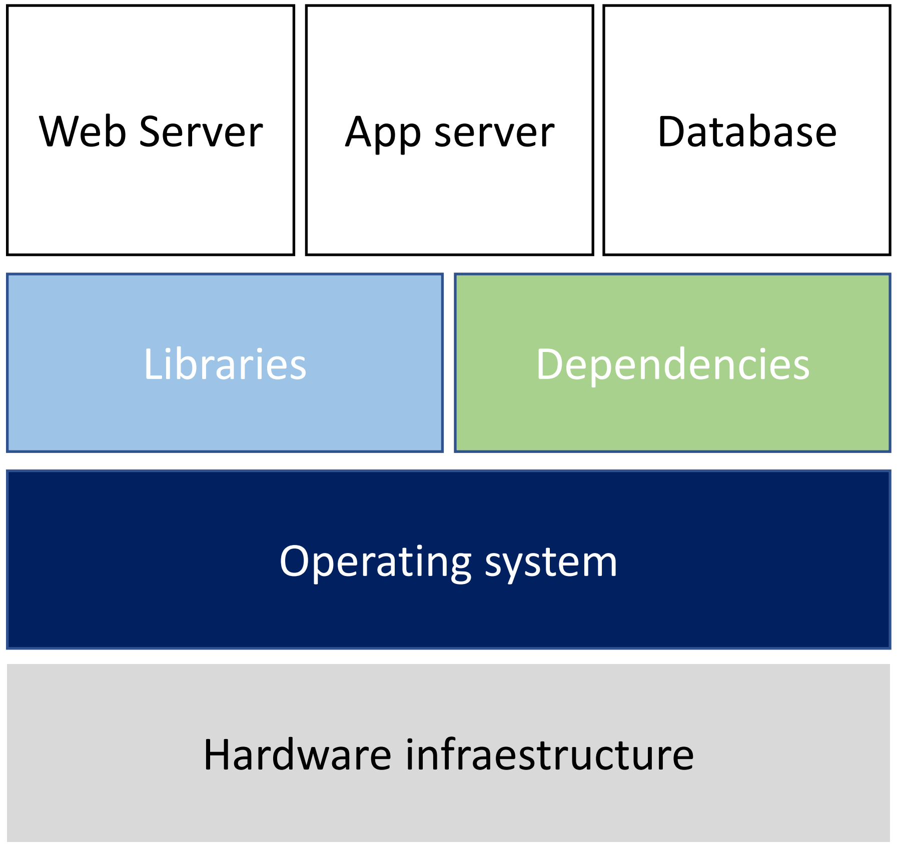
Docker Fundamentals

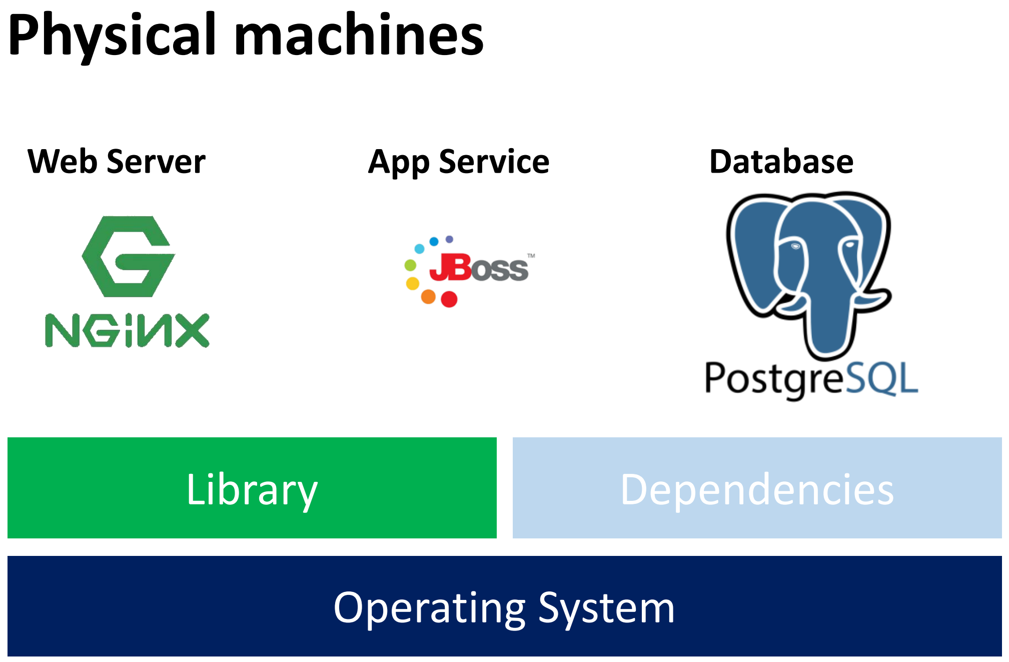
Ing. Pablo Campo

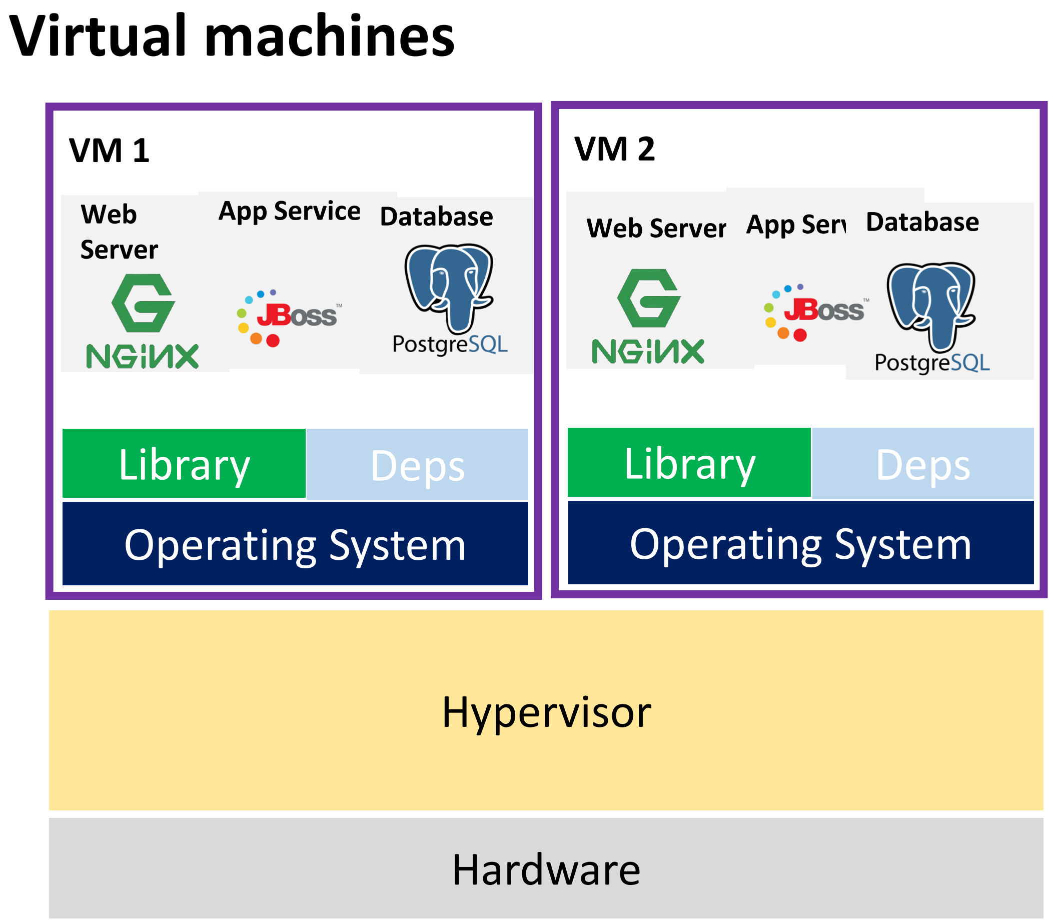
What problems we have with the traditional infra?

- Traditional approach
- Installation and configuration
  - Time consuming
  - Need to perform install/configs on every server and every enviroment (dev, qa, staging, production)
- Compatibility and dependency
  - Need to keep resolving issues relate to libraries and dependences.
- Inconsistencies across environments
  - Very hard to track changes across DEV/QA/Stagging and Prod environment and they end up with inconsistencies.
- Operational Support
  - Need more resources to handle operational issues on day to day basis (Server Support on hardware or software, and patching releases)
- Developer environments
  - When a new developer joins the teams, time it takes to provision his development environment in traditional approach is time talking.

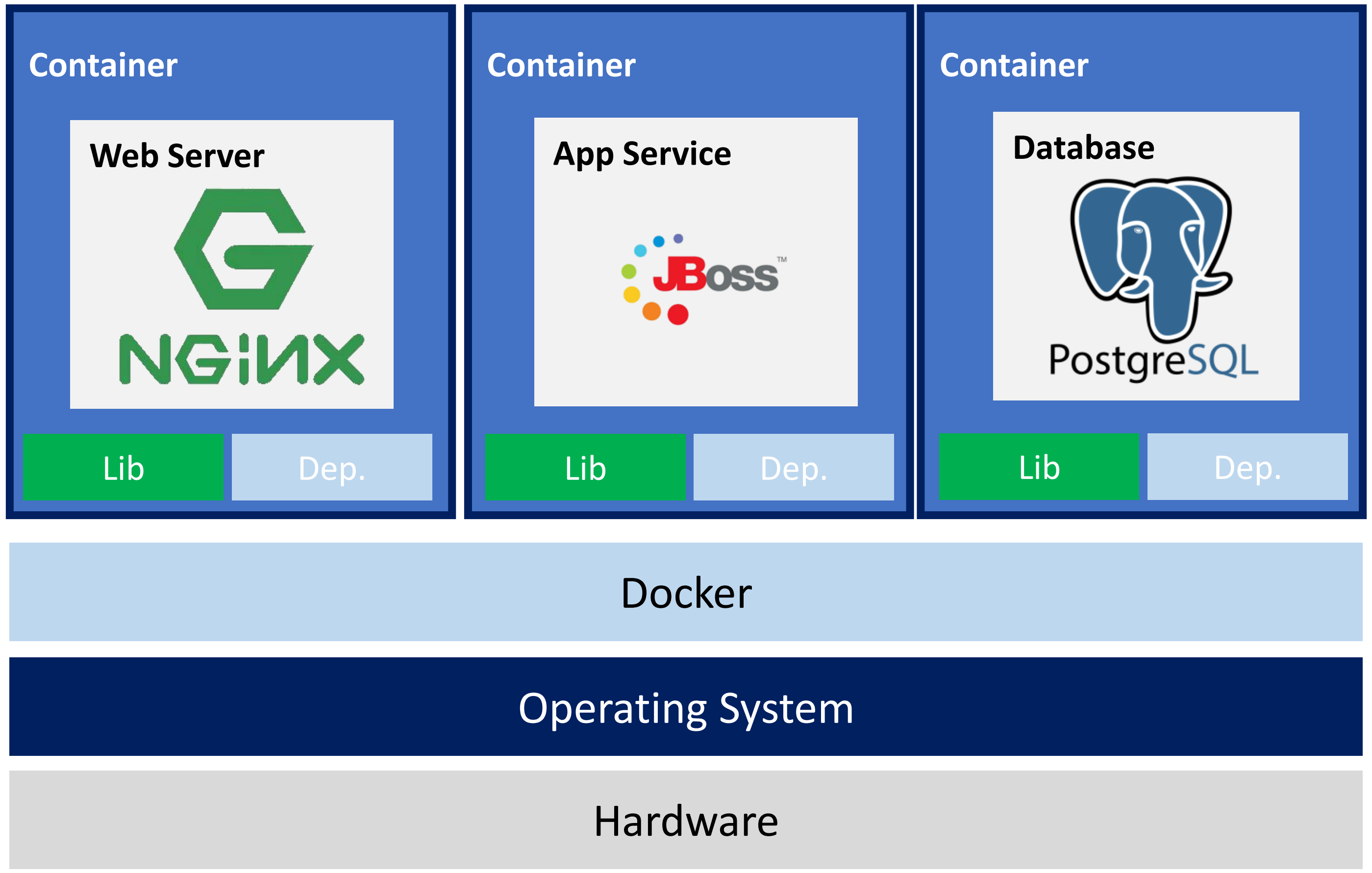


Traditional architectures

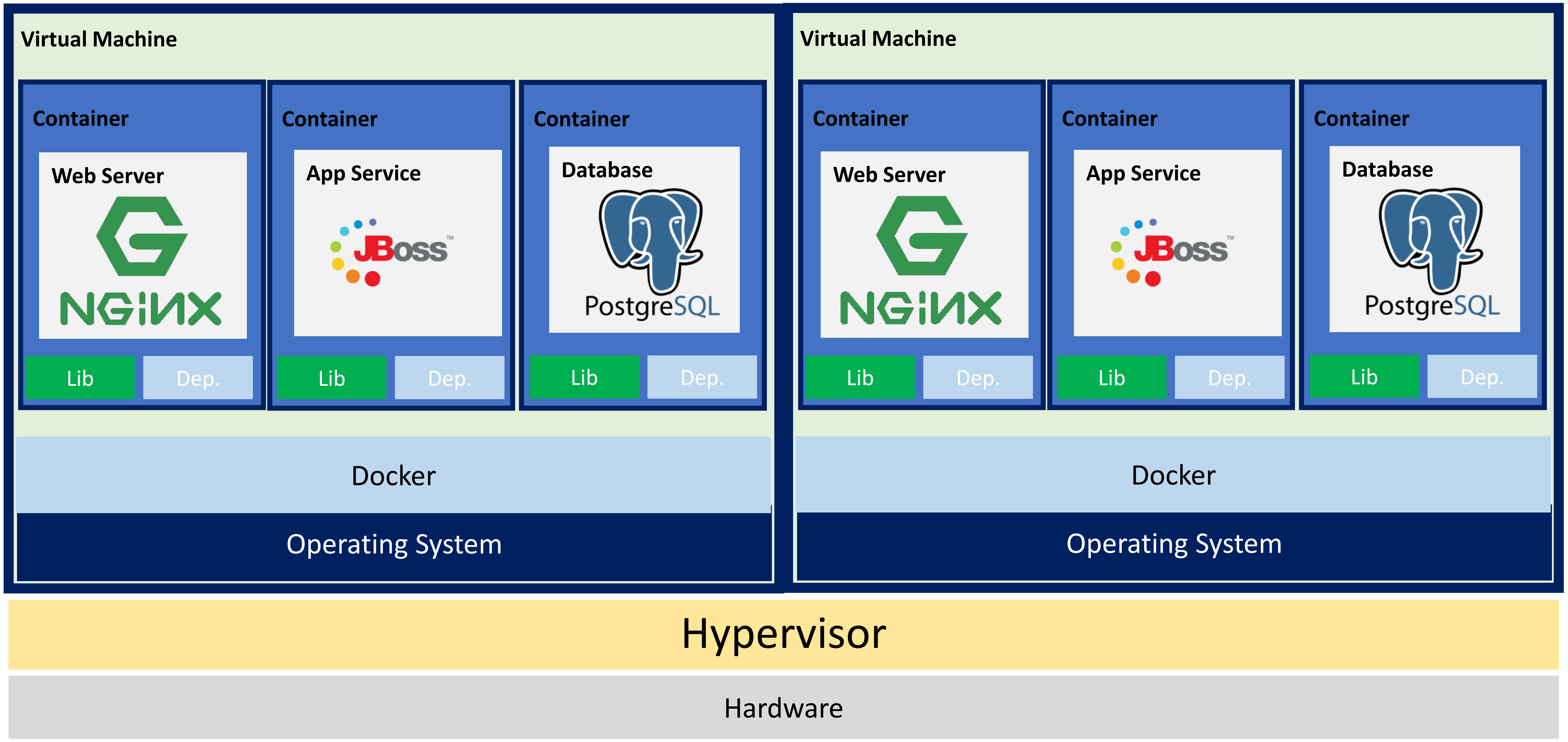




Physical machines with dockers

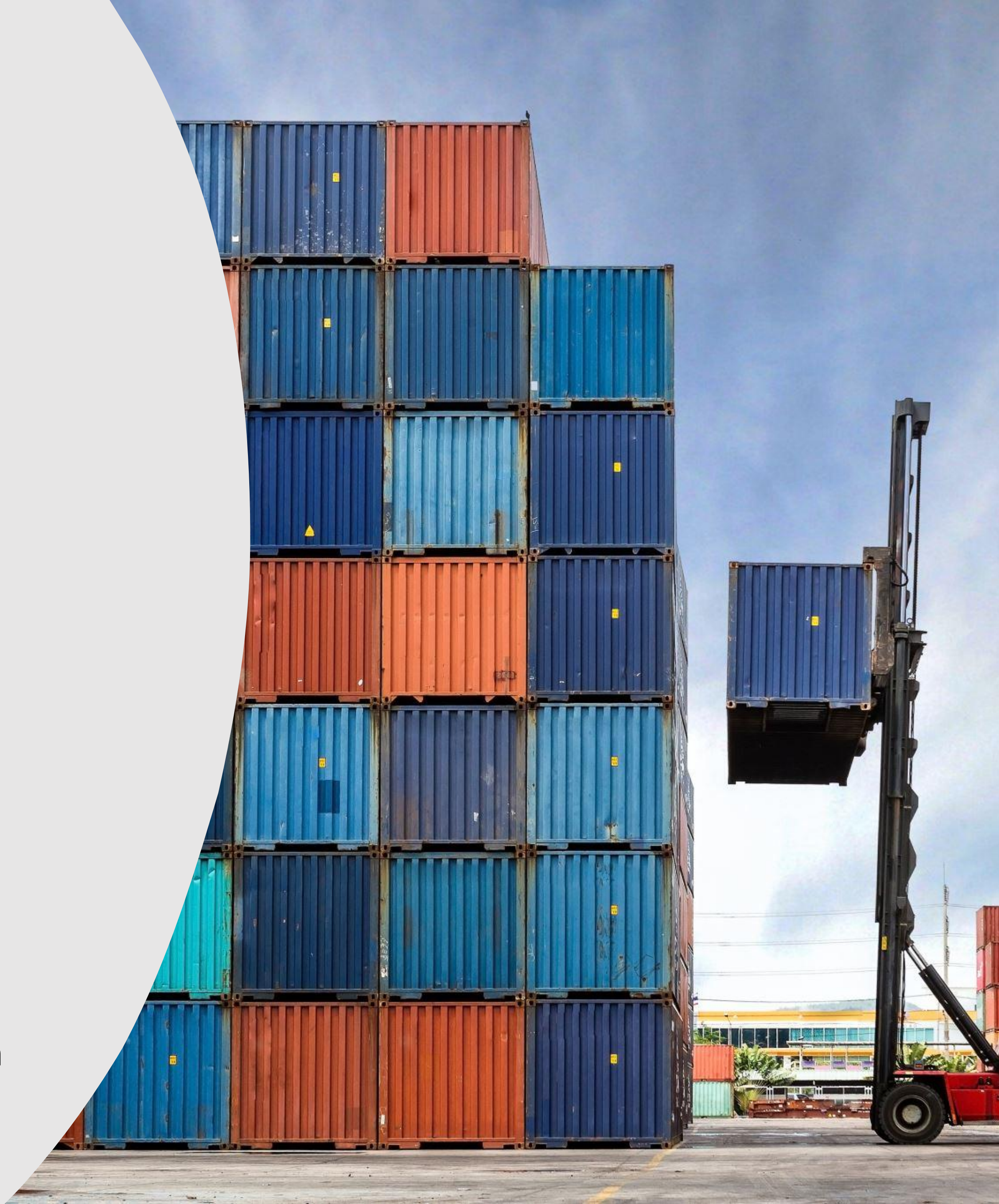


Virtual machines with docker

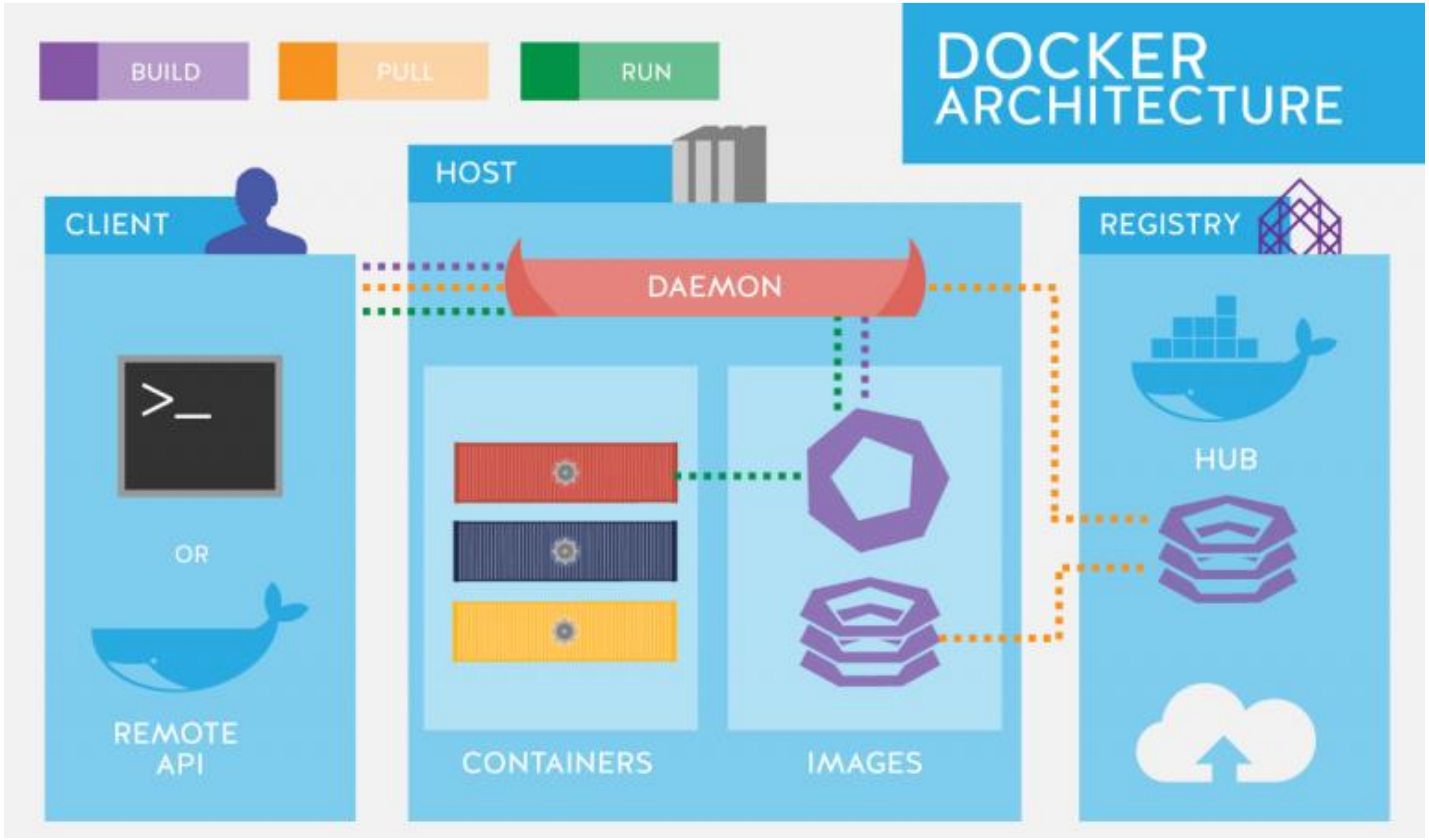


Why containers?

- **Flexible**: Even the most complex application can be containerized.
- **Lightweigh**: Container laverage and share a host kernel, making them much more efficient in term off system resources tan virtual machines.
- **Portable**: yo can build localy, deploy to the Cloud and run anywhare.
- **Loosely coupled**: Containers are highly self sufficient and encapsulated, allowing you to replace or upgrade without disrupting others.
- **Scalable**: You can increase and automatically distribute container replicas across a datacenter.
- **Secure**: Containers apply aggressive constraints and isolations to process without any configuration required on the part of the user.

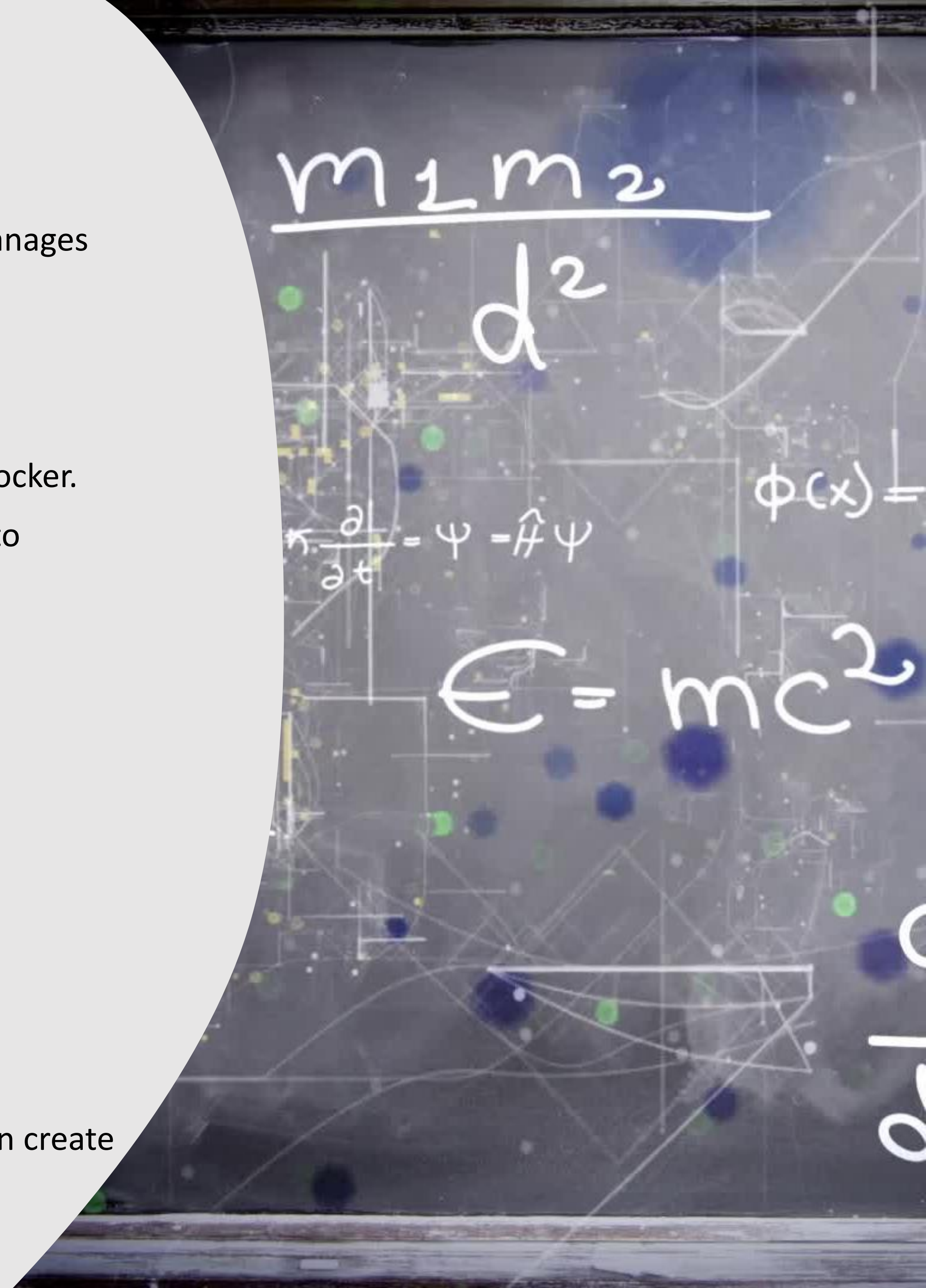


Docker architecture



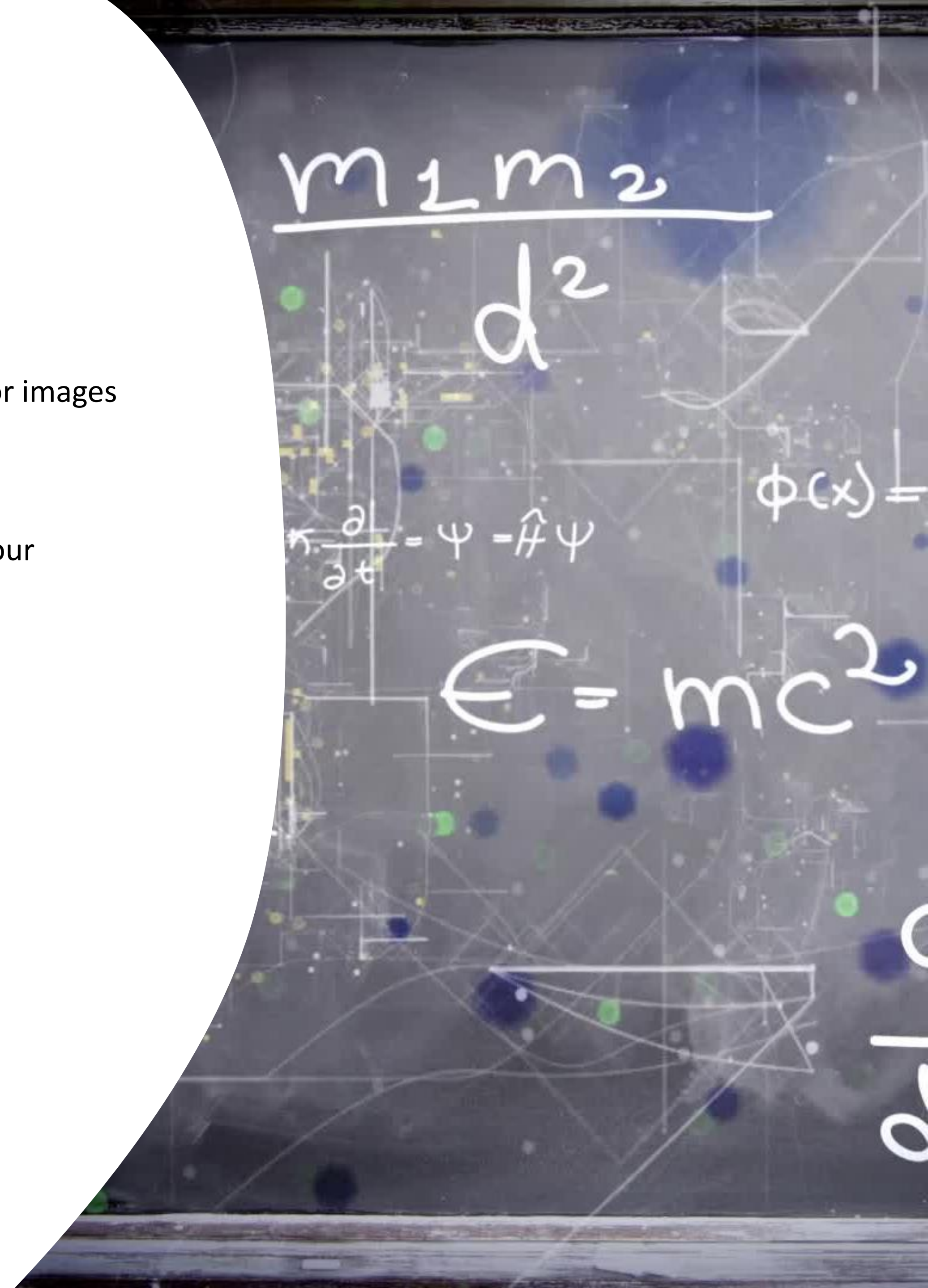
Docker terminology

- **Docker Daemon**: The Docker daemond (dockerd) listens for Docker Api request and manages Docker objects such as images, containers, networks and volumes.
- **Docker client**:
  - Docker Client can be present on either Docker Host or any other machine.
  - Docker Client (docker) is the primary way that many Docker users interact with docker.
  - When you use commands such as docker run, the client sends these commands to dockerd (docker daemond), which carries them out.
  - The docker command uses Docker Api
  - The docker client communicate with more than one Daemon.
- **Docker Images**:
  - An image is a read-only template with instructions for creating Docker container.
  - Often an image is based on another image, with some additional customization.
- **Docker Containers**:
  - A cointainer is a runnable instance of an image.
  - We can create, move, delete, stop a container using Docker Api or Cli.
  - We can connect a container to one or more networks, attach storage to it, or even create a new image based on its currente state.



Docker terminology

- **Docker registry**:
  - A Docker Registry stores Docker images.
  - Docker Hub is a public registry that anyone can use, and docker is configured to look for images on docker hub by default.
  - We can even run our own private registry.
  - When use docker pull or docker run commands, the required images are pulled from our configured registry.
  - When use docker push command, our image is pushed to our configured registry.



Thanks