

Manual de instalación

DAW | TFC | 2024 – 2025

Fernando Becerra Ortiz

Índice de contenidos

1. Requisitos del sistema	3
1.1 Sin Docker.....	3
1.2 Con Docker (recomendado).....	3
2. Instalación manual (Sin Docker)	3
2.1 Preparar entorno (XAMPP o Apache).....	3
2.2 Base de datos.....	6
2.3 Configuración del backend	7
2.4 Acceder a la aplicación.....	8
3. Instalación mediante Docker.....	9
3.1 Crear carpeta del proyecto	9
3.2 Crear archivo docker-compose.yml	9
3.3 Hacer pull a las 3 imágenes.....	10
3.4 Levantar los contenedores	11
3.5 Comprobar servicios	12
4. Configuración adicional	13
4.1 SMTP (PHPMailer)	13
4.2 Certificados PDF	13
5. Solución de problemas	14
5.1 MySQL no arranca	14
5.2 La app no conecta con la BD	14
5.3 Error 500.....	14
6. Actualizar versión	14
6.1 Actualizar imágenes desde Docker Hub	14
6.2 Reiniciar servicios	14

1. Requisitos del sistema

1.1 Sin Docker

- PHP 8 o superior
- MySQL 8
- Apache 2
- Extensiones:
 - mysqli
 - mbstring
- phpMyAdmin

1.2 Con Docker (recomendado)

- Docker Engine
- Docker Compose V2

2. Instalación manual (Sin Docker)

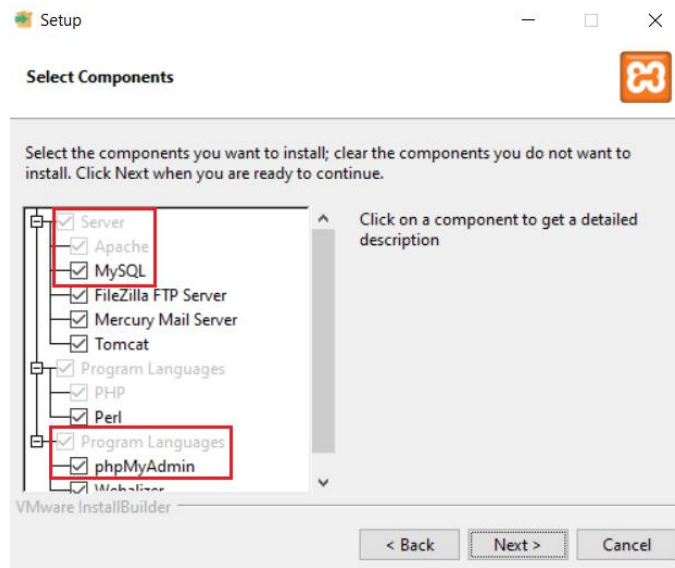
2.1 Preparar entorno (XAMPP o Apache)

1. Instalar XAMPP:

Hemos de dirigirnos a la página web oficial de [XAMPP](https://www.apachefriends.org/es/index.html) y hacer click en el botón de descargar.



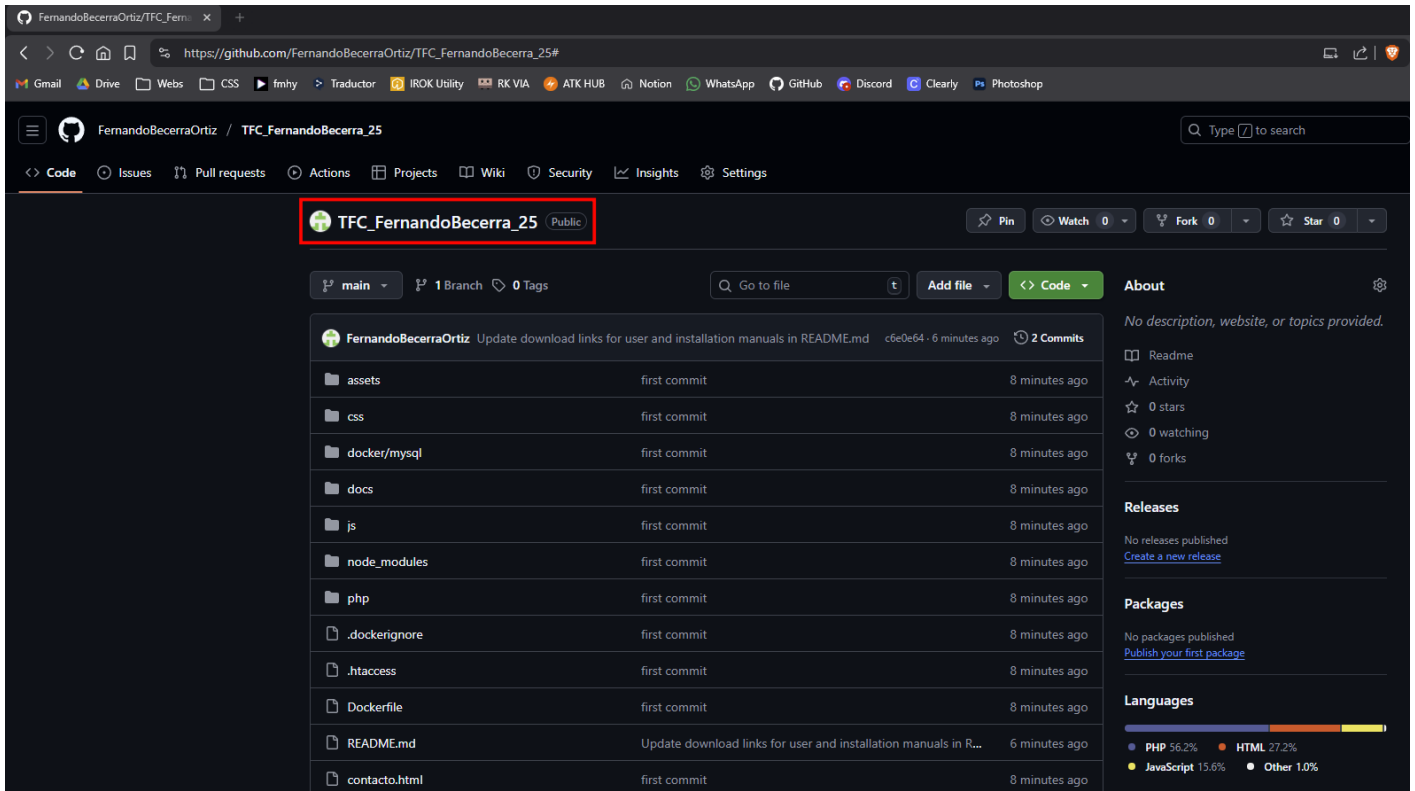
Hacemos click en siguiente a todo y una vez se haya terminado de instalar, podremos pasar al siguiente paso.



2. Copiar el repositorio en httdocs/foobar

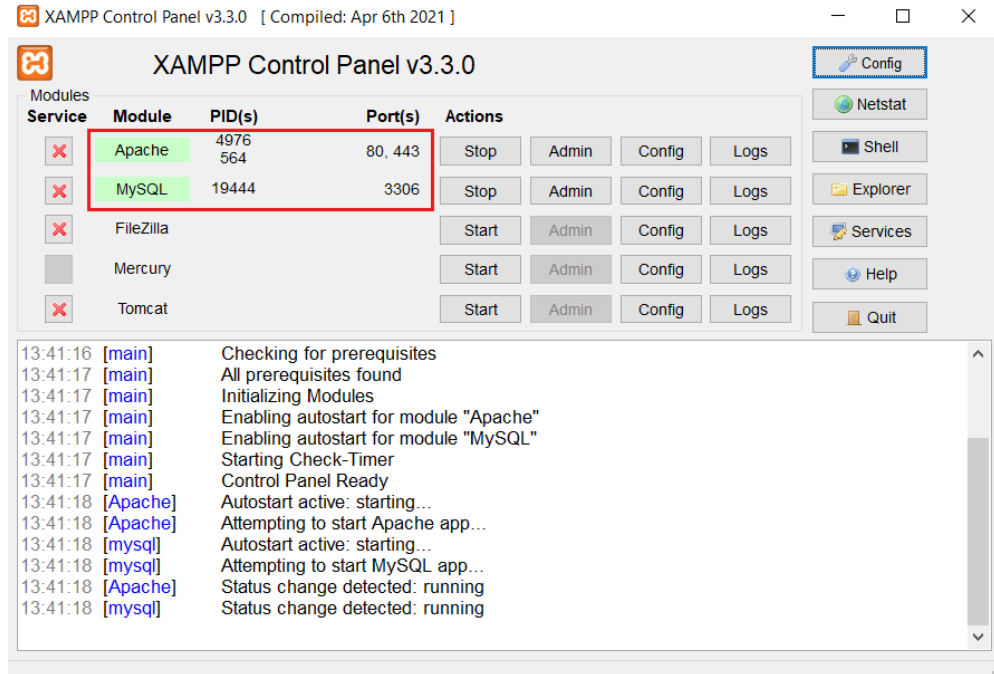
Para ello accederemos al repositorio

https://github.com/FernandoBecerraOrtiz/TFC_FernandoBecerra_25 y lo clonaremos usando `git` mediante el comando `git clone [URL]` en la carpeta `httdocs`

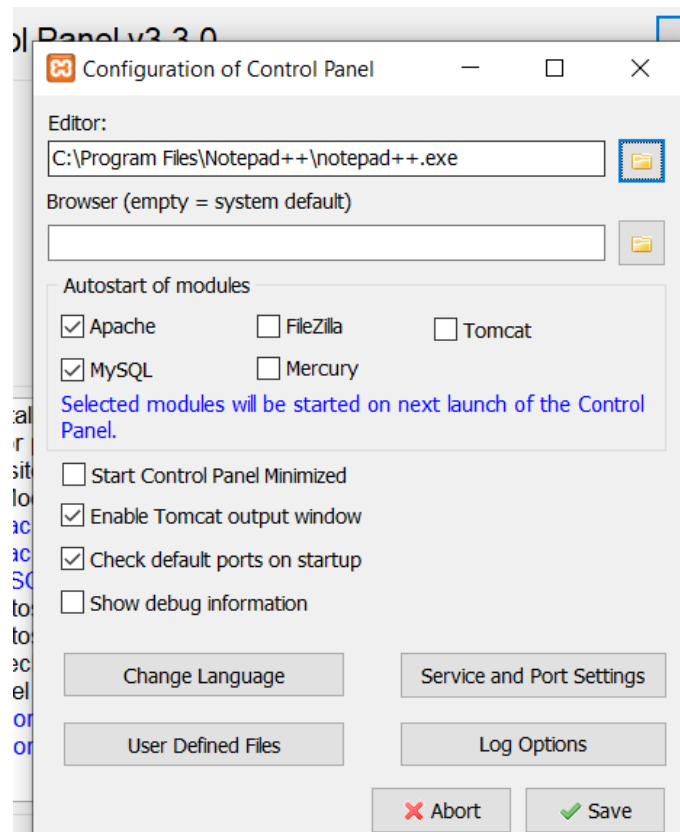


3. Arrancar Apache y MySQL

La primera vez que ejecutemos el programa deberemos configurar que se autoinicien los servicios de apache y mysql para no tener que ejecutarlos a mano cada vez.



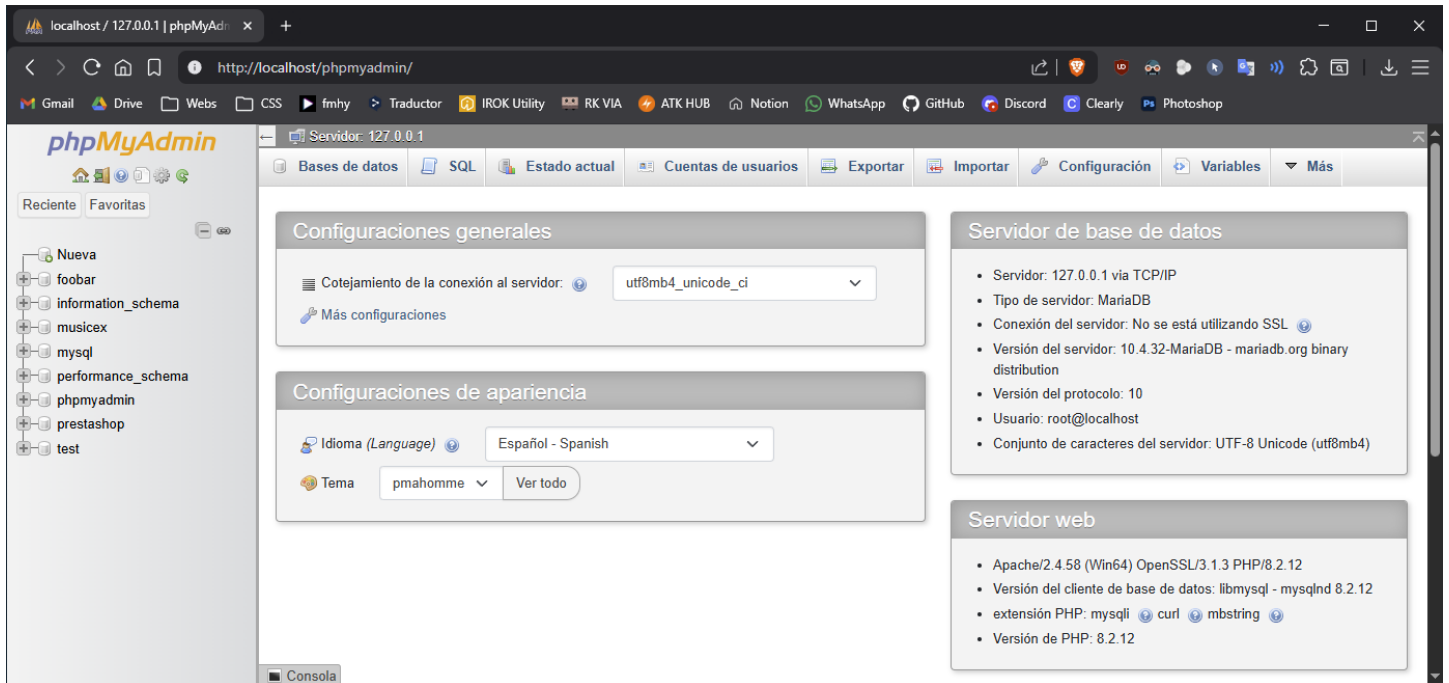
Para ello, haremos click en “Config” y marcaremos las casillas respectivas



2.2 Base de datos

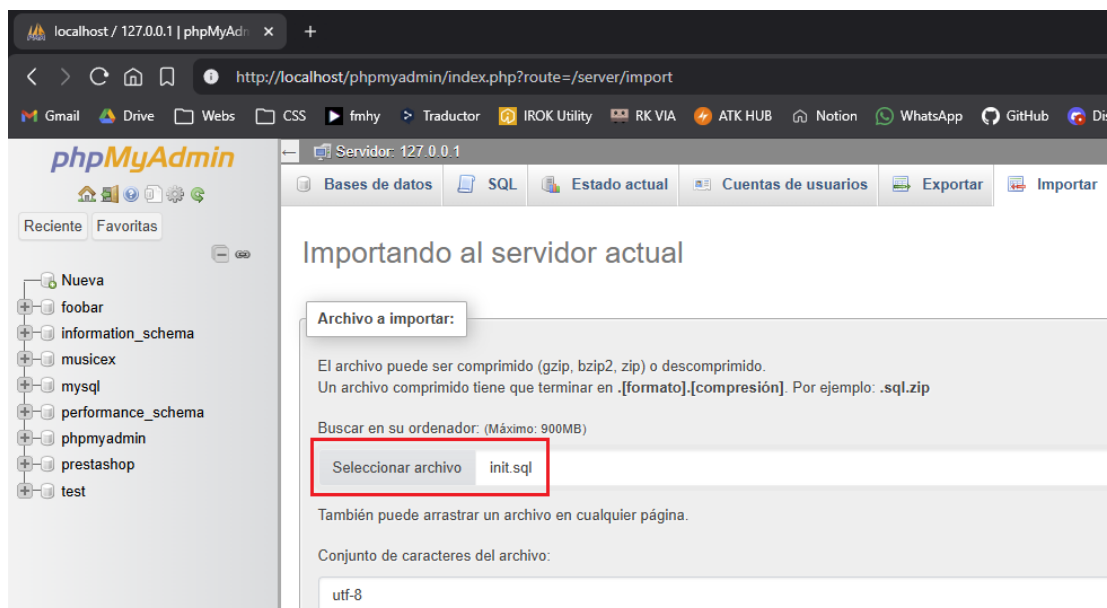
1. Abrir phpMyAdmin

Para ello, introduciremos la dirección web <http://localhost/phpmyadmin/>



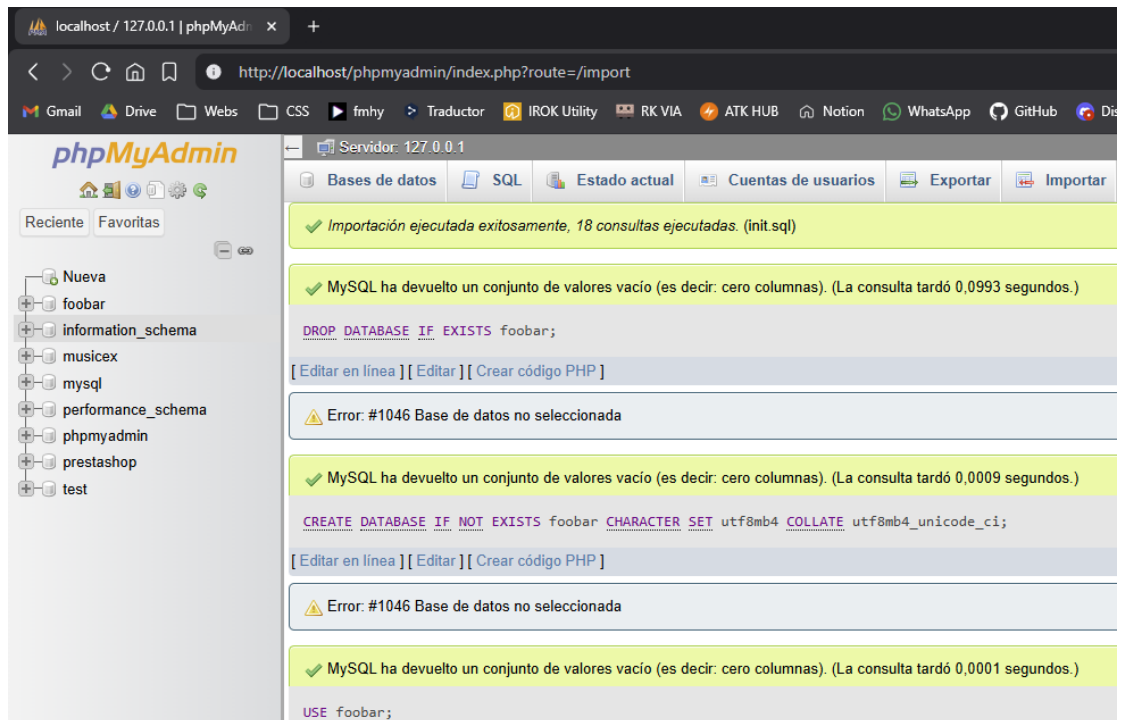
2. Importar init.sql

En la carpeta `foobar/docker/mysql` encontraremos el archivo `init.sql`, posteriormente haremos click en “Importar” situado en la barra de navegación de `phpmyadmin` y seleccionaremos dicho archivo `.sql`.

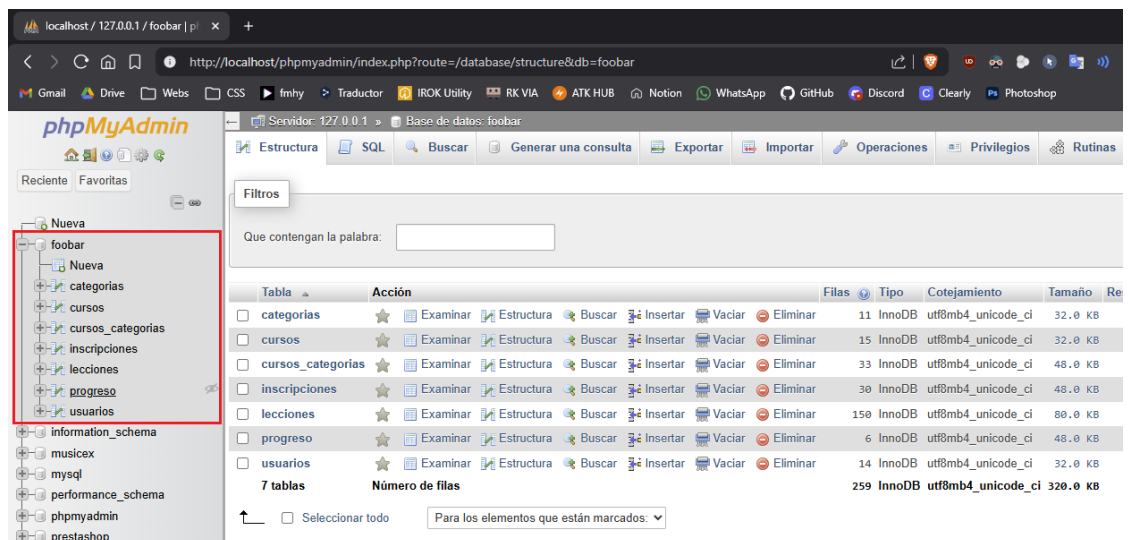


3. Verificar tablas creadas

Si miramos en la lista de las bases de datos ya aparecerá “foobar”.



Y, por último, revisaremos que se hayan creado todas las tablas y que las mismas tengan datos.



2.3 Configuración del backend

Para ello deberemos acceder a `/php/db.php` y colocar credenciales correctas correspondiente a un usuario con permisos de phpmyadmin:

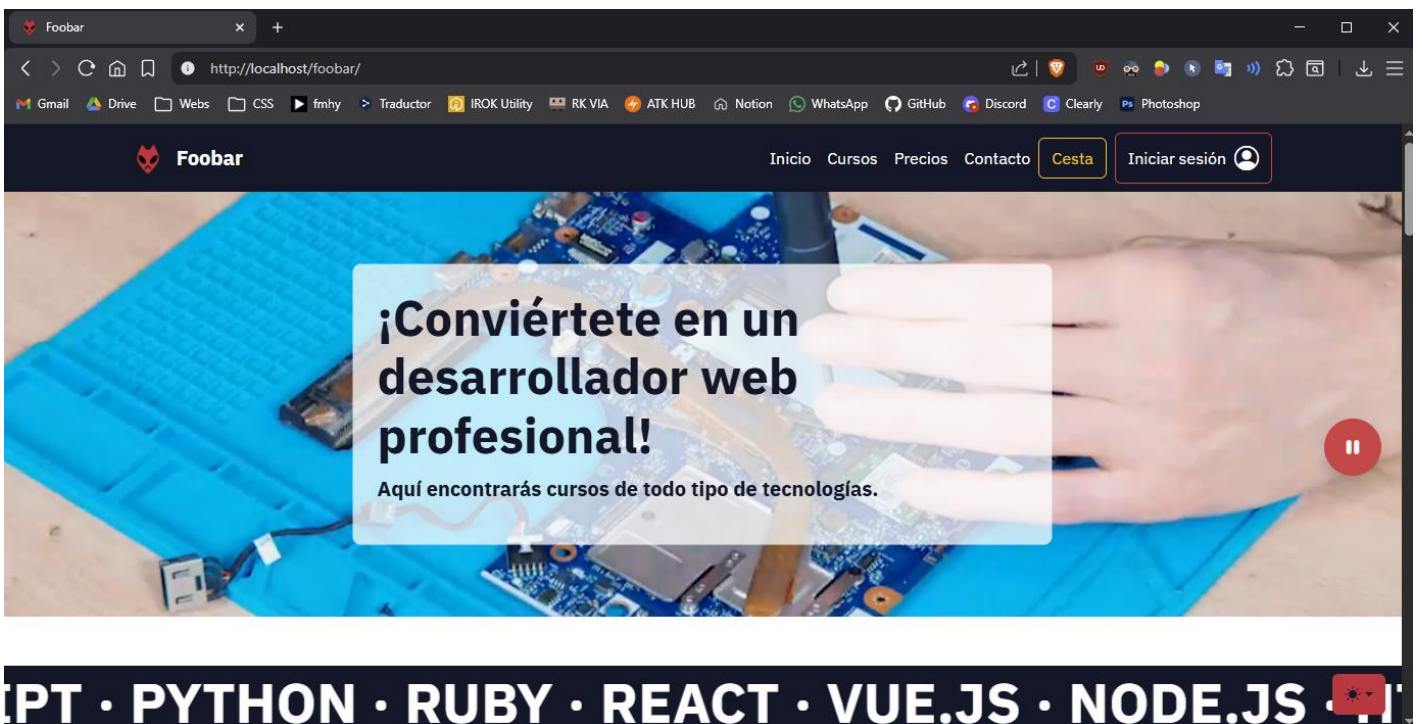
- host
- usuario

- contraseña
- nombre BD

```
$DB_CONFIG = [  
  'host' => 'db',  
  'username' => 'foobar_user',  
  'password' => 'foobar_pass',  
  'database' => 'foobar',  
  'charset' => 'utf8mb4',  
];  
  
function getConnection(): mysqli {  
  global $DB_CONFIG;  
  static $conn = null;  
  if ($conn == null) {  
    $conn = new mysqli(  
      $DB_CONFIG['host'],  
      $DB_CONFIG['username'],  
      $DB_CONFIG['password'],  
      $DB_CONFIG['database']  
    );  
    if ($conn->connect_error) {  
      sendJson(['error' => 'Error de conexión: '.$conn->connect_error], 500);  
    }  
    mysqli_set_charset($conn, "utf8mb4");  
    $conn->set_charset($DB_CONFIG['charset']);  
  }  
  return $conn;  
}
```

2.4 Acceder a la aplicación

Finalmente accederemos al index.html de la carpeta “foobar” y podremos ver el sitio web en funcionamiento <http://localhost/foobar>



3. Instalación mediante Docker

La plataforma se distribuye mediante 3 imágenes oficiales en Docker Hub:

- **fbecerrao/foobar_docker-app**: App completa con todo el código y los archivos necesarios para ejecutar el frontend y el servidor Apache.
- **fbecerrao/foobar-db**: Motor MySQL + init.sql integrado para la carga inicial de datos.
- **phpmyadmin/phpmyadmin**: Imagen docker oficial de phpmyadmin para disponer de una UI ligera web con la que administrar la base de datos.

3.1 Crear carpeta del proyecto

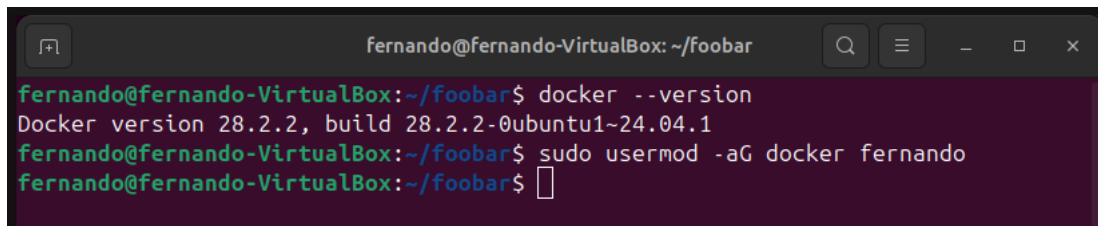
En primer lugar (en el caso de Linux), deberemos crear una carpeta destinada a alojar el archivo `docker-compose.yml`, para ello usaremos los comandos.

- `mkdir foobar`
- `cd foobar`

E introducir los siguientes 2 comandos para instalar Docker, así como asignar permisos al usuario correspondiente:

- `sudo apt install docker.io docker-compose-v2`
- `sudo usermod -aG docker [usuario]`

Para comprobar que Docker se ha instalado correctamente ejecutamos el comando `docker --version`



```
fernando@fernando-VirtualBox: ~/foobar
fernando@fernando-VirtualBox:~/foobar$ docker --version
Docker version 28.2.2, build 28.2.2-0ubuntu1~24.04.1
fernando@fernando-VirtualBox:~/foobar$ sudo usermod -aG docker fernando
fernando@fernando-VirtualBox:~/foobar$
```

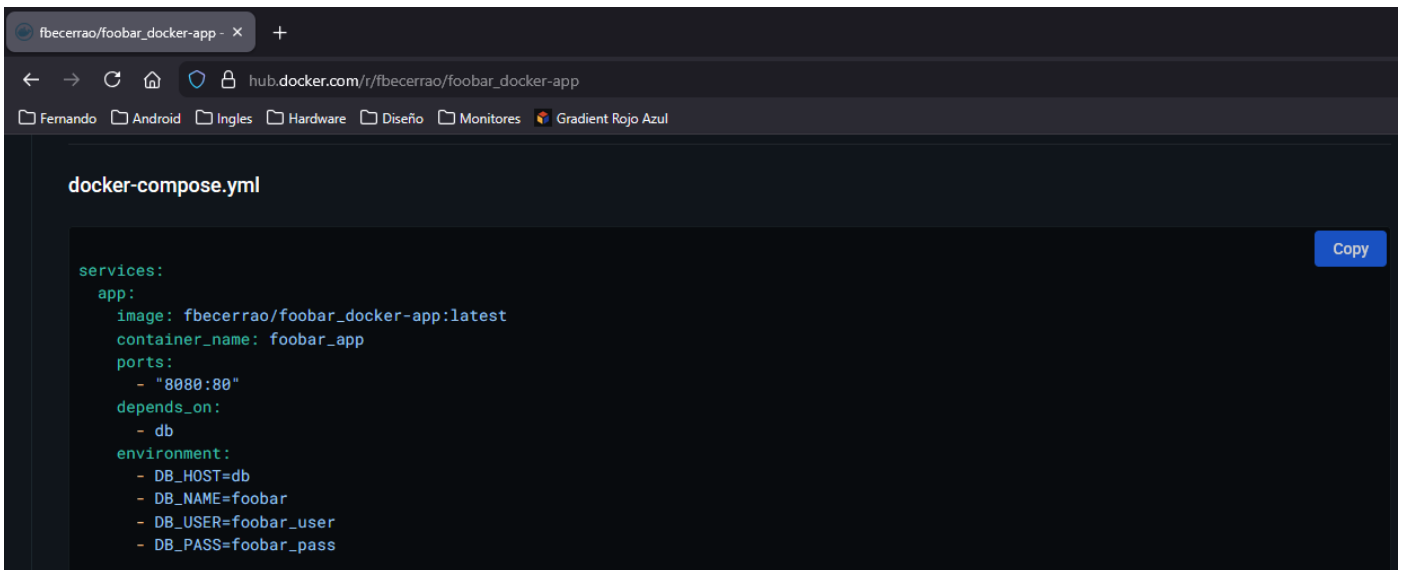
3.2 Crear archivo docker-compose.yml

El mismo incluye tres servicios:

- App
- DB con init.sql ya integrado
- phpMyAdmin

Para acceder a su contenido y replicarlo, bastará con acceder a la imagen principal de **docker hub** de la app donde se encuentra el archivo `.yml` y copiar su contenido

https://hub.docker.com/r/fbecerrao/foobar_docker-app



```
services:
  app:
    image: fbecerrao/foobar_docker-app:latest
    container_name: foobar_app
    ports:
      - "8080:80"
    depends_on:
      - db
    environment:
      - DB_HOST=db
      - DB_NAME=foobar
      - DB_USER=foobar_user
      - DB_PASS=foobar_pass
```

Una vez hayamos creado el archivo, copiamos el contenido.

```
fernando@fernando-VirtualBox:~/foobar$ cat docker-compose.yml

services:
  app:
    image: fbecerrao/foobar_docker-app:latest
    container_name: foobar_app
    ports:
      - "8080:80"
    depends_on:
      - db
    environment:
      - DB_HOST=db
      - DB_NAME=foobar
      - DB_USER=foobar_user
      - DB_PASS=foobar_pass

  db:
    image: fbecerrao/foobar-db:latest
```

3.3 Hacer pull a las 3 imagenes

Ahora procederemos a **descargar de forma manual** las 2 imágenes creadas a partir de contenedores modificados, el último contenedor de phpmyadmin se descargará con la imagen indicada (la oficial de phpmyadmin) en el archivo .yml en el momento de levantar los contenedores.

fbecerrao/foobar_docker-app:

```
Fernando@fernando-VirtualBox:~/foobar$ docker pull fbcerarrao/foobar_docker-app
Using default tag: latest
latest: Pulling from fbcerarrao/foobar_docker-app
0e4bc2bd6656: Downloading 13.66MB/29.78MB
bbb706d32abb: Download complete
a5a19df025b0: Downloading 9.706MB/117.8MB
7872b90dc36f: Download complete
b53319895a36: Download complete
bbfedcc04717: Download complete
93735e3787d7: Download complete
0d6623ff078a: Downloading 11.67MB/12.33MB
```

fbecerrao/foobar_docker-app:

```
fernando@fernando-VirtualBox:~/foobar$ docker pull fbecerrao/foobar-db
Using default tag: latest
latest: Pulling from fbecerrao/foobar-db
023a182c62a0: Extracting 5.243MB/49.5MB
4f78e34adfad: Download complete
a2ed1082d9e2: Download complete
c9ecfb07ed08: Download complete
4f94eaa123bf: Download complete
2a2d53254403: Download complete
```

3.4 Levantar los contenedores

Para ello, ejecutaremos el comando **docker compose up -d**

```
Fernando@fernando-VirtualBox:~/foobar$ docker compose up -d
[+] Running 13/22
  :: phpmyadmin [::: ] Pulling
  :: 8c7716127147 Downloading 28.9MB/29.78MB
  ✓ 24403a1f6855 Download complete
  :: e1cf44d6017a Downloading 23.67MB/117.8MB
  ✓ 2489d5e860a7 Download complete
  ✓ 0248257cbd51 Download complete
  ✓ dd53cf9bf4cf Download complete
  ✓ a139c2f3234a Download complete
  ✓ 6571cfdbe5b2 Download complete
  ✓ 8d83c968ca9a Download complete
  ✓ fddb92e888a7 Download complete
  ✓ 749b92ea0995 Download complete
  ✓ 4eed3454c20c Download complete
  ✓ 00ef78e422f0 Download complete
  ✓ 004f06ab2f6c Download complete
  ✓ 4f4fb700ef54 Download complete
  :: 273a09302c4a Downloading 19.01MB/30.86MB
```

Tal y como se había mencionado anteriormente, como no tenemos la imagen de phpmyadmin para crear contenedores, se descargará al momento.

Inmediatamente se mostrará un resumen del resultado del `docker compose`

```
[+] Running 5/5
✔ Network foo_bar_default      Created           0.3s
✔ Volume "foo_bar_db_data"     Created           0.0s
✔ Container foo_bar_db         Started           2.8s
✔ Container foo_bar_app        Started           3.5s
✔ Container foo_bar_phpmyadmin Started           3.0s
```

Para comprobar las imágenes que tenemos bastará con ejecutar docker images.

```
fernando@fernando-VirtualBox:~/foobar$ docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
fbecerrao/foobar-db latest      bb519da43db1  4 days ago    783MB
fbecerrao/foobar_docker-app latest      9d46568a3446  6 days ago    512MB
phpmyadmin/phpmyadmin latest      e66b1f5a8c58  7 weeks ago   742MB
```

Y para mostrar los contenedores que tenemos en este momento ejecutaremos docker container ls

```
fernando@fernando-VirtualBox:~/foobar$ docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                                                                 NAMES
9495b9db7b11   phpmyadmin/phpmyadmin              "/docker-entrypoint..." 3 minutes ago  Up 3 minutes  0.0.0.0:8081->80/tcp, [::]:8081->80/tcp  foobar_phpmyadmin
9865f4bf8945   fbecerrao/foobar_docker-app:latest "/docker-php-entrypoi..." 3 minutes ago  Up 3 minutes  0.0.0.0:8080->80/tcp, [::]:8080->80/tcp  foobar_app
c2b9d783a783   fbecerrao/foobar-db:latest         "docker-entrypoint.s..." 3 minutes ago  Up 3 minutes  0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp, 33060/tcp  foobar_db
```

3.5 Comprobar servicios

A continuación, si todo ha salido como debería al acceder a las siguientes direcciones web

- App: <http://localhost:8080>
- phpMyAdmin: <http://localhost:8081>

Podremos ver el sitio web y el gestor de bases de datos asociado con el contenedor donde está desplegada la base de datos.

The screenshot displays two web applications running locally. The left application is a course catalog website with a navigation bar, filters for various technologies (Astro, Tailwind, Bootstrap), and a list of course cards. The right application is the phpMyAdmin database management tool, showing a database named 'foobar' with several tables including 'categorias', 'cursos', 'cursos_categorias', 'inscripciones', 'lecciones', 'progreso', and 'usuarios'.

4. Configuración adicional

4.1 SMTP (PHPMailer)

Editar archivo php/enviar_contacto.php y configurar

- host
- usuario
- contraseña
- puerto

```
try {
    // CONFIGURACION SMTP
    $mail->isSMTP();

    // No sirve tu contraseña normal; requiere 2FA + App Password
    $mail->Host      = 'smtp.gmail.com';
    $mail->SMTPAuth  = true;
    $mail->Username  = 'fernandobecerraortiz@gmail.com'; // email Gmail
    $mail->Password  = 'tstpbdcfebwpddal';              // App Password (16 caracteres)
    $mail->Port      = 587;                              // STARTTLS
    $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;

    // Con Gmail, el setFrom debe ser el mismo que Username
    $mail->setFrom('fernandobecerraortiz@gmail.com', 'Contacto Foobar');
    $mail->addAddress('fbecerra01@suarezdefigueroa.es', 'Contacto Foobar');
    $mail->addReplyTo($mail, $nombre);

    // Contenido
    $mail->isHTML(true);
    $mail->Subject = "Nueva consulta: " . $asuntoF;
    $mail->Body    = $bodyHtml;

    // Debug a fichero
    $mail->SMTPDebug = 2; // 0=off, 2=info detallada
    $mail->Debugoutput = function ($str, $level) {
        @file_put_contents(__DIR__ . '/smtp_debug.log', '[' . date('c') . "] " . $str . "\n", FILE_APPEND);
    };

    $mail->send();
    respond(true, "Tu mensaje ha sido enviado correctamente.");
} catch (Exception $e) {
    // Guarda el motivo exacto
    @file_put_contents(
        __DIR__ . '/contacto_fallos.log',
        date('c') . " | ERROR SMTP: {$mail->ErrorInfo}\n",
        FILE_APPEND
    );

    // Mensaje genérico para el usuario; el detalle está en el log
    respond(false, "No se pudo enviar el mensaje. Revisa la configuración SMTP.");
}
```

4.2 Certificados PDF

La librería FPDF ya está integrada.
No requiere configuración.

5. Solución de problemas

5.1 MySQL no arranca

En caso de que no tengamos datos en la base de datos siempre podemos probar a eliminar todos los contenedores y volúmenes asociados al docker-compose.yml y volver a crearlos:

- Borrar: **docker compose down -v**
- Crear: **docker compose up -d --build**

5.2 La app no conecta con la BD

Revisar variables:

- DB_HOST
- DB_NAME
- DB_USER
- DB_PASS

5.3 Error 500

Revisar:

- permisos
- rutas incorrectas
- faltan extensiones PHP

6. Actualizar versión

6.1 Actualizar imágenes desde Docker Hub

En caso de que haya una nueva versión de los contenedores podemos actualizarlos y recrearlos (sin modificar los datos de los volúmenes) mediante:

- Actualizar: **docker compose pull**
- Re-crear: **docker compose up -d**

6.2 Reiniciar servicios

Para reiniciar los contenedores ejecutaremos **docker compose restart**