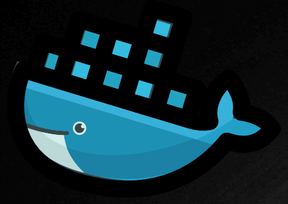
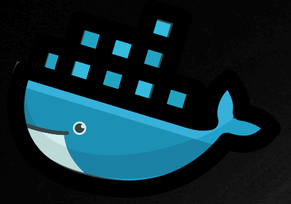


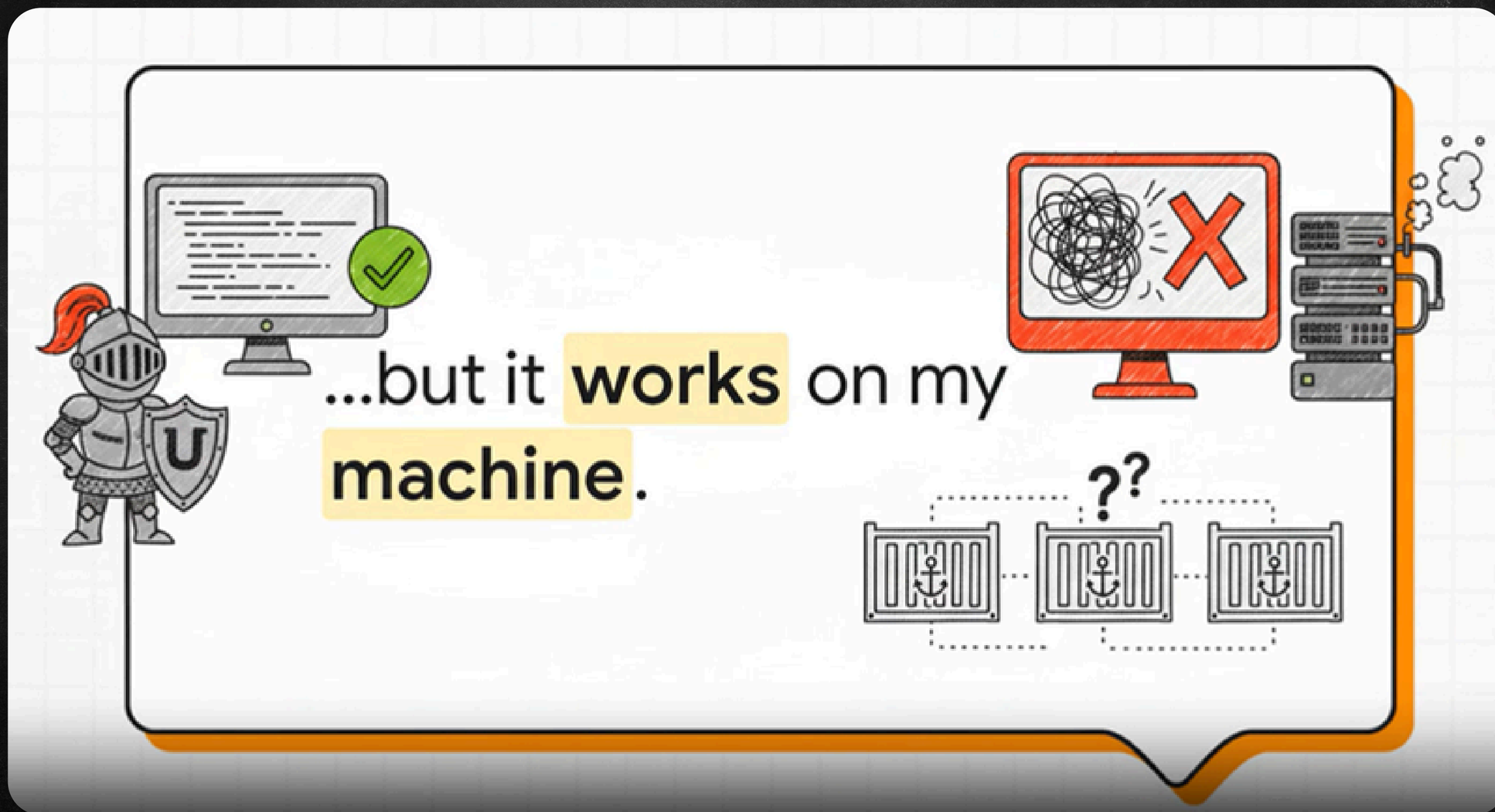
What is DOCKER?

Docker is a platform for packaging, distributing, and running applications inside lightweight, portable units called containers.

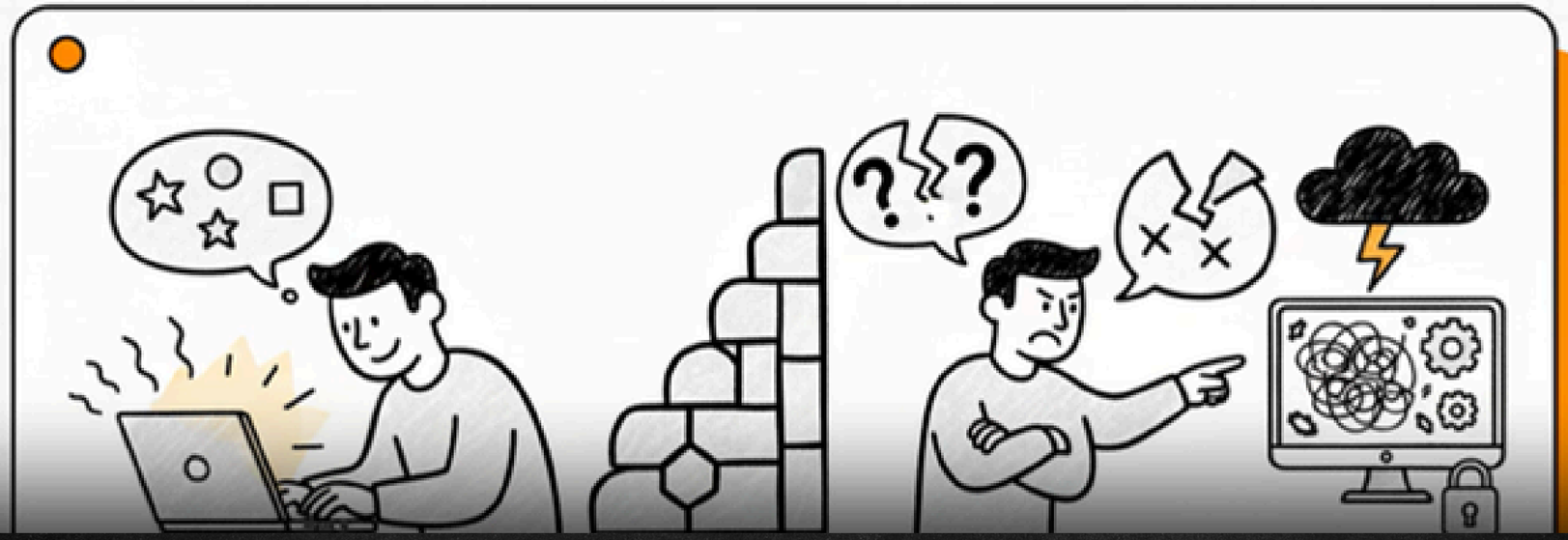


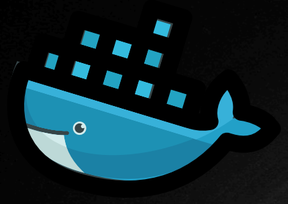
WHY DOCKER?

UMMMM.....


...but it works on my machine.
??

Docker: End "Works on My Machine"

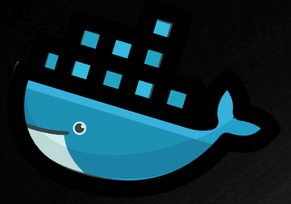




Packaging With DOCKER

Docker Container includes:

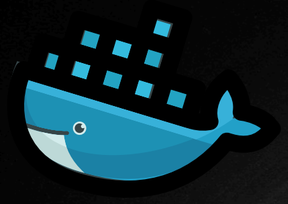
- Your code
- Python
- Libraries
- Dependencies
- Environment variables
- OS-level settings



DOCKER CONTAINER

A container is a lightweight isolated box that runs your app with:

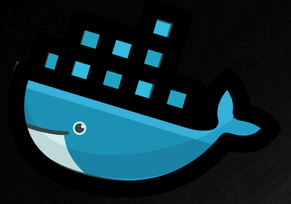
- Its own file system
- Its own dependencies
- Its own process
- But shares the host OS kernel



DOCKER VS Virtual Machines

A Virtual Machine is a full computer inside your computer.
It includes:

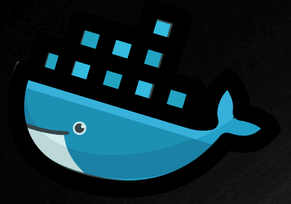
- Full OS (Linux/Windows)
- Kernel
- RAM
- CPU
- Drivers
- Applications



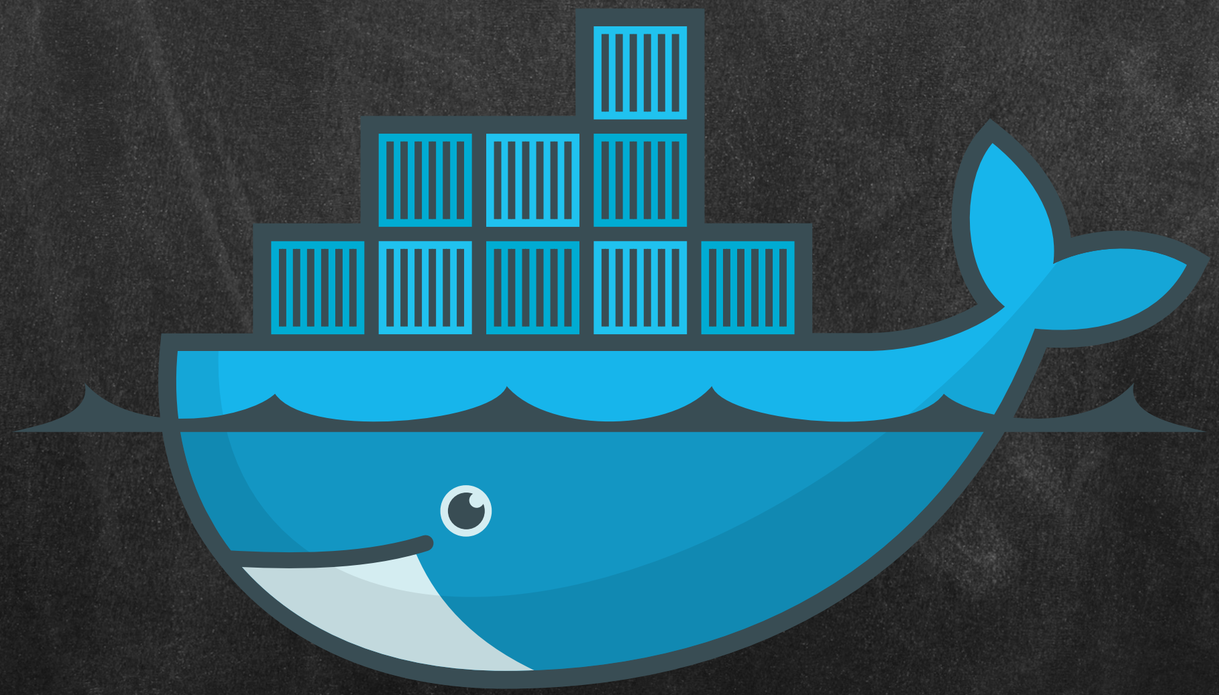
DOCKER BACKBONE

IMAGE

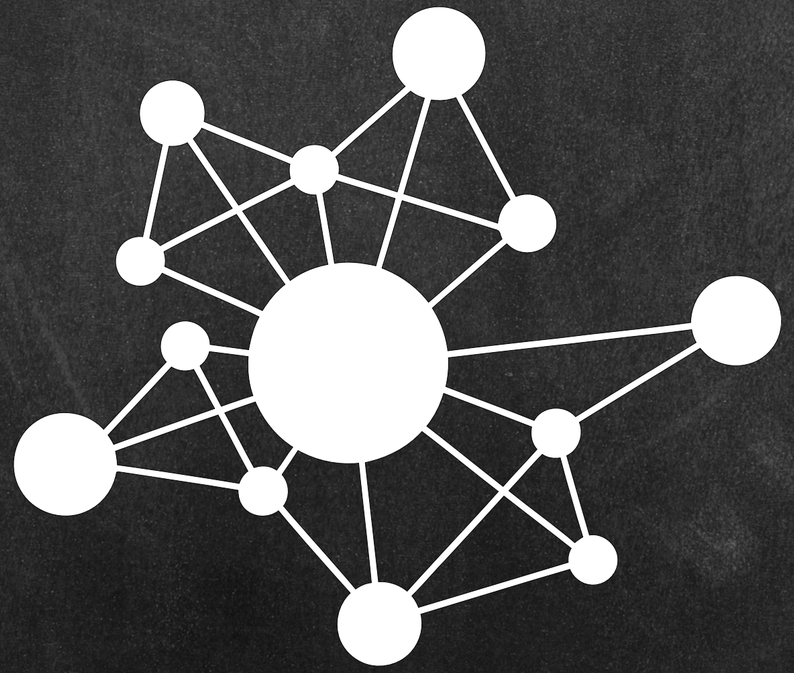
CONTAINER

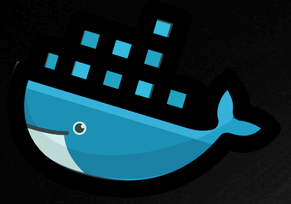


DOCKER INSTALLATION



DOCKER NETWORKS

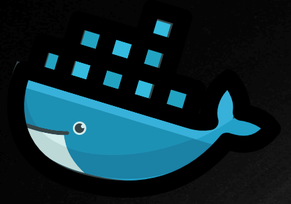




DOCKER NETWORKS

Whenever we create a Docker container, Docker automatically assigns it to a network called the `bridge` network. Now, let's say you have multiple containers running, each container is isolated from each other by default.

This means that they cannot communicate with each other because they are on different networks.



DOCKER COMPOSE

In the previous chapter, we learnt how to make multiple containers talk to each other using Docker Networks. However, managing multiple containers individually can be really hectic and time-consuming. What if I tell you that there is a way to manage multiple containers together as a single unit?

Yes, that's where `Docker Compose` comes into the picture!

DOCKER STORAGE