

# PHP

Sprint 0 Lab 4



Felipe Izquierdo Romero

## Índice

<u>1</u>	<u>Descripción del problema</u>
<u>2</u>	<u>Desarrollo</u>

## 1. Descripción del problema

Para el laboratorio de PHP se deberá realizar un formulario con php usando un contenedor con Docker con una imagen configurada con el material proporcionado para este laboratorio.

El formulario deberá de recoger del usuario

- Nombre
- Edad
- Descripción

Y tras enviar el formulario mostrar la edad de todos los usuarios que lo han completado.

## 2. Desarrollo

Para empezar este laboratorio hay que crear el contenedor con Docker.

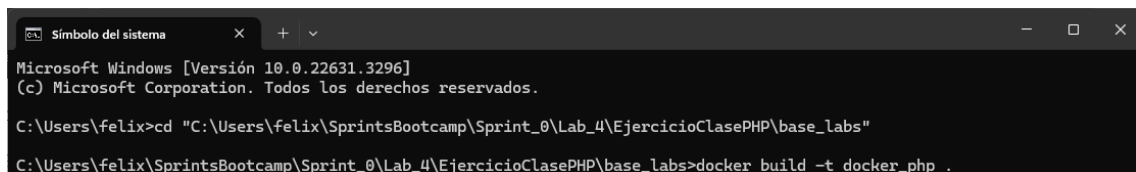
Estas son dos maneras que encontré para la configuración del entorno en mi máquina.

Primero me moví al directorio donde esta la “dockerfile” que descargue del bootcamp con el comando

```
cd <directorio_de_dockerfile>
```

y tras ese comando ejecute otro comando para montar la imagen

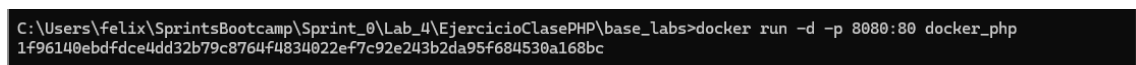
```
docker build -t <nombre_de_la_imagen>
```



*Ilustración 1- Creación de la imagen*

Tras crear la imagen construyo el contenedor con la imagen montada.

```
docker build -t <nombre_de_la_imagen> <ruta_de_la_imagen>
```



*Ilustración 2- Construir contenedor*

Para acceder en el navegador al index.php solo es necesario escribir <http://localhost:8080/> y podrás acceder al formulario

Otra manera de hacerlo usando el visual studio code.

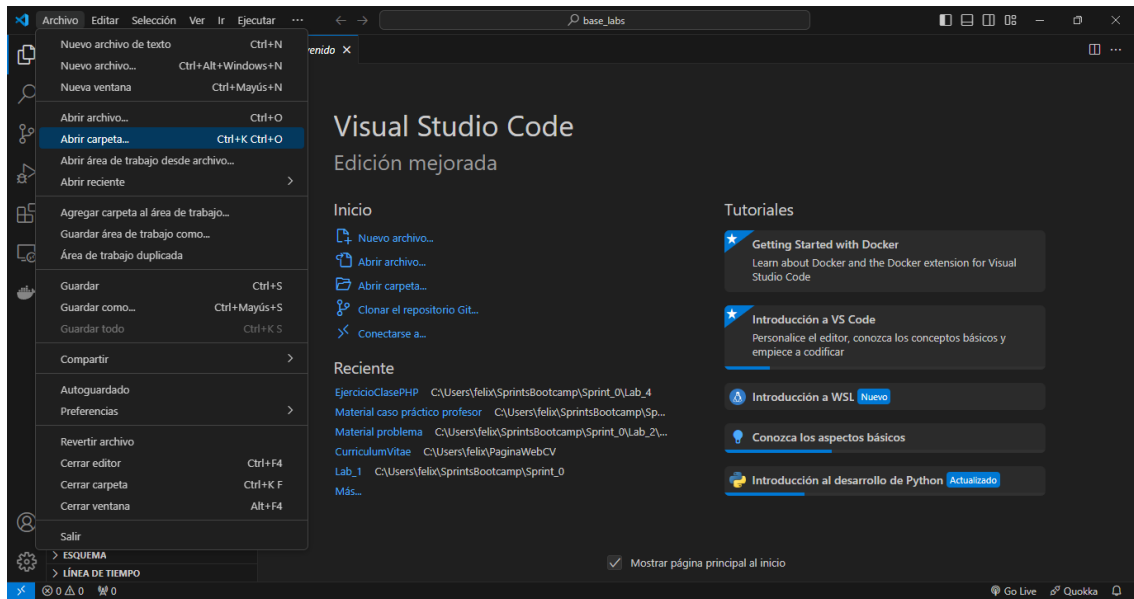


Ilustración 3- Abrir carpeta

Abres la carpeta donde está el proyecto y encima del index.php haces clic derecho y abrir un terminal integrado

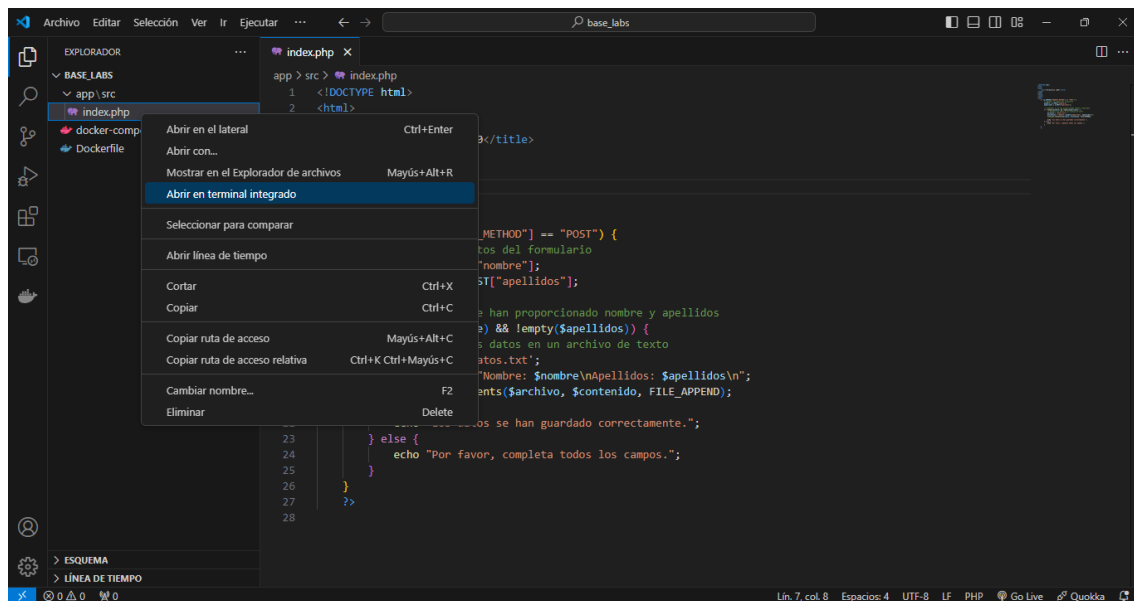


Ilustración 4- Abrir terminal

Abajo se abrirá un terminal y al escribir

```
docker init
```

saldrán diferentes opciones y pulsamos enter en

```
PHP with Apache – (detected) suitable for a PHP web application
```

Si se deseara otra se puede mover usando las flechas arriba y abajo del teclado.

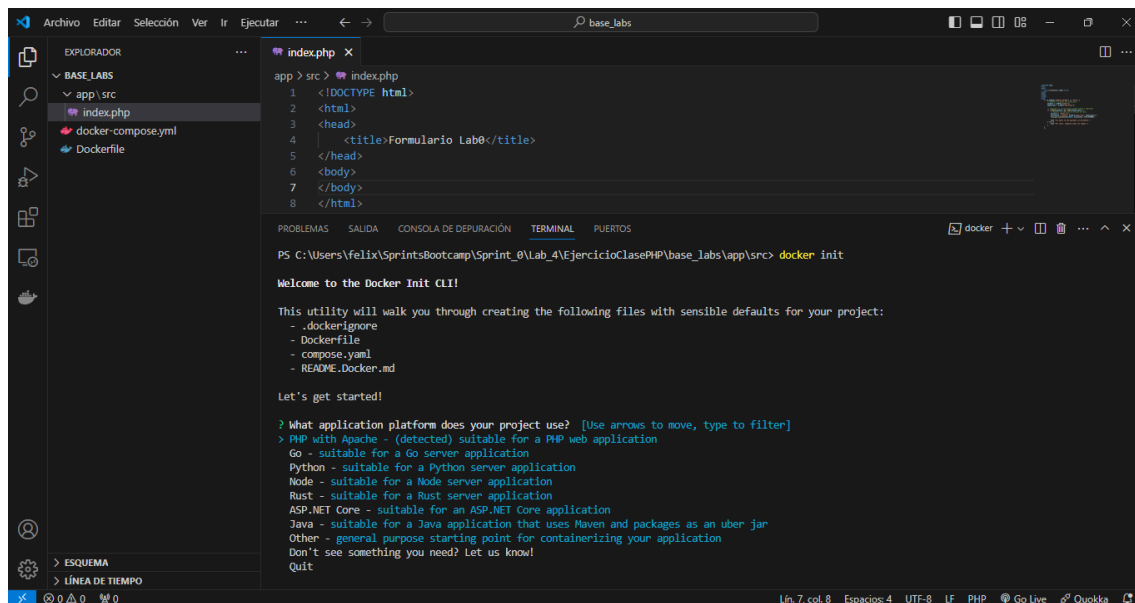


Ilustración 5- Crear contenedor en visual studio code

A continuación preguntara sobre que versión de php utilizar. En mi caso utilicé la versión estable mas reciente 8.3.4

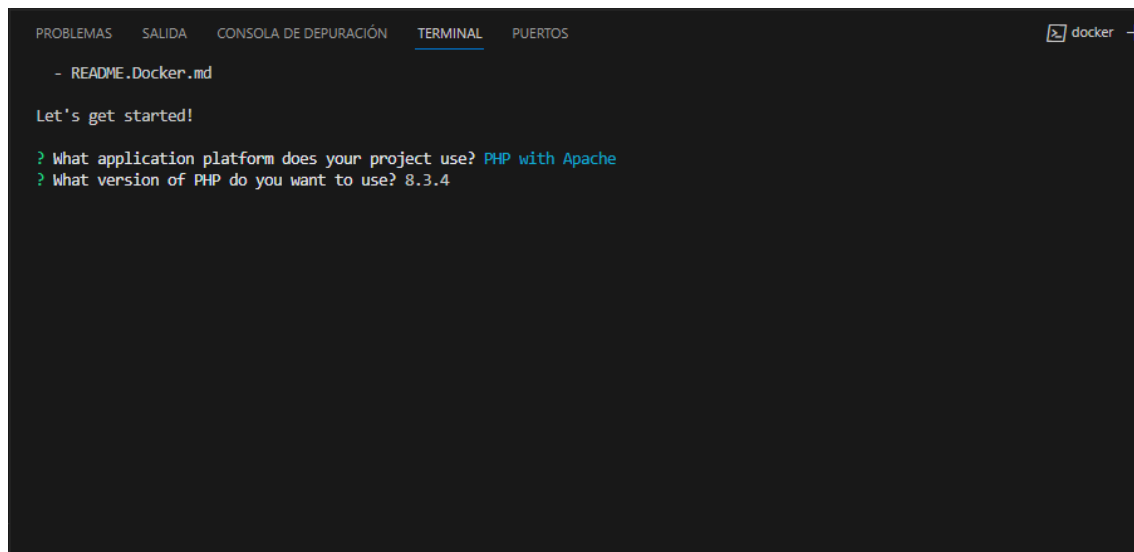
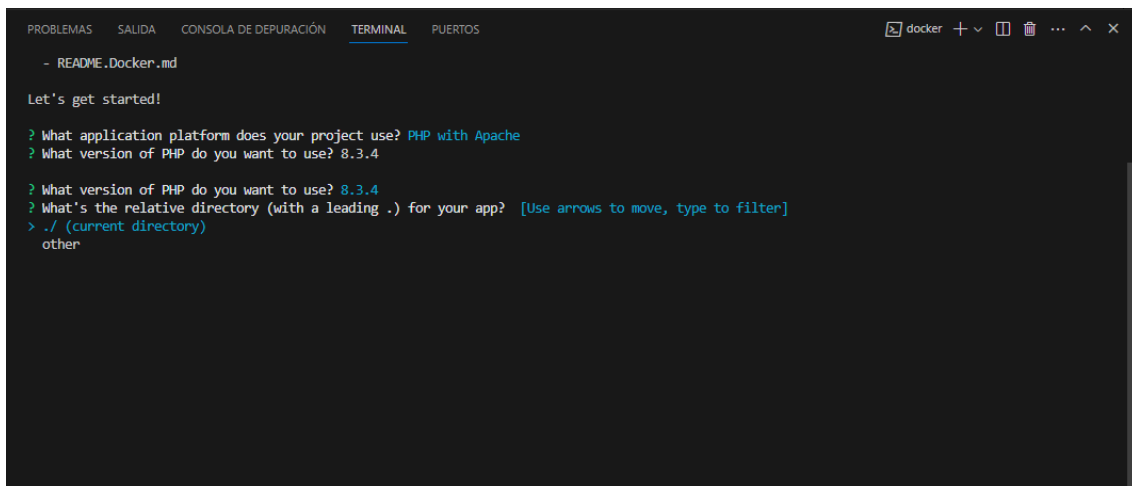


Ilustración 6- Versión de php

Te preguntara el directorio relativo donde ejecutar tu app que elegire el mismo directorio donde están alojados los archivos descargados del bootcamp que ofrecen el dockerfile y el Docker-compose.yml.



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS
- README.Docker.md

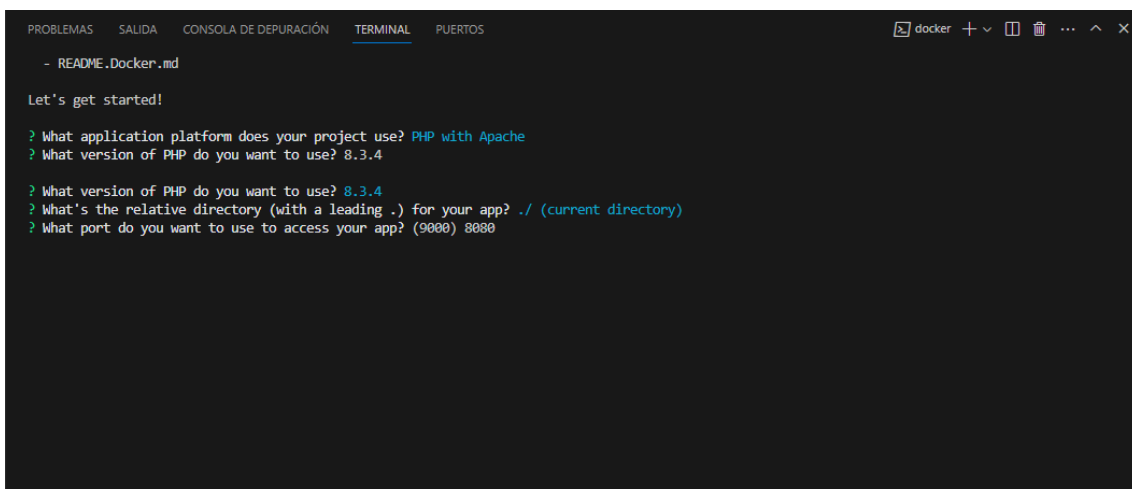
Let's get started!

? What application platform does your project use? PHP with Apache
? What version of PHP do you want to use? 8.3.4

? What version of PHP do you want to use? 8.3.4
? What's the relative directory (with a leading .) for your app? [Use arrows to move, type to filter]
> ./ (current directory)
other
```

*Ilustración 7- Directorio activo donde ejecutará el contenedor*

A continuación que puerto usara la aplicación yo escogí el 8080



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS
- README.Docker.md

Let's get started!

? What application platform does your project use? PHP with Apache
? What version of PHP do you want to use? 8.3.4

? What version of PHP do you want to use? 8.3.4
? What's the relative directory (with a leading .) for your app? ./ (current directory)
? What port do you want to use to access your app? (9000) 8080
```

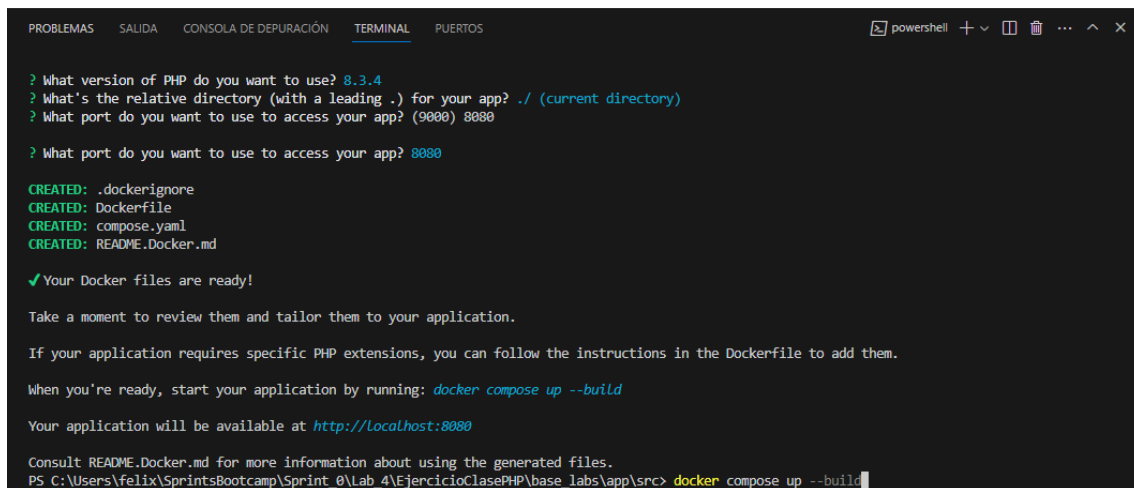
*Ilustración 8- Elección de puerto*

Para terminar de configurar Docker aparecerá este mensaje escribiendo y ejecutando el comando en la misma terminal se iniciaría la imagen el contenedor Docker pero antes de eso subiremos al directorio padre en el terminal con

```
cd ..
```

y después ya ejecutamos el comando

```
Docker compose up --build
```



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS
? What version of PHP do you want to use? 8.3.4
? What's the relative directory (with a leading .) for your app? ./ (current directory)
? What port do you want to use to access your app? (9000) 8080

? What port do you want to use to access your app? 8080

CREATED: .dockerignore
CREATED: Dockerfile
CREATED: compose.yaml
CREATED: README.Docker.md

✓ Your Docker files are ready!

Take a moment to review them and tailor them to your application.

If your application requires specific PHP extensions, you can follow the instructions in the Dockerfile to add them.

When you're ready, start your application by running: docker compose up --build

Your application will be available at http://localhost:8080

Consult README.Docker.md for more information about using the generated files.
PS C:\Users\felix\SprintsBootcamp\Sprint_0\Lab_4\EjercicioClasePHP\base_labs\app\src> docker compose up --build
```

*Ilustración 9- Fin del proceso de creación del contenedor*

La diferencia al hacerlo así es que será más fácil ver los ficheros de texto creados pues podremos verlos con el explorador de archivos de visual studio code ya que estará vinculado.

Si se hace de la otra manera con Docker Desktop habrá una consola con el directorio activo en

var/www/html

en ese directorio se encuentra index.php y donde se guardarán los archivos de texto si no se indica otra ruta y se escribe solamente el nombre del archivo.

Una vez terminada la configuración del entorno hay que crear el formulario en html y el script del formulario de php en index.php.

El formulario con las etiquetas form, label e input.



```
5 <form action="index.php" method="POST">
6   <label for="nombre">Nombre:</label><br>
7   <input type="text" id="nombre" name="nombre"><br>
8
9   <label for="edad">Edad:</label><br>
10  <input type="text" id="edad" name="edad"><br>
11
12  <label for="descripcion">Descripción:</label><br>
13  <input type="text" id="descripcion" name="descripcion"><br><br>
14
15  <input type="submit" value="Enviar">
16 </form>
```

*Ilustración 10- Formulario*

El script es un bloque if que verifica que existe la petición POST y dentro declara las variables y las guarda en un array. Otro bloque if comprueba que el array no contenga ningún valor vacío para guardar los datos en los ficheros.

```

if(existePost()){

    $nombre = isset($_POST['nombre']) ? $_POST['nombre'] : '';
    $edad = isset($_POST['edad']) ? $_POST['edad'] : '';
    $descripcion = isset($_POST['descripcion']) ? $_POST['descripcion'] : '';

    $array = generarArray($nombre,$edad,$descripcion);

    if(noVacios($array)){
        escribirNombreDescripcion($array);
        escribirEdadMostrarMedia($array);
        // echo $array['nombre'];
        // echo $array['edad'];
        // echo $array['descripcion'];
    }
    else{
        echo "ELEMENTOS VACIOS";
    }
}

```

*Ilustración 11- Verificación de la petición POST*

La función existePost() es un bloque if con cuatro condiciones que debe de cumplir y retornara true si se cumplen las cuatro.

```

22 function existePost(){
23     if($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['nombre']) && isset($_POST['edad'])
24     && isset($_POST['descripcion'])){
25         return true;
26     }
27     return false;
28 }

```

*Ilustración 12- existePost()*

La función generarArray() recibe los tres inputs nombre edad y descripción y retorna un array que contiene estas tres variables.

```

29 function generarArray($nombre,$edad,$descripcion){
30     return $array = [
31         "nombre" => $nombre,
32         "edad" => $edad,
33         "descripcion" => $descripcion
34     ];
35 }

```

*Ilustración 13- generarArray()*

La función noVacios() recibe un array como parámetro con un bucle foreach si un elemento de ese array está vacío retornará false.

```

36 function noVacios($array){
37     foreach ($array as $clave => $valor){
38         if($valor == ''){
39             return false;
40         }
41     }
42     return true;
43 }

```

*Ilustración 14- noVacios()*

Las dos funciones siguientes escribirNombreDescripcion() y escribirEdadMostrarMedia()



abrirán sus respectivos ficheros nombre\_y\_descripcion.txt y edades.txt o los crearán si no existen y escribirán los valores recibidos.

escribirEdadMostrarMedia() además leerá el fichero edades.txt recorrerá los valores recogidos con un bucle for para sumar todas las edades, tras el bucle sumara la recibida por el usuario y calculara y mostrara la media de edades.

```
44 function escribirNombreDescripcion($array){
45     $fichero = fopen('nombre_y_descripcion.txt','a');
46     fwrite($fichero,"Nombre: {$array['nombre']}\nDescripcion:{$array['descripcion']}\n");
47     fclose($fichero);
48 }
```

*Ilustración 15- escribirNombreDescripcion()*

```
49 function escribirEdadMostrarMedia($array){
50     $fichero = fopen('edades.txt','a');
51     $valores = file('edades.txt', FILE_IGNORE_NEW_LINES | FILE_SKIP_EMPTY_LINES);
52     $mediaEdades = 0;
53     $i = 0;
54     for ( ; $i < count($valores) ; $i++){
55         $mediaEdades+=$valores[$i];
56     }
57     $mediaEdades+=$array['edad'];
58     $i++;
59     $mediaEdades=$mediaEdades/$i;
60     echo "La media de edades es:",$mediaEdades;
61     fwrite($fichero,"{$array['edad']}\n");
62     fclose($fichero);
63 }
```

*Ilustración 16- escribirEdadMostrarMedia()*