

### Actividad Evaluable 3- GIT Y DOCKER

Ejercicio 1 - Trabajo con Imágenes

Servidor web

Servidor de base de datos

Ejercicio 2 - Almacenamiento

Bind mount para compartir datos

Ejercicio 3 - Redes

Despliegue de contenedores en red: Adminer y MariaDB

## Actividad Evaluable 3- GIT Y DOCKER

### Ejercicio 1 - Trabajo con Imágenes

#### Servidor web

Arrancamos un contenedor que ejecute una instancia de la imagen `php:7.4-apache`, al que llamamos `web` y hacemos que se pueda acceder al mismo desde el puerto 8000 de nuestro navegador.

Para ello ejecutamos el siguiente comando:

```
docker run -it -p 8000:80 --name web -v /adriana/Ejercicio1:/var/www/html/ -d php:7.4-apache
```

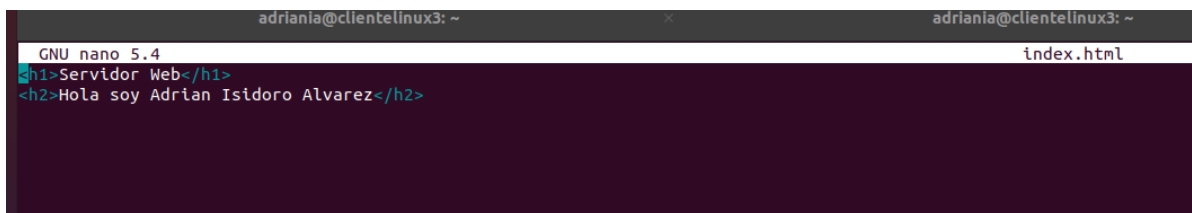
Nos conectamos a la consola del contenedor de docker con el comando:

```
docker exec -it web bash
```

Una vez en la consola del contenedor, instalamos el editor nano utilizando los siguientes comandos:

```
apt-get update  
apt-get install nano
```

Creamos nuestro fichero `index.html` y vemos el resultado desde el navegador:



```
adriana@clientlinux3: ~  
GNU nano 5.4  
index.html  
<h1>Servidor Web</h1>  
<h2>Hola soy Adrian Isidoro Alvarez</h2>
```

# Servidor Web

## Hola soy Adrian Isidoro Alvarez

Ahora creamos el fichero php `mes.php` que nos indique el mes actual en nuestro navegador, para ello simplemente creamos el fichero al igual que el `index.html` y comprobamos que funciona correctamente:

```
GNU nano 5.4 mes.php
<?php
setlocale(LC_TIME,"es_ES");
echo ("Estamos en " . date("F"));
?>
```

Estamos en April

Aquí podemos ver el tamaño del contenedor `web` después de haber creado ambos ficheros:

```
root@e03d299a51a5:/var/www/html# exit
exit
adriana@clienteLinux3:~$ docker ps --size
CONTAINER ID   IMAGE          COMMAND                  CREATED    STATUS    PORTS                               NAMES    SIZE
e03d299a51a5   php:7.4-apache "docker-php-entrypoi..." 20 minutes ago Up 20 minutes 0.0.0.0:8000->80/tcp, :::8000->80/tcp web      19.5MB (virtual 472MB)
```

Para finalizar tenemos que parar el contenedor y posteriormente borrarlo de la siguiente forma:

```
docker stop web
docker rm web
```

## Servidor de base de datos

Para este apartado usaremos un contenedor para ejecutar la imagen `mariadb`.

Para ello solamente tenemos que descargar esta imagen y lanzarla introduciendo los datos del enunciado:

```
docker pull mariadb
docker run --detach --name bdd --env MARIADB_USER=invitado --env
MARIADB_PASSWORD=invitado --env MARIADB_ROOT_PASSWORD=root --env
MARIADB_DATABASE=prueba mariadb:latest
```

Nos conectaremos de la misma forma que en el apartado anterior y comprobamos que podemos conectarnos a la base de datos utilizando el usuario que hemos creado anteriormente:

```
adriania@clientlinux3:~$ docker exec -it bdd bash
root@5c60d959a818:/# mysql --user=invitado --password=invitado
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.7.3-MariaDB-1:10.7.3+maria~focal mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> █
```

Una vez aquí utilizamos el siguiente comando para ver las bases de datos existentes:

```
show databases;
```

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| prueba |
+-----+
2 rows in set (0.000 sec)

MariaDB [(none)]> █
```

Ahora intentaremos borrar nuestra imagen `mariadb` sin antes parar el contenedor `bdd` para comprobar que no está permitido:

```
adriania@clientlinux3:~$ docker rm bdd
Error response from daemon: You cannot remove a running container 5c60d959a8183ee4ed8e7ded117938faf7710ed8c05034d506dc1b00308c8839a. Stop the container before attempting removal or force remove
```

Por último eliminamos el contenedor `bdd` parándolo anteriormente con los comandos `stop` y `rm`:

```
adriania@clientlinux3:~$ docker rm bdd
bdd
adriania@clientlinux3:~$ docker ps --size
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES   SIZE
adriania@clientlinux3:~$ █
```

## Ejercicio 2 - Almacenamiento

# Bind mount para compartir datos

Crearemos un directorio **saludo** con el comando `mkdir`, y en su interior creamos un fichero **index.html** y lo editamos con `nano`:



```
adriana@clienteLinux3: ~
GNU nano 4.8 index.html
<h1>Hola soy Adrián Isidoro</h1>
```

Arrancamos dos contenedores basados en la imagen `php:7.4-apache` haciendo un bind mount de la carpeta `saludo` en la carpeta `/var/www/html` de cada contenedor. Utilizaremos los puertos que nos indica el enunciado y sus nombres serán `c1` y `c2`:

```
sudo docker container run --name c1 -v
/Documentos/git/DAWActividad3/saludo:/var/www/html --publish 8181:80 --detach --
restart=always php:7.4-apache
```

Accedemos a nuestro fichero mediante el navegador utilizando el puerto **8181**.



Realizamos el mismo proceso para el segundo contenedor y comprobamos que editando el fichero **index.html** podemos acceder y ver los cambios:



Por último se nos pide demostrar que se puede acceder a la vez a ambos contenedores:

```

adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ sudo nano index.html
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker exec -it c1 bash
root@ce9e87eedb3b:/var/www/html# exit
exit
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker exec -it c2 bash
root@ad5732d2967f:/var/www/html# exit
exit
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker stop c1
c1
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker stop c2
c2
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker rm c1
c1
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker rm c2
c2
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$ docker ps --size
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS          NAMES      SIZE
adriania@clientellinux3:/Documentos/git/DAWActividad3/saludo$

```

## Ejercicio 3 - Redes

### Despliegue de contenedores en red: Adminer y MariaDB

Creamos una red **bridge**:

```
docker network create --driver bridge redbd
```

Ahora creamos un contenedor con la imagen **mariadb** en la red creada anteriormente. El enunciado nos indica que debe ser accesible por el puerto **3306** y también nos especifica el usuario y contraseña. Por lo tanto el comando a utilizar será el siguiente:

```
docker run -d --network redbd --name bbdd --env MARIADB_ROOT_PASSWORD=root -p 3306:3306 -v /Documentos/git/DAWActividad3 mariadb:latest
```

Para ejecutar la imagen **Adminer** usamos este comando:

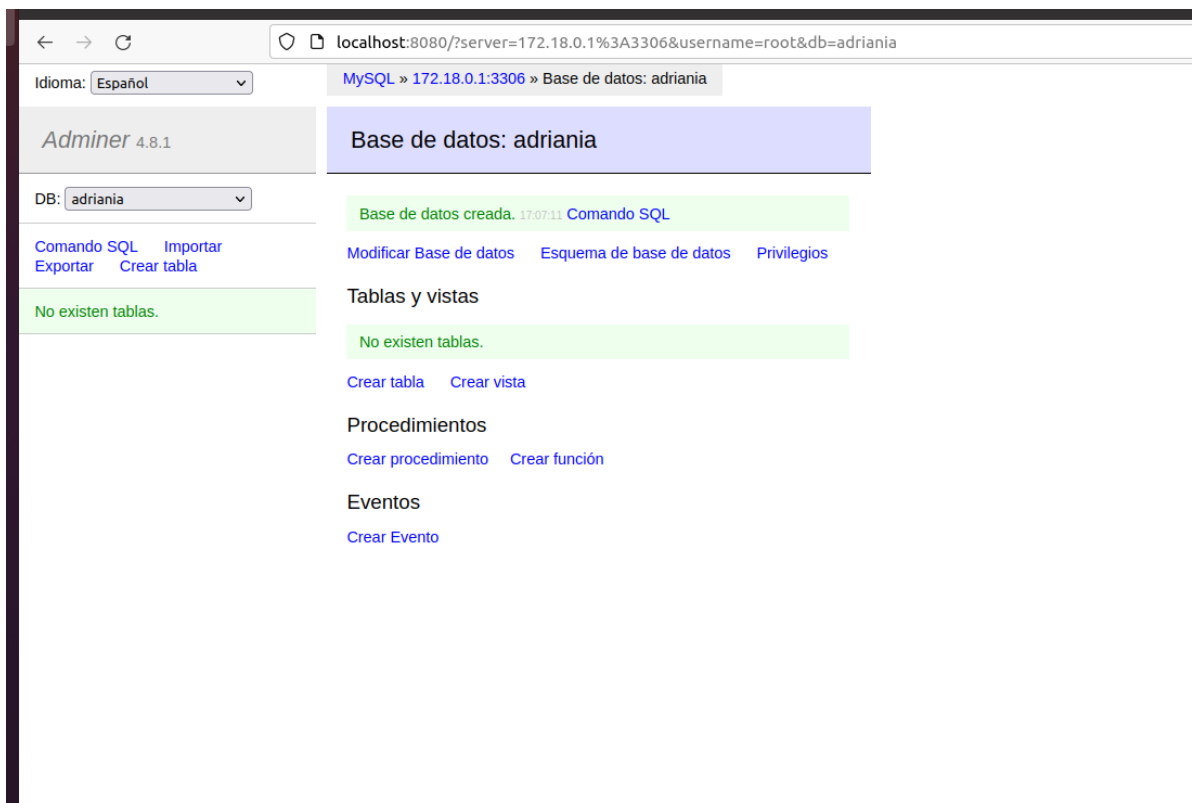
```
docker run --link bbdd:db -p 8080:8080 adminer
docker run -p 8080:8080 -e ADMINER_DEFAULT_SERVER=mysql adminer
```

Con esto ya podemos acceder a la base de datos a través de la interfaz web de **Adminer**:

The screenshot shows the Adminer web interface for a MySQL database. The interface is in Spanish and displays the 'Base de datos: mysql' section. On the left, there is a sidebar with a list of database tables and views. The main area shows a table with columns: Tabla, Motor, Colación, Longitud de datos, Longitud de índice, Espacio libre, Incremento automático, Registros, and Comentario. The table lists various MySQL system tables and their properties.

| Tabla              | Motor  | Colación           | Longitud de datos | Longitud de índice | Espacio libre | Incremento automático | Registros | Comentario                      |
|--------------------|--------|--------------------|-------------------|--------------------|---------------|-----------------------|-----------|---------------------------------|
| columns_priv       | Aria   | utf8mb3_bin        | 8 192             | 8 192              | 0             |                       | 0         | Column privileges               |
| column_stats       | Aria   | utf8mb3_bin        | 8 192             | 8 192              | 0             |                       | 0         | Statistics on Columns           |
| db                 | Aria   | utf8mb3_bin        | 8 192             | 8 192              | 0             |                       | 0         | Database privileges             |
| event              | Aria   | utf8mb3_general_ci | 8 192             | 8 192              | 0             |                       | 0         | Events                          |
| func               | Aria   | utf8mb3_bin        | 8 192             | 8 192              | 0             |                       | 0         | User defined functions          |
| general_log        | CSV    | utf8mb3_general_ci | 0                 | 0                  | 0             |                       | 2         | General log                     |
| global_priv        | Aria   | utf8mb3_bin        | 16 384            | 16 384             | 0             |                       | 3         | Users and global privileges     |
| gtid_slave_pos     | InnoDB | latin1_gedish_ci   | 16 384            | 0                  | 0             |                       | 0         | Replication slave GTID position |
| help_category      | Aria   | utf8mb3_general_ci | 16 384            | 24 576             | 0             |                       | 44        | help categories                 |
| help_keyword       | Aria   | utf8mb3_general_ci | 16 384            | 24 576             | 0             |                       | 16        | help keywords                   |
| help_relation      | Aria   | utf8mb3_general_ci | 16 384            | 24 576             | 0             |                       | 36        | keyword-topic relation          |
| help_topic         | Aria   | utf8mb3_general_ci | 1 531 904         | 40 960             | 0             |                       | 735       | help topics                     |
| index_stats        | Aria   | utf8mb3_bin        | 8 192             | 8 192              | 0             |                       | 0         | Statistics on Indexes           |
| innodb_index_stats | InnoDB | utf8mb3_bin        | 16 384            | 0                  | 0             |                       | -4        |                                 |
| innodb_table_stats | InnoDB | utf8mb3_bin        | 16 384            | 0                  | 0             |                       | -1        |                                 |
| plugin             | Aria   | utf8mb3_general_ci | 8 192             | 8 192              | 0             |                       | 0         | MySQL plugins                   |
| proc               | Aria   | utf8mb3_general_ci | 458 752           | 16 384             | 0             |                       | 50        | Stored Procedures               |

Ahora creamos una base de datos y comprobamos desde la consola del servidor web que se ha creado con éxito:



```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| adriania |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.001 sec)

MariaDB [(none)]> 
```