Ejercicio 1. (0,5 pts)

- **Para todos los contenedores que puedan estar o no activos en tu equipo en un único comando.**
- **Tras la ejecución comprueba que no queda ningún contenedor activo.**
- **Escribe los dos comandos que has ejecutado y sube una captura que demuestre que no hay ningún contenerdor en ejecución**

Respuesta:

docker stop $(docker ps -aq)

docker ps

Ejercicio 2. (1,5ptos)

- **Crea un contenedor a partir de una imagen “ubuntu” llamado “examendocker” de forma interactiva y abre la consola del contenedor.**
- **Escribe el comando que has utilizado**
- **Sal del contenedor**
- **Accede a la terminal desde el contenedor parado**
- **Escribe el comando que has utilizado**
- **Sal del contenedor**
- **Accede de nuevo al contenedor usando el comando exec**
- **Escribe los comandos que has utilizado**
- **Instala “nano”**
- **Crea un fichero llamado “ejercicio2_examen.txt” en la carpeta “/root/examen”**
- **Sal del contenedor**
- **Copia el fichero a la carpeta “/home/{dam/daw}/docker” de tu equipo**
- **Escribe el comando que has utilizado**
- **Crea una imagen a partir del contenedor y súbela a DockerHub con el nombre “{user}/examen” y con etiqueta de la versión “2022”**
- **Escribe los comandos que has utilizado**

Respuesta:

docker run -it --name examendocker ubuntu /bin/bash

docker start -ai examendocker

docker start examendocker

docker exec -it examendocker /bin/bash

apt update

apt install nano

mkdir /root/examen

touch /root/examen/ejercicio2_examen.txt

docker cp examendocker:/root/examen/ejercicio2_examen.txt /home/daw/docker/

docker commit -m "rosquilletas/examen" examendocker rosquilletas/examen:2022

docker push rosquilletas/examen:2022

Ejercicio 3. (2ptos)

Crea un fichero Dockerfile con las siguientes caracteristicas:

1. **Como imagen base: php:8.0-apache.**
2. **Ejecuta el comando para instalar las siguientes librerías o programas: tmux, vim, zip y libxrender.**
3. **Crea el directorio de trabajo /code.**
4. **Publica los puertos 443 y 80 del contenedor a los 4443 y 8080 de tu equipo.**
5. **Copia el fichero llamado examen.zip ubicado en la misma carpeta que el Dockerfile a la carpeta /root/examen del sistema de archivos de la imagen. El zip debe poder descomprimirse.**
6. **Al finalizar la imagen queremos que se ejecute el comando /bin/bash**
7. **Construye la imagen y llámala "examencustom"**
8. **Crea un contenedor llamado "customcont" a partir de la imagen construida en el paso anterior.**

Escribe el contenido del Dockerfile.

Escribe los comandos que has utilizado para crear la imagen y el contenedor a partir de la imagen.

Haz una captura de pantalla de la ejecución de docker ps -a para mostrar el contenedor construido a partir de la imagen.

Respuesta:

FROM php:8.0-apache

RUN apt update && apt install tmux && apt install vim && apt install zip

WORKDIR /code

EXPOSE 4443:443 8080:80

ADD ./examen.zip /root/examen

CMD ["/bin/bash"]

docker build -t examencustom ./

docker run -it --name customcont examencustom /bin/bash

docker ps -a

Ejercicio 4. (0,5pto)

- **Crea un red llamada “exanet”.**
- **Asocia la red al contenedor “examendocker”.**
- **Verifica que el contenedor “examendocker” está dentro de la red.**
- **Escribe los tres comandos que has utilizado.**

Respuesta:

docker network create exanet

docker network connect exanet examendocker

docker network inspect exanet

Ejercicio 5. (3 pts)

El Imperio ha sido atacado por un grupo de separatistas y ha conseguido hackear la web del imperio. Necesitan de forma rápida poder restaurarla para que las comunicaciones y noticias de sucesos sigan estando al alcance de los rebeldes.

El imperio ha contactado con un grupo de jóvenes padawan con los que quiere delegar dicha carga de trabajo y han dejado las siguientes instrucciones.

Nos han facilitado un zip "Episodio I - The Coder's Menace" en el que encontraremos los ficheros que se usarán para restaurar la web del imperio.

Sobre la carpeta raíz, tenemos que implementar el fichero encargado de configurar los servicios para poder restaurar el funcionamiento de la web. Es decir dentro de la carpeta que contiene los ficheros, es donde crearemos el fichero docker-compose.yml en versión 2.

Detalle de los servicios a implementar:

1. **web:**

El nombre del contenedor que generará se llamará "starwars". Y parte de la versión más reciente de la imagen httpd.

Creamos dos volúmenes. En uno mapearemos el contenido completo de la carpeta a la carpeta /usr/local/apache2/htdocs. El otro volumen copiará el contenido de la carpeta provisioning/etc/apache2/sites-enabled/vhost.conf a la carpeta /usr/local/apache2/hosts.

Abrimos el puerto http al puerto 8010.

Requerimos que se inicie de forma automática y el contenedor debe estar siempre siempre operativo/activo.

Finalmente lo asociaremos a la red tatooine_network.

2. **db:**

El nombre del contenedor que generará se llamará "starwars_db". Y parte de la versión "alpine3.17" de la imagen "postgres".

Se necesitan dos variables de entorno, con nombre POSTGRES_USER y POSTGRES_PASSWORD con los valores que creáis oportunos.

Creamos un volúmen docker llamado "dbs_data". En este volúmen se mapeará con el contenido del directorio: /var/lib/postgresql/data.

Abrimos el puerto local 5432 al 5432 para poder conectarnos a administrar la base de datos.

Requerimos que se inicie de forma automática.

Finalmente lo asociaremos a la red tatooine_network.

3. **Para acabar con los servicios, el último servicio a generar será una base de datos en memoria. El servicio se llamará "cache".**

cache:

El nombre del contenedor que generará se llamará "starwars_cache". Y parte de la versión más reciente de la imagen "redis".

Creamos un volúmen docker llamado "cache". En este volúmen se mapeará con el contenido del directorio: /data.

Abrimos el puerto local 6379 al 6379 para poder conectarnos a administrar la base de datos.

El contenedor, al iniciarse, tiene que lanzar el siguiente comando: redis-server --save 20 1 --loglevel warning --requirepass eYVX7EwVm

Requerimos que se inicie de forma automática.

Finalmente lo asociaremos a la red tatooine_network.

Para comprobar que todo está correctamente implementado, lanzamos el comando de composición de servicios. Si no nos da ningún error, accederemos a la siguiente url para comprobar si muestra el contenido de forma correcta.
http://localhost:8010/

Sube en la tarea tu fichero docker-compose.yml