KENNESAW STATE
UNIVERSITY

Module 4:
Containers and Dockers

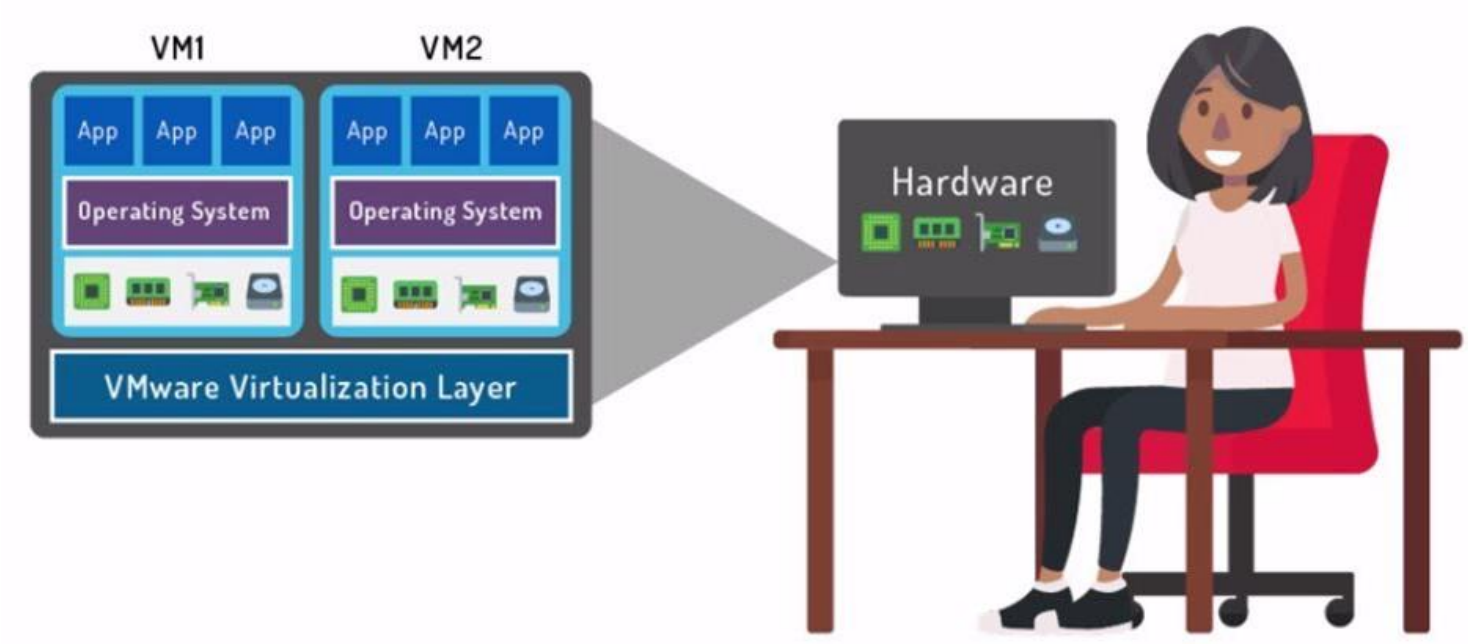
Dr. Maria Valero

Agenda

- What is virtualization
- Advantages of Virtualization
- Containers
- Docker and Docker Vocabulary
- Docker Architecture

Virtualization

- In computing, virtualization is the act of creating a virtual version of something, including virtual computer hardware platforms, storage devices, and computer network resources.

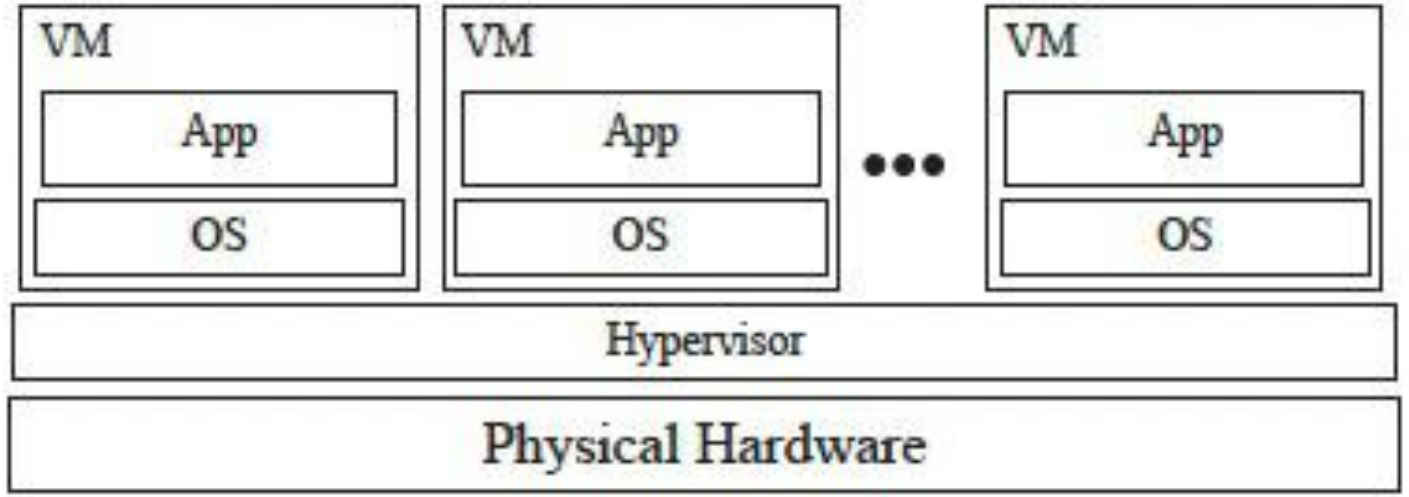


Advantages of Virtualization

- Minimize hardware costs
  - Multiple virtual servers on one physical hardware
- Easily move VMs to other data centers
  - Provide disaster recovery. Hardware maintenance
- Conserve Power
  - Free up unused physical resources
- Easier automation
  - Simplified provisioning/administration of hardware and software
- Scalability and Flexibility: Multiple operating systems

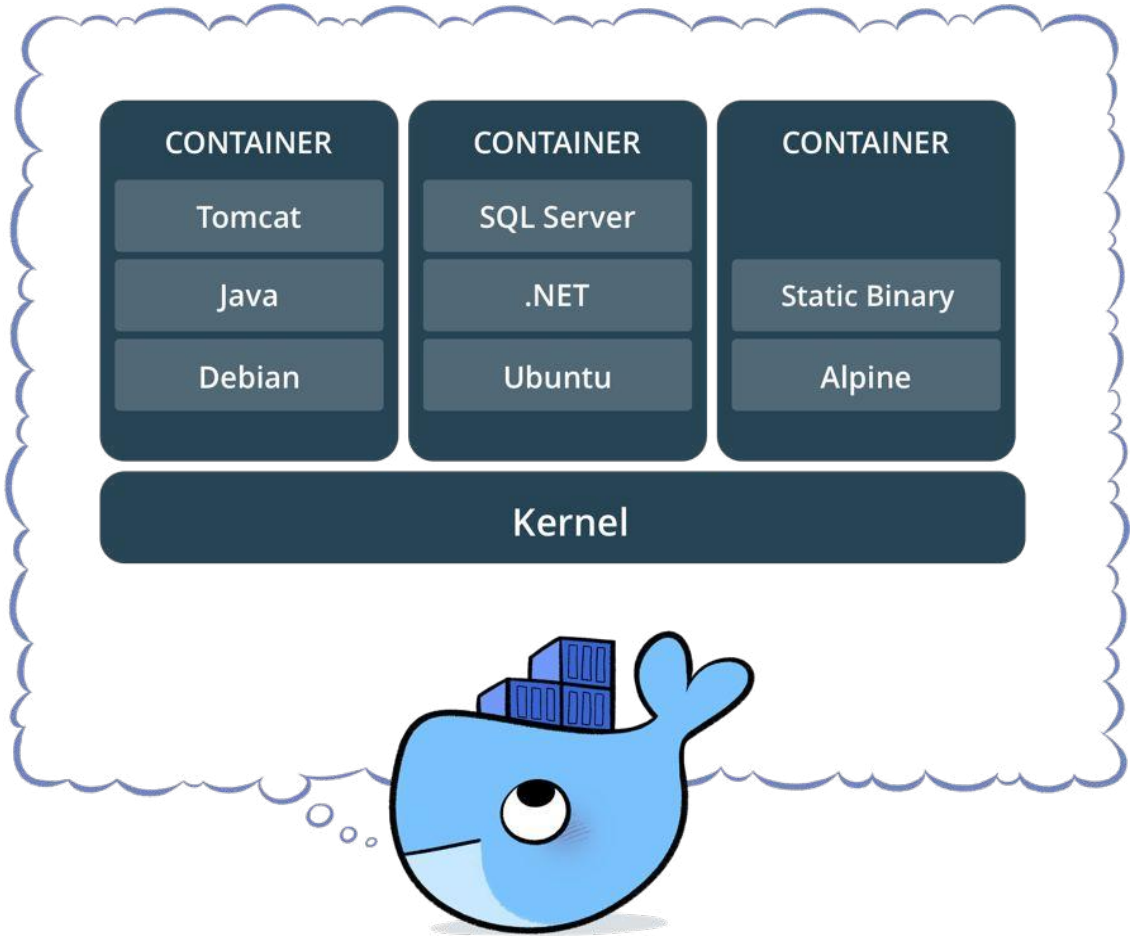
Problems of Virtualization

- Each VM requires an operating system (OS)
  - Each OS requires a license
  - Each OS has its own compute and storage overhead
  - Needs maintenance, updates

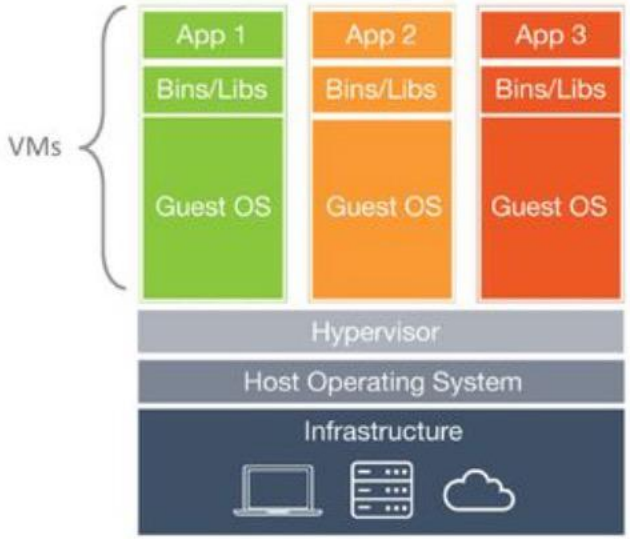


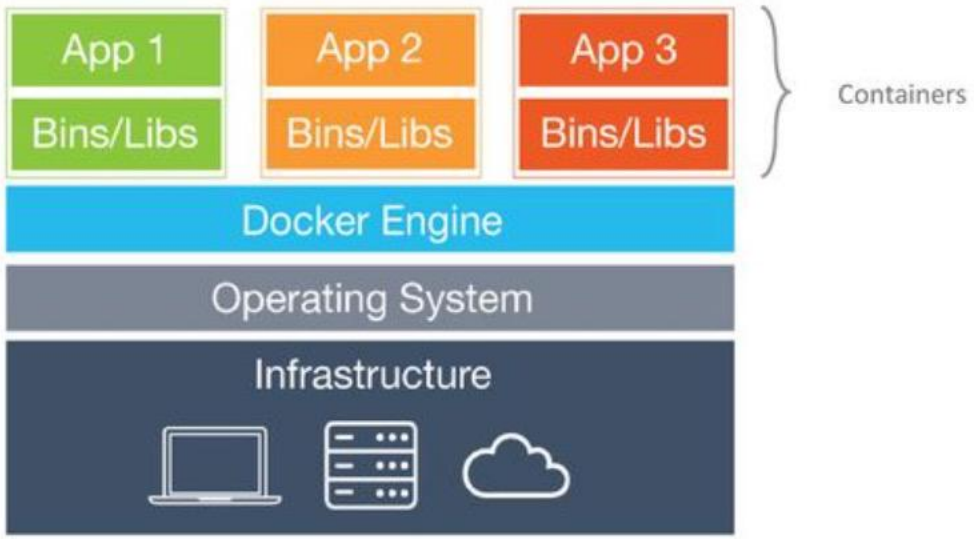
Solution: Containers

- Standardized packaging for software and dependencies
- Isolate apps from each other
- Share the same OS kernel
- Works for all major Linux distributions
- Works on other Operating Systems



Virtual Machines vs. Containers



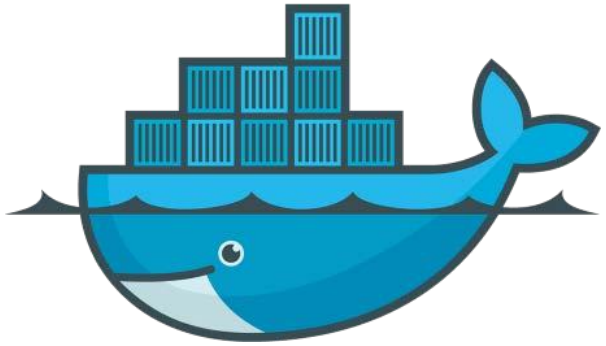


- Each VM includes the app, the necessary binaries, and the libraries and ***an entire guest operating system***

- Containers contain the app and all its dependencies but ***share the kernel*** with other containers
- Run as an isolated process
- ***Not tied*** to any specific infrastructure

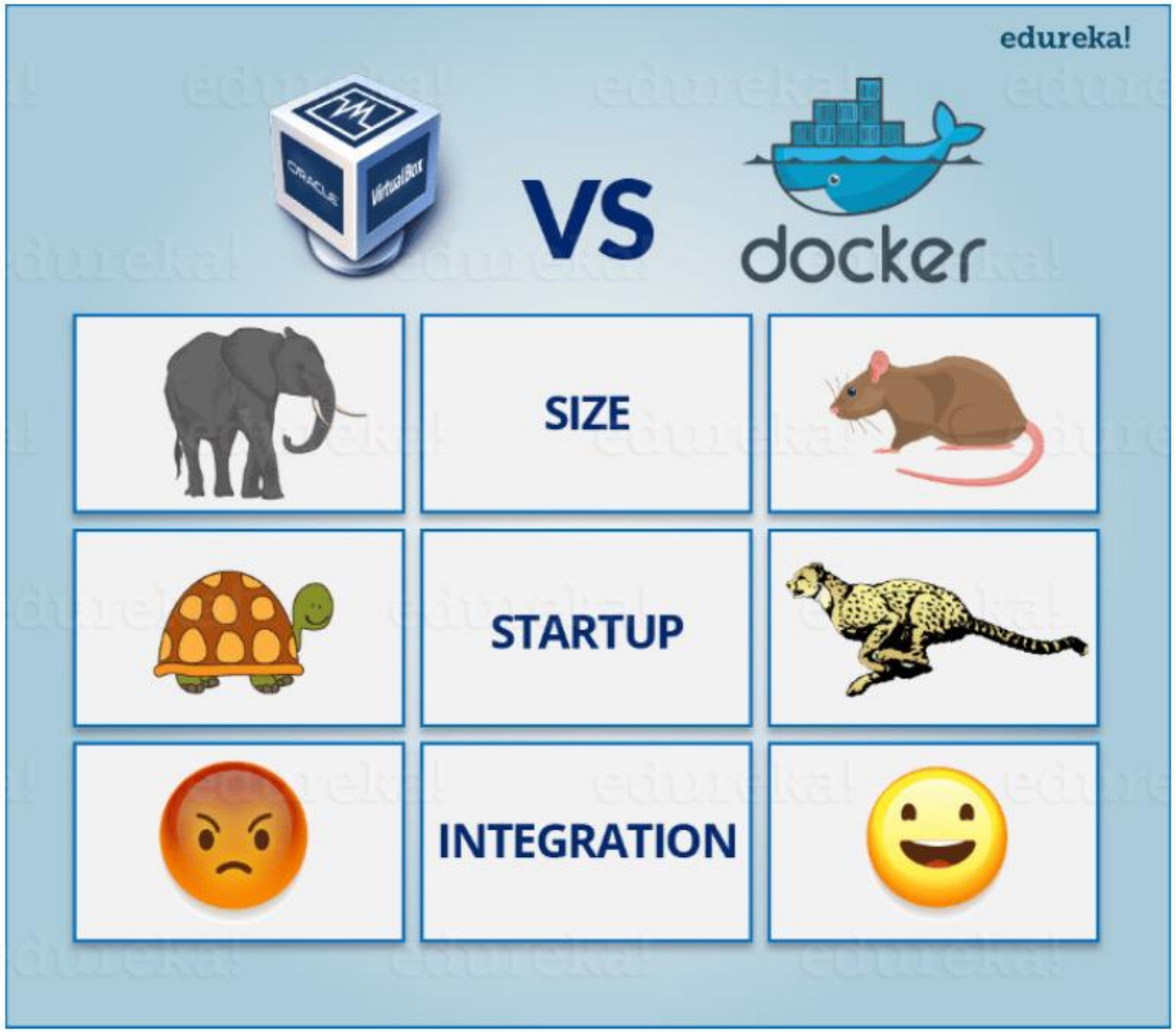
What is a Docker?

What Is Docker?

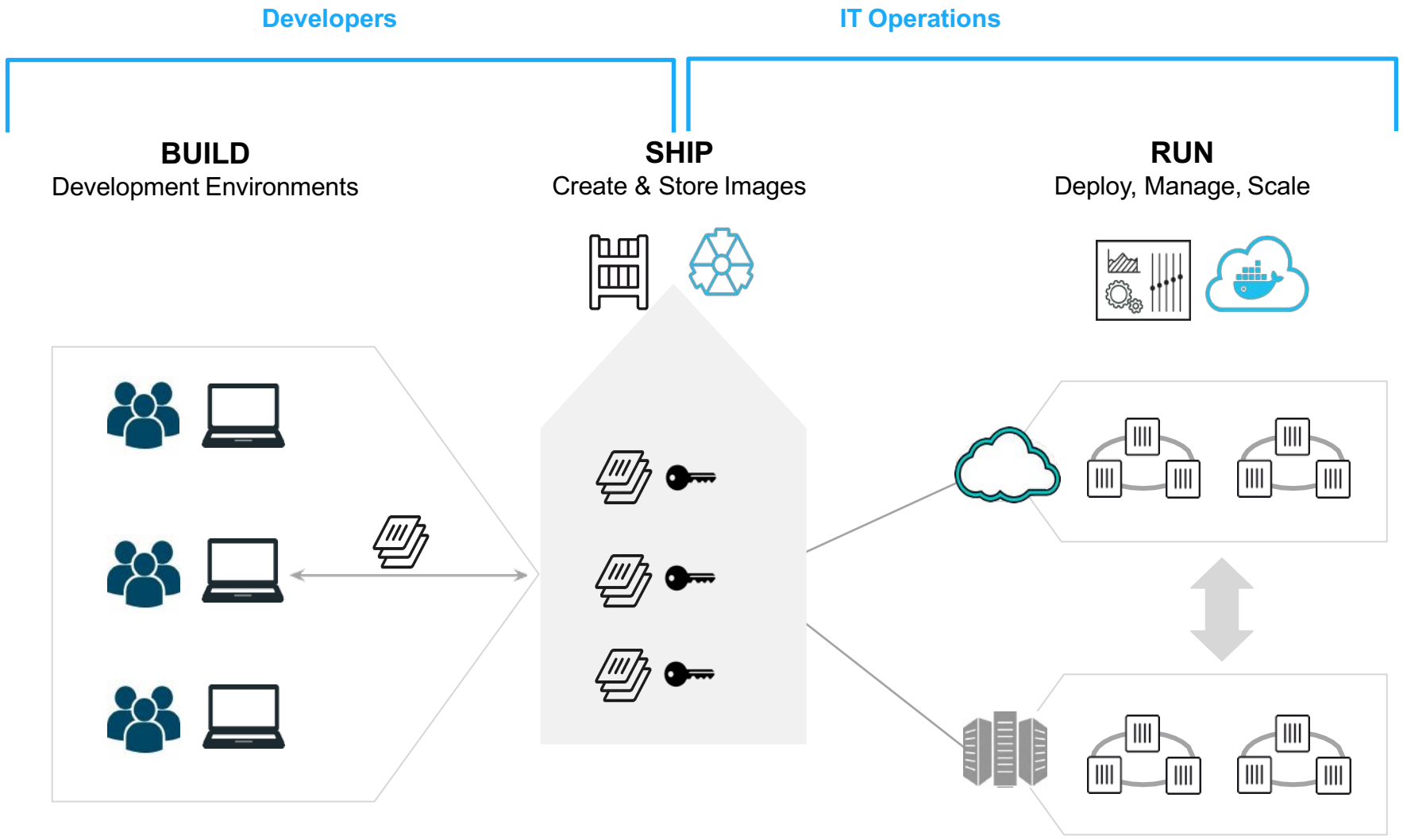


- Lightweight, open, secure platform.
- Simplify building, shipping, running apps
- Runs natively on Linux or Windows Servers
- Relies on “images” and “containers”

Dockers



Using Docker: Build, Ship, Run Workflow



Some Docker Vocabulary

Docker Image

The basis of a Docker container. Represents a full application

Docker Container

The standard unit in which the application service resides and executes

Docker Engine

Creates, ships and runs Docker containers deployable on a physical or virtual, host locally, in a datacenter or cloud service provider

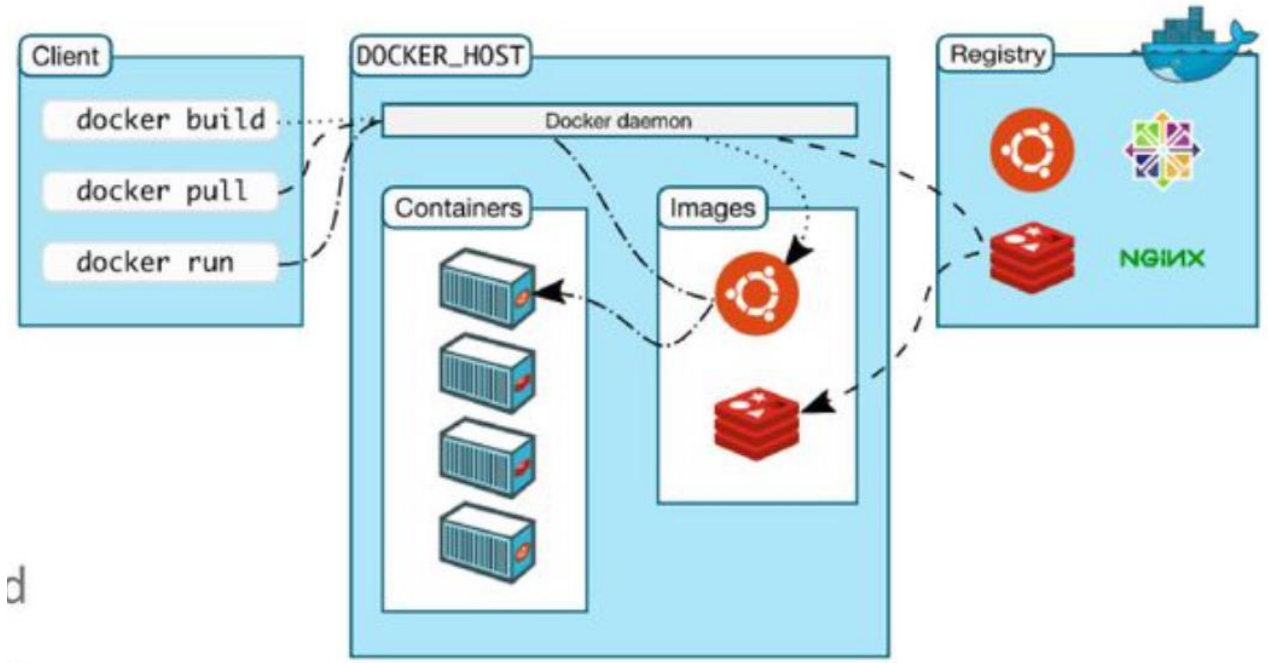
Registry Service (Docker Hub(Public) or Docker Trusted Registry(Private))

Cloud or server based storage and distribution service for your images

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 the Dockerfile.
- Container – Running instance of an Image using the docker run command
- Registry – Image repository

- https://labs.play-with-docker.com/



Docker File Example

Dockerfile

```
# Create image based on the official Node 6 image from dockerhub
FROM node:latest

# Create a directory where our app will be placed
RUN mkdir -p /usr/src/app

# Change directory so that our commands run inside this new directory
WORKDIR /usr/src/app

# Copy dependency definitions
COPY package.json /usr/src/app

# Install dependecies
RUN npm install

# Get all the code needed to run the app
COPY . /usr/src/app

# Expose the port the app runs in
EXPOSE 4200

# Serve the app
CMD ["npm", "start"]
```

- Instructions on how to build a Docker image
- Looks very similar to "native" commands
- Important to optimize your Dockerfile