PRÁCTICA FINAL DOCKER

EJERCICIO 1: RELLENAR LOS HUECOS

1- Hacer un Hola Mundo

docker run hellow-world

2- Levantar un servidor Apache en Docker, y ser capaz de acceder desde nuestro equipo por el navegador.

Docker run -d -p 80:80 –name caso\_practico httpd

3- Copiar un fichero index.html dentro de un contenedor, para modificar el que vemos desde el navegador (o si lo estamos haciendo por comandos, el que veamos con el comando curl)

docker cp archivo\_a\_copiar nombre\_contenedor:/ruta\_contenedor

EJERCICIO 2: CASO PRÁCTICO VISTO EN CLASE

* Crea tu propia imagen basada en httpd que muestre el index con el fondo rojo
* Crea tu propia imagen basada en httpd que muestre el index con el fondo verde
* Crea un fichero docker-compose.yml en el que haya 3 servicios

1. un servidor nginx que se construlla a partir de un Dockerfile

2. Un servidor httpd que muestre el fondo verde

3. Un servidor httpd que muestre el fondo rojo

Todos estos servicios estarán en la misma red, por lo que el servidor de nginx podrá acceder a los otros.

Además nginx será accesible desde localhost:8080, el servidor rojo desde localhost:8081, pero el verde no será accesible desde fuera; solamente lo podrán ver los contenedores.

* **NGINX**
* Hay que modificar el fichero /etc/nginx/conf.d/default.conf del contenedor antes de arrancarlo

OBJETIVO: Tener toda la infraestructura montada, simplemente ejecutando

docker compose -f "docker-compose.yml" up -d --build

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**Punto 1 & 2**

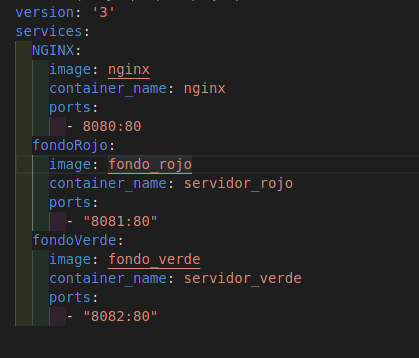
En un archivo dockerfile escribimos los siguiente.

****

En el mismo directorio, en la terminal escribimos **docker build -t nombre\_imagen .**

**Punto 3**

Creamos un archivo **docker-compose.yml** y escribimos



Aquí estamos indicando al docker-compose que levante 3 servicios. El pimero servicio con la imagen de nginx, el segundo y tercer servicio con las imagenes creadas anterior mente.

Luego levantamos el docker-compose escribiendo **docker-compose up**

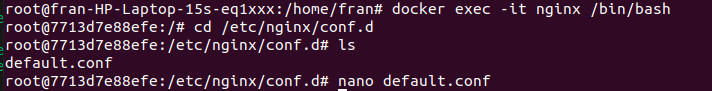
Si escribimos localhost:8080 nos mostrará que el servicio de nginx si está funcionando.



Como podremos notar el servicio nginx no está enlazando el resto del servicios, ya que si escribimos localhost:8080/rojo o verde, esto no funcionará.

Hay que modificar el archivo **default.conf** que se encuntra en el contendor de nginx.

Para ello porcedemos a:

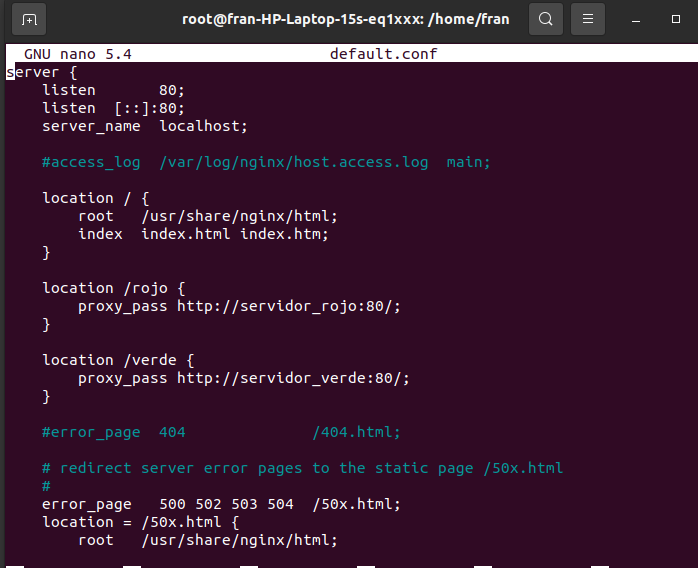
Despues hay que hacer un nano sobre el archivo default.conf, si no funciona habŕa que esintalar nano.

Para ello escribimos los comando

**apt-get update**

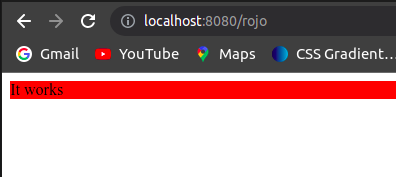
**apt-get install nano**

Una vez hecho nano sobre el archivo lo editamos de la siguiente manera.

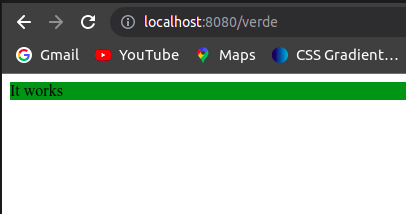


Una vez editado y guardado, volemos a arrancar el archivo **docker-compose.yml**

Una vez hecho esto podemos comprobar que si escribimos localhost:8080/rojo o localhost:8080/verde estos nos mostrará la imagen deseada.



Si escribimos localhost:8081 también se mostrará el fondo rojo



Entregar un ZIP, con toda la estructura de carpeta y ficheros empleados.

Para que pueda levantarlo TODO, simplemente haciendo el comando anterior, sin necesidad de tocar nada más.

A la hora de entregar los ejercicios prácticos, entregar DOCUMENTACIÓN, indicar por qué habéis tomado la decisión de emplear un parámetro u otro en el dockerfile y docker-compose.

Si habéis tenido problemas con alguna parte, cómo la habéis resuelto, o si no habéis sido capaces, indicar que situaciones habéis valorado para intentar resolverlo.

Adjuntar capturas de pantalla, trocitos de código, lo que necesitéis para explicaros.

EJERCICIO 3: RETO FINAL

**:80**

* Hay un servidor nginx haciendo redirecciones
* Con un volumen compartido con nuestro host, en el que podamos crear/modificar archivos.
* Por defecto, se creará el contenedor con los comando Python.
* Y se descargará un hello-world en Python de un repositorio GitLab [Ejemplo: <https://github.com/dbarnett/python-helloworld>]
* En el momento que entremos por consola al Docker, lo primero que se ejecutará será el hello-world

**:80/git**

* Aplicación git [gitlab/gitlab-ce]
* El usuario y contraseña por defecto son:
  + root
  + cat /etc/gitlab/initial\_root\_password

Desde el servidor peticiones se tendrá que poder hacer peticiones git al servidor de GitLab

Documentación de ayuda para este paso: http://localhost/help/user/ssh.md

*PARTE OBLIGATORIA:* Si no somos capaces de hacer que ese docker de GITLAB sea funcional. Esas dos máquinas tienes que ser capaces de interactuar entre si.

*MINIPUNTO I:* Que desde el docker principal, pueda mandar ficheros al otro; debe existir una comunicación. (Ejemplo con el comando scp, o habilitando algún servicio con sus puertos, etc)

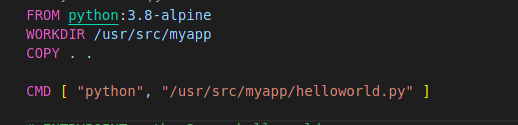
*MINIPUNTO II:* Que se puedan hacer peticiones git desde un servidor a otro (Mucha atención a los servicios expuestos en cada uno de los contenedores, y de los puertos que necesitan.

Aquí muy importante la DOCUMENTACIÓN, sobre todo la parte de los MINIPUNTOS, que de verdad lo habéis intentado, y os habéis peleado con docker, para que se comuniquen entre sí.

**Parte 1:**

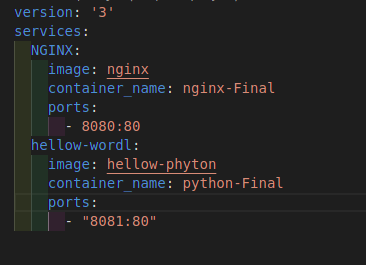
Creo un dockerfil con las siguientes indicaciones. En la parte de de COPY se le indica que copie todo lo que hay en el direcotorio en el que se encuentra el dockerfile. En nuestro caso copia la aplicación de python descargada de github.

Con WORkDIR le decimos que cree el direcotorio myapp donde se guardará la aplicación y con CMD le indicamos que ejecute la aplicación helloworld.py

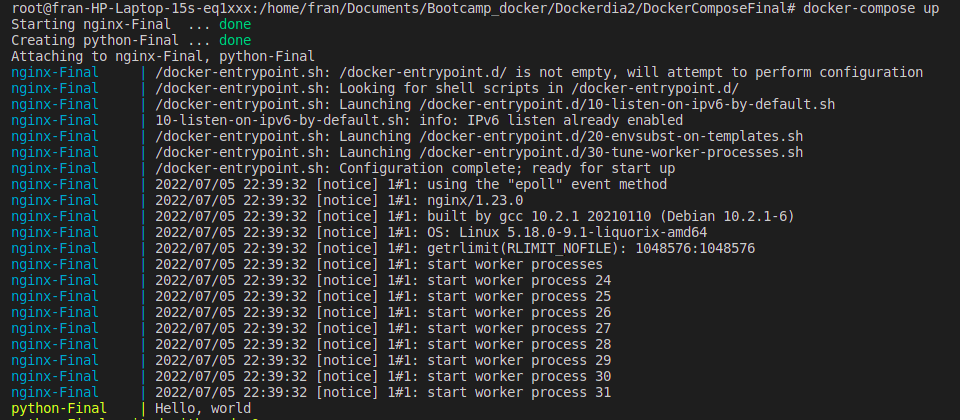
****

Ejecutamos el comando **docker build -t hellow-python**(En mi caso) **.**

Unavex construida la imagen creamos un docker-compose en el que le indicamos que levante un servicio nginx en el puerto 80 y la imagen creada posterior mente en otro puerto distinto.



Ejecutamos comando **docker-compose up** en el directorio en el que este se encuentra.

Como se observa, al final aparece la frase Hello, world.

**Parte 2: En blanco**

EJERCICIO 4: SUGERENCIAS, OPINIONES DE LA FORMACIÓN. PUNTOS QUE TE HAN GUSTADO, PUNTOS A MEJORAR.

La formación de docker me ha gustado ya que la formadora se ha implicado mucho ya que aparte de dar la teoría le ha dedicado mucho tiempo ha explicar en la práctica como se hace, lo que ha ayudado a entender mucho mejor como se hace.

Un punto negativo de esto es que hemos estado ocupando horas fuera de la formación, lo que es comprensible ya que teniamos muchas dudas y no daba tiempo en una hora y medía a resolverlas todas.