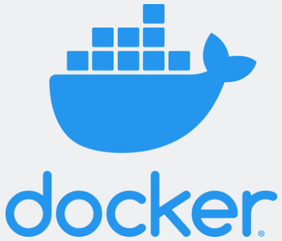
Sopes 1

https://docs.docker.com/



DOCKERFILE

MULTISTAGE

Josue Gramajo - 202000895

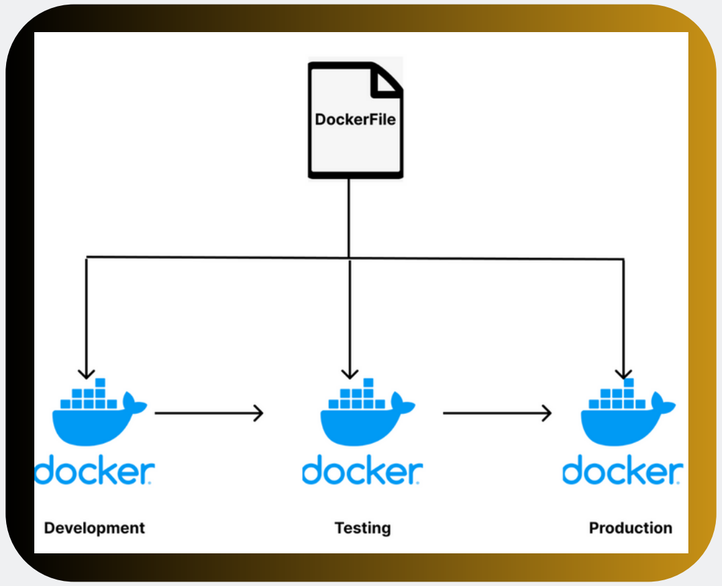
Sopes 1 | Introducción

https://docs.docker.com/

Definición de DockerFile Multistage

¿Qué es?

Dockerfile Multistage es una característica que permite dividir la construcción de imágenes Docker en múltiples etapas. Esto mejora la eficiencia, la seguridad y reduce el tamaño final de las imágenes, abordando problemas comunes en el desarrollo y despliegue de aplicaciones en contenedores.



¿Por qué?

La construcción tradicional de imágenes Docker a menudo involucra la inclusión de herramientas de desarrollo y dependencias en la imagen final, aumentando innecesariamente su tamaño. Esto puede comprometer la seguridad y eficiencia del despliegue. Dockerfile Multistage aborda esta problemática al permitir la creación de imágenes en múltiples etapas.

Sopes 1 | Introducción | Conceptos

https://docs.docker.com/

Conceptos Clave

En Dockerfile Multistage, varios conceptos son fundamentales para comprender cómo funciona y cómo puede mejorar el proceso de construcción de imágenes.

- FROM: Establece la imagen base para cada etapa, permitiendo la selección de imágenes especializadas.
- AS: Etiqueta las etapas del build, facilitando la organización y referencia de las fases.
- COPY: Transfiere archivos entre etapas, evitando la inclusión innecesaria de archivos de construcción en la imagen final.
- WORKDIR: Define el directorio de trabajo para las instrucciones siguientes, permitiendo establecer el contexto en cada etapa.
- RUN: Ejecuta comandos en el contexto actual, utilizado en cada etapa para tareas específicas como instalación de dependencias o construcción de aplicaciones.

conceptos.txt

- **FROM**: La declaración FROM inicial.
- **AS**: Nombrar cada etapa del build.
- **COPY**: Copiar archivos entre etapas.
- **WORKDIR**: Establecer el directorio de trabajo.
- **RUN**: Ejecutar comandos en cada etapa.

#Docker #DockerFile

https://docs.docker.com/

Ejemplo

Dockerfile

Dockerfile realiza la compilación de una aplicación en una etapa y luego copia el ejecutable resultante a una imagen más ligera de Alpine para su ejecución. Esto ayuda a reducir el tamaño final de la imagen, ya que la imagen de construcción puede contener herramientas y dependencias de compilación que no son necesarias para la ejecución de la aplicación.

DockerFile

```
# Etapa 1: Build
FROM alpine AS builder
RUN apk update && apk add build-base
COPY . /app
WORKDIR /app
RUN make

# Etapa 2: Ejecución
FROM alpine
COPY --from=builder /app/myapp
/usr/local/bin/myapp
CMD ["myapp"]
```