Docker Image >> Es como un instalador. Docker nos va proveer de instaladores como por decir: Mysql, MongoDB, MariaDB, Apache, etc. estas image vienen con un OS para que puedan ejecutarse.

$docker rum “image” >> Descarga la imagen y la ejecuta. También podemos primero descargar la image y luego ejecutarla.

$docker images >> Lista de las imagenes que tengo instaladas

$docker search “image” >> Busca la image en hub.dockeer.com y nos la muestra en una lista

$docker ps >> Lista de los contenedores que se estan ejecutando con base a una image.

$docker ps -a >> Lista de los contenedores que estan detenidos (stop). Muestra una especie de historial.

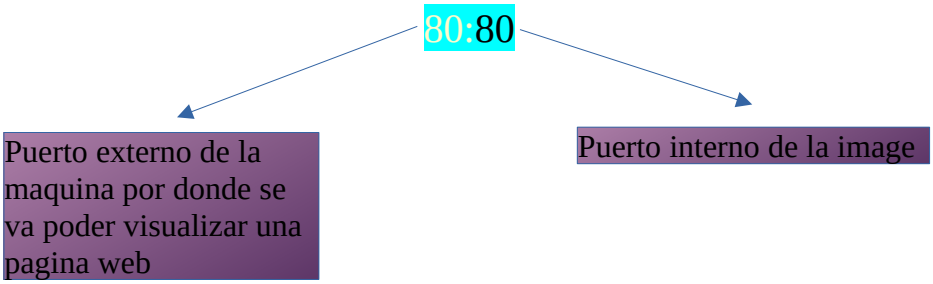
$docker ps -aq >> Muestra todos los ID de los contenedores

$docker rm“id” >> Borra el contenedor con el ID especificado

$docker rm”name” >> Borra el contenedor con en NAME especificado

$docker rm ($docker ps -aq) >> Borra todos los contenedores

$docker run -p 80:80 -d mginx >> Esta image ya esta descargada, con esta instrucción se ejecuta la image “mginx” , -d quiere decir que se ejecutara en segundo plano, -p indica por que puerto se va a poder ver el contenido de la image ya que este tiene un puerto interno.




$docker stop “id” >> Deteine al contenedor. Puede introducirse los 3 primeros caracteres y con eso bastara

$docker stop $(docker ps -aq) >> Detiene a todos los contenedores

$docker rm $(docker ps -aq) -f >> Borra a todos los contenedores. Lo primero que hace es detener al contenedor si se esta ejecutando, y luego lo borra. -f fuerza el cierre del contenedor

$docker rmi “image” >> Elimina la image

$docker rmi “id” >> También elimina la image solo que se la específica por el ID

$docker images >> Lista todas las imagenes

$docker images -aq >> Lista todos los ID de las imagenes
$docker rmi $(docker images -aq) >> Borra todas las imagenes. Sin embargo si se estan ejecutando lo primero que se debe hacer es parar los contenedores que funcionan con esa base de imagen.
$docker rm $(docker ps -aq) -f con esto borramos los contenedores, este se ejecuta antes que el comando anterior

$docker run -p 80:80 -d nginx
$docker run -p 3000:80 -d nginx
$docker run -p 4000:80 -d nginx
$docker run -p 5000:80 -d nginx

Crea varios contenedores con diferentes Ids lo que nos permite tenerlos separados uno del otro

$docker run -p 80:80 -p 3000:80 -p 4000:80 -p 5000:80 -d nginx Hace lo mismo del anterior solo crea un solo ID para todos

$docker run -d -p 80:80 –name website -v $(pwd):/usr/share/nginx/html:ro
>> -v $(pwd) ruta que se va copiar
>> /*usr/*share/nginx/html lugar donce se va pegar
>> ro cuando se le añade esto quiere decir que solo sera de lectura, es decir, que no podra modificar

$docker run -d -p 3000:80 -v $(pwd):/usr/share/nginx/html –name website nginx
>> con esto se podra modificar

$docker exec -it website bash >> Conectarme a este contenedor. Entra en la image, es decir que entrara en el OS que contiene a la image, en este caso tendra acceso en la terminal

$docker build -t “contenedor” . >> Construye una imagen

$docker build -t “usuario/contenedor” >> Construye una imagen, con este se podra subir a docker hub

Subir a Docker Hub

$docker login >> Con este comando nos perdirea nuestro usuario y contraseña

$docker push “usuario/image” >> Una vez logeado lo que hacemos es colocar nuestro usuario y el nombre de la image