

TAREA DOCKER

1.Instala docker en una máquina y configúralo para que se pueda usar con un usuario sin privilegios.

Si queremos usar el cliente de docker con un usuario sin privilegios, usaremos el comando:

```
usermod install docker.io
```

2.Ejecuta un contenedor a partir de la imagen hello-word.

Primero ejecutamos la imagen hello-world ejecutando el comando de docker:

```
docker run hello-world
```

Comprueba que nos devuelve la salida adecuada.

```
daw@daw-docker:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Comprueba que no se está ejecutando.

Para comprobar que contenedores se están ejecutando, usamos el comando:

```
docker ps
```

```
daw@daw-docker:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
daw@daw-docker:~$
```

Lista los contenedores que están parados.

```
docker ps -a
```

```
daw@daw-docker:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
daw@daw-docker:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
6e92af96c87e   hello-world   "/hello"   9 minutes ago   Exited (0) 9 minutes ago   PORTS   NAMES
amazing_euclid
```

Borra el contenedor.

```
docker rm amazing_euclid
```

3.Crea un contenedor interactivo desde una imagen debian.

```
docker run -it debian bash
```

```
daw@daw-docker:~$ docker run -it debian bash
Unable to find image 'debian:latest' locally
latest: Pulling from library/debian
bbeef03cda1f: Pull complete
Digest: sha256:534da5794e770279c889daa891f46f5a530b0c5de8bfb5e40394a0164d9fa87
Status: Downloaded newer image for debian:latest
root@cdb39004f7a7:/#
```

Instala un paquete (por ejemplo nano).

Para instalar un paquete, primero actualizamos la base de repositorios de linux:

```
apt-get update
```

Después instalamos el paquete nano:

```
apt-get install nano
```

Sal de la terminal, ¿sigue el contenedor corriendo? ¿Por qué?.

```
root@cdb39004f7a7:/# exit
exit
daw@daw-docker:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
cdb39004f7a7   debian   "bash"    2 minutes ago   Exited (0) 4 seconds ago           agitated_chatelet
daw@daw-docker:~$
```

El contenedor no está corriendo. ya que no lo hemos arrancado como demonio.

Vuelve a iniciar el contenedor y accede de nuevo a él de forma interactiva. ¿Sigue instalado el nano ?.

```
daw@daw-docker:~$ docker start -i agitated_chatelet
root@cdb39004f7a7:/# nano
root@cdb39004f7a7:/#
```

Nano sigue instalado en el contenedor, ya que se guardan los datos de la imagen.

Sal del contenedor, y bórralo. Crea un nuevo contenedor interactivo desde la misma imagen. ¿Tiene el nano instalado?

```
daw@daw-docker:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nextcloud     latest    88161e9426f1   2 days ago    1.02GB
nginx         latest    a99a39d070bf   3 days ago    142MB
httpd         2.4      463980270363   3 days ago    145MB
debian        latest    5c8936e57a38   3 days ago    124MB
busybox       latest    66ba00ad3de8   10 days ago    4.87MB
mariadb       latest    a748acbaccac   5 weeks ago    410MB
ubuntu        latest    6b7dfa7e8fdb   5 weeks ago    77.8MB
hello-world   latest    feb5d9fea6a5   15 months ago  13.3kB
daw@daw-docker:~$ docker run -it 5c893
root@98acfd5b977:/# nano
bash: nano: command not found
root@98acfd5b977:/# S
```

Creamos un nuevo contenedor interactivo usando la id de la imagen que ya tenemos en local, cuando intentamos ejecutar nano, no existe.

Es una imagen nueva por lo que todos los datos que teníamos antes, se pierden.

4.Crea un contenedor demonio con un servidor nginx, usando la imagen oficial de nginx. Al crear el contenedor, ¿has tenido que indicar algún comando para que lo ejecute?

Creamos el contenedor con el comando `-d`. Este comando hace que NGINX se esté ejecutando en segundo plano. Usamos también el comando `--name` para indicarle un nombre al contenedor.

```
docker run -d --name miNginx nginx
```

```
daw@daw-docker:~$ docker run -d --name miNginx nginx
4debbe8605764c4d4a54dcb06348d274b9f69b560f1f1aa3d662e905ce546632
daw@daw-docker:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
4debbe860576   nginx    "/docker-entrypoint..." 8 seconds ago  Up 6 seconds  80/tcp      miNginx
daw@daw-docker:~$ S
```

Accede al navegador web y comprueba que el servidor esta funcionando.

Usamos el comando inspect para saber la ip y el puerto:


```
docker inspect miNginx
```

```

},
"NetworkSettings": {
  "Bridge": "",
  "SandboxID": "801aa734d50ecd8d5c97ff26c5aa29612f1e5a309c3cd286168453f235bdd7b5",
  "HairpinMode": false,
  "LinkLocalIPv6Address": "",
  "LinkLocalIPv6PrefixLen": 0,
  "Ports": {
    "80/tcp": null
  },
  "SandboxKey": "/var/run/docker/netns/801aa734d50e",
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "9dc556a954ede052ff2bffe27827a81b981743efc52eddf636c00e94f27fc717",
  "Gateway": "172.17.0.1",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "172.17.0.2",
  "IPPrefixLen": 16,
  "IPv6Gateway": "",
  "MacAddress": "02:42:ac:11:00:02",
  "Networks": {
    "bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": null,
      "NetworkID": "a528555b1b4833e3d6d3b34d41095d246453b20782d4b29d7360d5000076e6fe",
      "EndpointID": "9dc556a954ede052ff2bffe27827a81b981743efc52eddf636c00e94f27fc717",
      "Gateway": "172.17.0.1",
      "IPAddress": "172.17.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:11:00:02",
      "DriverOpts": null
    }
  }
}

```

Desde el navegador entramos en la dirección ip que nos indica:

 172.17.0.2

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Muestra los logs del contenedor.

Para ello usamos el comando logs:

`docker logs miNginx`

```

root@aws-docker:~# docker logs miNginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/01/14 17:37:00 [notice] 1#1: using the "epoll" event method
2023/01/14 17:37:00 [notice] 1#1: nginx/1.23.3
2023/01/14 17:37:00 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/01/14 17:37:00 [notice] 1#1: OS: Linux 5.15.0-57-generic
2023/01/14 17:37:00 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/01/14 17:37:00 [notice] 1#1: start worker processes
2023/01/14 17:37:00 [notice] 1#1: start worker process 28
2023/01/14 17:37:00 [notice] 1#1: start worker process 29
172.17.0.1 - - [14/Jan/2023:17:39:35 +0000] "GET / HTTP/1.1" 200 615 "-" Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:108.0) Gecko/20100101 Firefox/108.0 "-"
172.17.0.1 - - [14/Jan/2023:17:39:35 +0000] "GET /favicon.ico HTTP/1.1" 404 153 "http://172.17.0.2/" Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:108.0) Gecko/20100101 Firefox/108.0 "-"
2023/01/14 17:39:35 [error] 28#28: *1 open() "/usr/share/nginx/html/favicon.ico" failed (2: No such file or directory), client: 172.17.0.1, server: localhost, request: "GET /favicon.ico HTTP/1.1", host: 172.17.0.2, referer: "http://172.17.0.2/"

```

5.Crea un contenedor con la aplicación Nextcloud, mirando la documentación en docker Hub, para personalizar el nombre de la base de datos SQLite que va a utilizar.

Nos vamos a la página https://hub.docker.com/_/nextcloud, donde encontraremos toda la documentación. Encontramos que para darle un nombre a nuestra BBDD SQLite, tendremos que incluir esta variable de entorno:

SQLite:

- `SQLITE_DATABASE` Name of the database using sqlite

MYSQL / MariaDB:

```
docker run -d --name myNextCloud -e SQLITE_DATABASE=javibbdd nextcloud
```