**Dockerfile**

**docker build -t nombre directorio\_Dockerfile**  
Genera una nueva imagen denominada nombre a partir de un fichero Dockerfile

**FROM imagen**  
La imagen base

**MAINTAINER nombre usuario**  
El usuario que gestiona esa imagen

**RUN comandos**  
Ejecuta comandos en la imagen

**ENV variables de entorno**  
Define variables de entorno

**EXPOSE puerto**  
Puerto a exponer en la imagen

**ADD fichero\_origen fichero\_destino**  
Añade un fichero a la imagen

**COPY fichero\_origen fichero\_destino**  
Copia un fichero a la imagen

**ENTRYPOINT fichero\_origen fichero\_destino**  
Comando a ejecutar al iniciar la imagen

**VOLUME fichero**  
Crea un fichero de volumen a la imagen

**WORKDIR directorio**  
Establece el directorio inicial al arrancar la imagen

**Docker CLI**

**Rendimiento y logs**

**docker logs nombre**  
Muestra los logs del contenedor nombre

**docker inspect nombre**  
**Muestra la información completa del contenedor denominado nombre**

**Imagenes**

**docker images**  
Lista las imagenes almacenadas en el host

**docker inspect id/nombre**  
Muestra los detalles de una imagen a partir de un id o un nombre

**docker pull nombre**  
Descarga la imagen correspondiente a ese nombre

**docker rmi id/nombre**  
Elimina la imagen correspondiente a ese id o nombre

Redes

**docker network ls**  
Lista las redes disponibles

**docker network inspect nombre\_red**  
Muestra los detalles de la red correspondiente a nombre\_red

**docker network create -d bridge nombre\_red**  
Crea una red de tipo bridge denominada nombre\_red

**Volumenes**

**docker volume ls**  
Listar los volumenes

**docker volume create --name nombre**  
Crear un volumen con nombre

**docker volume rm**  
Borrar un volumen con nombre

**docker volume inspect id**  
Ver detalles de un volumen

**Imágenes**

**docker push servidor:5000/imagen**  
Sube una imagen al repositorio

**docker pull servidor:5000/imagen**  
Descarga una imagen del repositorio

**Contenedores**

**docker create -it --name nombre imagen**  
Crea un contenedor denominado name a partir de imagen

**docker start nombre**  
Arranca el contenedor denominado nombre

**docker stop nombre**  
Para el contenedor denominado nombre

**docker restart nombre**  
Rearranca el contenedor denominado nombre

**docker rm nombre**  
Elimina el contenedor denominado nombre

**docker ps -a**  
Lista todos los contenedores en ejecución (-a incluye los parados)

**docker run -it imagen comando**  
Arranca y ejecuta el comando en un contenedor de esa imagen en modo interactivo

**docker exec -it nombre comando**  
Ejecuta el comando en el contenedor nombre

**docker run -d**  
Arranca el contenedor en modo daemon

**docker run -P**  
Arranca el contenedor y expone los puertos del contenedor en puertos aleatorios del host

**docker run -p puerto\_host:puerto\_contenedor**  
Arranca el contenedor y expone el puerto\_contenedor en el puerto\_host

**docker run -v directorio\_host:directorio\_contenedor**  
Asigna el directorio\_host para mapearlo en el directorio\_contenedor

**docker run --hostname nombre\_host**  
Arranca el contenedor y asigna nombre\_host al contenedor

**docker run --net=mired**  
Arranca el contenedor y lo asigna en la red denominada mired

**DockerCompose**

version: '3'

services:

app:

image: node:latest

container\_name: app\_main

restart: always

command: sh -c "yarn install && yarn start"

ports:

- 8000:8000

working\_dir: /app

volumes:

- ./:/app

environment:

MYSQL\_HOST: localhost

MYSQL\_USER: root

MYSQL\_PASSWORD:

MYSQL\_DB: test

mongo:

image: mongo

container\_name: app\_mongo

restart: always

ports:

- 27017:27017

volumes:

- ~/mongo:/data/db

volumes:

mongodb:

Comandos

* version refers to the docker-compose version (Latest 3)
* services defines the services that we need to run
* app is a custom name for one of your containers
* image the image which we have to pull. Here we are using node:latest and mongo.
* container\_name is the name for each container
* restart starts/restarts a service container
* port defines the custom port to run the container
* working\_dir is the current working directory for the service container
* environment defines the environment variables, such as DB credentials, and so on.
* command is the command to run the service

source: https://www.freecodecamp.org/news/what-is-docker-compose-how-to-use-it/#:~:text=Docker%20Compose%20is%20a%20tool,with%20NodeJS%20and%20MongoDB%20together.