



DOCKER

- INSTRUCTOR
- I.S.C. E.T.W. M.T.W. CARLOS URIEL DE JESÚS SÁNCHEZ GONZÁLEZ
- [HTTPS://WWW.LINKEDIN.COM/IN/CARLOS-URIEL-DE-JESUS-SANCHEZ-GONZALEZ-8920A1372/](https://www.linkedin.com/in/carlos-uriel-de-jesus-sanchez-gonzalez-8920a1372/)

¿QUÉ ES DOCKER?

- Docker es una plataforma que permite crear, ejecutar y gestionar contenedores.
- Un contenedor es como una “cajita” que contiene:
 - tu aplicación,
 - más todas sus dependencias,
 - pero sin incluir un sistema operativo completo (a diferencia de una máquina virtual).
- ➡ Así, tu aplicación corre igual en cualquier lugar (tu laptop, un servidor, la nube), sin importar el sistema operativo o las librerías instaladas afuera.

¿CÓMO FUNCIONA DOCKER?

- Docker se basa en 2 conceptos principales:
- 1. Imágenes (Images)
 - Son “plantillas de solo lectura” (ejemplo: ubuntu, mysql, node:18).
 - Contienen el sistema base + dependencias necesarias.
 - Se crean a partir de un Dockerfile.
- Ejemplo de Dockerfile:
 - FROM node:18
 - WORKDIR /app
 - COPY package*.json .
 - RUN npm install
 - COPY ..
 - CMD ["npm", "start"]

¿CÓMO FUNCIONA DOCKER?

- 2. Contenedores (Containers)
 - Son instancias en ejecución de una imagen.
 - Ligeros y rápidos (usan el kernel del host, no arrancan un SO completo como una VM).
 - Puedes correr muchos contenedores en paralelo sin que choquen.
- Ejemplo:
 - # Crear y correr un contenedor de Nginx
 - docker run -d -p 8080:80 nginx
 - Esto levanta un servidor web Nginx en <http://localhost:8080>.

ARQUITECTURA DE DOCKER

- Docker Engine:
 - El “motor” que corre contenedores.
 - Tiene el daemon (`dockerd`) que gestiona imágenes y contenedores.
- Docker CLI:
 - El cliente (`docker run`, `docker ps`, etc.).
- Docker Hub:
 - Repositorio público de imágenes (como GitHub, pero para contenedores).

DIFERENCIA CON MÁQUINA VIRTUAL

- VM: necesita un SO completo dentro de otra (pesado, lento).
- Docker: solo empaqueta tu app + dependencias, usa el mismo kernel → más ligero y rápido.