Garcia Canul Guillermo De Jesus

¿Cómo aplicar estos comandos a un entorno productivo?

En este documento veremos un número considerable de comandos que podemos usar para realizar nuestras actividades dentro de docker, sin embargo si queremos ver de manera práctica estos comandos te invitamos a leer la documentación de **CÓMO DESPLEGAR EN DOCKER**, de esta manera podrás llegar a subir tus proyectos a un entorno productivo cómo AWS.

Liga de la documentación:
https://drive.google.com/drive/folders/1I7gjvp_yZ8hUEMVQSKsvkAVrYr8l-TCO?usp=sharing

---

Comandos para manejar contenedores

Estos comandos son los que vamos a emplear cuándo queramos levantar, parar o entrar a un contenedor.

> **Consideraciones**
>
> Este comando solamente va a servir cuándo tengamos una imagen creada de manera previa

```
docker run imagen-creada
```

Corre un contenedor con la imagen que especificaste. Si no existe, la intentará bajar de las imágenes existentes.

```
docker run -d imagen-creada
```

Igual que el anterior, pero lo corre en modo **detached** (segundo plano), en un puerto especificado.

```
docker run -p 3000:3000 imagen-creada
```

Mapea el puerto local al del contenedor. Ideal para exponer el servicio.

```
docker ps
```

Te dice qué contenedores están corriendo.

```
docker ps -a
```

Muestra todos los procesos, no importa si el estado de estos está off o si le hicimos kill de manera previa.

```
docker stop id_contenedor
```

Detiene el contenedor que esté corriendo.

> **Consideraciones**
>
> Es importante que nosotros sepamos el id del contenedor que queremos detener, para eso haremos uso de docker ps, una vez que sepamos el id exacto lo podremos detener, en caso del siguiente comando también será útil conocer el id de nuestro contenedor

```
docker start id-contenedor
```

Ejecuta nuevamente un contenedor que ya habías detenido.

```
docker restart id-contenedor
```

Reinicia un contenedor.

```
docker rm id-contenedor
```

Elimina un contenedor, siempre y cuándo este no se encuentre en ejecución

```
docker exec -it id-contenedor bash
```

Te permite explorar tu contenedor como si fuera una terminal Linux.

Comandos para trabajar con imágenes

A continuación vamos a empaparnos en las opciones que tenemos para trabajar con nuestras imágenes

```
docker build -t nombre-imagen .
```

Construye la imagen a partir del Dockerfile actual.

```
docker build -f ruta/dockerfile/
```

Construye desde nuestro Dockerfile en una ruta específica.

```
docker images
```

Lista todas las imágenes que tenemos en local.

```
docker rmi imagen-nombre
```

Borra una imagen local.

```
docker tag imagen-nombre usuario/imagen-nombre
```

Etiqueta tu imagen para poder subirla a Docker Hub.

```
docker push usuario/imagen-nombre
```

Sube la imagen a Docker Hub. Necesitas estar logueado (`docker login`) .

```
docker pull usuario/imagen-nombre
```

Descarga la imagen desde Docker Hub a tu máquina.

Comandos para redes - Comunicación Intra Contenedor

Cuándo manejamos más de un contenedor, ya sea para desplegar nuestro back, front, db, APIS, necesitamos que estos se comuniquen para que nuestra app funcione, por lo que los

siguientes comandos nos ayudarán a entender cómo podemos hacerlo.

```
docker network ls
```

Te muestra las redes creadas por Docker.

```
docker network create mi-red
```

Crea una red personalizada.

> **Consideraciones**
>
> Este comando aplica en los casos que no usamos docker compose

```
docker network connect mi-red contenedor
```

Conecta un contenedor a una red (Intranet).

```
docker network inspect mi-red
```

Muestra información de la red y los contenedores conectados.

```
docker network rm mi-red
```

Elimina la red si ya no se está usando.

Comandos para volúmenes (persistencia de datos)

Muchas veces la información que estamos manejando se puede perder si no creamos volúmenes específicos para su contención, por lo que cuándo reiniciemos el servidor, podríamos estar perdiendo datos importantes, podemos evitar esto de la siguiente manera.

```
docker volume ls
```

Lista los volúmenes que tienes.

```
docker volume create mi-volumen
```

Crea un volumen nuevo.

```
docker run -v mi-volumen:/app/data imagen-nombre
```

Asocia ese volumen a un contenedor (los datos se guardan en `mi-volumen` aunque el contenedor muera).

```
docker volume inspect mi-volumen
```

Muestra información detallada del volumen.

```
docker volume rm mi-volumen
```

Borra el volumen (solo si no está en uso).

Comandos básicos de Linux/Bash

Estos son los comandos básicos cuándo trabajamos con la terminal de nuestro VPS o servidor.

```
cd carpeta
```

Nos permite movernos a una carpeta dentro de nuestro directorio.

```
ls -la
```

Listar archivos con detalle (incluso los ocultos).

```
pwd
```

Muestra nuestra ruta actual.

```
cat archivo.txt
```

Muestra el contenido de un archivo plano en consola.

Consideraciones

Si queremos editar ese archivo en específico podemos usar:

```
nano archivo.txt
```

```
mkdir nombre-carpeta
```

Creamos una carpeta

Para Finalizar

Estos comandos serán parte del día a día como desarrollador, es probable que usemos más algunos que otros, lo importante es conocer para que sirve cada comando, ya tenemos las bases, ahora lo que resta es empezar a aplicarlas, nuevamente te invitamos a poner en práctica estos comandos e ir mecanizando el uso de estos, aunque es importante saber que todos estos procesos se pueden ir automatizando, jamás será menos importante el saber para qué sirven y en qué momento podemos usarlos.