

INSTALACIÓN, ADMINISTRACIÓN Y CONFIGURACION (3)



Azael Morell Martínez
Sistemas Informáticos
1º DAM



ÍNDICE:

INTRODUCCIÓN:	3
OBJETIVOS:	3
Material utilizado del centro (OBRIGADO):	3
DESARROLLO:	4
PROBLEMAS ENCONTRADOS Y SUGERENCIAS:	17
CONCLUSIÓN:	17
WEB GRAFÍA:	17

INTRODUCCIÓN:

Docker es una plataforma de código abierto que automatiza la construcción, implementación y ejecución de aplicaciones en contenedores. Los contenedores son unidades aisladas de software que contienen todo lo necesario para ejecutar una aplicación, incluyendo el código, un sistema operativo, bibliotecas y herramientas de desarrollo. Docker permite a los desarrolladores empaquetar sus aplicaciones y sus dependencias en un contenedor, lo que garantiza que la aplicación se ejecute de la misma manera en cualquier entorno, ya sea en una máquina local, en un servidor en la nube o en un entorno de producción. Docker también facilita la administración y la escalabilidad de las aplicaciones, ya que los contenedores se pueden crear, eliminar y mover fácilmente entre diferentes hosts. Además, Docker ofrece una forma eficiente de utilizar los recursos del sistema, ya que los contenedores comparten el mismo kernel y los recursos del sistema operativo, lo que reduce el uso de memoria y CPU en comparación con las máquinas virtuales tradicionales.

OBJETIVOS:

- ❖ Aprender los conceptos básicos de Docker, incluyendo contenedores, imágenes, Dockerfile y Docker Compose.
- ❖ Aprender a construir y ejecutar contenedores Docker utilizando la línea de comandos.
- ❖ Aprender a crear imágenes Docker personalizadas utilizando Dockerfile.
- ❖ Aprender a administrar y escalar contenedores Docker utilizando Docker Compose.

Material utilizado del centro (OBRIGADO):

- Ordenador completo en funcionamiento que cada alumno se responsabiliza.

Indicar características principales del equipo (benchmark):

DATOS :	VALOR:
Marca *y modelo del Procesador	Modelo 94 y marca: Nombre del modelo: Intel(R) Pentium(R) CPU G4400 @ 3.30GHz
Capacidad de Memoria RAM	Capacidad RAM: 8192 MB
Tipo de dispositivo de almacenamiento y capacidad (*GiB)	HDD (8 GiB)

DESARROLLO:**(PARTE 2)**

1. The Docker client contacted the Docker daemon.

Este comando inicia el servicio docker en un sistema Linux.

```
azael@azael-virtualbox:~$ sudo systemctl start docker
```

2. The Docker daemon pulled the "hello-world" image from the Docker Hub. (amd64)

Descargamos una imagen mediante este comando

```
azael@azael-virtualbox:~$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:266b191e926f65542fa8daec01a192c4d292bff79426f47300a046e1bc576fd
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
azael@azael-virtualbox:~$
```

3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.

Arrancamos y creamos una imagen

```
azael@azael-virtualbox:~$ docker run hello-world

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

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

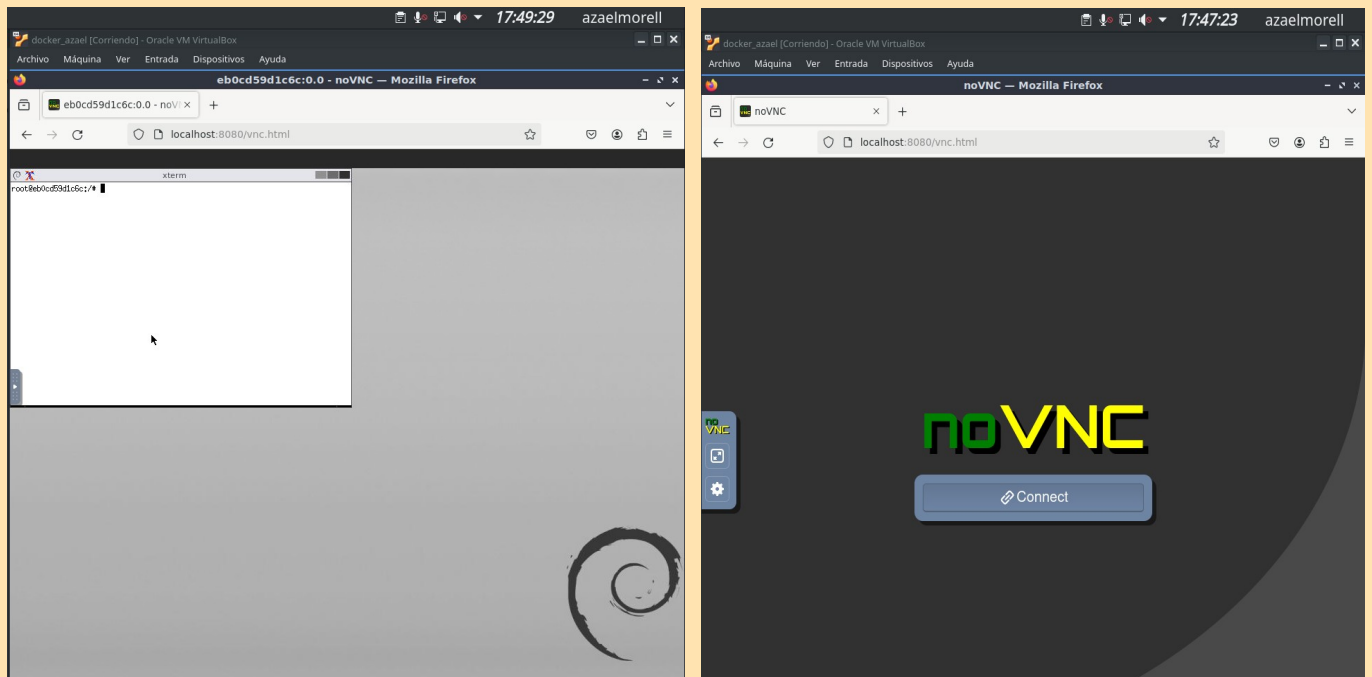
For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal. To try something more ambitious, you can run an Ubuntu container with: \$

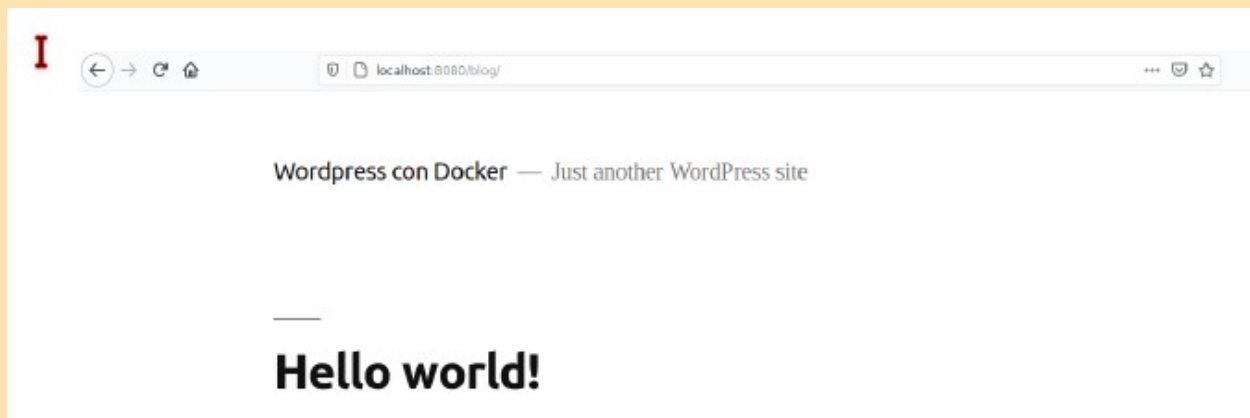
```
azael@azael-virtualbox:~$ docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
49b384cc7b4a: Pull complete
Digest: sha256:3f85b7caad41a95462cf5b787d8a04604c8262cdcdf9a472b8c52ef83375fe15
Status: Downloaded newer image for ubuntu:latest
root@239d17265892:/#
```

(PARTE 3)

En esta parte he reunido las capturas finales donde se demuestra que he logrado acceder a NoVNC mediante el caso practico



También he hecho esta captura demostrando que conseguí acceder al wordpress siguiendo el caso practico que me aparece en la practica.



(PARTE 4 | CASO PRACTICO CREANDO IMAGEN EN NODE)

En este caso practico cree la imagen en node mediante estos comandos:

`docker build -t sampledocker ./`: Construye una imagen de Docker con el nombre "sampledocker" desde el directorio actual.

`docker run -dp 3000:3000 sampledocker:` Ejecuta un contenedor desde la imagen "sampledocker" y expone el puerto 3000 del contenedor al puerto 3000 del host.

```
azael@azael-virtualbox:~/getting-started/app$ docker build -t sampledocker ./
[+] Building 12.1s (10/10) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 468B                                0.0s
=> [internal] load metadata for docker.io/library/node:12-alpine  0.8s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [1/5] FROM docker.io/library/node:12-alpine@sha256:d4b15b3d48f42059a15bd659be60afe21762aae9d6cbea6f124440895c27db6 0.0s
=> [internal] load build context                                  0.0s
=> => transferring context: 2.68kB                                    0.0s
=> CACHED [2/5] RUN apk add --no-cache yarn                       0.0s
=> CACHED [3/5] WORKDIR /app                                       0.0s
=> [4/5] COPY . .                                                 3.6s
=> [5/5] RUN yarn install --production                           7.1s
=> exporting to image                                              0.4s
=> => exporting layers                                              0.3s
=> => writing image sha256:a09c41bf444bb1751cd8f1da23f1defa20161eabeff9bfb6a9fdd37b6984fff 0.0s
=> => naming to docker.io/library/sampledocker                    0.0s
azael@azael-virtualbox:~/getting-started/app$ docker run -dp 3000:3000 sampledocker
3a0d8945afe6b99dc961ee94aa006e0b99fe8c5bc57613c1dd48c95e90c3dfd6
azael@azael-virtualbox:~/getting-started/app$
```

(Part 5)

Esta captura de pantalla muestra una serie de comandos de Docker para ejecutar un contenedor de MySQL y un contenedor de WordPress.

Primero se crea una red llamada "redwp" para conectar los dos contenedores.

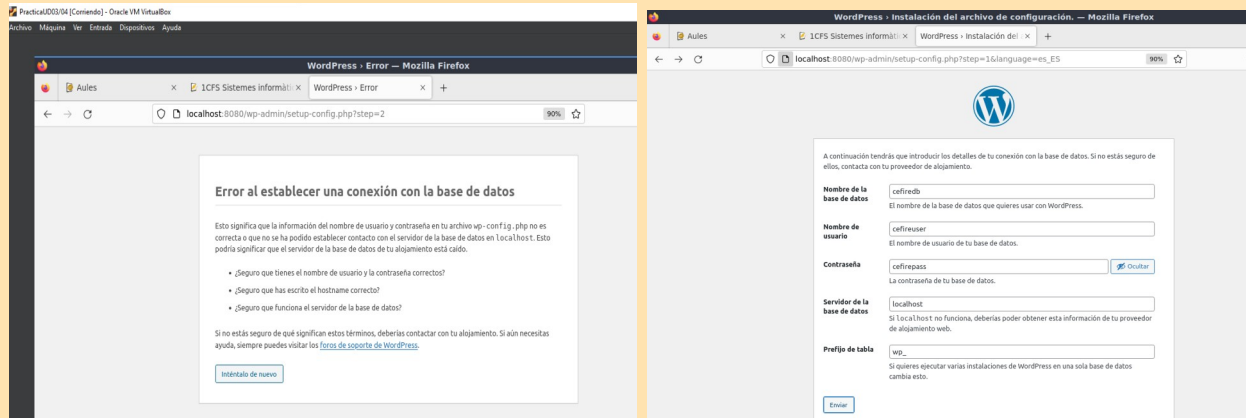
Luego se inicia un contenedor de MySQL con el nombre "nuestromysql" conectado a la red "redwp" y se le configuran las variables de entorno para la contraseña, el usuario y la base de datos.

Finalmente, se intenta iniciar un contenedor de WordPress con el nombre "nuestrowp" conectado a la red "redwp", pero falla porque no se encuentra la imagen de "wordpress:latest".

La última parte del código intenta descargar la imagen de "wordpress:latest" de la biblioteca de Docker.

```
azael@azael-virtualbox:~/getting-started/app$ docker run -dp 3800:3800 sampledocker
3a0d8945afe6b99dc961ee94aa006e0b99fe8c5bc57613c1dd48c95e90c3dfdf6
azael@azael-virtualbox:~/getting-started/app$ docker network create redwp
2195bae8b4d7364cc5bb6975ec1c45901016fc99b1e9ed63a50dc4dfe7be4b5
azael@azael-virtualbox:~/getting-started/app$ docker run --name nuestromysql --network redwp -v /home/sergi/mysqldata:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=cefireroot -e MYSQL_USER=cefireuser -e MYSQL_PASSWORD=cefirepass -e MYSQL_DATABASE=cefiredb -d mysql:5.6
Unable to find image 'mysql:5.6' locally
5.6: Pulling from library/mysql
35b2232c987e: Pull complete
fc55c00e48f2: Pull complete
0030405130e3: Pull complete
e1fef776a8d1: Pull complete
1c76272398bb: Pull complete
f57e698171b6: Pull complete
f5b825b269c9: Pull complete
dcb0af686073: Pull complete
27bbfeb886d1: Pull complete
6f70cc868145: Pull complete
1f6637f4600d: Pull complete
Digest: sha256:20575e3e6e6216036d25dab5903808211f1e9ba63dc7825ac20cb975e34cfcae
Status: Downloaded newer image for mysql:5.6
b3736a73884c069c124ed9265b6a654219351b91d5e8a27902a96cb15b1f72c0
azael@azael-virtualbox:~/getting-started/app$ docker run --name nuestrowp --network redwp -p 8080:80 -d wordpress
Unable to find image 'wordpress:latest' locally
latest: Pulling from library/wordpress
09f376ebb190: Pull complete
76afcdc86551: Pull complete
ceed4541c527: Pull complete
9ec84be954b0: Pull complete
ff0e278869f9: Pull complete
```


PROBLEMA ENCONTRADO SIN CONCLUIR UNA SOLUCIÓN



Acabando esta parte 6 probé a detener un contenedor llamado "nuestromysql", luego lo elimine. Después, lance un nuevo contenedor llamado "nuestromysql" con una configuración específica, incluyendo la conexión a una red llamada "redwp" y el montaje de un volumen local para los datos de MySQL. Finalmente, inicie el contenedor de MySQL.

```
azael@azael-virtualbox:~/getting-started/app$ docker stop nuestromysql
nuestromysql
azael@azael-virtualbox:~/getting-started/app$ docker rm nuestromysql
nuestromysql
azael@azael-virtualbox:~/getting-started/app$ docker run --name nuestromysql --network redwp -v /home/sergi/mysqldata:/var/lib/mysql -d mysql:5.7
Unable to find image 'mysql:5.7' locally
5.7: Pulling from library/mysql
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Pull complete
ffc89e9dfd88: Pull complete
43d05e938198: Pull complete
064b2d298fba: Pull complete
df9a4d85569b: Pull complete
Digest: sha256:4bc6bc963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
5f060e0e670ee4f3b55e1e2469d75841463c0526e0689706ad11c7d92a42542f
azael@azael-virtualbox:~/getting-started/app$
```

(Parte 6)

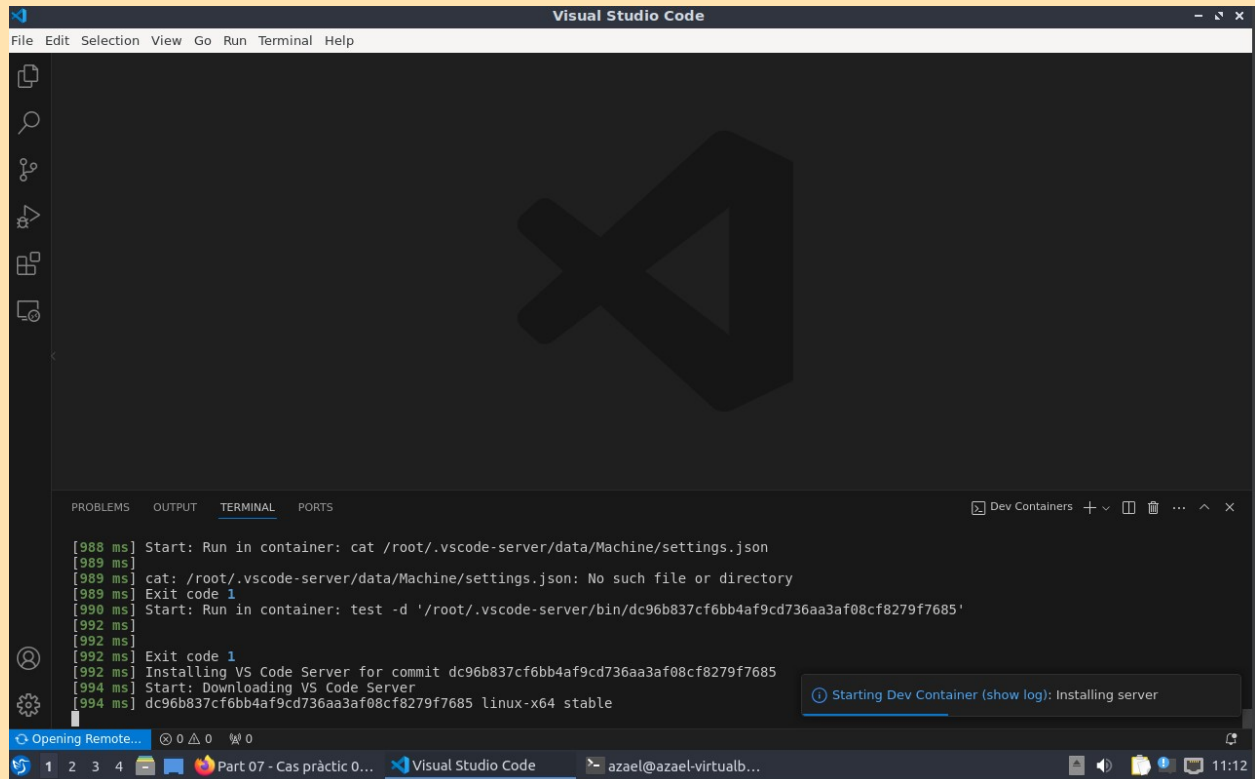
```
azael@azael-virtualbox:~/getting-started/app$ docker-compose up -d
Creating network "getting-started_default" with the default driver
Creating volume "getting-started_db_data" with default driver
Creating getting-started_db_1 ... done
Creating getting-started_wordpress_1 ... done
azael@azael-virtualbox:~/getting-started/app$ docker-compose down
Stopping getting-started_wordpress_1 ... done
Stopping getting-started_db_1 ... done
Removing getting-started_wordpress_1 ... done
Removing getting-started_db_1 ... done
Removing network getting-started_default
azael@azael-virtualbox:~/getting-started/app$ docker-compose up -d
Creating network "getting-started_default" with the default driver
Creating getting-started_db_1 ... done
Creating getting-started_wordpress_1 ... done
azael@azael-virtualbox:~/getting-started/app$
```

(Parte 7)

el comando que se indica en el caso practico estaba mal. El correcto es el que muestra en la captura

```
azael@azael-virtualbox:~$ docker run -d --name servidordesarrollo -p 8080:80 php:7.2-apache
Unable to find image 'php:7.2-apache' locally
7.2-apache: Pulling from library/php
6ec7b7d162b2: Download complete
db606474d60c: Download complete
afb30f0cd8e0: Downloading [=====>] 76.65MB/76.65MB
3bb2e8051594: Download complete
4c761b44e2cc: Download complete
c2199db96575: Download complete
1b9a9381eea8: Download complete
fd07bbc59d34: Download complete
72b73ab27698: Download complete
983308f4f0d6: Download complete
6c13f026e6da: Download complete
e5e6cd163689: Download complete
5c5516e56582: Download complete
154729f6ba86: Download complete
docker: write /var/lib/docker/tmp/GetImageBlob2402726004: no space left on device.
See 'docker run --help'.
azael@azael-virtualbox:~$
```

Al ajuntar el contenedor al visual studio haciendo uso de las extensiones me surge un problema . No me dice ningún error pero se que cargando infinitamente el remoto. Ahora muestro una captura de lo que ocurre:



The screenshot shows the Visual Studio Code interface. The main editor area is dark with a large, faint Visual Studio logo. The bottom panel contains a terminal window with the following output:

```
[988 ms] Start: Run in container: cat /root/.vscode-server/data/Machine/settings.json
[989 ms] cat: /root/.vscode-server/data/Machine/settings.json: No such file or directory
[989 ms] Exit code 1
[990 ms] Start: Run in container: test -d '/root/.vscode-server/bin/dc96b837cf6bb4af9cd736aa3af08cf8279f7685'
[992 ms]
[992 ms] Exit code 1
[992 ms] Installing VS Code Server for commit dc96b837cf6bb4af9cd736aa3af08cf8279f7685
[994 ms] Start: Downloading VS Code Server
[994 ms] dc96b837cf6bb4af9cd736aa3af08cf8279f7685 linux-x64 stable
```

Below the terminal, a status bar indicates "Starting Dev Container (show log): Installing server". The bottom of the window shows the taskbar with various application icons and the system clock at 11:12.

PROBLEMAS ENCONTRADOS Y SUGERENCIAS:

A lo largo de esta práctica he realizado varios pasos en los que tenido que solucionar ejercicios. Durante el camino es cierto que en algunas partes he tenido problemas que posteriormente no se al 100% si de verdad es correcto. Como por ejemplo en la parte 7 hay un comando que se pone en caso practico que no funciona correctamente. Posteriormente encontré el comando adecuado. También debo de mencionar que en esa misma parte 7 la ultima captura no entiendo el motivo de que tras 30 minutos de espera no cargue el remoto. Dejándose ese tema de lado el resto me ha ido bien, aunque en algunos apartados me haya costado un poco mas de tiempo.

CONCLUSIÓN:

Ahora he de hablar sobre esta practica, en mi opinión ha sido de gran ayuda para poder mejorar y aprender conocimientos. Dejando ese tema de lado, he podido ver bastantes comandos, en general los procesos de Docker son curiosos e intrigantes. A medida que vayamos avanzando en temario sobre este tema imagino que se irá complicando, pero hasta el momento aun teniendo contratiempos creo que voy por buen camino.

WEB GRAFÍA:

La documentación para poder realizar esta actividad que he utilizado yo personalmente es el “pdf” que subió mi profesor Oscar.