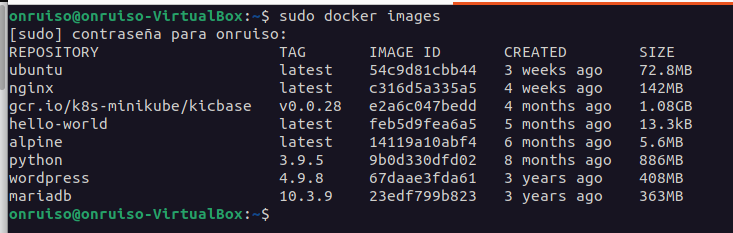
Docker Comandos

Luis Felipe Narváez Gómez. E-mail: luis.narvaez@usantoto.edu.co. Cod: 2312660. Facultad de Ingeniería de Sistemas

En el siguiente documento se podrá observar la sucesión de comandos correspondientes a la introducción de comando de la herramienta de Docker. El primer comando que veremos corresponde a ver las imágenes que tenemos instaladas dentro de nuestro Docker. El comando es el siguiente:

*sudo docker images*



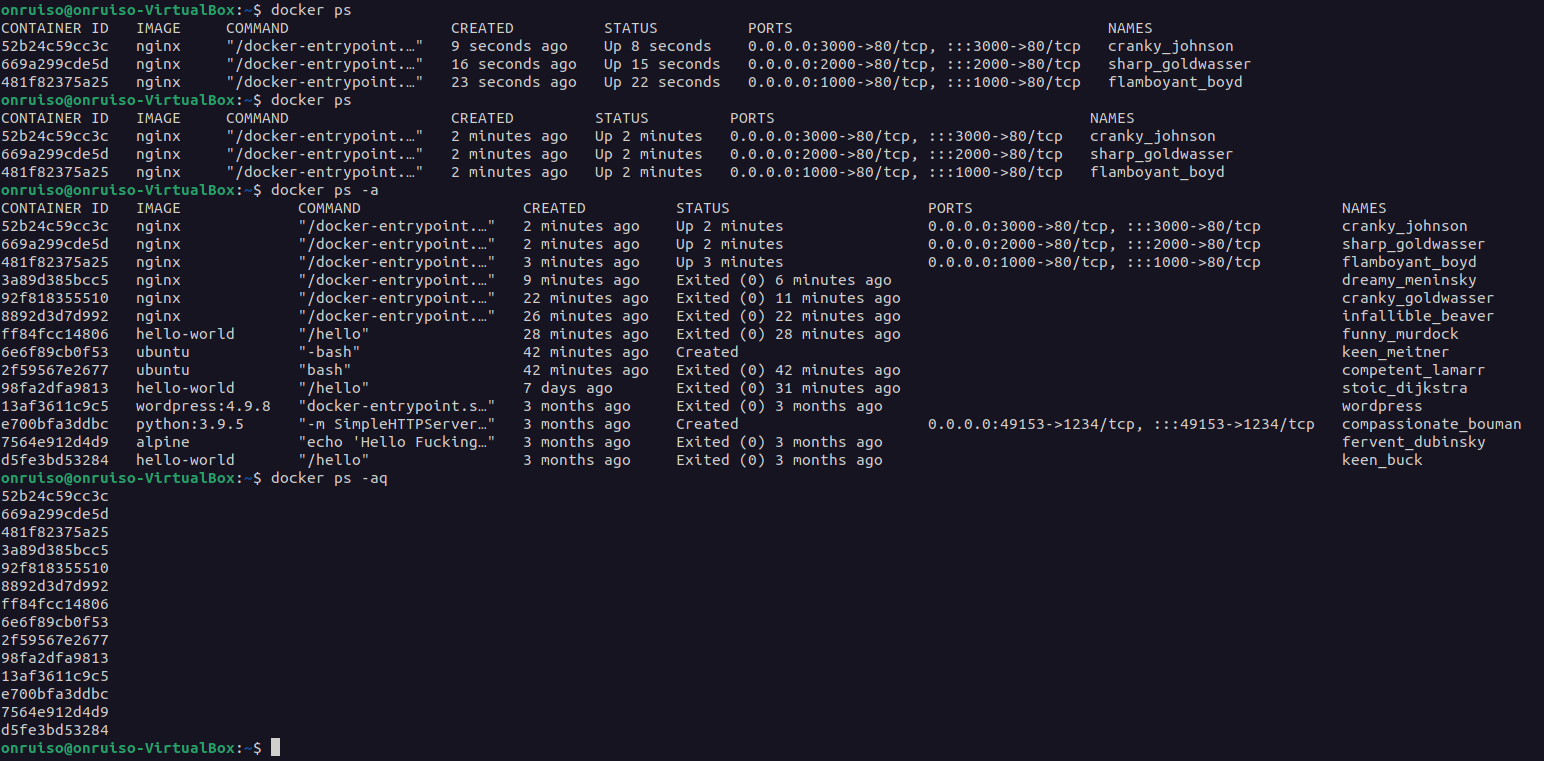
Las imágenes de docker pueden hallarse por internet con sus respectivos comandos para poder ser instalas, es recomendables in embargo buscarlas en el catalogo oficial de docker el cual se encuentra en el enlace: <https://hub.docker.com/> en estas imágenes nos daremos cuenta que la mayoría de imágenes se instalan con el comando “pull”.

Docker puede estar o haber ejecutado ya en primer o segundo plano imágenes o servicios, para ver este historial de los mismos podemos utilizar alguno de los siguientes comandos:

*sudo docker ps*

*sudo docker ps -a*

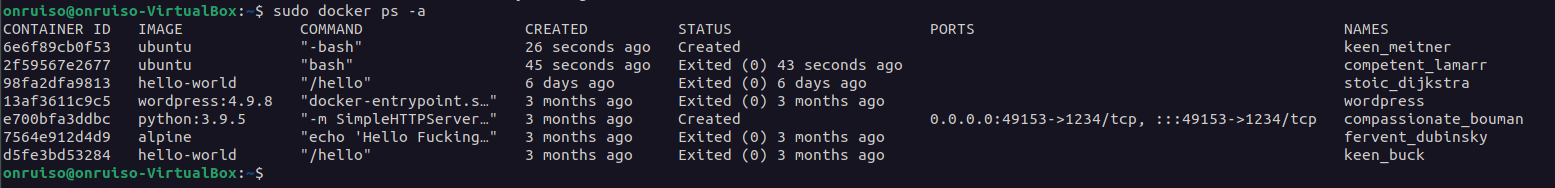
*sudo docker ps -aq*



La diferencia entre cada comando radica en la información que nos ofrece, el primer comando ofrece solo un listado de los servicios actualmente en ejecución, el segundo comando ofrece un listado detallado de los servicios que se han ejecutado desde la instalación de Docker y el tercer comando nos dice solo el CONTAINER ID de los servicios que se han ejecutado hasta el momento.

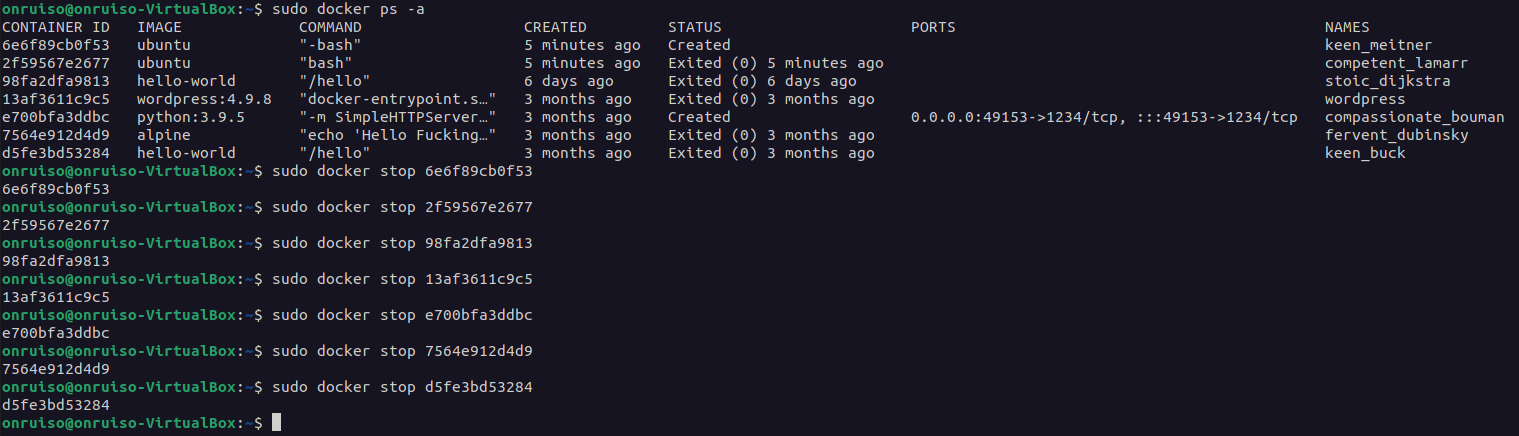
En la siguiente imagen vemos las imágenes que se han ejecutado hasta el momento con docker, utilizando el comando:

*Sudo docker ps -a*



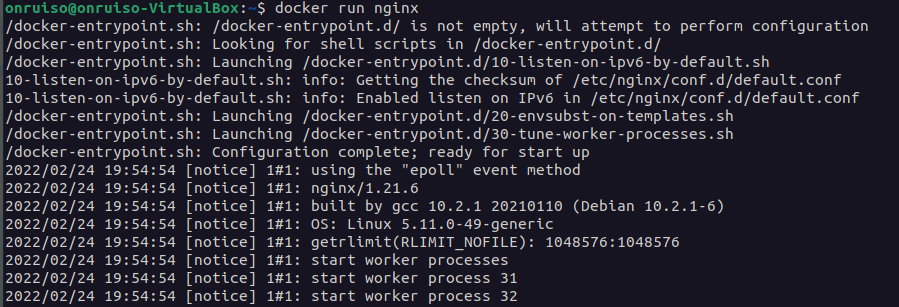
Los servicios que se están ejecutando, pueden ser parados, esto quiero decir que dejen de ejecutar sus funciones. Para esto utilizamos el comando “stop”, esto de manera básica se da sobre el CONTAINER ID del servicio que queremos detener. En la siguiente imagen procedemos a parar cada uno de los diferentes servicios que nos aparecen.

*Sudo docker stop CONTAINER ID*



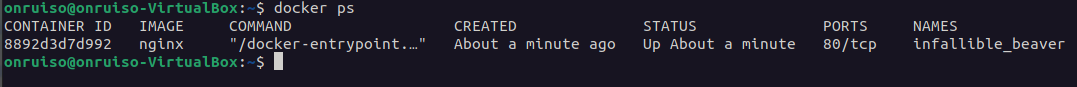
Para inicializar o correr un servicio basta con utilizar el comando “run”, esto pondrá en un puerto por defecto a accionar el servicio que especifiquemos.

*Sudo docker run SERVICIO*



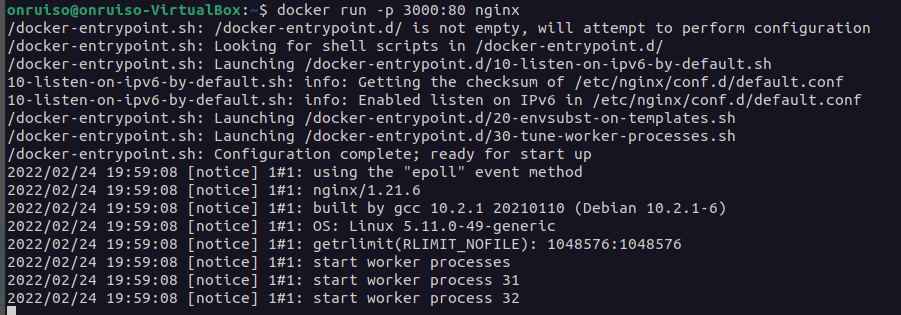
Muchas veces nos encontraremos que la terminal en la que inicialicemos el servicio queda ocupada, basta con inicializar nuevamente otra terminal para ver los servicios que estamos ejecutando en el momento con el comando:

*Sudo docker ps*

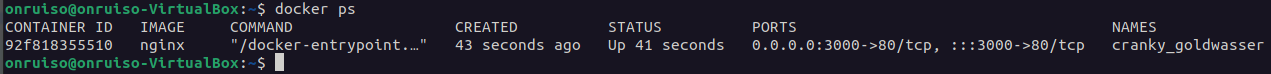


La inicialización de un servicio puede darse también en un puerto que nosotros mismos le especifiquemos, para esto utilizamos en el comando de “run” la sucesión de “-p” y a continuación el puerto en el que lo necesitamos. En la siguiente imagen pondremos de ejemplo poner a correr NGINX en un puerto especifico.

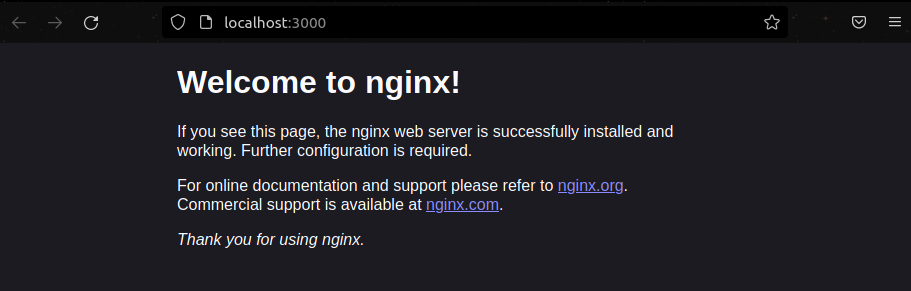
*Sudo docker run -p 3000:80 nginx*



Rectificamos que este funcionando el servicio actualmente con “docker ps”.

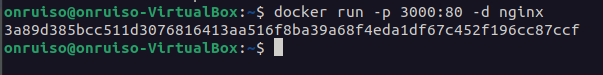


Puesto que conocemos el puerto en el que fue lanzado este servicio dentro de nuestra red local, podemos acceder por cualquier buscador de red como Google Chrome o bien Software especifico como Postman y/o Insomnia; verificando la ejecución actual del servicio. Se debe tener en cuenta que a veces en el cache de un navegador web, en caso de haber ejecutado un servicio previo en el mismo puerto, la información de esta anterior queda cargada allí, así se monte un servicio diferente posteriormente.

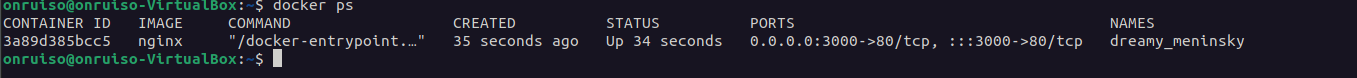


El anterior servicio se esta ejecutando en primer plano, esto lo constatamos por la inutilización posterior al lanzamiento del servicio de la terminal. Para poder ejecutar una utilidad en segundo plano, podemos utilizar la parte del comando “-d”, como en el siguiente comando.

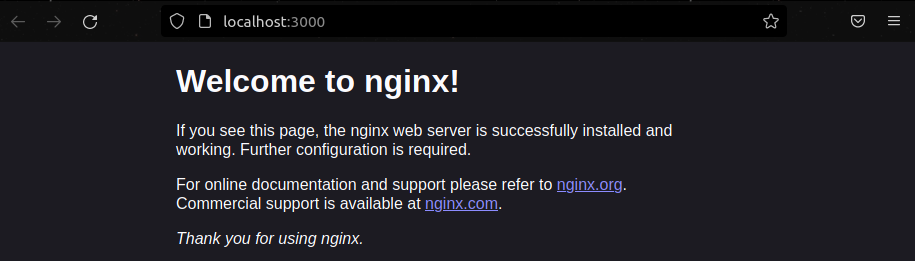
*Sudo docker run -p 3000:80 -d nginx*



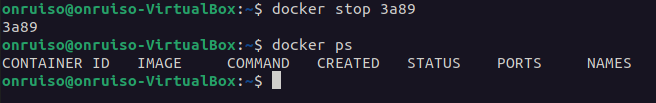
Verificación del servicio en ejecución con el comando “docker ps”.



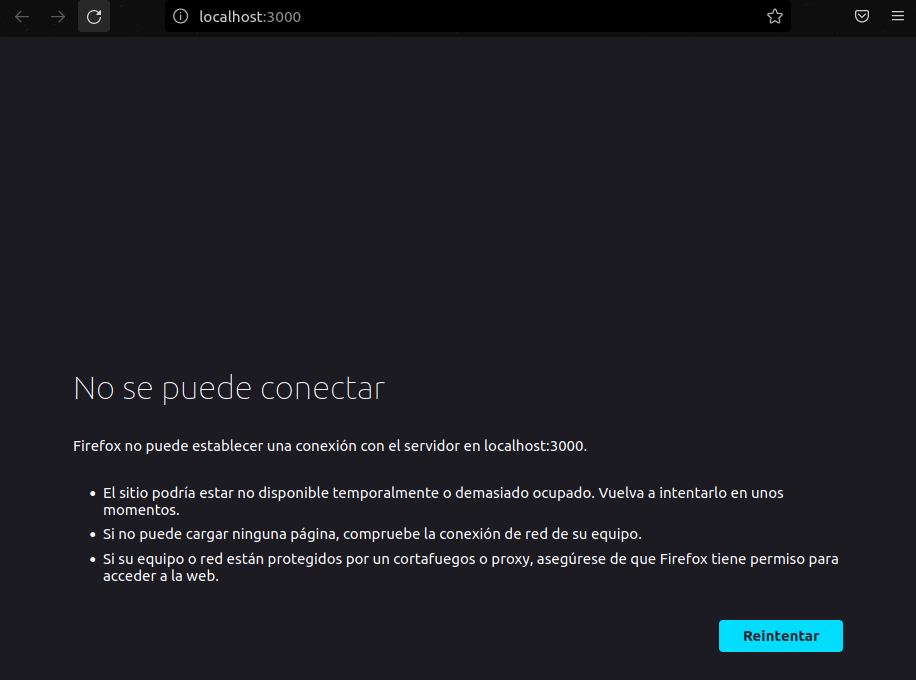
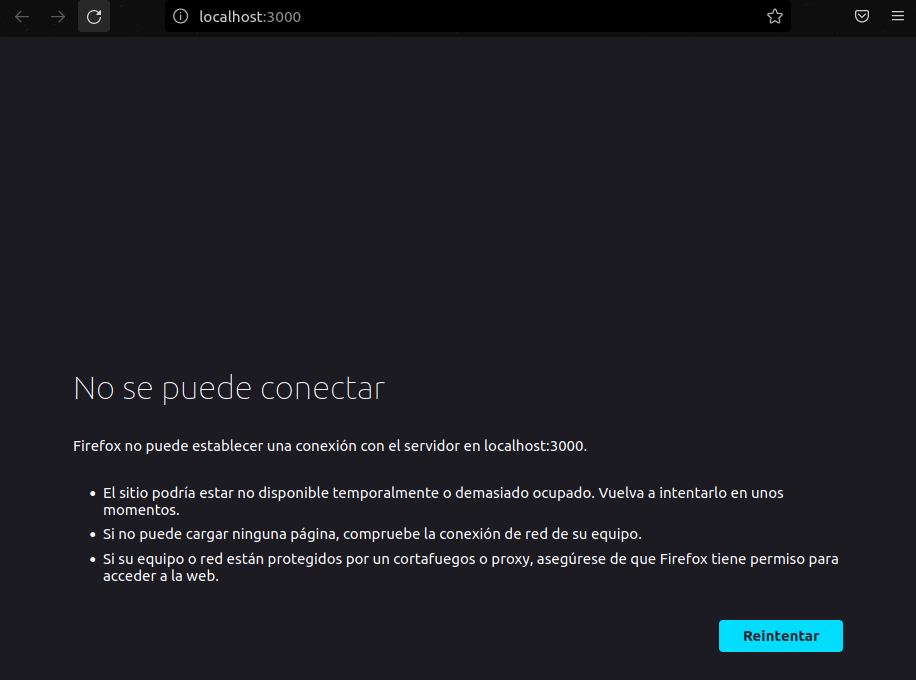
Verificación del servicio en un navegador web, ingresando el puerto en el que se ejecutó.



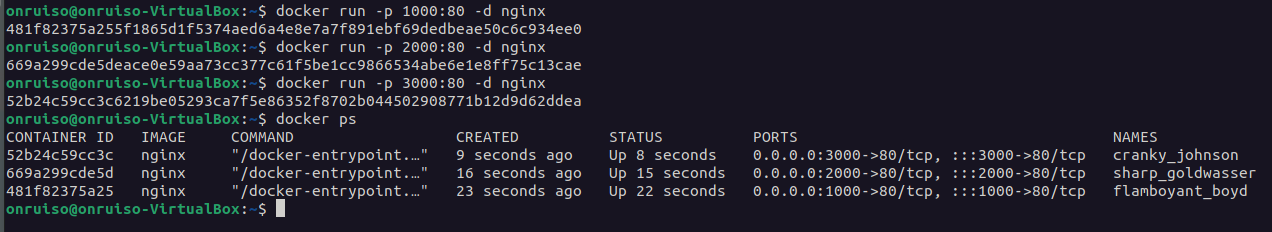
Ahora bien podemos parar el servicio con el CONTAINER ID, asi como vimos anteriormente, sin embargo este comando también funciona con la discriminación de los tres a cuatro primeros caracteres del servicio especifico que necesitamos detener.



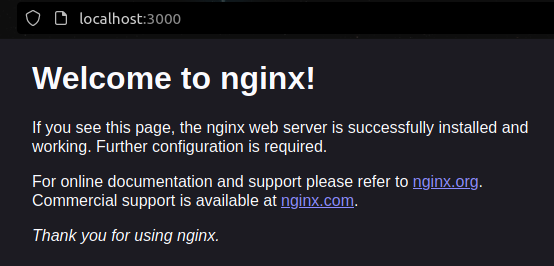
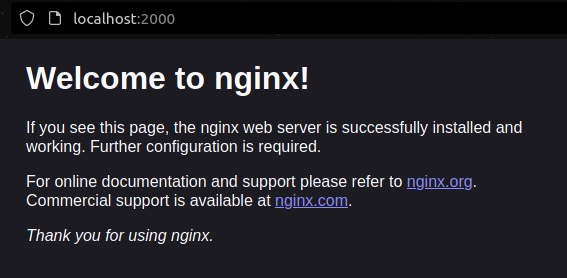
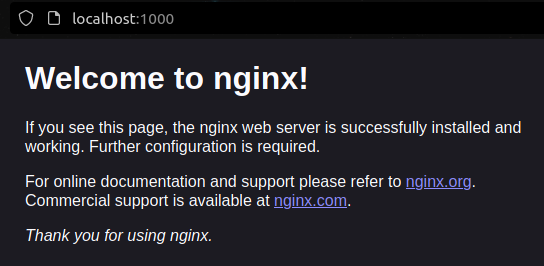
Una vez detenido el servicio, podemos comprobar su no accionalidad con el comando “docker ps” el cual no mostrara el servicio previamente parado, de la misma manera en el navegador web no se podrá acceder al servicio.



Los servicios pueden ser ejecutados tantas veces queramos, esto asignando un puerto distinto a cada servicio que ejecutemos, esto puede hacerse de manera tradicional, escribiendo el comando una vez por cada puerto que queramos.

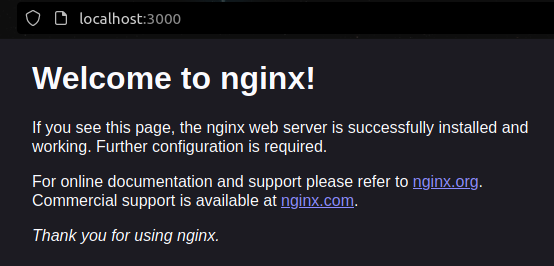
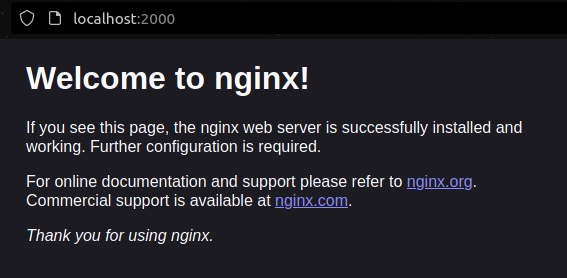
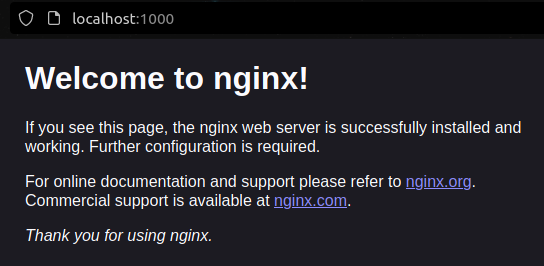
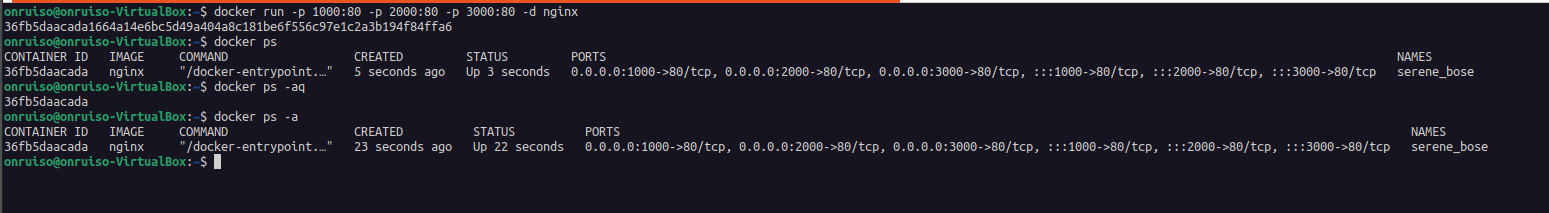


Como podemos ver en la anterior imagen de comando, se inicializo tres servicios NGINX en puertos distintos y se comprobó por consola su estado activo de funcionamiento, lo mismo puede ser corroborado mediante el uso del navegador web como se muestra a continuación.



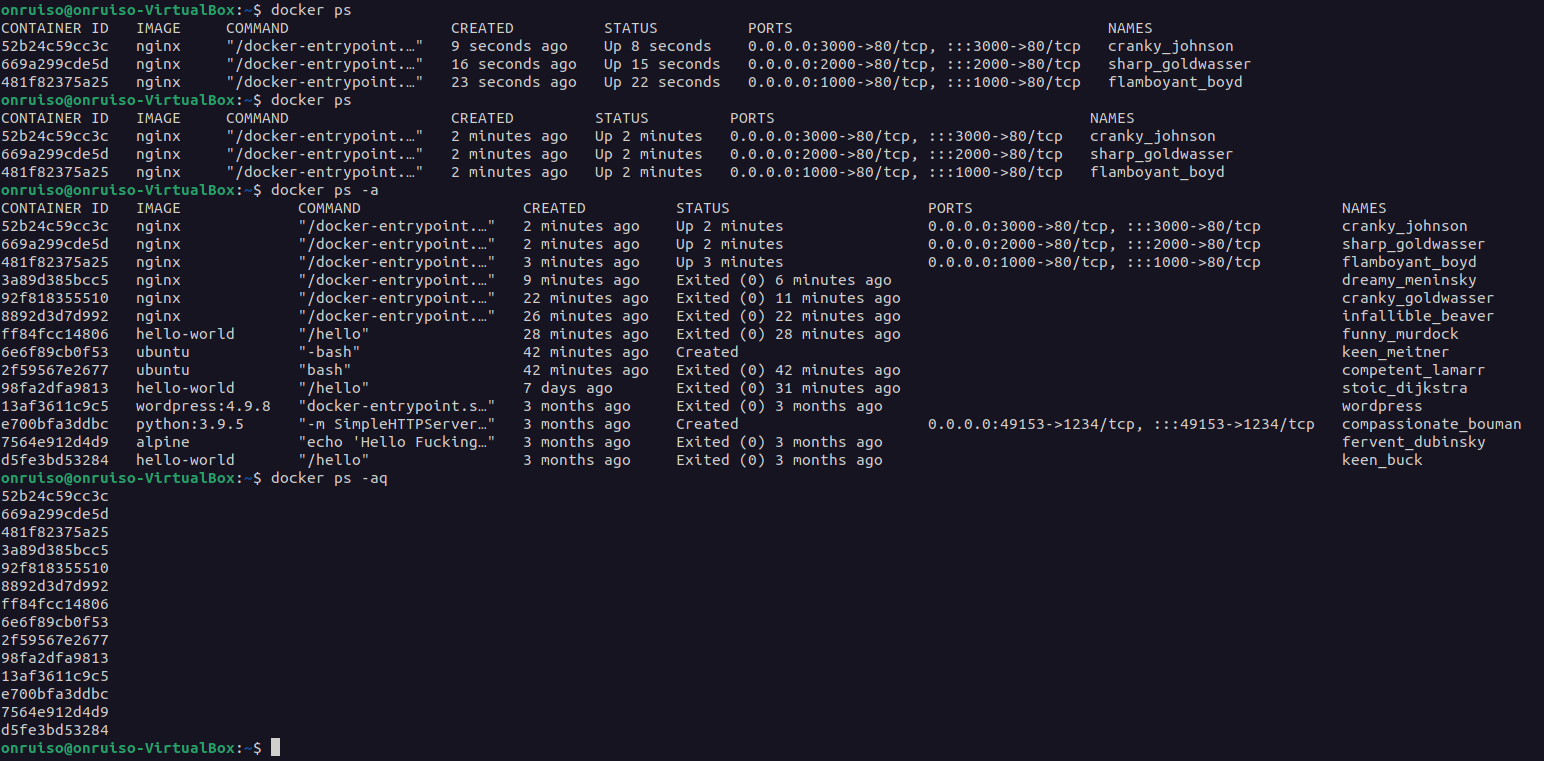
También existe la propiedad de poder inicializar esta serie de servicios de manera consecutiva utilizando un dolo comando como se muestra a continuación poniendo cada puerto de manera sucesiva separada por el comando “-p” que estipula el uso de puerto, se debe tener en cuenta que esto se logra estableciendo que los puertos que ingresemos utilizaran un único servicio en común.

*Sudo docker run -p 1000:80 -p 2000:80 -p 3000:80 -d nginx*



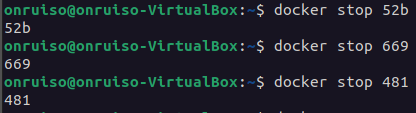
Podemos remover por completo los servicios que tenemos en ejecucion o del historial de ejecucion de docker, para esto utilizamos el comando “rm”. Para poder remover un servicio debemos tener encuenta que este no este en ejecucion en el momento, si es el caso con el comando “stop” podemos detenerlo y posteriormente retirarlo.

Primero veremos que servicios tenemos dentro de docker con el comando “sudo docker ps -aq” o bien otro método de “ps”, de ahí veremos que servicio o servicios queremos retirar.

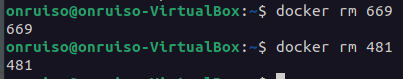


Una vez localizados podemos iniciar su detención con el comando “stop” y luego su remoción con el comando “rm”.

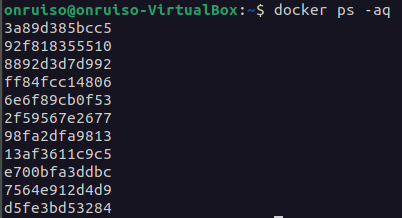
*Sudo docker stop CONTAINER ID*



*Sudo docker rm CONTAINER ID*

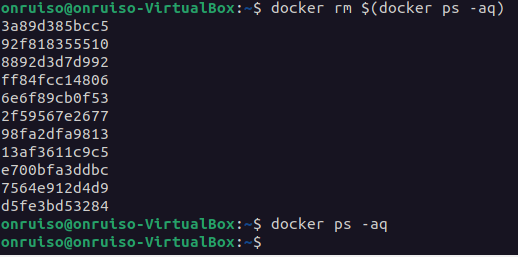


Una vez removidos los servicios podemos comprobarlo con alguno de los comandos de listado de “ps”.



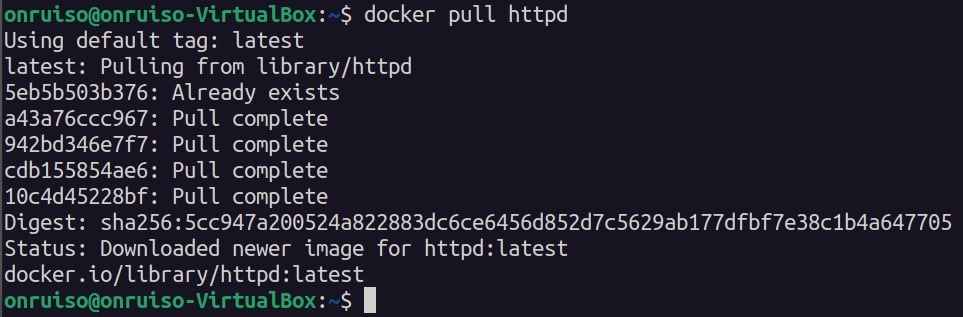
Como podemos darnos cuenta puede ser un poco engorroso eliminar uno por uno los servicios que queremos, mas cuando estos son varios. Si es el caso que queremos remover todos los que tenemos en docker, podemos utilizar el siguiente comando. (recordemos que estos deben estar previamente detenidos).

***Sudo docker rm $(docker ps -aq)***



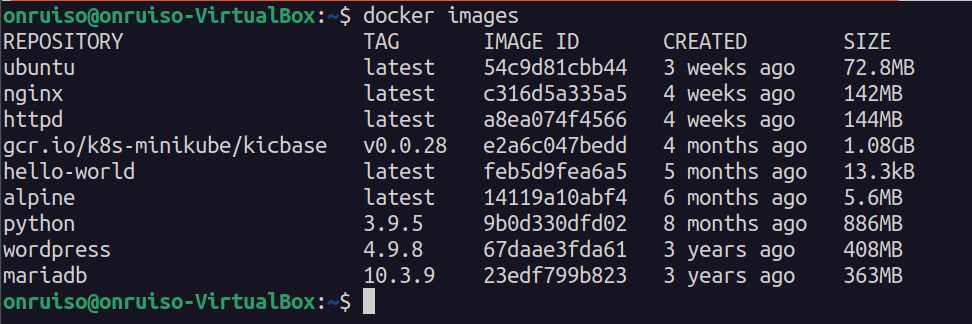
Ahora bien, del mismo modo que como venimos trabajando hasta el momento, podemos instalar otro tipo de imágenes como lo es la de httpd, la cual podemos conseguirla en el enlace: <https://hub.docker.com/_/httpd> y que instalamos en Docker con el comando “pull”.

***Sudo docker pull httpd***

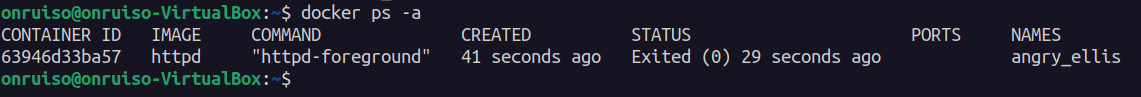
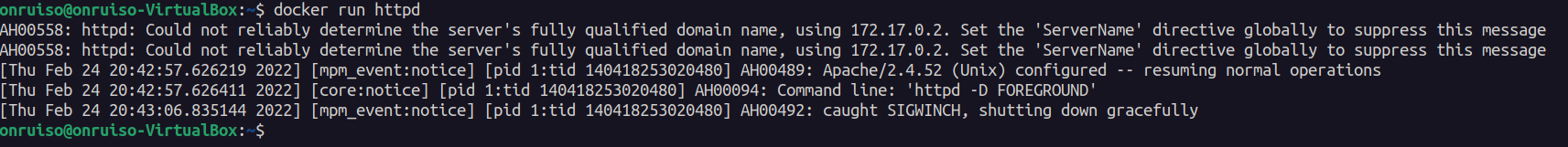


Una vez instalada la nueva imagen, podemos comprobar que la misma este en docker con el comando “images” el cual muestra a detalle las imágenes que tenemos instaladas dentro de Docker.

*Sudo docker images*



Procedemos a hacer el mismo proceso de ejecución de servicios que se llevaron con NGINX pero ahora con HTTPD. Primero tenemos que inicializar el servicio con el comando con RUN y verificar por listado.



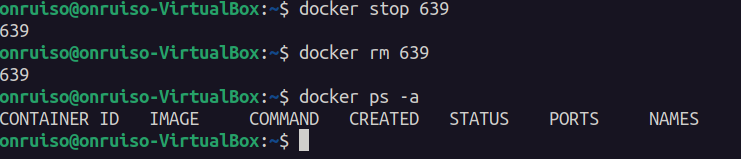
Podemos parar el mismo servicio con el comando STOP



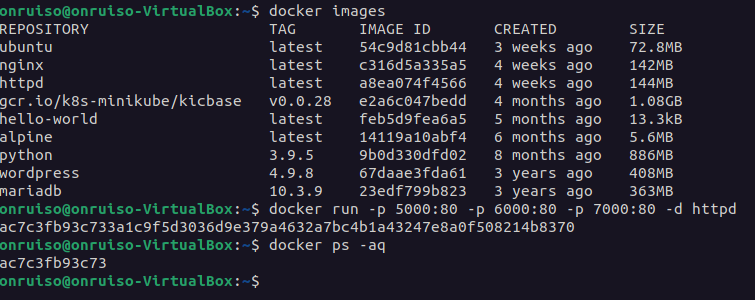
Asi como volverlo a poner en acción con el comando START



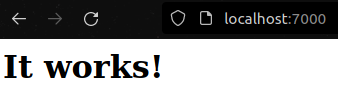
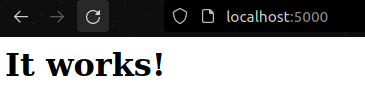
De la misma manera que con el servicio de NGINX, podemos detener y remover el servicio activo con el comando RM el cual es la simplificación de REMOVE.



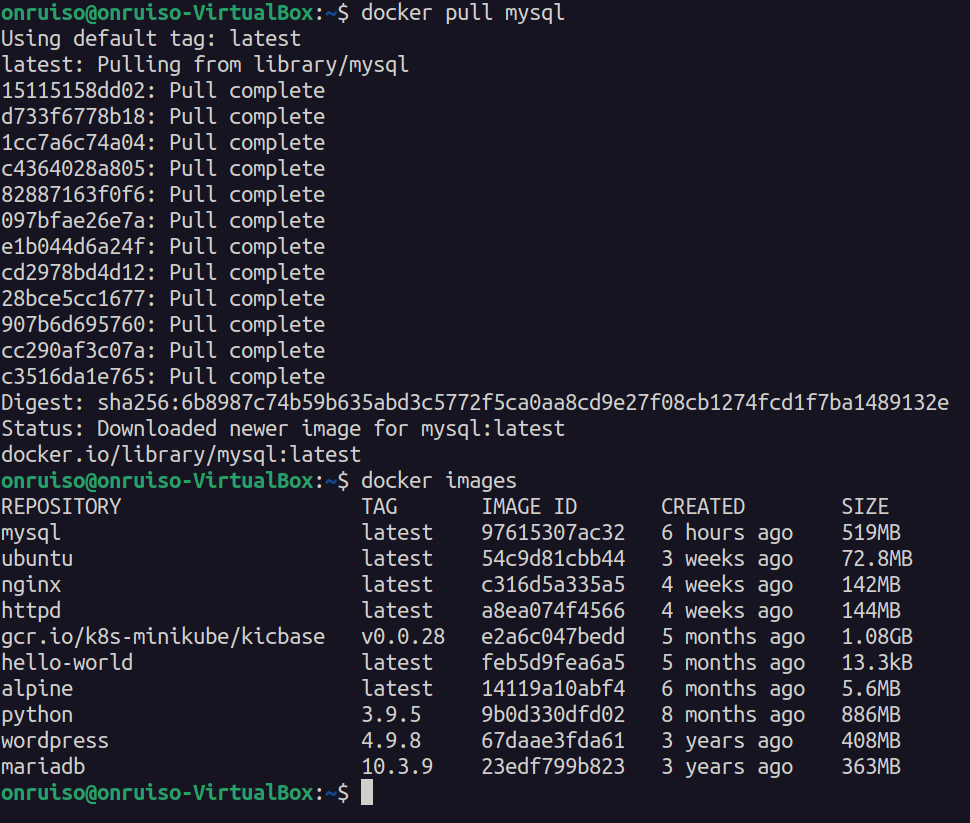
Asi que como con el anterior servicio, podemos inicializar tantos servicios de HTTPD como puertos virtuales tengamos disponibles en nuestra maquina virtual.



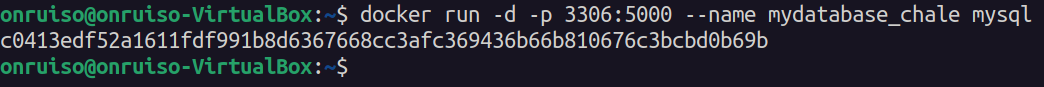
Y del mismo modo comprobar que los mismos funcionen también en el navegador web o un software especializado como Postman o Insomnia.



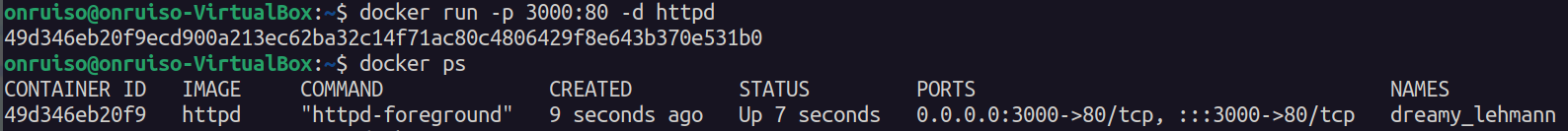
Repetimos el proceso para la instalación de la imagen de MYSQL en Docker, utilizando nuevamente el comando de PULL.



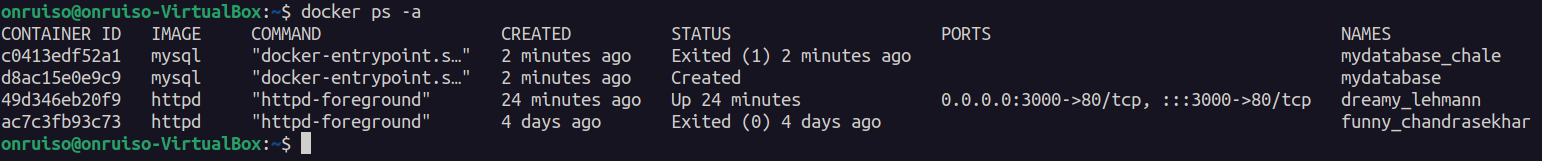
Inicializar una database en mysql

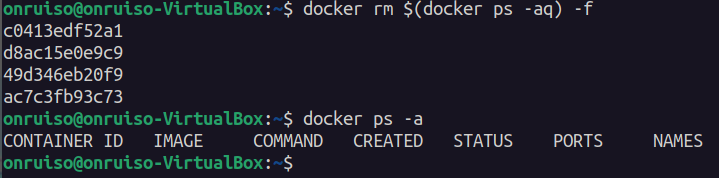


Inicializar un puerto httpd.

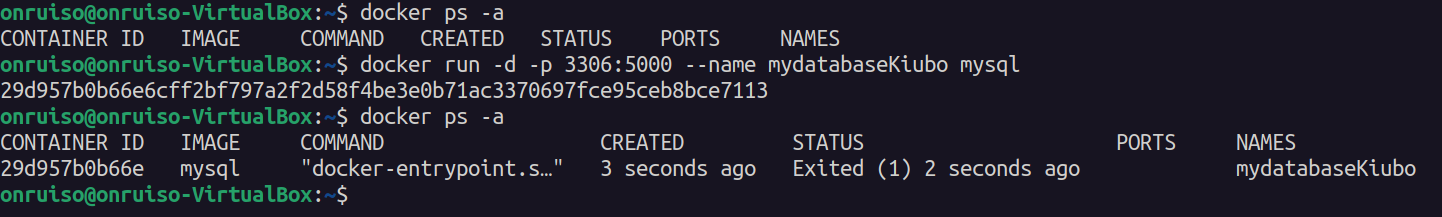


Ver que procesos se están ejecutando y forzar cierre (-f)

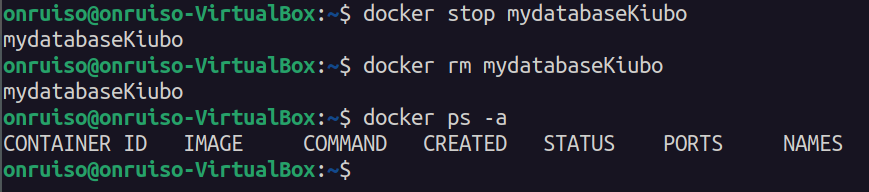




Reiniciar un puerto con MySQL, nombre de database libre



Parar y remover esta database con el nombre que le asignamos



Llamar atributos específicos

