

Este archivo php lo que nos permitirá es conectarnos a la base de datos para poder obtener la información de esta y poder mostrar los datos por pantalla.

```
index.php
1  <?php
2  $mysqli = @new mysqli('db', 'root', 'rootpass', 'demo');
3  if ($mysqli->connect_error) {
4      die("Error de conexión: " . $mysqli->connect_error);
5  }
6  $result = $mysqli->query("SELECT * FROM users");
7  while ($row = $result->fetch_assoc()) {
8      echo $row['id'] . " - " . $row['nombre'] . "<br>";
9  }
10 ?>
11
```

Aqui con el archivo yml tendremos dos servicios, el de la base de datos y el de apache, luego usaremos una red interna para que se puedan ver entre ellos, aparte tambien crearemos un volumen para los datos de la base de datos persistan. Definiremos los perfiles dev y prod y por ultimo añadiremos el healthcheck en la base de datos con un depends\_on con un service\_healthy.

```
docker-compose.yml
1  version: '3.9'
2
3  services:
4      db:
5          image: mysql:8.0
6          environment:
7              MYSQL_ROOT_PASSWORD: rootpass
8              MYSQL_DATABASE: demo
9          volumes:
10             - db_data:/var/lib/mysql
11             - ./init.sql:/docker-entrypoint-initdb.d/init.sql
12          networks:
13             - internal
14          healthcheck:
15              test: ["CMD", "mysqladmin", "ping", "-h", "localhost"]
16              interval: 5s
17              retries: 5
18              profiles: ["dev"]
19
20      web:
21          image: php:8.2-apache
22          volumes:
23             - ./src:/var/www/html
24          networks:
25             - internal
26          depends_on:
27              db:
28                  condition: service_healthy
29          profiles: ["dev", "prod"]
30
31  volumes:
32      db_data:
33
34  networks:
35      internal:
```

Aqui creamos el el archivo sql para poder crear la base de datos al ejecutar los comandos pertinentes.

```
init.sql
1  CREATE DATABASE IF NOT EXISTS demo;
2  USE demo;
3
4  CREATE TABLE users (
5      id INT AUTO_INCREMENT PRIMARY KEY,
6      nombre VARCHAR(100) NOT NULL
7  );
8
9  INSERT INTO users (nombre) VALUES ('Mon3tr'), ('Laevatain'), ('Kakekuri');
10
```

--profile dev: levanta db y web (porque ambos tienen el profile dev).

--profile prod: levanta solo web (la BD no tiene profile prod).

-d: modo detached (en segundo plano).

--build: fuerza reconstrucción si cambias algo (aunque aquí usamos imágenes oficiales, sigue siendo válido).

docker ps: para capturas:

En dev: deben salir 2 contenedores (db y web).

En prod: solo web (y la web debe fallar al conectar a la BD).

docker compose down: para parar y limpiar contenedores y red (el volumen persiste).

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker compose --profile dev up -d --build
time="2026-01-26T18:34:45+01:00" level=warning msg="C:\Users\deividsonic2\Desktop\Practica5-Docker-Deivid Martinez\docker-compose.yml: the attribute
`version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] up 17/18
[+] up 22/22:8.2-apache [#####] 180.2MB / 182.8MB Pulling 23.6s
✓ Image php:8.2-apache Pulled 23.6s
✓ 4f4fb700ef54 Pull complete 0.0s
✓ 119d43e0c815 Pull complete 9.4s
✓ f13d132b4043 Pull complete 0.5s
✓ 4b8a03043cb8 Pull complete 19.9s
✓ 578d1b3b19d8 Pull complete 20.3s
✓ 11f3d9b643c8 Pull complete 20.1s
✓ c38fcd831a5 Pull complete 21.0s
✓ 19d8887b0138 Pull complete 0.7s
✓ 1bfd1029d57c Pull complete 0.8s
✓ cdcbbc956c Pull complete 1.0s
✓ 2910362eca76 Pull complete 1.0s
✓ 5e50fcd76d61 Pull complete 20.9s
✓ a270794a48bb Pull complete 0.8s
✓ 89a19b403240 Pull complete 1.1s
✓ 598093b4ce56 Pull complete 0.6s
✓ b9137a3dab72 Download complete 0.7s
✓ 79fd30d4334e Download complete 0.0s
✓ Network practica5-docker-deividmartinez_internal Created 0.1s
✓ Volume practica5-docker-deividmartinez_db_data Created 0.0s
✓ Container practica5-docker-deividmartinez-db-1 Healthy 22.3s
✓ Container practica5-docker-deividmartinez-web-1 Created 0.1s
```

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
c687d4501b9f   php:8.2-apache "docker-php-entrypoi..." 6 minutes ago  Up 5 minutes  80/tcp                             practica5-docker-deividmartinez-web-1
3a3c18b88bdb   mysql:8.0     "docker-entrypoint.s..." 6 minutes ago  Up 6 minutes  3306/tcp, 33060/tcp                practica5-docker-deividmartinez-db-1
b12405995ced   mysql:8.0     "docker-entrypoint.s..." 7 days ago    Up 6 minutes  3306/tcp, 33060/tcp                mysql_db
```

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker compose --profile prod up -d --build
time="2026-01-26T18:58:24+01:00" level=warning msg="No services to build"
[+] up 2/2
✓ Container practica5-docker-deividmartinez-db-1 Healthy
✓ Container practica5-docker-deividmartinez-web-1 Running
```

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
c687d4501b9f   php:8.2-apache "docker-php-entrypoi..." 24 minutes ago  Up 23 minutes  80/tcp                             practica5-docker-deividmartinez-web-1
3a3c18b88bdb   mysql:8.0     "docker-entrypoint.s..." 24 minutes ago  Up 24 minutes  3306/tcp, 33060/tcp                practica5-docker-deividmartinez-db-1
b12405995ced   mysql:8.0     "docker-entrypoint.s..." 7 days ago     Up 24 minutes  3306/tcp, 33060/tcp                mysql_db
```

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker compose down
[+] down 2/2
✓ Container practica5-docker-deividmartinez-db-1 Removed
! Network practica5-docker-deividmartinez_internal Resource is still in use
```

```
deividsonic2@A-11-PC10 MINGW64 ~/Desktop/Practica5-Docker-Deivid Martinez
$ docker compose --profile prod up -d --build
service "web" depends on undefined service "db": invalid compose project
```

## Explicación del docker-compose.yml

El archivo docker-compose.yml define dos servicios: db, que ejecuta MySQL y crea la base de datos con datos iniciales, y web, que ejecuta PHP con Apache para mostrar la información en el navegador. El volumen db\_data sirve para que los datos de MySQL no se pierdan aunque el contenedor se elimine. La red interna permite que los servicios se comuniquen entre sí usando sus nombres. El healthcheck comprueba si MySQL está listo antes de permitir que la aplicación web arranque en el perfil de desarrollo. Finalmente, los perfiles controlan qué servicios se levantan: en dev se inician db y web para trabajar normalmente, mientras que en prod solo se inicia web, provocando intencionadamente un error de conexión al no existir la base de datos.

## Conclusion Personal

Una práctica como esta enseña mucho en poco tiempo porque combina varios conceptos del mundo profesional. En esencia, te permite entender cómo se organizan varios servicios con Docker Compose. También aprendes a separar entornos mediante perfiles. Aprendes a garantizar que un servicio no arranque hasta que otro esté listo usando healthchecks. Aprendes cómo una aplicación web se comunica con una base de datos dentro de una red interna. Además, trabajas con volúmenes para mantener datos persistentes.