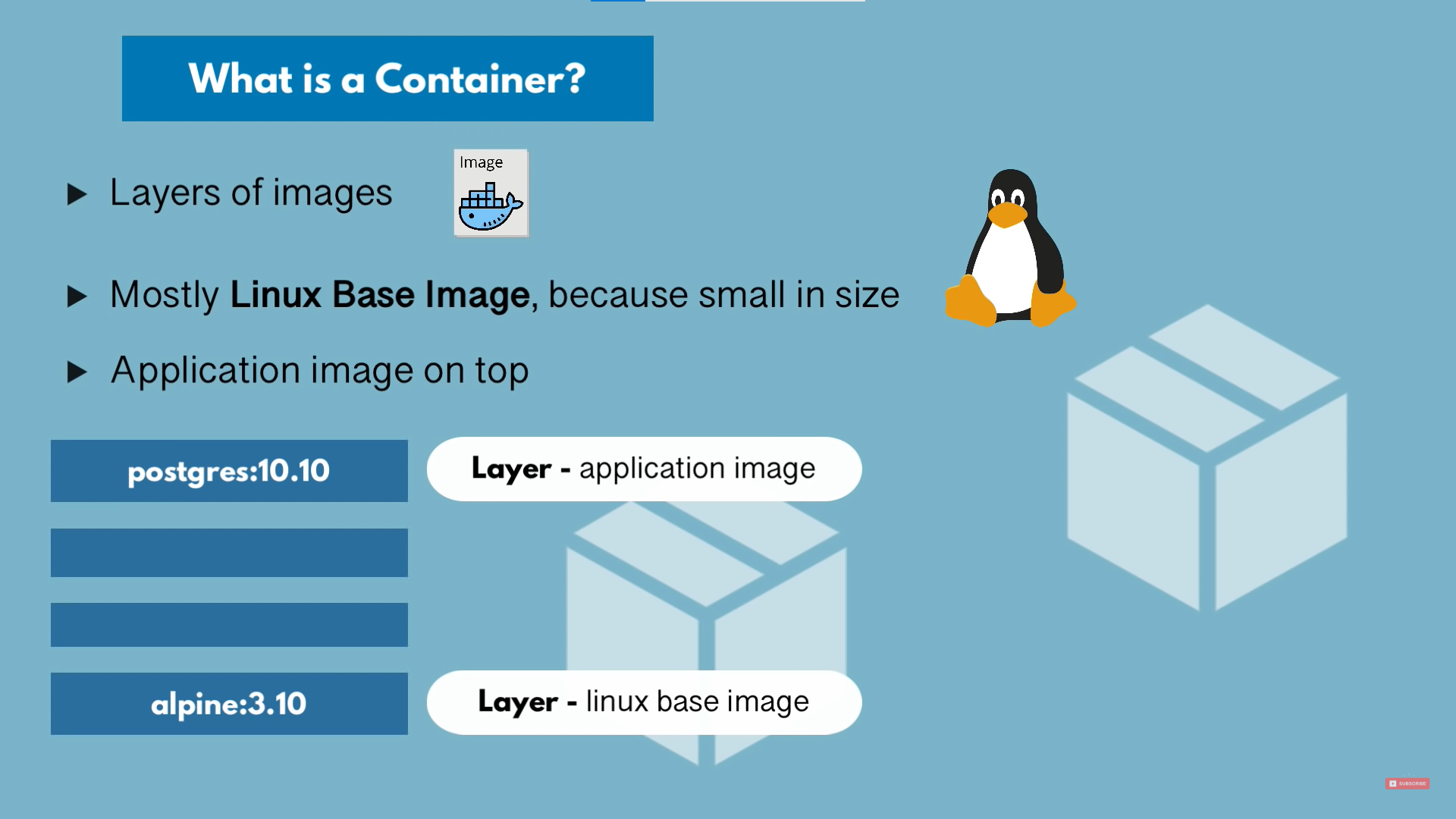


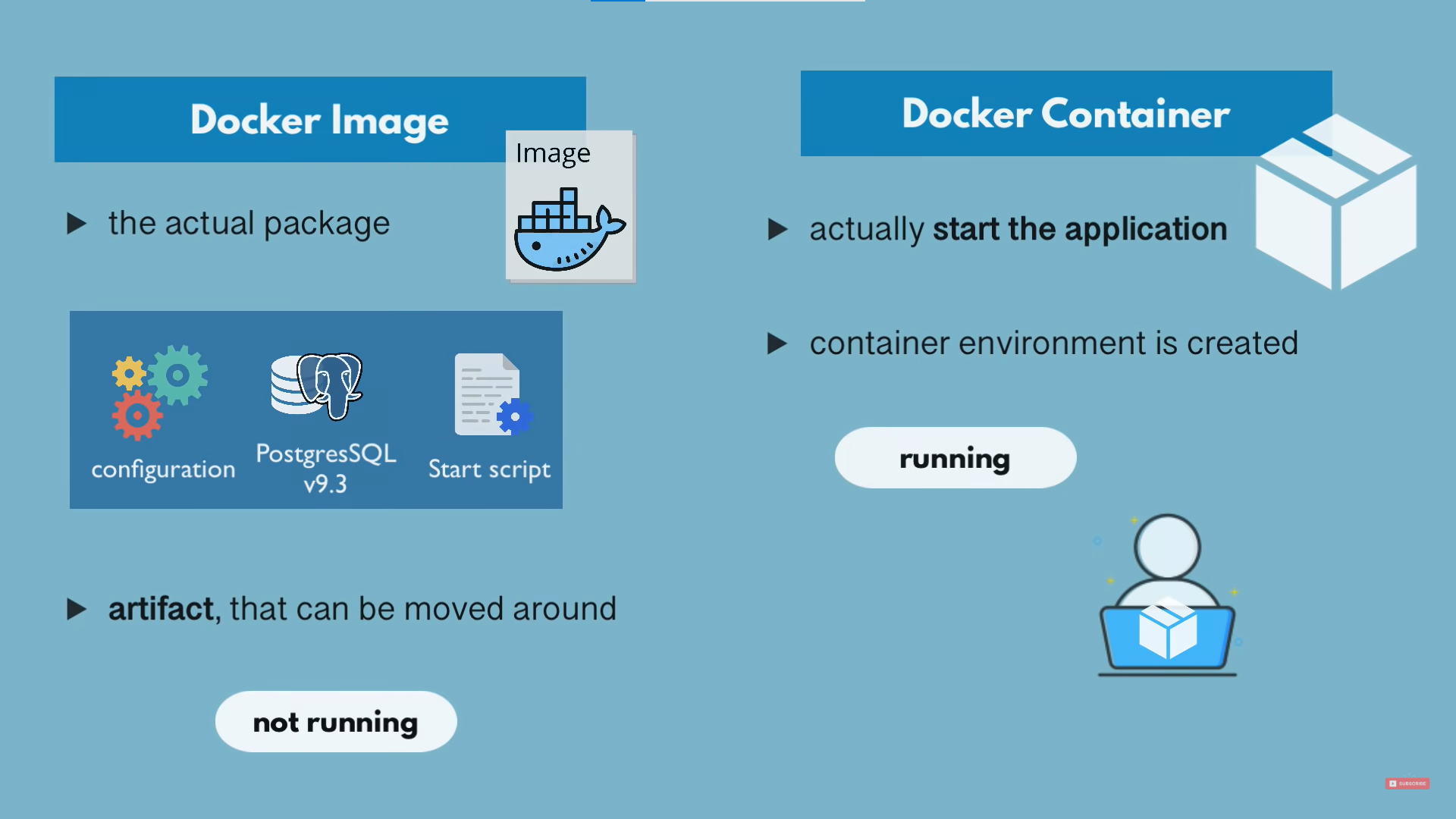


**WHAT IS A CONTAINER?**



Containers are images which run in the machine.

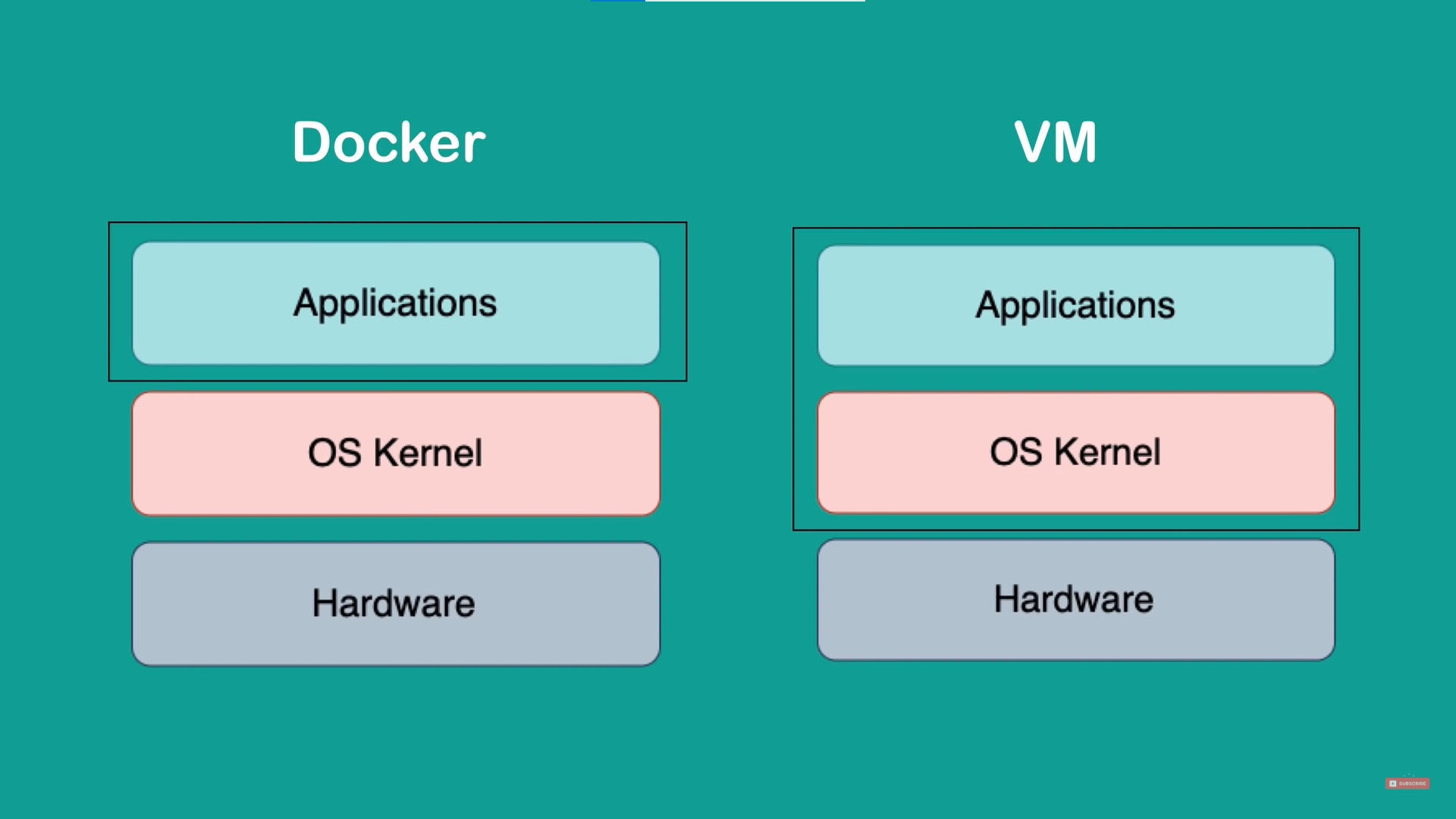




**DOCER INSTALLATION**

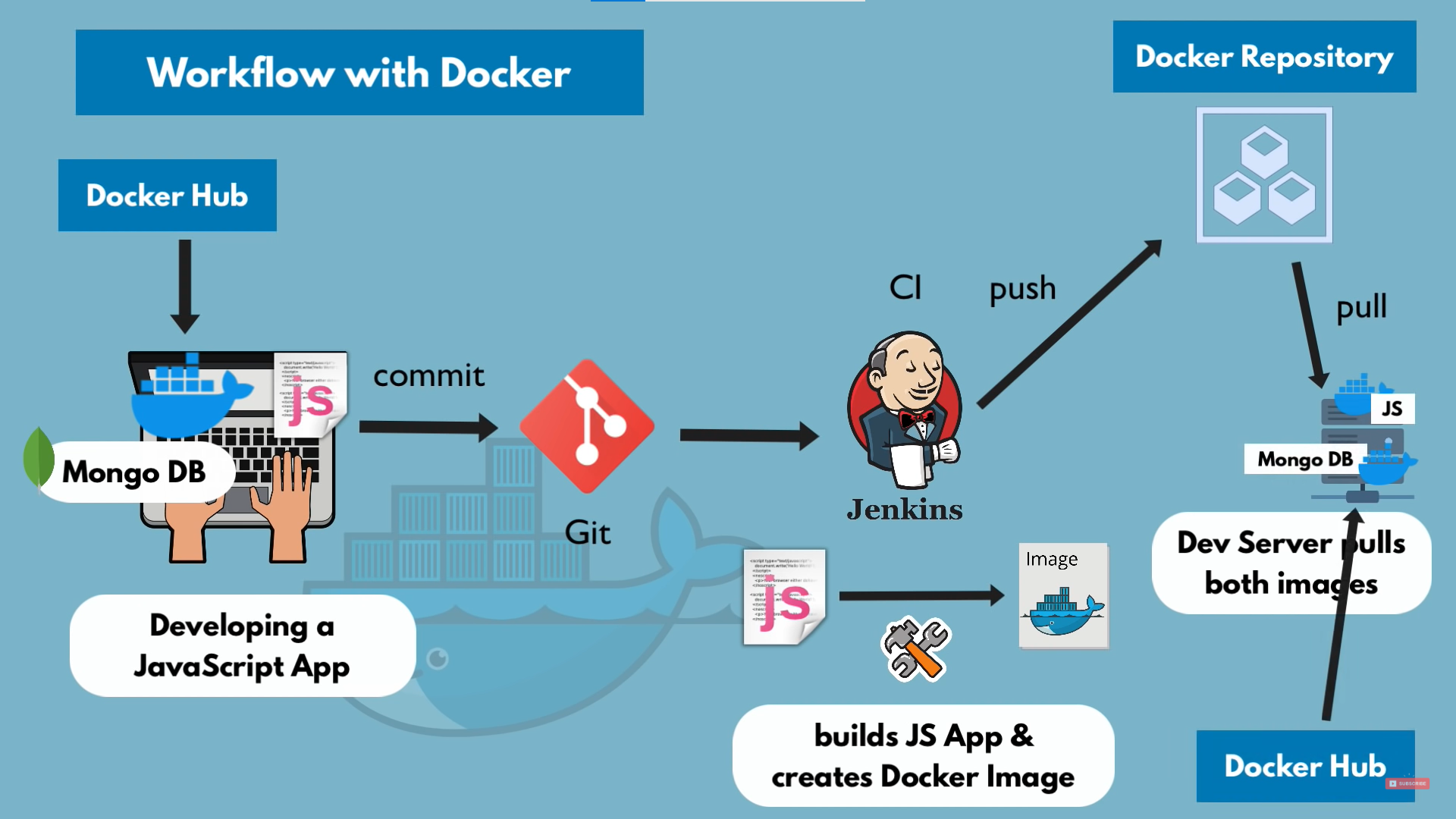
* Refer official Docker Documentation for better understanding
  + <https://docs.docker.com/>

**DOCKER vs. VM**



|  |  |
| --- | --- |
| **DOCKER** | **VM** |
| Size: Docker image size is smaller. | Size: VM size is larger. |
| Speed: Docker containers start and run much faster | Speed: VM takes time to start and boot up. |
| Compatibility: Older versions of Windows can’t host Linux based Docker images. | Compatibility: VM of any OS can run on any OS host. |

**WORKFLOW OF DOCKER**



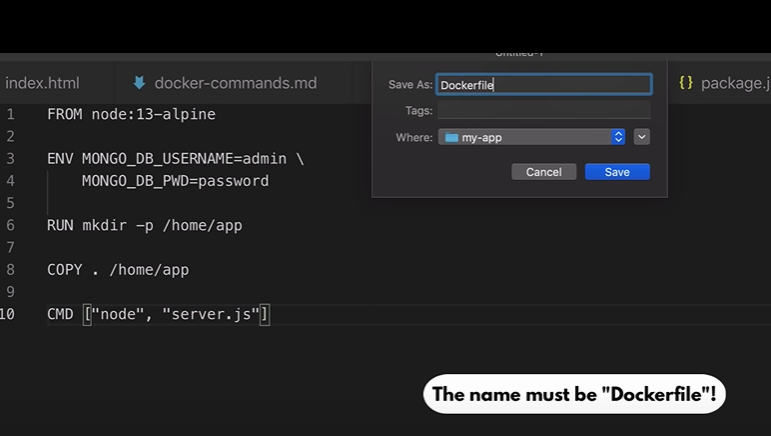
**DOCKER COMPOSE**

It is used to define and run multi container Docker application.

|  |  |
| --- | --- |
| Docker run command | Mongo-docker-compose.yaml |
| docker run -d \  --name mongodb \  -p 27017:27017 \  -e MONGO-INITDB\_ROOT\_USERNAME=admin \  -e MONGO-INITDB\_ROOT\_PASSWORD=admin \  --net mongo-network \  mongo (image name) |  |
| docker run -d \  --name mongo-express \  -p 8080:8080 \  -e ME\_CONFIG\_MONGODB\_ADMINUSERNAME=admin \  -e ME\_CONFIG\_MONGODB\_ADMINPASSWORD=admin \  -e ME\_CONFIG\_MONGODB\_SERVER=mongodb \  --net mongo-network \  mongo-express |

**CREATE A DOCKER FILE**

FROM node:13-alpine (The image container must have)



**TO BUILD AND IMAGE USING DOCKERFILE**

* + docker build -t <image\_name:version> <location>
  + docker build -t my-app:1.0 . (here the dockerfile location is the same directory as the cmd)
* To pull an image:
  + docker pull image\_name:version
  + docker pull postgres:9.6
* To run an image
  + docker run container\_name:version
  + docker run postgres:9.6

**\*\* If the image doesn’t exist in the system then it will automatically download the image and then run it.\*\***

* To see the container status
  + docker ps
* To stop a container
  + docker stop <Image ID>
* To start a container
  + docker start <Image ID>
* To search all the containers running or not running
  + docker ps -a
* To bind a container with a port
  + docker run -p6000:6379 <image\_name>
* To run a container in detached more
  + docker run -d <image\_name>
* To check for logs of a container
  + docker logs <image ID>
* To name a container name
  + docker run -d -port6001:6379 --name redis-older redis:4.0
* To get inside a container as a root user
  + docker exec -it <container ID> /bin/bash
* docker network ls
* To create a docker network
  + docker network create mongo-network
* docker run -d \

-p27017:27017 mongo \

-e MONGO\_INITDB\_ROOT\_USERNAME=admin \

-e MONGO\_INITDB\_ROOT\_PASSWORD=admin \

--name mongodb \

--net mongo-network \

mongo

* To run a docker container using docker compose file
  + docker-compose -f mongo.yaml up

**mongo.yaml is the name of the file with docker compose instructions**

* To stop a docker container using docker compose file
  + docker-compose -f mongo.yaml down
* To build a docker image using dockerfile
  + docker build -t <image\_name:version> <location>
  + docker build -t my-app:1.0 . (here the dockerfile location is the same directory as the cmd)
* To see the images
  + docker images
* To delete an image
  + docker rmi <image\_name>
* To delete a container
  + docker rm <container\_name>
* To get details of an image
  + docker image inspect <image\_name>
* To get details of a docker container
  + docker inspect <container\_name>
* To check logs of a container
  + docker logs <container\_name>
* To get the output of the commands use
  + DOCKER\_BUILDKIT=0 docker build…….
* To remove all the docker images
  + docker rmi $(docker images -aq)
* To remove all docker containers
  + docker rm $(docker pas -aq)
* To stop all containers
  + docker stop $(docker ps -aq)
* To maintain log rotate
  + docker run --log-opt max-size=10m --log-opt max-file=5 my-app:latest