Arquitetura de software – Dia 2

F0012 - O que é Docker?

O Docker é uma plataforma de virtualização leve que permite criar, distribuir e executar aplicações em containers. Um container é uma unidade de software que empacota o
código da aplicação e todas as suas dependências (bibliotecas, configurações, etc.), garantindo que ela funcione de forma consistente em qualquer ambiente,

Conceitos Fundamentais para Entender o Docker

Namespaces

Os namespaces são uma funcionalidade do kernel do Linux usada pelo Docker para isolar os recursos de um container do restante do sistema operacional e de outros containers. Eles garantem que cada container tenha sua própria visão dos recursos, como;

- **PID (Process ID):** Isola os processos entre containers.
- **NET (Network):** Isola as interfaces e configurações de rede.
- **MNT (Mount):** Isola o sistema de arquivos.
- **UTS (Unix Timesharing System):** Permite que cada container tenha seu próprio nome de host.
- **IPC (Interprocess Communication):** Isola os mecanismos de comunicação entre processos.

Com os namespaces, cada container parece um sistema independente, mesmo estando no mesmo kernel do host.

Cgroups

Os cgroups (ou control groups) são outra funcionalidade do kemel do Linux que permite controlar e limitar o uso de recursos por processos ou grupos de processos. No Docker,
isso é usado para gerenciar:

- **CPU:** Limita o uso do processador.
- **Memória:** Define quanta memória um container pode usar.
- **Disco:** Restringe o uso de |/O do disco.
- **Rede:** Controla o consumo de recursos de rede.

Isso garante que cada container tenha recursos suficientes e que um container não monopolize os recursos do host.

File System

O file system no Docker é uma combinação de camadas que formam o sistema de arquivos do container. Ele utiliza a tecnologia UnionFS (Union File System), que permite:

- **Camadas de leitura e escrita:** As imagens do Docker são compostas por várias camadas de somente leitura, e os containers adicionam uma camada de leitura e escrita por
  cima.

- **Reaproveitamento de camadas:** Quando várias imagens compartilham camadas em comum, elas são reutilizadas para economizar espaço.

O sistema de arquivos no Docker é modular e eficiente, permitindo que containers sejam criados rapidamente.

Docker Host

O Docker Host é a máquina onde o Docker está instalado e em execução. Ele pode ser:

- Uma máquina física (bare metal).
- Uma máquina virtual.
- Um servidor em nuvem.

No Docker Host, o daemon do Docker (processo principal) gerencia as imagens, containers, volumes e redes. Ele também interage com o kemel do sistema operacional.

Esses conceitos (namespaces, cgroups, file system e Docker Host) são os pilares que tornam o Docker eficiente, isolado e leve para executar aplicações em containers.

Principais Conceitos

1. Containers: Ambientes isolados e portáteis que contêm a aplicação e suas dependências.
2. Imagens: Um template imutável que define como um container deve ser criado (como um "snapshot").
3. Dockerfile: Um arquivo que descreve passo a passo como construir uma imagem.
4. Docker Hub: Um repositório público para compartilhar e baixar imagens de containers.
5. Volumes: Permitem persistir dados fora do container.

Benefícios

Portabilidade: Funciona da mesma forma em diferentes sistemas operacionais e servidores.
Leveza: Containers compartilham o kemel do sistema operacional, usando menos recursos que máquinas virtuais.
Rapidez: Criação e inicialização de containers são muito mais rápidas do que VMs.
Escalabilidade: Ideal para ambientes de microsserviços e DevOps.

Comandos Básicos

docker build: Cria uma imagem a partir de um Dockerfile.
docker run: Executa um container baseado em uma imagem.
docker ps: Lista os containers em execução.
docker stop: Para um container em execução.
docker pull: Baixa uma imagem do Docker Hub.

Exemplo Simples de Dockerfile

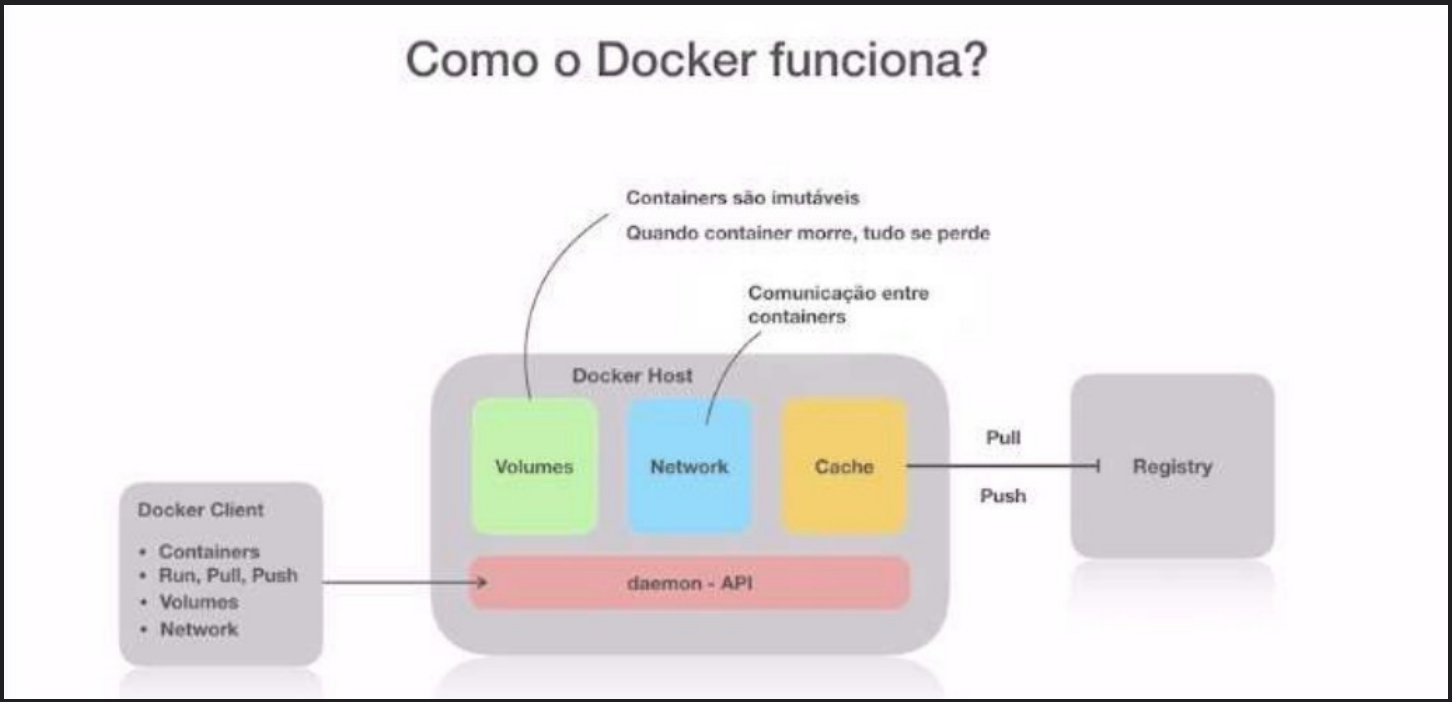
```
# Usa uma imagem base
FROM python:3.9

# Define o diretório de trabalho
WORKDIR /app

# Copia os arquivos para o container
COPY . /app

# Instala as dependências
RUN pip install -r requirements.txt

# Comando para iniciar a aplicação
CMD ["python", "app.py"]
```



Como Instalar o Docker

1. Instalar no Linux (Ubuntu/Debian)

Passo 1: Atualizar os pacotes do sistema

```
sudo apt update && sudo apt upgrade -y
```

Passo 2: Instalar dependências necessárias

```
sudo apt install -y ca-certificates curl gnupg
```

Passo 3: Adicionar a chave GPG oficial do Docker

```
sudo install —-m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/Linux/debian/gpa | sudo gpg —-dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpa
```

Passo 4: Configurar o repositório Docker

```
echo
“deb [arch=$(dpkg—print-architecture) \
signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/Linux/debian \
$(lsb release -cs) stable” | sudo tee /etc/apt/sources.list.d/docker.list
> /dev/null
```

Passo 5: Instalar o Docker

```
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin
```

Passo 6: Verificar a instalação

```
docker --version
```