EDEM
Escuela de Empresarios

GFT

Docker

September 2020

Agenda

1. Microservice Architecture
2. Containers
3. Docker
4. Docker Compose
5. Kubernetes

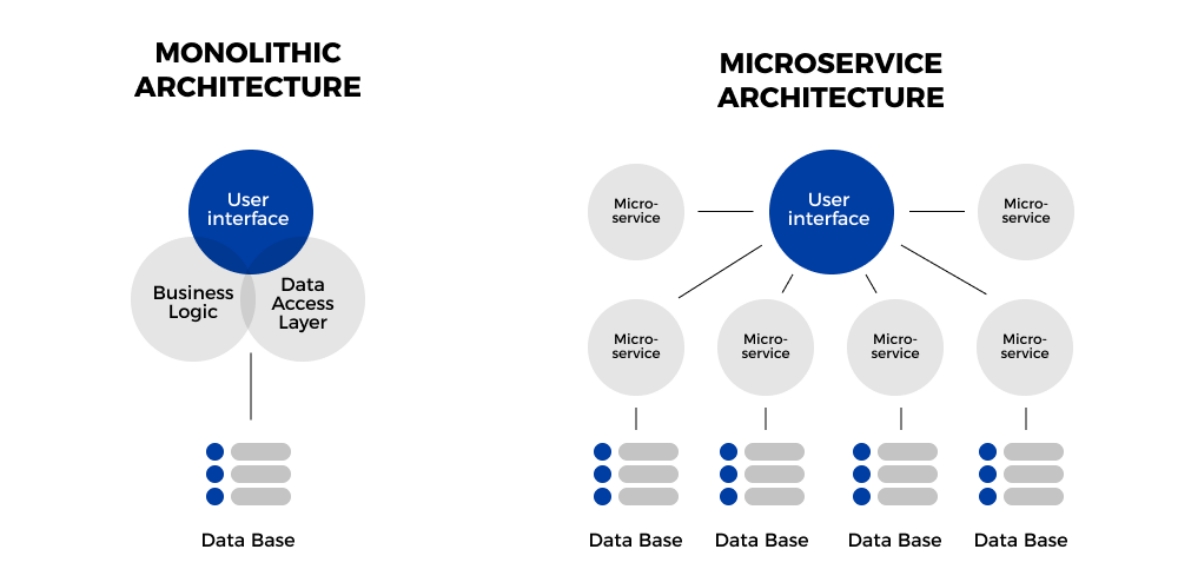
Microservice architecture

Microservices

- Microservice Architecture is an architectural style that structures an application as a collection of services that are
  - Highly maintanible and testable
  - Loosely coupled
  - Independently deployable
  - Organized around business capabilities
  - Owned by a small team

- The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications
  - It also enables an organization to evolve its technology stack

Microservice Architectura

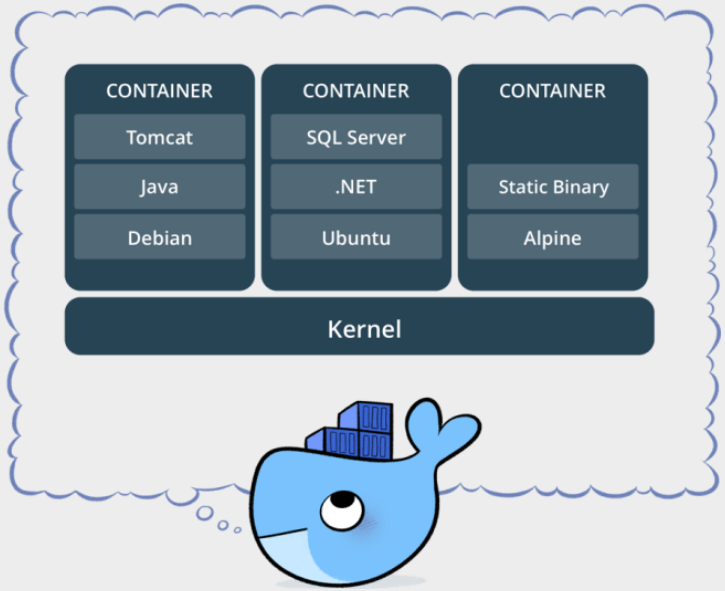


- **Developer issues:**
  - Minor code changes require full re-compile and re-test
  - Application becomes single point of failure
  - Application is difficult to scale
- **Microservice:**
  - Break application into separate operations
  - Make the app independently, scalable, stateless, highly available by design

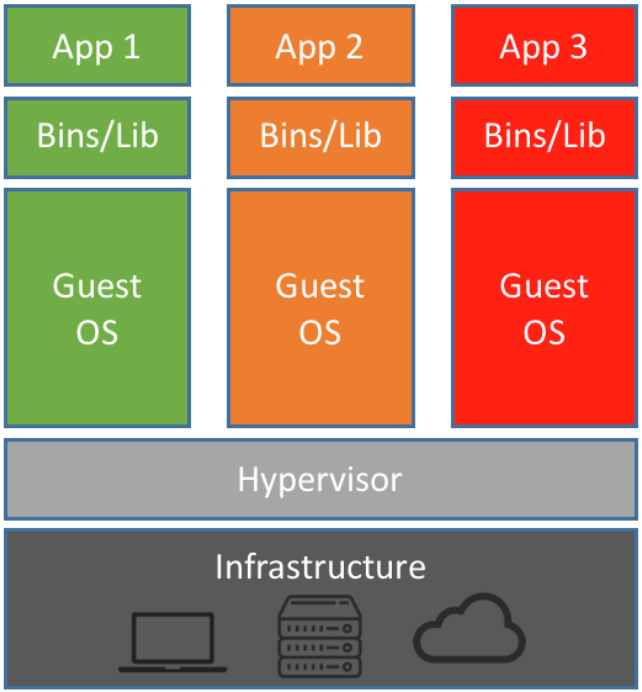
Containers

What is a container?

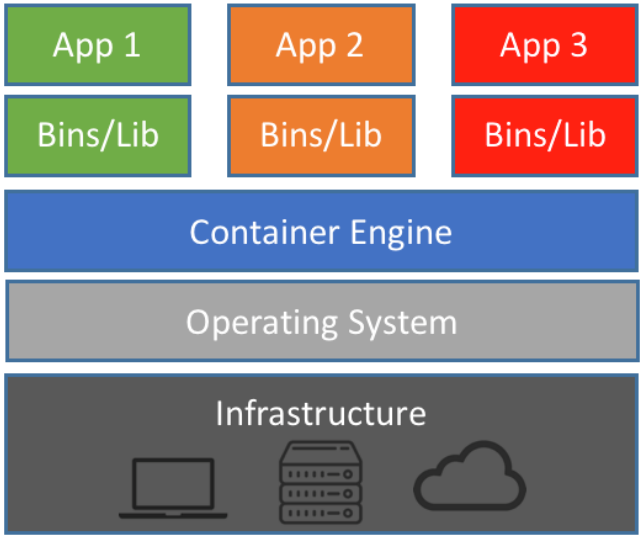
- Standardized packaging for software and dependencies
- Isolate apps from each other
- Share the same OS Kernel
- Works with all major Linux and Windows Server



Containers vs VMs

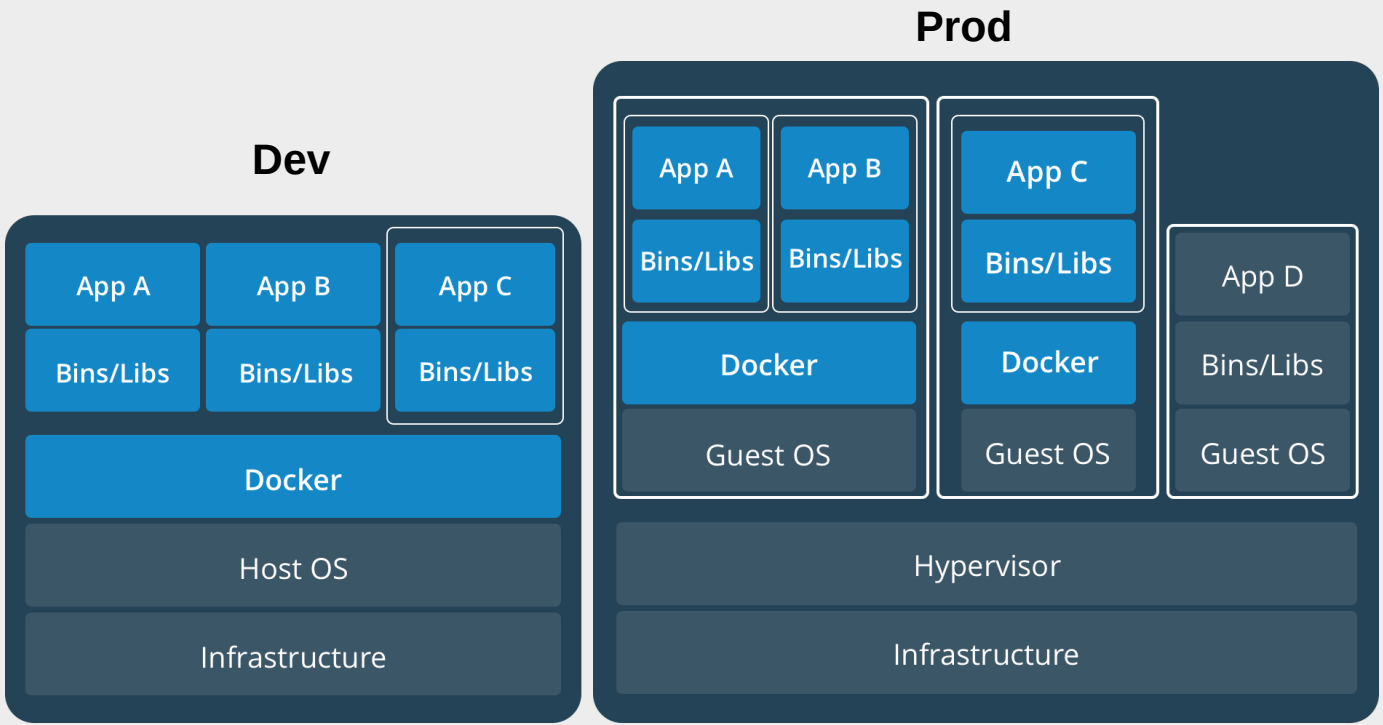


Machine Virtualization



Containers

Containers and VMs together

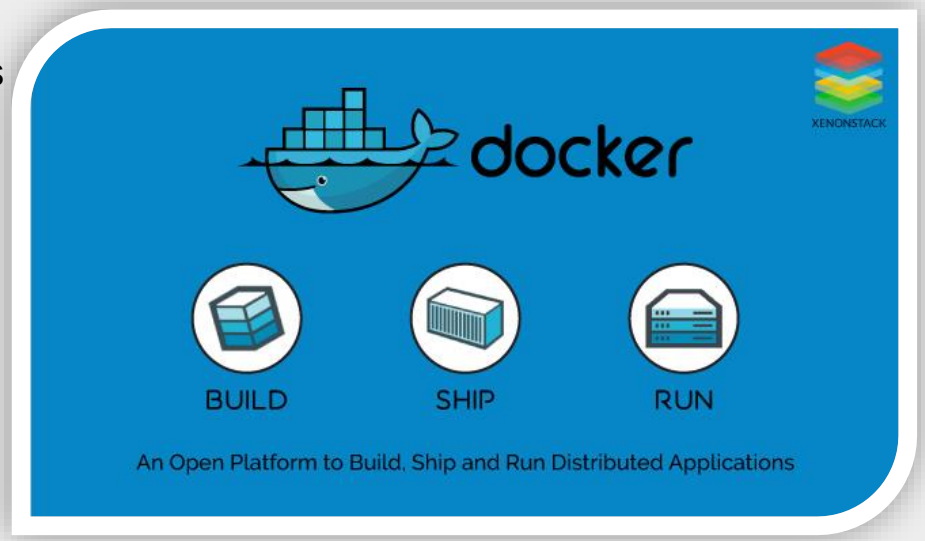


Container vs VMs

Docker

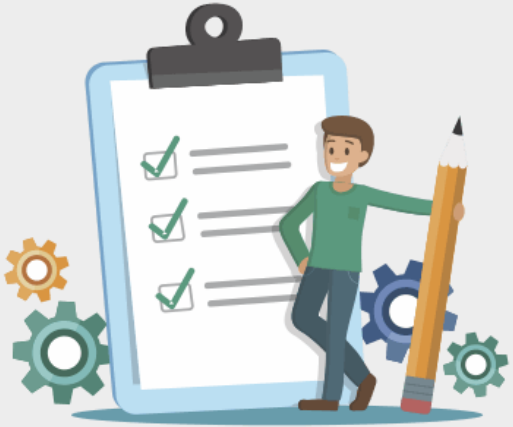
Docker

- Docker is an open platform for developing, shipping, and running containerized applications
- With Docker, you can manage your infrastructure in the same way you manage your applications
- No OS to boot → applications online in seconds



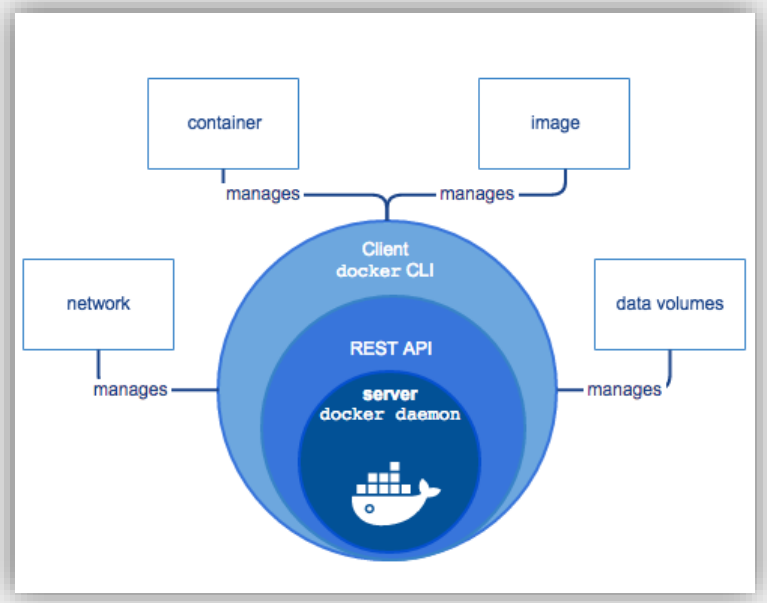
Hands-on

- Docker in your laptop
  - **Windows Users (Windows 10 Enterprise):**
    - https://enmilocalfunciona.io/instalando-y-probando-docker-en-windows-10/
    - https://github.com/docker/kitematic/releases (Kitematic)
  - **Mac Users:**
    - https://www.thegeekdiary.com/how-to-install-docker-on-mac/
    - https://github.com/docker/kitematic/releases (Kitematic)
  - **Ubuntu Users:**
    - https://docs.docker.com/engine/install/ubuntu/
    - https://github.com/docker/kitematic/releases (Kitematic)



Docker Engine

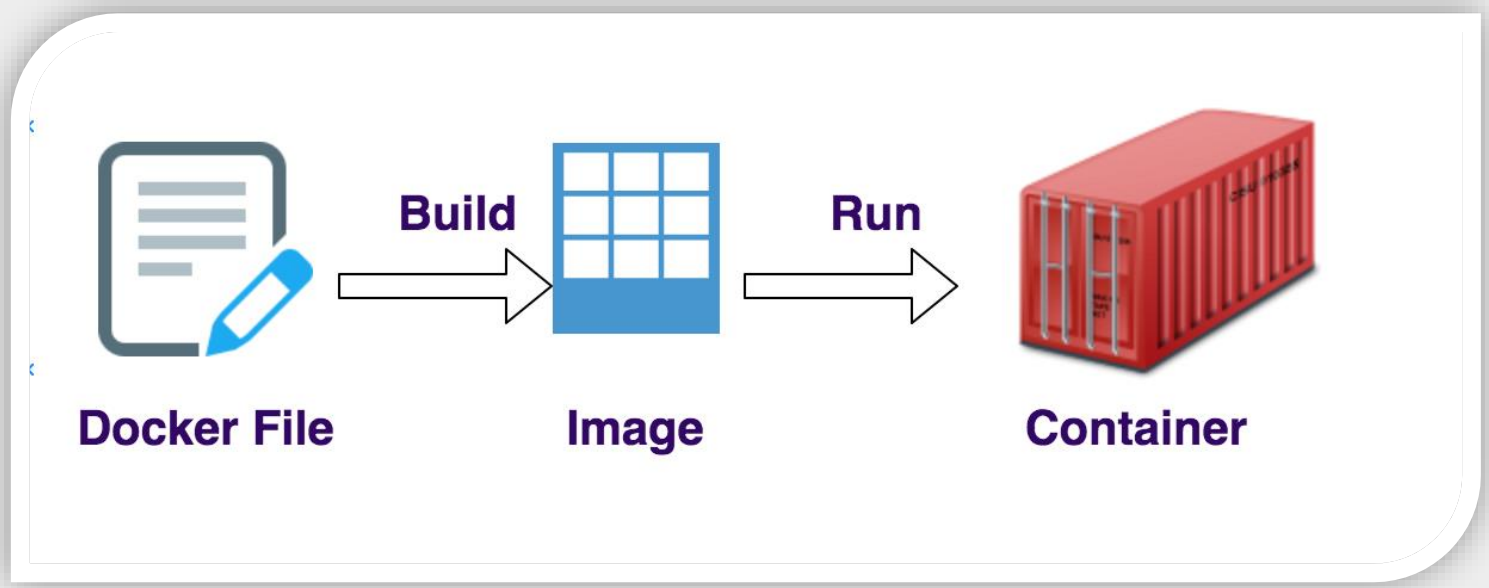
- Docker Engine is a client-server application with these major components:
  - Server
  - REST API
  - Command Line Interface (CLI)



Hands-on

- Docker Google Cloud:
  - **Create an VM instance** with the following features:
    - Zone: us-central1-a
    - Machine: e2-micro
    - OS: Container optimized OS
    - Allow HTTP/HTTPS
  - ~ $ docker version
  - ~ $ docker run –dp 80:80 docker/getting-started
  - **Your browser**: http://[VM-IP]
  - **Execute the same instructuios in your local machine** (Optional)

Docker Concepts



DockerFile

- A DockerFile is a text document that contains all the commands a user could call on the command line to assemble an image
  - You can consider DockerFile as blueprint of Docker Image

- DockerFile as a **sequential set of instruction** for Docker Engine
  - Order of sequence is important!!
  - **Each instruction creates a layer**
  - **A stack of such sequenced layers** managed by a filesystem **becomes a docker image**
  - Layers can be cached and reused by Docker

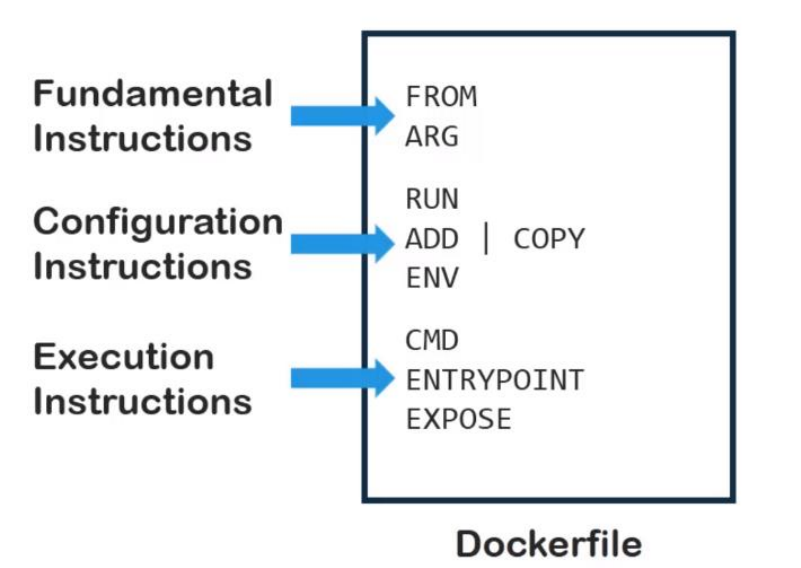
- Primary way to interacting with Docker

Dockerfile

```
FROM microsoft/dotnet:sdk AS build-env
WORKDIR /Docker

# Copy csproj and restore as distinct layers

RUN dotnet restore

# Copy everything else and build
COPY . ./
RUN dotnet publish -c Release -o out


# Build runtime image
FROM microsoft/dotnet:aspnetcore-runtime
WORKDIR /app
COPY --from=build-env /app/out .
ENTRYPOINT ["dotnet", "aspnetapp.dll"]
```

DockerFile Structure

- It’s a file with no extension called “Dockerfile”
- The instructions can be generally divided into three categories:
  - Fundamental
  - Configuration
  - Execution

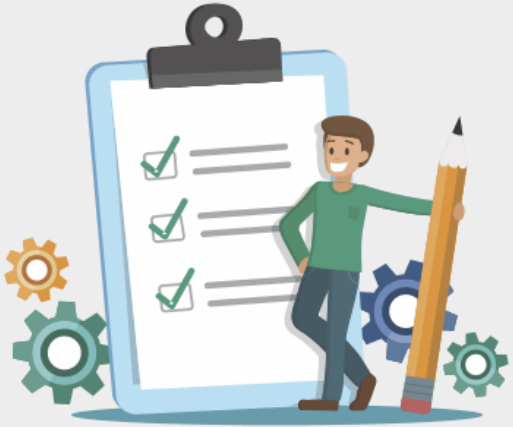


Demo 1

- DockerFile
  - Creates a new DockerFile
  - ~ $ docker build –t first_edem_img .
  - ~ $ docker images

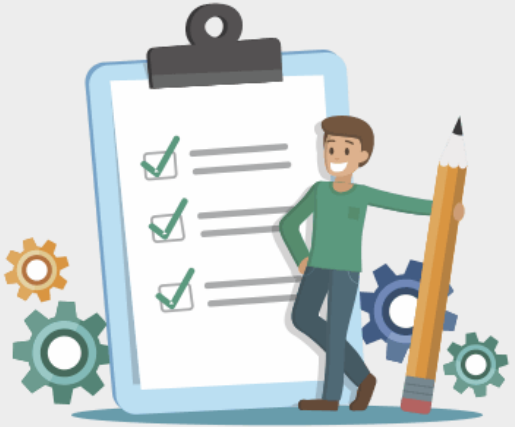
Demo 2

- DockerFile
  - Creates a new DockerFile
  - ~ $ docker build –t second_edem_img .
  - ~ $ docker images
  - ~ $ docker run –itd --name cont_second_edem second_edem_img
  - ~ $ docker ps –a
  - ~ $ docker exec –it cont_second_edem bash



Demo 3

- DockerFile
  - Create a new DockerFile
  - ~ $ docker build –t third_edem_img .
  - ~ $ docker images
  - ~ $ docker run –itd --name cont_third_edem –p 8080:80 third_edem_img
  - ~ $ docker ps –a
  - With browser http://localhost:8080

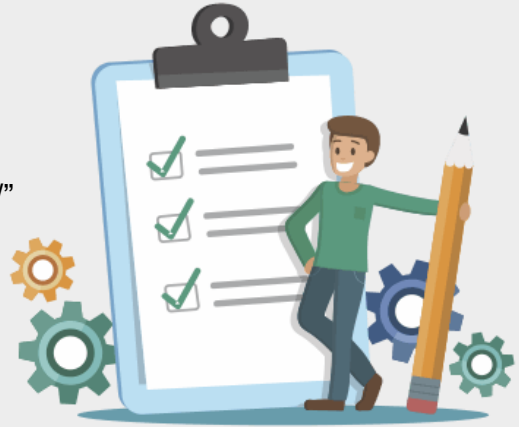


Hands On 4

- DockerFile
  - Create a file called “index.html” which contains the following:

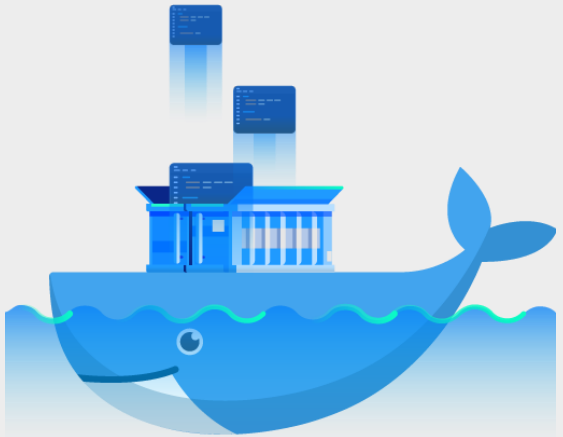
```
<p>Tu primer párrafo.</p>
<p>Tu segundo párrafo.</p>
<p>Un enunciado.
<br> EDEM.</p>
```

  - Modify DockerFile from Demo 3 to COPY file “index.html” into “/var/www/html/”
    - https://docs.docker.com/engine/reference/builder/#copy
  - Generate container with the previous web page



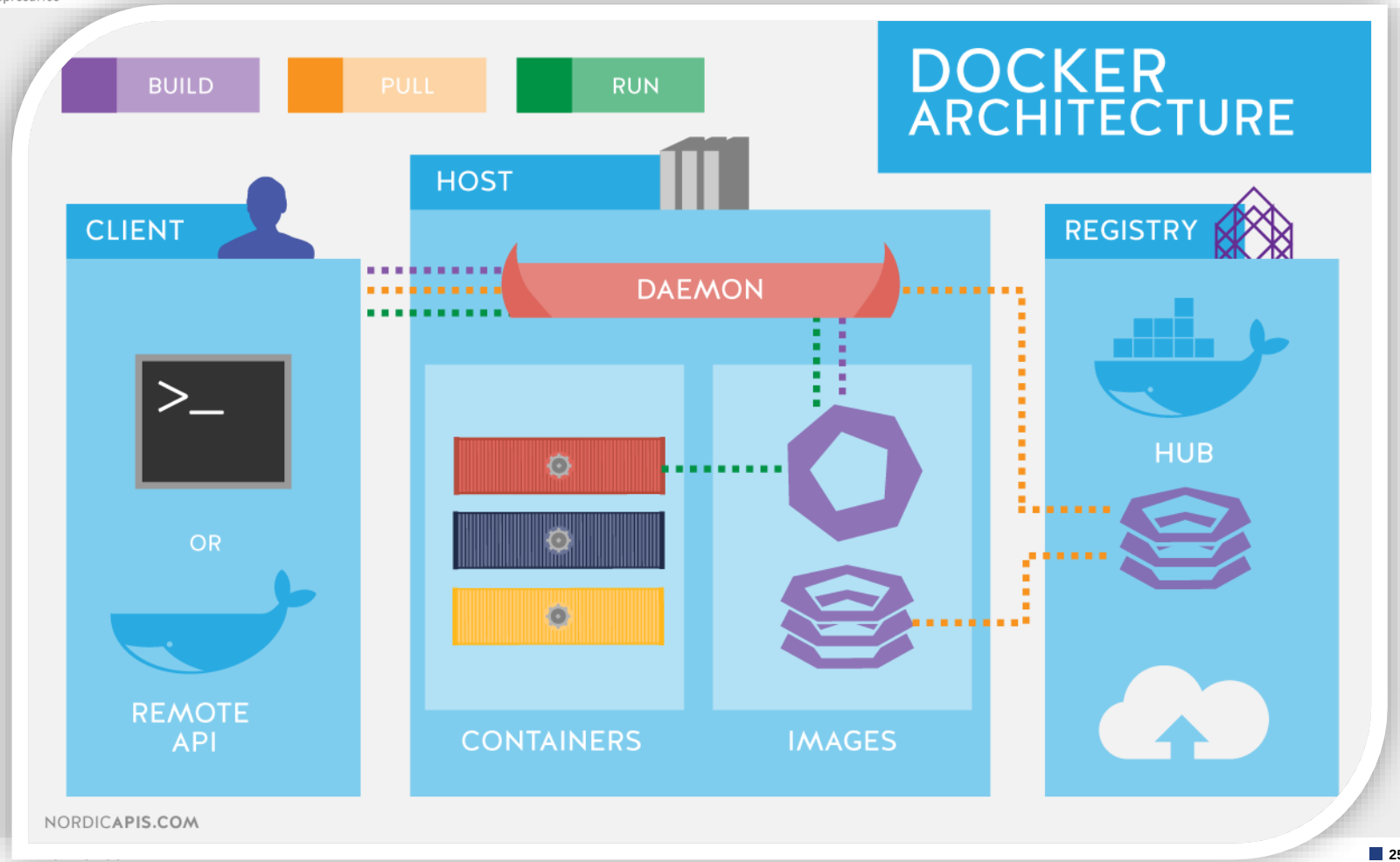
Docker Image

- A stack of multiple layers created from DockerFile instructions
- Recognized by name or Image ID
- The are pushed to and can be pulled from Docker Hub



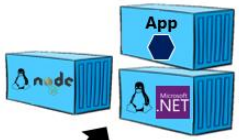
Docker Registry

- The Registry is a stateless, highly scalable server side application that stores and lets you distribute Docker images
  - Fully own your images distribution pipeline
  - Locally or using Docker Hub


BUILD
PULL
RUN
DOCKER ARCHITECTURE
HOST
CLIENT
DAEMON
REGISTRY
>_
OR
REMOTE API
CONTAINERS
IMAGES
HUB
NORDICAPIS.COM

Basic taxonomy in Docker

A **Registry**
Stores many
static images



Images
Static, persisted container image

Container
Image-instance running
an app process (service/web)

Hosted Docker Registry

Docker Trusted Registry on-prem.

On-premises
('n' private organizations)

Docker Hub Registry

Docker Trusted Registry on-cloud

Azure Container Registry

AWS Container Registry

Google Container Registry

Quay Registry

Other Cloud

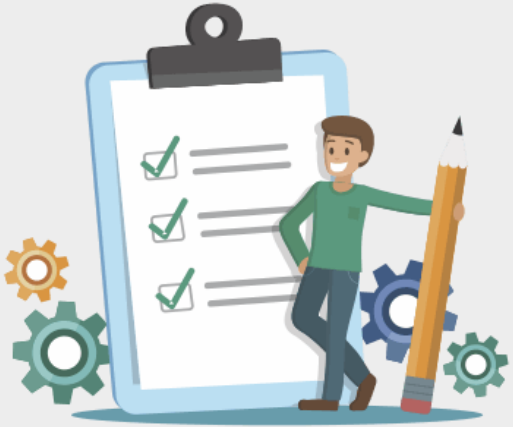
Public Cloud
(specific vendors)

Demo 5 & Hands on

- Docker Image & Docker Hub
  - Stop all containers from Kitematic UI
  - Removing last image:
    - ~ $ docker image rm third_edem_img
    - Any issue?
    - Remove all containers from Kitematic and test again
  - Remove all images

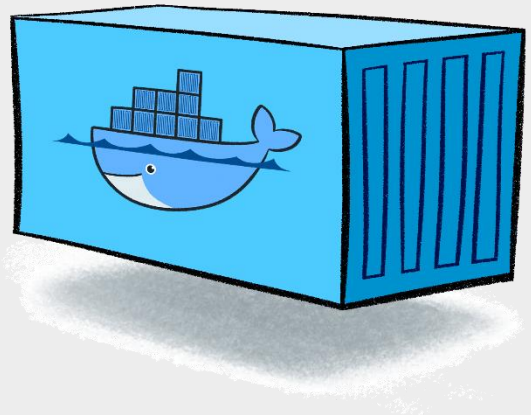
Demo 6 & Hands on

- Docker Hub
  - Pull Wordpress image from Docker Hub https://hub.docker.com/_/wordpress
    - ~ $ docker pull wordpress
    - ~ $ docker run --name my-wordpress –p 8080:80 –d wordpress
  - Browser http://localhost:8080



Docker Container

- Running instance of a Docker Image
- Provides similar isolation to VMs but lighter!
- Adds writable layer on top of image layers and works on it
- Can talk to other containers like processes in Linux
- Provide resources to an image



Demo 7 & Hands on

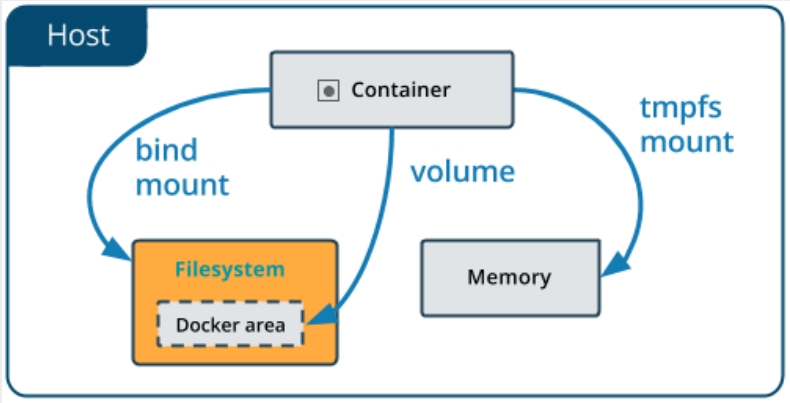
- Docker Containers & Docker Hub
  - Convert web page container into image
    - Repeat Hands On 4 changing some line from “index.html”
    - Execute container

  - ~ $ docker ps –a
  - Add new line in “/var/www/html/index.html”
    - ~ $ docker exec –it cont_hands_on_8 bash
    - ~ $ cd /var/www/html
    - ~ $ echo “ New file >> ./index.html

  - ~ $ docker commit cont_hands_on_8 new_hands_on_8
  - ~ $ docker login –username=xxxxx
  - ~ $ docker tag 4350cd3f7ffd yourhubusername/myfirstimage:latest
  - ~ $ docker push yourhubusername/myfirstimage

Docker Volume

- What happens to the data if a container crash o removed?
  - Data could be lost!!!
- Docker has two options for containers to store files in the host machine:
  - Volumes
  - Bind mounts
- Volumes have the following advantages:
  - Easier to back up or migrate
  - Managed using Docker CLI
  - More safely shared among multiple containers
  - Isolated from the host file system

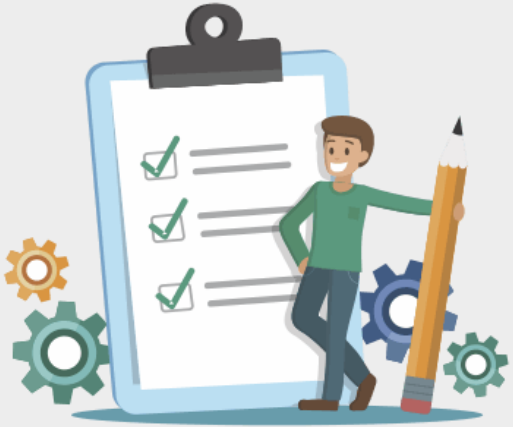


Demo 8

- Docker Volumes

  - ~ $ docker volume create my-vol

  - ~ $ docker volume ls
  - ~ $ docker volume inspect my-vol
  - ~ $ docker run –d --name volume_test \
    --mount source=my-vol,target=/app dlpgft/myfirstimage
  - ~ $ docker exec --it volume_test bash



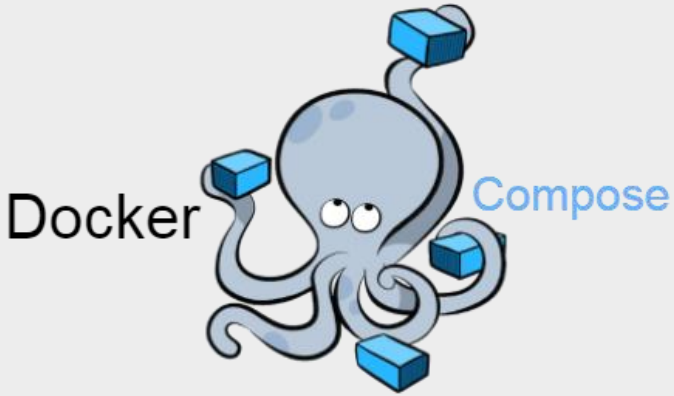
Demo 9 and Hands on

- Docker Containers
  - Remove all containers
    - Tip: ~ $ docker container --help
  - Remove all images except wordpress

Docker Compose

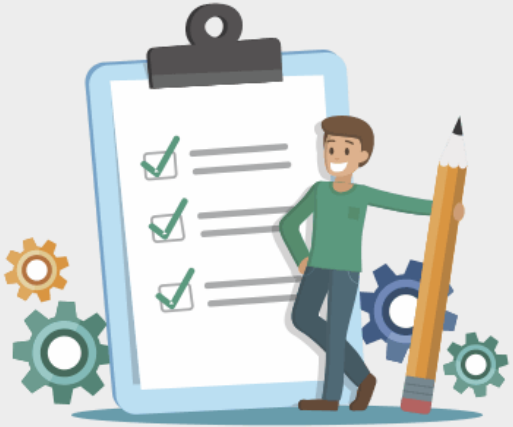
Docker Compose

- Compose is a tool for **defining and running complex applications** with docker
- Without Docker Compose, multiple DockerFiles will be needed for a full or complex application
  - Separate files for front-end, back-end...
- With Docker Compose, you can define a **multi-container application in a single file**
- Usually the file is called “docker-compose.yml”



Demo 10 (Only for Linux Users)

- Docker Compose Installation
  - https://docs.docker.com/compose/install/



Demo 11 & Hands on

- Docker Compose – Full Wordpress

  - Create a docker-compose file
  - Fill docker-compose file using the following link:
    - https://docs.docker.com/compose/wordpress/
  - ~ $ docker-compose up -d

Hands on 12

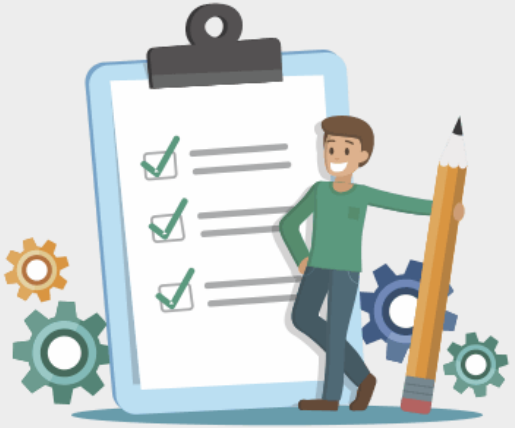
- Docker Compose – Full Wordpress
  - Add Ubuntu con Nginx from Demo 4 into the previous docker-compose file
  - Execute this docker-compose again with new changes

Hands on 13

- Remove all containers
- Remove all images
- Remove all volumes

Hands on 14

- Install Jupyter from Docker



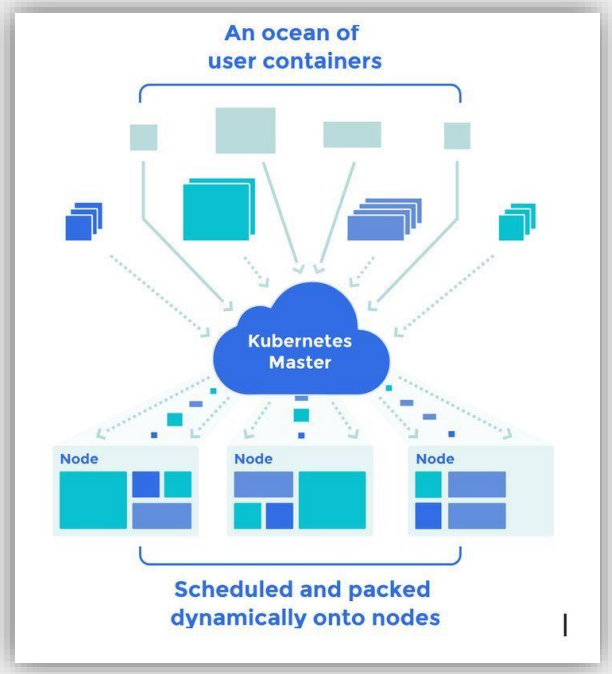
Kubernetes

Kubernetes

- Large and small software companies deploying thousands of container instances daily
  - How can we manage this complexity?
- Originally developed by Google.
- Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications
- Kubernetes makes it easy to deploy and operate applications in a microsevice architecture

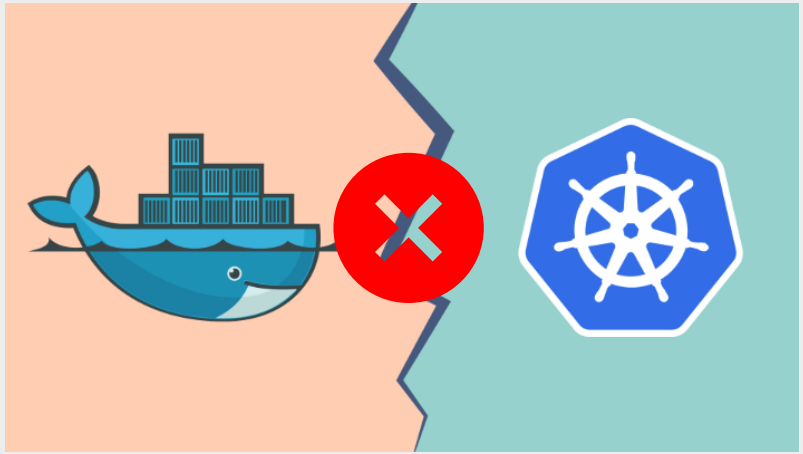
Kubernetes

- Features:
  - Controlling resource consumption by application or team
  - Evenly spreading application load across a host infrastructure
  - Automatically load balancing requests across the different instances of an application
  - Monitoring resource consumption and resource limits
  - Moving an application instance from one host to another
  - Automatically leveraging additional resources made available when a new host is added

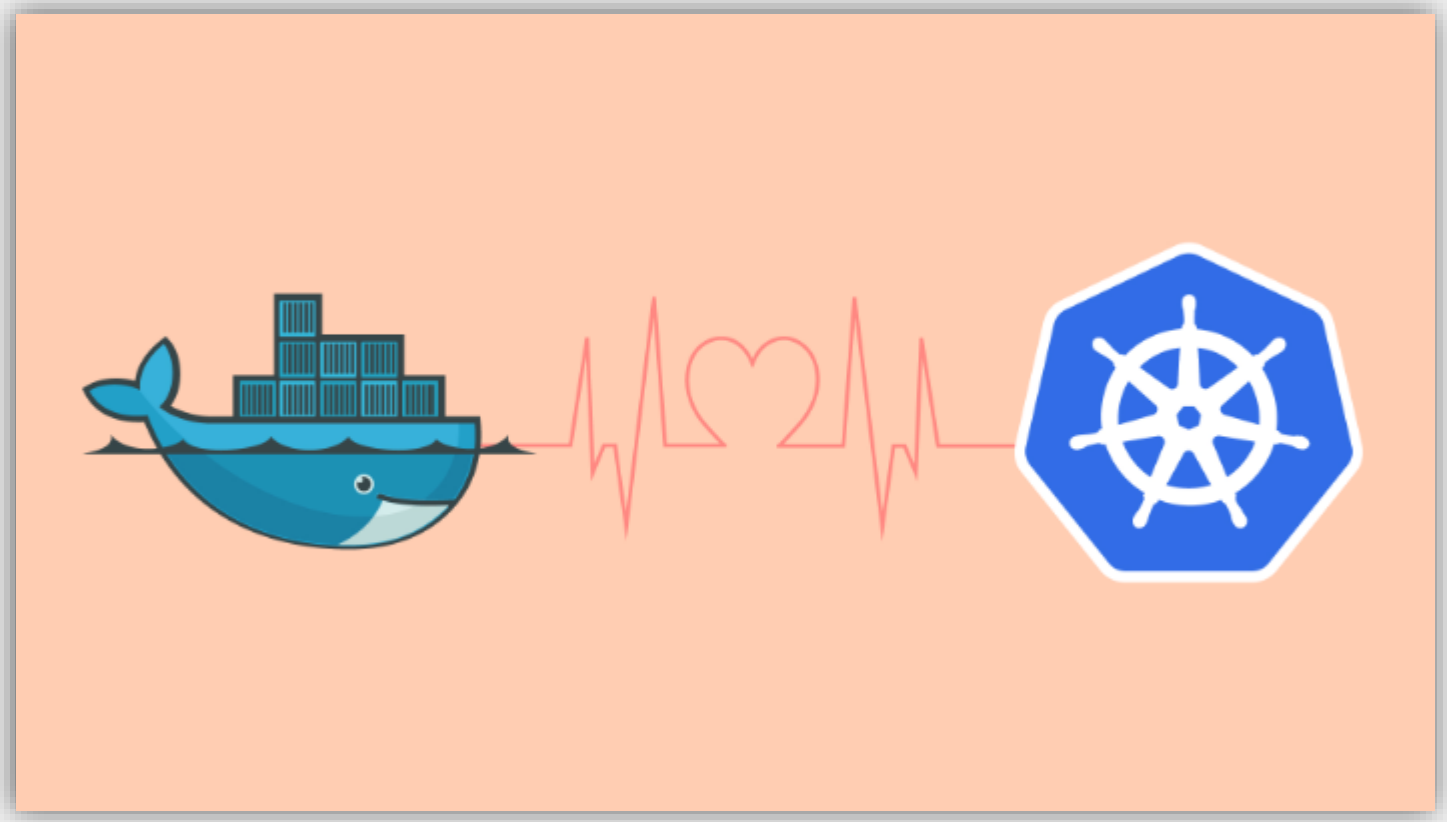


Kubernetes - Docker

- Docker is used to isolate your application into containers
- Kubernetes, on the other hand, is a container scheduler. It’s used to deploy and scale your application



Kubernetes - Docker



Miguel Ángel Sotomayor

Senior Data Engineer – Data Architect

mlsy@gft.com

https://github.com/masfworld

https://www.linkedin.com/in/miguelsotomayorf/